Clasificación: árboles de decisión

Ramon Sangüesa i Solé

PID_00165729

Universitat Oberta de Catalunya
www.uoc.edu

Índice

Introducción

Los árboles de decisión son uno de los modelos de clasificación más utilizados. Su interpretación bastante natural, su relación con técnicas de clasificación establecidas y probadas, en particular, y también los distintos métodos existentes para traducir los modelos resultados en colecciones de reglas de clasificación los hacen versátiles y aplicables.

Objetivos

El estudio de los materiales asociados a este programa hará que el estudiante alcance los objetivos siguientes:

1. Conocer el origen y las ideas fundamentales que hay detrás de los árboles de decisión.

2. Saber los detalles de los métodos de construcción más frecuentes y utilizados.

1. Introducción: la estructura de los árboles decisión

La mejor manera de empezar es con el árbol de decisión resultante de aplicar un método bastante conocido, el método ID3, a un conjunto de datos sobre los clientes del gimnasio.

El conjunto de datos sobre los clientes del gimnasio consiste en una serie de registros que reúnen los valores de los datos siguientes:

- Identificador del cliente.

- Nivel físico del cliente. Este atributo nos indica, según los resultados de las pruebas físicas a que ha sido sometido el cliente al inscribirse al gimnasio, si su estado físico es lo suficientemente bueno. Los valores que toma este atributo son 'Bajo', 'Normal' y 'Alto', valores que corresponden a un estado físico cada vez mejor.

- Intensidad de la actividad 1. Hace referencia al nivel de esfuerzo que corresponde a la actividad deportiva principal que desarrolla al cliente. Este atributo adopta valores en las categorías 'Baja', 'Media' y 'Alta'.

- Intensidad de la actividad 2. Este atributo indica lo mismo que el atributo anterior, pero en referencia a la segunda actividad que desarrolla el cliente.

- Clase. Atributo que toma los valores 0 ó 1, y que indica si el cliente efectúa o no una combinación correcta de nivel y actividad. Esta asignación a cada clase ha sido efectuada manualmente un especialista en preparación deportiva.

Cliente	Nivel físico	Intensidad de la actividad 1	Intensidad de la actividad 2	Clase
1	Normal	Media	Baja	0
2	Bajo	Baja	Baja	1
3	Normal	Alta	Alta	1
4	Alto	Alta	Alta	1
5	Alto	Baja	Baja	0
6	Bajo	Media	Media	1
7	Normal	Media	Media	0
8	Alto	Alta	Alta	1

El problema que se plantean los métodos de construcción de árboles de decisión es el siguiente: ¿cuál es la mejor secuencia de preguntas para saber, a partir de la descripción de un objeto en términos de sus atributos, a qué clase corresponde? Evidentemente, "la mejor secuencia" puede entenderse como aquella que, con el mínimo número de preguntas, devuelve una respuesta lo suficientemente detallada.

En otras palabras, la "mejor secuencia" es aquella que efectúa las preguntas más discriminantes, es decir, aquéllas cuyas respuestas permiten descartar un número más amplio de objetos diferentes del que estamos considerando y, por tanto, llegar con mayor rapidez a decidir a qué clase pertenece.

Ejemplo de mejor secuencia de preguntas

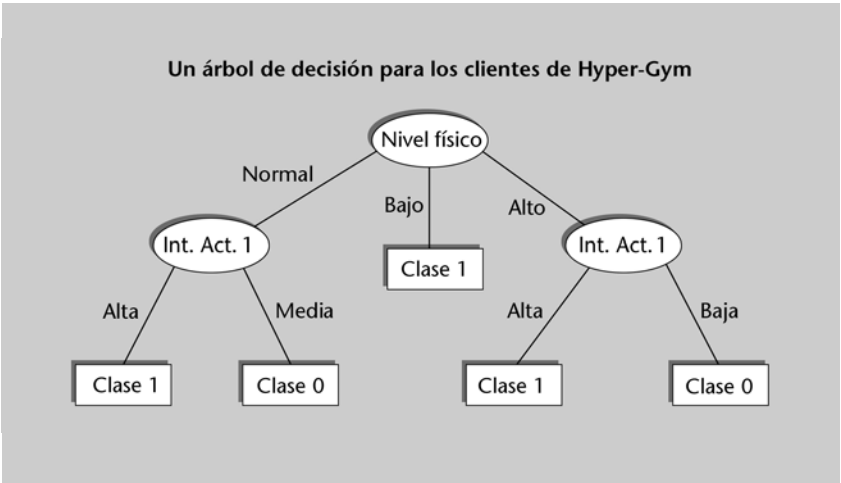
¿Con el fin de establecer, por ejemplo, si un cliente pertenece a la clase 0 o a la clase 1, ¿qué es mejor? ¿Preguntar primero por su estado físico o por la intensidad de la primera actividad que desarrolla?

¿Qué tiene que ver todo esto con un árbol? Pues que la manera de organizar las preguntas conduce con bastante naturalidad a una estructura arbórea. En efecto, un árbol de decisión está formado por nodos (nodos de decisión) que efectúan una pregunta sobre el valor de un atributo. Las diferentes respuestas que pueden darse generan otros nodos de decisión. De esta manera, pues, se va construyendo el árbol.

El interés de estos modelos es que, además, permiten una traducción bastante natural a listas de reglas de decisión.

Ejemplo de construcción de un árbol de decisión

En el caso de los niveles físicos e intensidades de ejercicio de un cliente de Hyper-Gym, podemos obtener la estructura siguiente:



Será suficiente con repasar el camino que va dentro del árbol desde la raíz hasta cada una de las hojas para ir construyendo una conjunción de condiciones sobre los valores de los atributos (lo que se pregunta a cada nodo) y, de esta manera, construir el antecedente de una regla de decisión. Este árbol puede traducirse en el conjunto de reglas siguiente:

Si (*Nivel_físico* = 'Normal') & (*intensidad_actividad_1* = 'Media') entonces *clase* = 0
Si (*Nivel_físico* = 'Normal') & (*intensidad_activitat_1* = 'Alta') entonces *clase* = 1
Si (*Nivel_físico* = 'Bajo') entonces *clase* = 1
Si (*Nivel_físic'* = 'Alto') & (*intensidad_actividad_1* = 'Alto') entonces *clase* = 1
Si (*Nivel_físic'* = 'Alto') & (*intensidad_actividad_1* = 'Baja') entonces *clase* = 0

Los métodos de construcción de árboles de decisión empiezan por tratar de obtener una condición, una prueba sobre los valores de un atributo, tal que efectúe la mejor partición sobre los conjuntos de datos, es decir, que separe los distintos casos en grupos tan diferentes entre sí como sea posible, pero de manera que los casos que se hallan dentro de cada partición tengan la máxima semejanza posible entre sí, ya que ése es el objetivo de un procedimiento de clasificación. Se consigue la máxima homogeneidad cuando todos los casos que aparecen en una partición pertenecen a la misma clase.

Así pues, lo importante en los métodos de construcción de árboles de decisión es elegir el atributo que ofrece una mejor separación de los casos existentes.

Con pequeñas variaciones en función de los distintos métodos, el procedimiento de construcción del árbol procede de manera iterativa. Así, si el primer nodo que se considera es un atributo que posee tres valores posibles (por ejemplo, 'Alto', 'Bajo' y 'Normal'), entonces el conjunto de datos se parte en tres subconjuntos que corresponden a aquellos casos para los cuales el atributo correspondiente tiene el valor 'Alto', el valor 'Normal' y el valor 'Bajo'. La pregunta para el valor 'Alto' se convierte en un nodo de decisión, y también las preguntas o condiciones sobre los valores 'Bajo' y 'Normal'. En este momento, el árbol ya tiene un nivel.

A las particiones de los datos que resultan se les vuelve a aplicar el procedimiento hasta que se cumple una condición de final, diferente según cada método, pero que en general podemos decir que intenta obtener particiones en las que todos los casos que le corresponden pertenecen a la misma clase, o en las que la mayoría de los casos son de la misma clase.

Por ejemplo, si prestamos atención a esta relación entre los valores de los atributos y los casos dentro de la base de datos en que, efectivamente, se cumple lo que acabamos de plantear, podemos ver que al primer nivel del árbol de los clientes e intensidades de ejercicio tenemos esta primera partición:

Condición	Casos (clientes)	Clase a la que pertenece
Nivel físico = 'Normal'	1	0
	3	1
	7	0
Nivel físico = 'Bajo'	2	1
	6	1
Nivel físico = 'Alto'	4	1
	5	0
	8	1

Podemos observar que en este nivel el valor 'Bajo' del atributo *Nivel físico* es suficiente para agrupar correctamente dentro de la misma partición dos casos

que pertenecen a la misma clase, es decir, que da lugar a una partición (un subconjunto de los datos originales) completamente homogénea porque todos sus casos pertenecen a la clase 1.

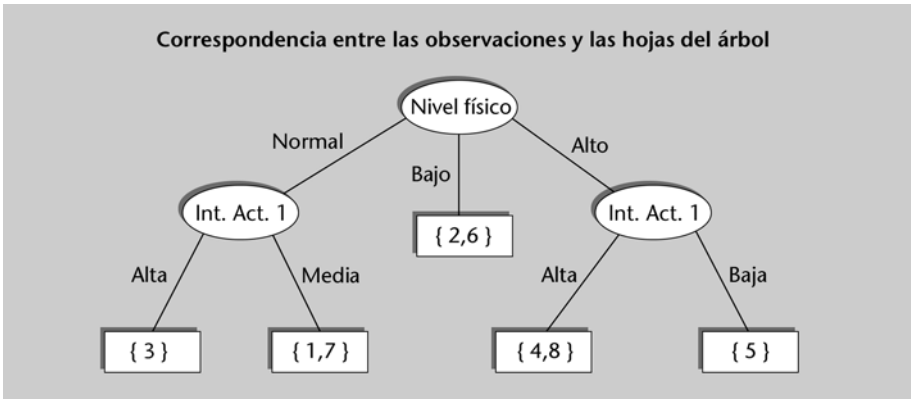
De esta partición podemos deducir que el valor 'Bajo' de ese atributo discrimina bastante bien, pero los otros dos ('Normal' y 'Alto'), no tanto, ya que las particiones correspondientes reúnen casos de la clase 0 y de la 1. Es preciso utilizar los otros atributos para ver si podemos obtener particiones igualmente homogéneas.

Fijémonos en las condiciones que corresponden a las preguntas sobre los valores del atributo *Intensidad de la actividad 1*. Colocamos aquí las combinaciones de valores correspondientes a los caminos posibles dentro del árbol:

Condición	Casos (clientes)	Clase a la que pertenece
Nivel físico = 'Normal' & *Intensidad de la actividad 1* = 'Medio'	1	0
	7	0
Nivel físico = 'Normal' & *Intensidad de la actividad 1* = 'Alta'	3	1
Nivel físico = 'Alto' & *Intensidad de la actividad 1* = 'Alta'	4	1
	8	1
Nivel físico = 'Alto' & *Intensidad de la actividad 1* = 'Baja'	5	0

En este caso, podemos ver que todas las particiones que inducen las distintas condiciones son completamente homogéneas. Todos los casos que tienen un nivel físico medio y una primera actividad con intensidad media pertenecen a la clase 1; todos los casos con nivel físico medio y primera actividad con intensidad alta pertenecen a la clase 0. Y así sucesivamente.

En la figura siguiente podemos ver que a cada rama le corresponden los distintos casos:



En realidad, pues, los algoritmos de construcción de árboles de decisión efectúan una partición del espacio determinada por los atributos que se conside-

ren. En el caso del ejemplo, en el que se consideran, obviando el identificador de cliente, tres atributos (*Nivel físico*, *Intensidad de la actividad 1 e Intensidad de la actividad 2*) determinan regiones en el espacio de tres dimensiones. La respuesta que dan las cadenas de preguntas que están representadas por las distintas reglas es en qué subregión de este espacio hay que ubicar un caso nuevo.

Actividad

1.1. Meditad un momento por qué en el ejemplo que acabamos de exponer no hay que preguntar por la intensidad de la segunda actividad física desarrollada por el cliente.

La clave de la cuestión para cada método es cómo hay que encontrar, sobre qué atributos hay que preguntar y en qué secuencia para obtener árboles que tengan la máxima capacidad predictiva y, al mismo tiempo, cumplan otros parámetros de calidad (por ejemplo, que no tengan demasiados niveles).

En otras palabras, la clave reside en cómo conseguimos que en cada paso se obtenga lo que hemos llamado **la mejor partición**.

Los distintos métodos que se utilizan para alcanzar este objetivo se distinguen por las características siguientes:

- Capacidad para utilizar atributos con más de dos valores.
- Uso de diferentes medidas de homogeneidad de las clases.
- Capacidad para preguntar en cada nodo sobre más de un solo atributo.

A continuación revisaremos los métodos más conocidos: ID3, C4.5, CART, CHAID y LMDT.

2. Métodos de construcción de árboles de decisión para clasificación: ID3 y C4.5

El método ID3 se debe a J. Ross Quinlan, un investigador australiano que lo propuso en 1986 (Quinlan, 1986). El acrónimo que da nombre al método corresponde a la expresión inglesa *Iterative Dichotomizer 3*, es decir: 'dicotomizador iterativo 3'. En otras palabras, es un proceso que repite un "corte en dos" hasta que se cumple una determinada condición.

Introduciremos algo de nomenclatura y notación:

Nomenclatura y notación.

Supongamos que disponemos de un conjunto de m casos. Designamos este conjunto de casos por $X = \{x_j\}_{j = 1,m}$. Debemos recordar que cada x_j es, en realidad, una tupla definida con respecto al conjunto de atributos de partida.

- El **conjunto de atributos de partida** es $A = \{A_k\}_{k = 1,n}$. En el caso del ejemplo tenemos:

$$A_1 = \textit{Nivel físico}$$
$$A_2 = \textit{Intensidad de la actividad 1}$$
$$A_3 = \textit{Intensidad de la actividad 2}$$

- El **conjunto de casos** es $X = \{x_j\}_{j=1,8}$, donde, por ejemplo, x_3 = {'Normal', 'Alta', 'Alta'}.

- El **conjunto de valores** de un atributo, A_i, se denota por $V(A_i)$. Por ejemplo, $V(A_1)$ = {'Alto', 'Normal', 'Bajo'}. El valor que toma un A_i para un caso x_j se denota por $A_i(x_j)$. Por ejemplo, el valor del atributo 3 para el caso 2 es $A_3(x_2)$ = Baja.

- El conjunto de casos que adoptan un determinado valor v para un atributo A_i determinado se denota por $A_i^{-1}(v)$.

$$A_i^{-1}(v) = \{x \in X: A_i(x) = v\}$$

Por ejemplo, el conjunto de casos con nivel físico bajo corresponde a los casos en que el primer atributo es igual a 'Bajo'. Formalmente, con esta notación:

$$A_1^{-1} = (\text{'Bajo'}) = \{2, 6\}$$

- El conjunto de clases se denota por $C = \{C_i\}_{i = 1,k}$. En nuestro ejemplo $k = 2$.

El **método ID3** trata de encontrar una partición que asegure la máxima capacidad predictiva y la máxima homogeneidad de las clases.

2.1. Medidas de homogeneidad

Para medir la homogeneidad de cada subconjunto hay varias medidas. Estas medidas buscan, de una manera u otra, asignar valores extremos al caso en que la partición que se considere contenga sólo casos de una única clase. También nos permiten comparar particiones no del todo uniformes, teniendo en cuenta que su diversidad interna queda reflejada numéricamente, y que eso suministra una base para la comparación.

A continuación comentaremos las diferentes medidas de desorden de que hemos hablado. De hecho, se trata de ver qué grado de variedad o de desorden hay en la partición que resulta de fijar un valor v determinado entre los distintos valores que puede tomar un atributo A_i.

Ejemplo de medida de desorden

Podemos elegir el atributo *Nivel físico*. Vemos que el valor 'Bajo' genera una partición que es más homogénea que la que genera el valor 'Alto' o la generada por el valor 'Normal':

Valor del atributo *Nivel físico*	Casos (*Clientes*)	Clase a las que pertenece
Nivel físico = 'Normal'	1	0
	3	1
	7	0
***Nivel físico* = 'Bajo'**	**2**	**1**
	6	**1**
Nivel físico = 'Alto'	4	1
	5	0
	8	1

Si fijamos la proporción de casos de la clase 0 como:

$$\frac{\text{Número de casos de clase 0}}{\text{Número de casos de la partición}}$$

tenemos que cada partición presenta las proporciones siguientes como medidas primitivas de diversidad de cada una de éstas:

Partición	Valor del atributo *Nivel físico*	Casos	Proporción
1	Normal	{1, 3, 7}	2/3
2	Bajo	{2, 6}	0
3	Alto	{4, 5, 8}	1/3

Bajo esta medida de desorden, la partición 2, que corresponde a los casos en que el nivel físico es bajo, presenta la máxima homogeneidad (valor de proporción 0). Las otras dos particiones todavía no son bastante homogéneas, pero según el criterio que hemos utilizado, la partición 1 parece más desordenada o menos homogénea que la partición 2 (2/3 > 1/3). ¿Qué comportamiento presentan los otros atributos? Lo vemos en las tablas que encontramos a continuación:

Valor del atributo *Intensidad de la actividad 1*	**Casos** (*Clientes*)	**Clase a la que pertenece**	**%**
Baja	2	1	50
	5	0	
Media	1	0	66
	6	1	
	7	0	
Alta	3	1	0
	4	1	
	8	1	

Valor del atributo *Intensidad de la actividad 2*	**Casos** (*Clientes*)	**Clase a la que pertenece**	**%**
Baja	1	0	66
	2	1	
	5	0	
Media	6	1	50
	7	0	
Alta	3	1	0
	4	1	
	8	1	

La proporción que hemos propuesto en este ejemplo no es más que una medida orientativa a modo de explicación.

Normalmente, se asigna a este caso el valor 0, y cuanto más heterogénea sea la partición, es decir, cuantos más casos procedentes de clases diferentes tenga, entonces la medida adquiere un valor mayor. De alguna manera, intenta medir el "desorden" existente dentro de cada partición.

La entropía.

La medida del desorden típica es la **entropía**, que es una propiedad de las distribuciones de probabilidad de los distintos atributos y que ahora comentamos con más detalle.

La **medida del desorden** es una "medida de información". En efecto, la información se ha asociado con el grado de "sorpresa" que aporta una señal. Si esperamos que un dato tenga un valor determinado, si, por ejemplo, sabemos que normalmente la renta de un cliente es de 3,5 millones y aparece un cliente con treinta millones de renta, entonces tenemos que considerar este hecho

como "sorprendente"; es decir, es un valor más informativo que otros más próximos a los tres millones.

La propiedad anterior se expresa relacionando los valores que puede tomar una variable con la probabilidad de aparición que tiene, esto es, con la **distribución de probabilidad de una variable**.

El desorden se mide en bits. Si tenemos una variable X que puede adoptar tomar los valores $x_1, \ldots, x_n$ según una distribución $P(X)$, entonces el grado por sorpresa de cada valor tiene que estar en relación con la probabilidad de aparición de dicho valor. En principio, valores más bajos de probabilidad tendrían que ser más sorprendentes.

La forma de encontrar los bits de información contenidos en la distribución de probabilidad de un atributo viene determinada por la expresión que damos a continuación:

$$I(X) = -\sum_{1}^{n} \log_2 p(x_i)$$

Teniendo en cuenta que los logaritmos para valores reales entre 0 y 1 son negativos, la suma total es positiva.

¿Qué tiene que ver el desorden con la homogeneidad de una clase? Fijaos en que esta propiedad se encuentra relacionada con un atributo que tomamos como decisivo para calificar una clase de homogénea. Tenemos que intentar obtener una clase homogénea, o sea, en la que predomine uno de los valores del atributo. En consecuencia, necesitamos una medida que refleje el hecho de que, si todos los valores son lo mismo, la calidad es máxima. Eso es precisamente lo que hace la medida de información.

La ganancia de información.

De hecho, en cada paso de la construcción de un árbol no nos interesa el atributo que aporta más información, sino aquel que nos da la máxima diferencia de información con respecto a la partición actual si efectuamos la partición teniendo en cuenta sus valores. Es decir, nos interesa la máxima **ganancia de información**. Se trata, pues, de elegir en cada paso aquel atributo que maximice la ganancia de información.

Dada una partición, hallaremos la **ganancia de información** haciendo la aproximación de que la probabilidad de cada valor es su frecuencia de aparición observada.

El **ganancia de información** queda definida de la manera siguiente:

$$G(X,A_k) = I(X,C) - E(X,A_k)$$

donde cada elemento se explica a continuación:

a) $I(X,C)$ es la información asociada a las particiones inducidas por un conjunto de clases dado, C, con respecto al conjunto de casos X. Se define de la manera siguiente:

$$I(X,C) = -\sum_{C_i \in C} p(X,c_i) \log_2 p(X,c_i)$$

donde $p(X,c_i)$ es la probabilidad de que un ejemplo pertenezca a una clase dada c_i. Se acostumbra a dar la aproximación de que esta probabilidad es la frecuencia de aparición observada de casos que pertenecen a la clase c_i, que es el estimador de esta probabilidad:

$$p(X, c_i) = \frac{\#c_i}{\#X}$$

En la expresión anterior, $\#c_i$ es el número de casos del conjunto de casos X que pertenecen a la clase c_i, y $\#X$ es el número de casos total del conjunto de casos.

Ejemplo de obtención de la probabilidad

En el ejemplo que hemos ido siguiendo a lo largo de este apartado sobre el nivel físico de los clientes de la cadena de gimnasios Hyper-Gym, el número de casos sería $\#X = 8$. Entonces, la probabilidad de que un caso pertenezca a la clase 0 es:

$$P(X,C_0) = \frac{3}{8}$$

b) $E(X,A_k)$ es la información esperada del atributo A_k en lo referente al conjunto de casos X. Es decir, indica el nivel de diversidad de este atributo (los distintos valores que puede tomar) en el conjunto de casos X.

$E(X,A_k)$ es una medida de la diversidad que introduce en las particiones el hecho de escoger el atributo A_k. Se expresa de la manera que vemos a continuación:

$$E(X, A_k) = \sum_{v_i \in v(A_k)} p(X, A_k^{-1}(v_i)) \cdot IA_k^{-1}(v_i), C)$$

donde $p(X, A_k^{-1}(v_i))$ es la probabilidad de que un caso tenga el valor v_i en el atributo A_k. Corresponde a hacer la aproximación a partir de las frecuencias observadas siguientes:

$$p(X, A_k^{-1}(v_i)) = \frac{\#A_k^{-1}(v_i)}{\#X}$$

Es decir, corresponde al número de casos que muestran el valor v_i en el atributo A_k con respecto al total de casos existentes.

La **ganancia de información** mide en qué grado un atributo determinado hace aumentar o disminuir el desorden de las particiones que puede haber a partir de comparar el desorden que induce con respecto al existente en un momento determinado. La función $I(X,C)$ mide la diversidad de valores con respecto a las clases existentes.

Problemas con la ganancia.

La medida de ganancia de información presenta un problema: los atributos que adoptan muchos valores diferentes. Entre los atributos X_1, que toma los valores {'Sí', 'No'}, y X_2, que toma valores {'Alto', 'Bajo', 'Medio'}, normalmente preferirá el segundo. ¿Por qué? Lo veremos ahora mismo.

Supongamos el caso de un atributo que tenga un valor diferente para cada observación del conjunto de datos. En una situación como ésta, ¿cuál sería su información?

$$I(X) = -\sum_{1}^{n} \log_2 p(x_i) = 0$$

porque cada partición que generáramos sería completamente homogénea. En consecuencia, sería el atributo que nos daría siempre la máxima ganancia con respecto al último test efectuado.

Lectura recomendada

Encontraréis más información con relación a otros métodos que utilizan las medidas de homogeneidad y las repercusiones que comporta la elección en la obra siguiente:

R. López de Màntaras (1991). "A Distance-Based Attribute Selection Measure for Decision Tree Induction". *Machine Learning* (vol. 6, núm. 1, pág. 81-92).

Una medida que intenta compensar esta tendencia es la **medida de razón de la ganancia de información** (López de Màntaras, 1991). Esta medida compensa el efecto mencionado teniendo en cuenta el número de particiones posibles que generaría un atributo y su tamaño respectivo (sin tener en cuenta ninguna información sobre la clase).

Las medidas comentadas se utilizan para otros métodos, y la elección de una o de otra presenta repercusiones importantes en cuanto a los tipos de atributos que se escogen y la forma final del árbol que se obtiene.

2.2. Formalización del algoritmo ID3

Formalizamos, a continuación, el algoritmo ID3 de construcción de árboles de decisión siguiendo la notación que hemos propuesto antes. Lo presentamos en forma de función recursiva. La función ID3 toma como parámetros de entrada dos conjuntos, X y A. El primero corresponde al conjunto de casos, y el segundo, al conjunto de atributos. Mantenemos la notación que hemos presentado al principio. La función crear_árbol(A_k) crea un árbol que tiene como nodo raíz el atributo A_k. Cuando este atributo corresponde a la etiqueta de una clase, lo denotamos por crear_árbol(C_i).

Haremos un seguimiento de un pequeño ejemplo con el fin de ver cómo se va construyendo progresivamente el árbol a partir de un conjunto de datos minúsculo y ficticio. Aquí lo tenemos:

Función ID3 (**ent** X, A **son** conjuntos) **retorna** árbol_de_decisión
var *árbol1*, *árbol2* **son** árbol_de_decisión;
A_{max} **es** atributo
opción
  caso: $(\exists C_i \forall x_j \in X \Rightarrow x_j \in C_i)$ **hacer**
  {Todos los casos son de una misma clase}

árbol1 := crear_árbol(C_i)
caso: **no** ($\exists C_i \forall x_j \in X \Rightarrow x_j \in C_i$) **hacer**
opción:
caso $A \neq \varnothing$ **hacer** {Hay algún atributo para tratar}
$A_{max} := max_{A_k \in A} \{G(X, A_k\}$
{Escoger el atributo A_{max} que maximiza el beneficio}
árbol1 := crear_árbol(A_{max}) {Crear un árbol con raíz A_{max}}
Para todo $v \in V(A_{max})$ **hacer**
árbol2 := ID3($A^{-1}{}_{max}(v)$, $A - \{A_{max}\}$);
{Aplicar ID3 al conjunto de casos con valor v al
atributo A_{max}, pero sin considerar este atributo}
árbol1 := añadir_rama(*árbol1*,*árbol2*,*v*);
{conectar el árbol resultante}
fpara
caso A = $\varnothing$ **hacer**
árbol1 := crear_árbol(clase_mayoritaria(X))
fopción
regresa *árbol1*
ffunción

Caso	Horario	Sexo	Nivel físico	Clase
1	Mañana	Hombre	Alto	0
2	Mañana	Mujer	Normal	0
3	Mediodía	Mujer	Normal	1
4	Tarde	Mujer	Normal	1
5	Tarde	Mujer	Alto	0
6	Mediodía	Mujer	Bajo	1
7	Tarde	Hombre	Bajo	1
8	Mañana	Mujer	Normal	0

1) Empezamos el proceso. Consiste en los cálculos que veremos a continuación.

a) Medimos, en primer lugar, la información contenida en el conjunto inicial de datos:

$$I(X, C) = -\frac{1}{2}\log_2\left(\frac{1}{2}\right) - \frac{1}{2}\log_2\left(\frac{1}{2}\right) = 1$$

El primer término corresponde a los casos de la clase 0 (1, 2, 5 y 8), y el segundo, a los de la clase 1 (3, 4, 6 y 7).

b) Medimos la entropía aportada por cada uno de los atributos:

- $E(X, Sexo) = \frac{2}{8}(-\frac{1}{2}\log_2\frac{1}{2} - \frac{1}{2}\log_2\frac{1}{2}) + \frac{6}{8}(-\frac{3}{6}\log_2\frac{3}{6} - \frac{3}{6}\log_2\frac{3}{6}) = 1$

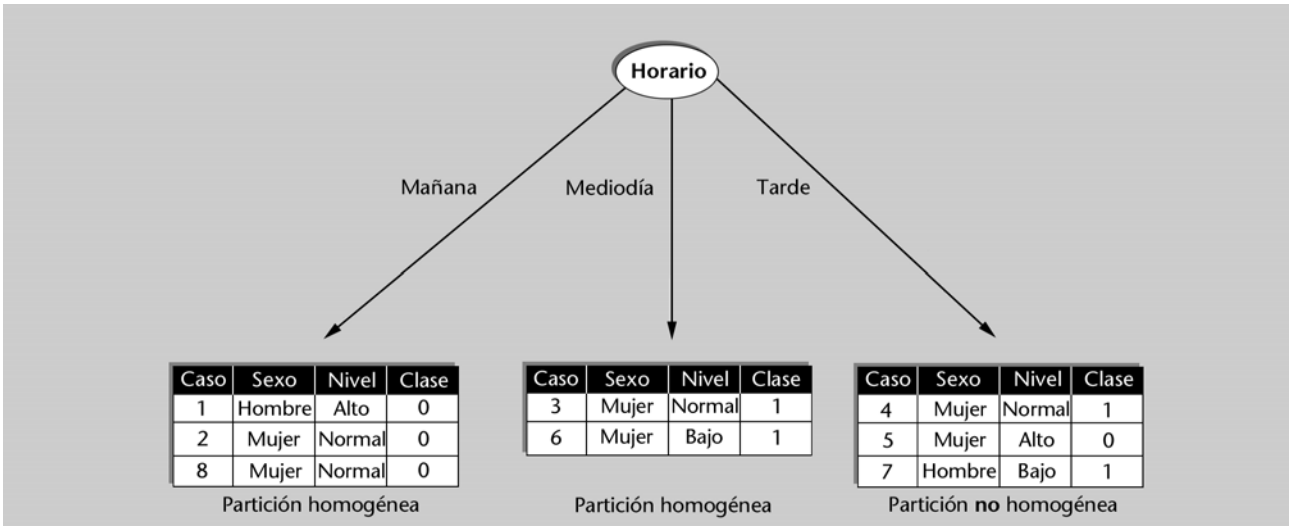
- $E(X, Horario) = \frac{3}{8}(-1\log_2 1 - 0\log_2 0) + \frac{2}{8}(-0\log_2 0 - 1\log_2 1) +$
$+ \frac{3}{8}(-\frac{1}{3}\log_2\frac{1}{3} - \frac{2}{3}\log_2\frac{2}{3}) = 0{,}344$

- $E(X, Nivel\ físico) = \frac{2}{8}(-1\log_2 1 - 0\log_2 0) + \frac{4}{8}(-\frac{1}{2}\log_2\frac{1}{2} - \frac{1}{2}\log_2\frac{1}{2}) +$
$+ \frac{2}{8}(-0\log_2 0 - 1\log_2 1) = 0{,}5$

c) En la tabla que vemos a continuación tenemos la ganancia para cada uno de los atributos:

Atributo	$I(X,C) - E(X,A_k)$
Horario	1 – 0,344 = 0,656
Sexo	1 – 1 = 0
Nivel físico	1 – 0,5 = 0,5

Así pues, el atributo que aporta un cambio más importante hacia la consecución de particiones más homogéneas es *Horario*. Por lo tanto, la construcción del primer nivel del árbol corresponde al esquema siguiente:



Como podemos ver, no todas las particiones son homogéneas, de manera que será necesario iterar el proceso con los atributos que todavía no se han tratado.

En primer lugar, tenemos que encontrar cuál de los atributos {*Sexo*, *Nivel*} es el que tiene más ganancia.

2) Iteramos la aplicación de las fórmulas que ya conocemos:

a) Calculamos la información inicial de la partición que interesa:

$$I(X, C) = -\frac{1}{3}\log_2\left(\frac{1}{3}\right) - \frac{2}{3}\log_2\left(\frac{2}{3}\right) = 0{,}918$$

El primer término corresponde al caso 5, y el segundo, a los casos 4 y 7.

b) Calculamos ahora la entropía que aporta cada atributo:

- $E(X, Sexo) = \frac{1}{3}(-1\log_2 1 - 1\log_2 1) + \frac{2}{3}(-\frac{1}{2}\log_2\frac{1}{2} - \frac{1}{2}\log_2\frac{1}{2}) = \frac{2}{3}$

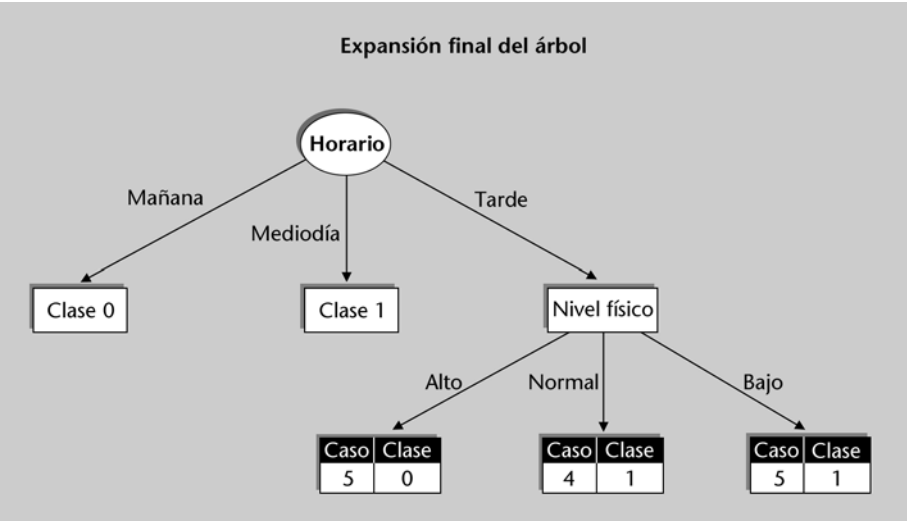
- $E(X, Nivel) = \frac{1}{3}(-0\log_2 0 - 1\log_2 1) + \frac{1}{3}(-1\log_2 1 - 0\log_2 0) +$

  $+ \frac{1}{3}(-0\log_2 0 - 1\log_2 1) = 0$

c) Finalmente, calculamos la ganancia correspondiente:

Atributo	$I(X,C) - E(X,A_k)$
Sexo	0,918 – 0,666 = 0,252
Nivel físico	0,918 – 0 = 0,918

3) Desarrollamos el árbol de la manera correspondiente:



En este momento todas las hojas del árbol corresponden a particiones homogéneas. En el caso del subárbol que se encuentra más a la derecha, las particio-

nes son trivialmente homogéneas, ya que contienen un único caso, que por fuerza no puede pertenecer más que a una única clase.

Horario = 'Mañana' => Clase 0
Horario = 'Mediodía' => Clase 1
Horario = 'Tarde' & *Nivel físico* = 'Alto' => Clase 0
Horario = 'Tarde' & *Nivel físico* = 'Normal' => Clase 1
Horario = 'Tarde' & *Nivel físico* = 'Bajo' => Clase 1

El ejemplo que acabamos de ver es muy sencillo, pero ha puesto en evidencia una serie de aspectos de los métodos de construcción de árboles de decisión:

a) Los árboles de decisión hacen una partición de un espacio con tantas dimensiones como atributos se utilizan para describir los casos.

b) Encuentran condiciones lógicas para describir las características de los casos que entran dentro de cada partición.

c) Utilizan medidas que evalúan el desorden de cada partición aportada por los atributos en cada paso del algoritmo.

Podéis ver las reglas de asociación en el módulo "Reglas de asociación" de esta asignatura.

d) Son fácilmente traducibles a reglas.

Fijémonos en que este mismo ejemplo ya nos señala alguno de los problemas comunes a otros métodos de clasificación que pueden aportar los árboles: las clases de un caso único quizá sean demasiado específicas y no deben servir para generalizar. También es cierto que este pequeño ejemplo es demasiado reducido para alcanzar generalizaciones significativas.

EL ID3 también presenta una serie de problemas que le son propios. Básicamente, genera demasiadas ramas en el árbol, lo cual provoca lo siguiente:

- Que las reglas sean claramente sobreespecializadas.
- Que se reduzca innecesariamente el número de casos que corresponden a cada nodo, con lo que la información que se extrae tiene cada vez menos soporte estadístico porque corresponde a una muestra más y más pequeña.

El hecho de que se generen árboles con estas características también afecta a la capacidad predictiva del modelo resultante.

En nuestro ejemplo consideramos la capacidad predictiva de cada rama del árbol.

Recordamos las reglas que han derivado del árbol. Las presentamos en la tabla que vemos a continuación:

Id.	Regla
1	Horario = 'Mañana' => Clase 0
2	Horario = 'Mediodía' => Clase 1
3	Horario = 'Tarde' & Nivel físico = 'Alto' => Clase 0
4	Horario = 'Tarde' & Nivel físico = 'Normal' => Clase 1
5	Horario = 'Tarde' & Nivel físico = 'Bajo' => Clase 1

Para encontrar la capacidad predictiva de cada rama, tenemos que contrastar cada una de las reglas obtenidas que resultan de hacer un recorrido por todas las ramas desde la raíz hasta la hoja correspondiente contra un conjunto de datos de prueba. Aquí lo tenemos:

Caso	Horario	Sexo	Nivel físico	Clase	Clase predicha	Regla
101	Mañana	Hombre	Alto	0	0	1
324	Mañana	Hombre	Bajo	1	0	1
5344	Mediodía	Hombre	Bajo	1	1	2
23	Mañana	Mujer	Bajo	0	0	1
28	Mañana	Hombre	Normal	1	0	1
29	Mañana	Mujer	Normal	0	0	1
333	Mediodía	Mujer	Bajo	1	1	2
442	Mañana	Mujer	Normal	1	0	1
32	Mediodía	Hombre	Bajo	1	1	2
112	Mediodía	Hombre	Normal	0	1	2
187	Mediodía	Hombre	Normal	0	1	2
54	Tarde	Hombre	Bajo	1	1	5
588	Tarde	Mujer	Alto	0	0	4
6536	Mediodía	Hombre	Bajo	0	1	2
72	Mañana	Hombre	Normal	0	0	1
811	Tarde	Hombre	Normal	0	1	4

Podemos tener una aproximación del valor predictivo de cada regla calculando la proporción de casos en los que la regla ha realizado una predicción adecuada con respecto al total de los casos que cubría:

Id.	Regla	Casos cubiertos	Predicciones incorrectas	Error
1	Horario = Mañana => Clase 0	7	3	42%
2	Horario = Mediodía => Clase 1	6	3	50%
3	Horario = Tarde & Nivel físico = Alto => Clase 0	0	0	-
4	Horario = Tarde & Nivel físico = Normal => Clase 1	2	1	50%
5	Horario = Tarde & Nivel físico = Bajo => Clase 1	1	1	0%

Evidentemente, hemos de tomar esta aproximación a la capacidad predictiva con mucha precaución. Por ejemplo, la regla 5 no es que sea el 100% predictiva. No hay los suficientes casos en el conjunto de prueba para cubrir realmente todas las reglas. Lo que hay que poner en evidencia, por ejemplo, es que la regla 2, con un 50% de capacidad predictiva, no es muy buena (¡podríamos tener el mismo resultado lanzando una moneda al aire!). La regla 1 tampoco es mucho mejor. La pregunta es si eliminando las ramas que corresponden a reglas con poca capacidad predictiva podemos mejorar la capacidad global de predicción del modelo.

El **error global** de todo el modelo es la suma ponderada de los errores de todas las hojas del árbol, es decir, el error de cada hoja multiplicado por la probabilidad de que un caso vaya a parar a la partición representada por la hoja.

Hacemos la aproximación de que la probabilidad es el número de casos cubiertos por la hoja (o regla) correspondiente con respecto al total de los casos de evaluación. Para un árbol con n hojas:

$$Error_{Global} = \sum_{i=1}^{n} w_i \cdot error_i$$

donde cada elemento de la expresión se define a continuación.

- w_i: es el peso o probabilidad de la hoja, es decir, la probabilidad de que un caso sea clasificado en la partición representada por la rama que acaba en la hoja i. En otras palabras, la probabilidad de que se utilice la regla i para clasificar un caso.

- $Error_i$: es el error correspondiente a la rama que acaba en la hoja i, o, si se prefiere, el error de clasificación propio de la regla i.

En nuestro caso tenemos que el número de hojas es 5, y, utilizando los valores calculados anteriormente, podemos ver que el error global de este árbol de decisión es el siguiente:

$$Error_{Global} = \frac{7}{16}0{,}42 + \frac{6}{16}0{,}5 + \frac{0}{16}0{,}0 + \frac{2}{16}0{,}5 + \frac{1}{16}0{,}0 = 0{,}43$$

Este valor para el error global es una cifra bastante alta, de manera que también tendremos un error bastante elevado.

Podemos observar que las diferentes reglas obtenidas para cada uno de los posibles recorridos poseen un valor predictivo diferente. Es más, según determinadas condiciones, eliminando las ramas que corresponden a reglas con una

capacidad predictiva menor (lo que se denomina **podar el árbol**), se obtienen modelos globales, árboles enteros con un mejor comportamiento predictivo conjunto.

2.3. Métodos de poda

Los métodos de poda intentan conseguir árboles que no tengan más niveles ni particiones de los que son necesarios para alcanzar un buen nivel de predicción.

Aquí veremos uno de los ejemplos típicos, y para ello utilizaremos uno de los conjuntos públicos de datos del repositorio de la UCI en Irvine. Se trata del problema de la decisión de recomendar el uso de lentes de contacto a varios pacientes según la edad que tengan (categorizada en 'Joven', 'Prepresbicia' y 'Presbicia'), la diagnosis actual ('Miope' e 'Hipermétrope') y la presencia o no de astigmatismo ('Sí' o 'No'). La recomendación final puede ser llevar lentes de contacto duras, blandas, o no llevar. Éste es, pues, el atributo por clasificar.

Lectura complementaria

Encontraréis el problema que tratamos en este subapartado en la obra siguiente:

J. Cendrowska (1987). "PRISM: an Algorithm for Inducing Modular Rules". *International Journal of Man-Machine Studies* (vol. 4, núm. 27, pág. 349-370).

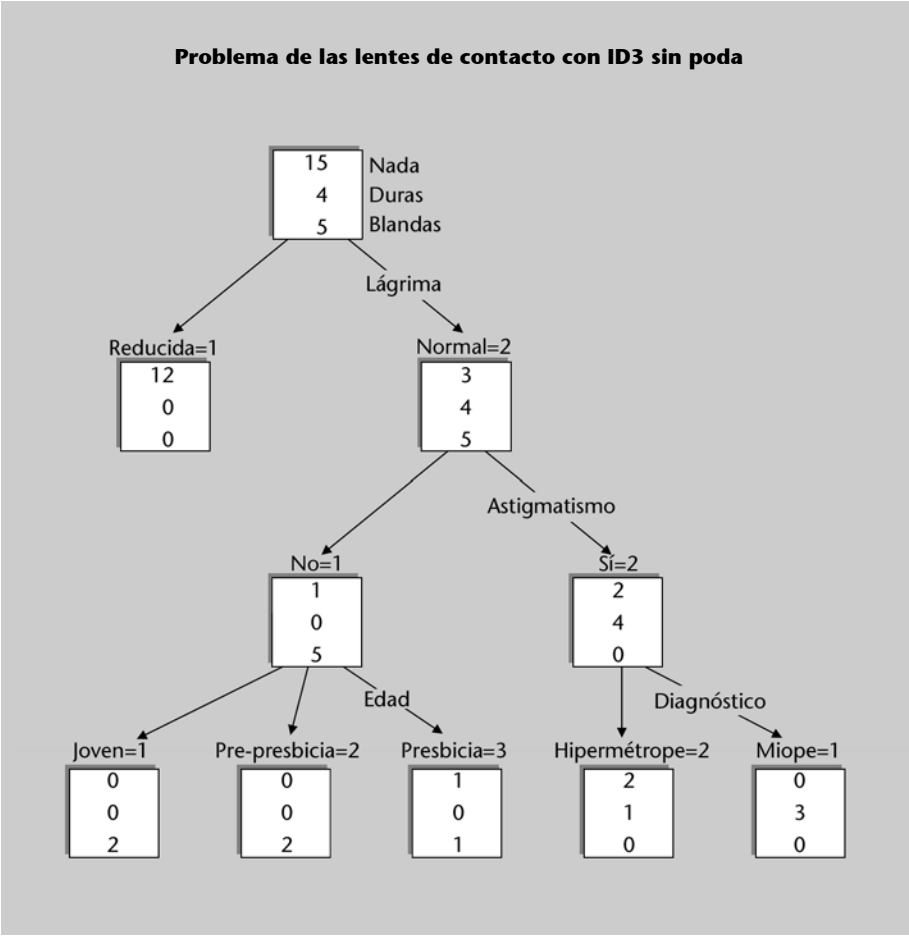
A continuación tenemos la tabla de veinticuatro ejemplos que cubre el dominio:

Encontraréis los datos utilizados en este ejemplo en la dirección siguiente: *http://www.gmd.de/mlarchive/frames/datasets/datasets-frames.html.*

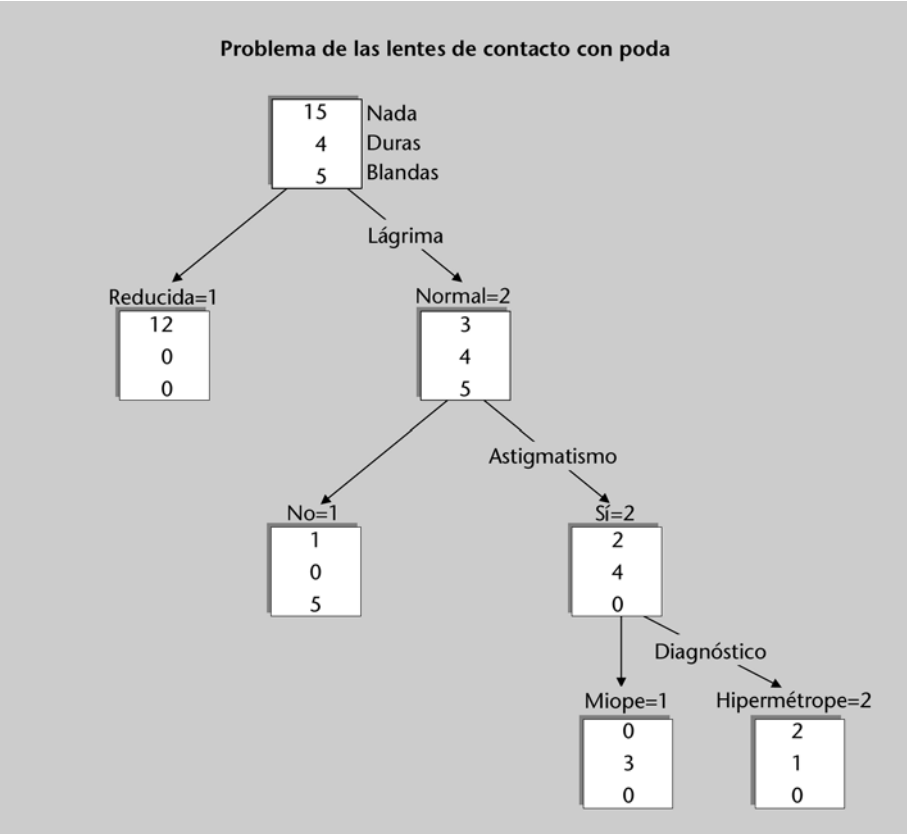
Edad	Diagnóstico	Astigmatismo	Lágrima	Recomendación
Joven	Miope	Sí	Normal	Duras
Joven	Hipermétrope	Sí	Normal	Duras
Prepresbicia	Miope	Sí	Normal	Duras
Presbicia	Miope	Sí	Normal	Duras
Joven	Miope	No	Reducida	Nada
Joven	Miope	Sí	Reducida	Nada
Joven	Hipermétrope	No	Reducida	Nada
Joven	Hipermétrope	Sí	Reducida	Nada
Prepresbicia	Miope	No	Reducida	Nada
Prepresbicia	Miope	Sí	Reducida	Nada
Prepresbicia	Hipermétrope	No	Reducida	Nada
Prepresbicia	Hipermétrope	Sí	Reducida	Nada
Prepresbícia	Hipermétrope	Sí	Normal	Nada
Presbicia	Miope	No	Reducida	Nada

Edad	Diagnóstico	Astigmatismo	Lágrima	Recomendación
Presbicia	Miope	No	Normal	Nada
Presbicia	Miope	Sí	Reducida	Nada
Presbicia	Hipermétrope	No	Reducida	Nada
Presbicia	Hipermétrope	Sí	Reducida	Nada
Presbicia	Hipermétrope	Sí	Normal	Nada
Joven	Miope	No	Normal	Blandas
Joven	Hipermétrope	No	Normal	Blandas
Prepresbicia	Miope	No	Normal	Blandas
Prepresbicia	Hipermétrope	No	Normal	Blandas
Presbicia	Hipermétrope	No	Normal	Blandas

En la figura siguiente vemos un primer resultado de haber aplicado el método ID3 sin poda:



En cada nodo aparece la cantidad de casos correspondientes a cada clase final, lo cual nos ofrece una idea de la heterogeneidad de las particiones que genera este nodo. Cuando aplicamos la poda, obtenemos un árbol más sencillo:



Los métodos de poda* dan como resultado árboles más fáciles de interpretar y más detallados. Quizá en este ejemplo la diferencia no sería demasiado significativa, pero en otros que incorporan más atributos acostumbra a ser mucho más apreciable.

* En inglés, *pruning*.

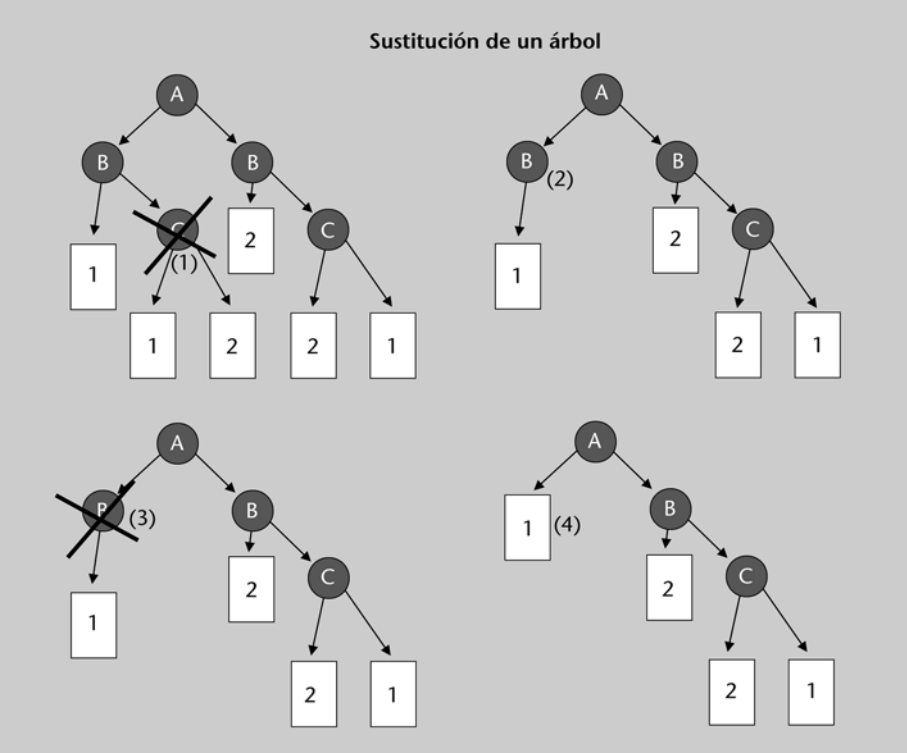
Una manera de alcanzar este objetivo es la que consiste en construir primero el árbol, y después analizar la tasa de predicción que obtenemos cuando eliminamos parte del árbol. Bien, este procedimiento recibe el nombre de **método de pospoda**.

Sin embargo, puede parecer más útil ahorrarse el proceso de construcción de los árboles poco prometedores y preguntar a cada nivel cuál es el valor de predicción que nos asegura el árbol parcialmente construido. Si en un determinado nodo lo suficientemente informativo obtenemos una evaluación de la tasa de predicción de las particiones resultantes inferior a un límite de calidad prefijado, entonces detenemos la expansión del árbol. Éste es un **método de prepoda**.

Ambos métodos se utilizan, aunque parece que la pospoda es más detallada que la prepoda. Las operaciones que se utilizan en pospoda son la promoción de un subárbol y la sustitución de un árbol:

a) La **operación de sustitución** consiste en analizar el subárbol a partir del nodo en el que el método efectúa la construcción y sustituirlo por un conjunto de hojas finales. Eso supone establecer las categorías correctas para etiquetar las ramas. En principio, este hecho podría rebajar la calidad final de un árbol

construido con ID3, que trata de tener hojas tan homogéneas como sea posible, pero daría un árbol más comprensible, más fácil de calcular y menos sobreespecializado. Esta última característica es importante, ya que nos interesa poder generalizar nuevos ejemplos. La figura siguiente muestra la operación de sustitución:

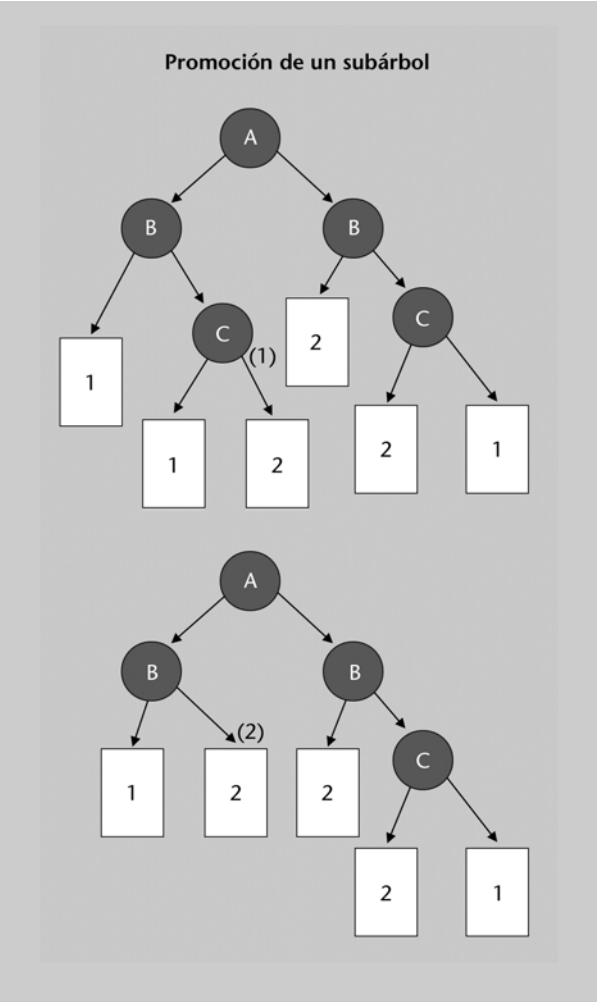


En la figura anterior los números entre paréntesis indican el orden en que se consideran los nodos del árbol. El subárbol que cuelga del subárbol izquierdo *B* se ha sustituido por una partición en la que las observaciones se asignan mayoritariamente a la clase 1.

b) La **operación de promoción** funciona en sentido contrario al de la sustitución. Si aquélla actuaba en sentido descendente,* ésta lo hace de manera ascendente, es decir, de las hojas hacia la raíz. El método C4.5 y su sucesor, el C5.0 (ambos muy parecidos a ID3) incorporan un método de pospoda por promoción.

* En inglés, *top-down*.

Con la operación de promoción se detecta que un subárbol no es útil o interesante (por ejemplo, tiene una capacidad de predicción baja) y es sustituido por un subárbol que ofrezca mejores expectativas. Claro está, la citada operación implica una reclasificación de ejemplos. Todos los ejemplos que se encontraban "bajo el paraguas" del nodo original han de ser ubicados bajo el nuevo y, por tanto, hay que crear particiones nuevas. La complejidad de esta operación no es negligible.



Todo el subárbol que dependía del nodo izquierdo con la etiqueta *B* se sustituye por el subárbol que cuelga de *C*, y hay que reclasificar las observaciones de las particiones de *B* y de *C*.

La forma más sencilla de calcular a fin de decidir si hay que hacer la promoción o sustitución de un árbol consiste en anticipar la tasa de error que se obtiene sin sustitución (o promoción) y con ésta.

¿Como hay que decidir si hay que hacer la promoción o sustitución de un árbol?

Ya hemos dicho que, cada vez que nos planteamos cambios de este tipo, hace falta reclasificar las observaciones que quedan por debajo del nodo sujeto de análisis.

Comentaremos dos métodos de poda: el típico C4.5 y un método basado en el principio MDL.

2.3.1. Poda con el método C4.5

A la hora de calcular la tasa de error nos encontramos ante una situación comprometida. Lo más correcto sería utilizar un conjunto de datos diferente del

que utilizamos para construir el árbol, es decir, un conjunto diferente del de entrenamiento. De lo contrario, como mínimo se pueden reservar algunos de los datos del conjunto original con ese fin. Finalmente, y es lo que hace el método C4.5, se pueden utilizar los mismos datos del conjunto de entrenamiento. Contra toda lógica, funciona bastante bien.

Antes, sin embargo, tendremos que hacer una excursión estadística para conocer otra distribución de probabilidad.

La distribución binomial

El problema del cálculo de la tasa de error (o, al revés, de la precisión) de un método de clasificación podemos entenderlo como el de la estimación de una proporción en una población.

Queremos saber, visto que hemos calculado a partir de los datos una tasa de error del e%, cuál es la tasa que podríamos anticipar en ejemplos nuevos procedentes de la misma población.

En principio deberíamos tener preferencias con respecto a las tasas calculadas a partir de muestras mayores, lo cual es particularmente relevante cuando hay que comparar las tasas de error en varios niveles de un árbol, ya que, cuanto más hacia abajo del árbol nos encontramos, menos observaciones hay (menores son las particiones).

> La predicción de la tasa de error vista de esta manera puede entenderse como el cálculo de la proporción de errores o éxitos que ofrecerá un determinado proceso de clasificación con respecto a una población.

El resultado del modelo de clasificación para cada observación puede ser correcto o incorrecto (bien clasificado, mal clasificado). En cuanto a la observación, pues, el proceso tiene dos resultados: cierto o falso, correcto o incorrecto, cara o cruz. Sobre un conjunto de n observaciones, efectuaremos una secuencia de n pruebas que pueden dar como resultado correcto o incorrecto. Queremos encontrar cuál es esa proporción.

Este tipo de procesos, que son análogos a lanzar n veces una moneda al aire e intentar inferir cuál es la proporción de caras y cruces que esperamos ver (el 50%, si la moneda no está trucada), sigue una **distribución binomial** o **distribución de Bernoulli**.

El problema de estimación que nos planteamos ante fenómenos como el de lanzar una moneda al aire o clasificar observaciones es estimar, a partir de la proporción observada p, cuál sería la proporción real. Por lo tanto, nos encon-

tramos ante un problema típico de estimación. De nuevo, tendremos que establecer intervalos de confianza y decidir dónde se sitúa la proporción en la población. Pero ahora la distribución no es normal, sino binomial.

Esta distribución, para una tasa de éxito de p, sigue una ley de probabilidad que tiene por término medio p y de varianza, $1 - p$. En un proceso de n pruebas de este tipo, en el que hemos observado k éxitos (clasificaciones correctas), la variable aleatoria p', que representa la proporción de éxitos esperados:

$$p' = \frac{k}{n}$$

sigue una ley binomial con media p y varianza $\frac{p(1-p)}{n}$.

Cuando n es grande, esto se aproxima a la distribución normal. En consecuencia, para encontrar un intervalo de confianza del 90%, nos interesa encontrar el valor z tal que:

$$p[-z \leq p' \leq z] = 90\%$$

Podemos comprobar que el valor correspondiente de z es 1,65.

Para el caso que nos interesa, tenemos que normalizar la variable p' para que tenga media 0 y desviación típica, 1. Aplicamos la transformación que ya conocemos y resulta:

$$P\left[-z \leq \frac{p' - p}{\sqrt{p(1-p)/n}} \leq z\right]$$

A fin de encontrar los límites del intervalo de confianza, hacemos lo de siempre: consultamos las tablas correspondientes y después reconvertimos el valor obtenido al rango de valores de p' mediante la transformación siguiente (que no nos entretendremos en derivar):

$$p = \left(k + \frac{z^2}{2n} \pm z\sqrt{\frac{k}{n} - \frac{k^2}{n} + \frac{z^2}{4n^2}}\right) \Big/ \left(1 + \frac{z^2}{n}\right)$$

Todavía tenemos que ver cómo se relaciona todo esto con C4.5. En tal caso, intentamos estimar el contrario de p, que era la proporción esperada de éxitos. Recordamos que queremos estimar la proporción de error. La cosa es fácil, consiste en hacer lo siguiente:

$$\%\ \text{de éxitos} + \%\ \text{de errores} = 1$$

Lectura complementaria

Con relación a hacer una estimación pesimista de e, encontraréis más información en la obra siguiente:

J.R. Quinlan (1987). "Simplifyng Decisions Trees". *International Journal of Man-Machine Studies* (núm. 27, pág. 221-234).

Así pues, se trata de estimar la proporción de $e = (1 - p)$. Una última modificación muy importante es que ahora trabajamos directamente con los datos de entrenamiento, de manera que hacemos una estimación pesimista de e tomando el nivel superior de confianza. El nivel de confianza que toma el algoritmo C4.5 es por defecto el 25%. Se trata de encontrar (normalizando de nuevo para obtener una distribución con media 0 y desviación estándar 1):

$$P\left[\frac{k-e}{\sqrt{e(1-e)/n}} > z\right]$$

donde k ahora es la tasa de error observada.

La z correspondiente al nivel de confianza del 25% es 0,69. Para ver cuál es su valor en términos de proporciones, tenemos que deshacer el cambio de variable y, teniendo en cuenta que ahora calculamos la proporción de error, resulta una expresión un tanto diferente a la de antes:

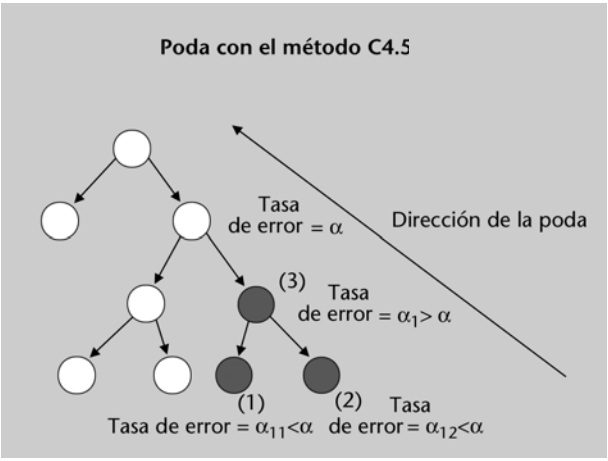
$$p = \frac{\left(k + \frac{z^2}{2n} \pm z\sqrt{\frac{k}{n} - \frac{k^2}{n} + \frac{z^2}{4n^2}}\right)}{1 + \frac{z^2}{n}}$$

Después de todo esto, si por ejemplo apreciamos una proporción de error del 15% a partir de cien mil observaciones, ya estamos en condiciones de decir que la proporción verdadera de error oscilará entre el 12,09% y el 16,03%.

Se trata de hacer lo siguiente:

1) Para cada nodo observamos la proporción de error (e) y calculamos la proporción según la estimación pesimista (e'); después combinamos estos estimadores ponderándolos según el número de valores de cada partición.

2) Si la estimación pesimista para un nodo es menor que la de los hijos, semillas eliminamos los hijos.



2.3.2. Poda con el método MDL

Se ha comprobado experimentalmente que los métodos de poda basados en la estimación pesimista que hemos descrito generan árboles demasiado grandes y tasas de error más altas de lo necesario. Una manera alternativa de plantearse la poda de árboles, que además genera árboles más compactos, es la que utiliza el método de mínima descripción de longitud. Otras intentan crear varios árboles con respecto a diferentes conjuntos de datos y después evalúan con validación cruzada (que tendremos ocasión de ver en otro módulo) los distintos modelos, para acabar quedándose, finalmente, con el mejor (con el consiguiente incremento de la complejidad de todo el proceso).

Lecturas complementarias

Podéis ver los resultados experimentales de los métodos de poda basados en la estimación pesimista y el método de mínima descripción de longitud, respectivamente, en las obras siguientes:

J. Mingers (1989). "An Empirical Comparison of Pruning Methods for Decision Tree Induction". *Machine Learning* (núm. 4, pág. 227-243).

M. Rissanen (1985). "The Minimum Description Length Principle". En: S. Kotz; N.L. Johnson (ed.). *Encyclopedia of Statistical Sciences* (vol. 5). Nueva York: John Wiley & Sons.

A continuación apuntaremos la explicación del método de mínima descripción de longitud.

1) La **formalización del problema** en el marco del MDL es la que presentamos ahora:

Supongamos que tenemos un atributo de clase, c_t, que puede tomar 0, ..., $m - 1$ valores. El conjunto de datos está formado por una sucesión de n observaciones. Cada una de éstas posee los valores correspondientes a los distintos atributos X_1, ..., X_n más el correspondiente al atributo de clase. Supondremos que cada atributo toma valores en un conjunto de valores determinado. La variable categórica X_i adopta valores x_i en un conjunto de valores $\{1, \ldots, r_i\}$.

El **método de mínima descripción de longitud** (MDL) intenta encontrar el modelo, dentro de cada clase, que permita la mínima codificación de la secuencia de la clase c_n dados los valores observados.

Hay que encontrar, dentro del espacio de posibles subárboles que se pueden generar a partir de un árbol T, los mínimos. Cada nodo del árbol T posee un atributo con el número de observaciones correspondientes y el valor de corte para el atributo correspondiente dentro del árbol. De esta manera, tendremos que codificar, para cada nodo, los valores de los atributos para cada uno de los subárboles que genera (o el punto de corte correspondiente, en variables continuas) y las probabilidades de la clase.

2) La codificación del problema se basa en utilizar como código la longitud de la cadena necesaria para representar cada subárbol con la convención de guardar, para cada nodo, la información que hemos comentado. La fórmula para calcular esta longitud tiene un aspecto bastante impresionante:

$$\log_2 n_i + \log_2 \frac{t!}{t_0! + \ldots + t_n!} + \log_2 \binom{t + m - 1}{m - 1}$$

En la fórmula que acabamos de ver, n_i es el número de veces que el símbolo i (un valor de un atributo categórico, por ejemplo) aparece en la secuencia de variables de clase c_t –recordamos que puede haber t clases–. El tercer término representa la longitud de codificación necesaria para codificar el número de veces que aparece cada valor, y se puede considerar el coste del modelo para el modelo (subárbol) de la clase correspondiente. Los términos primero y segundo representan la longitud de codificación necesaria para codificar los datos observados. El inconveniente principal es que ambos términos se aproximan al mismo orden de magnitud cuando algunos de los recuentos de apariciones de valores son próximos a 0 o al número de clases, t.

Una mejora de este modelo pasa por considerar la longitud siguiente:

$$L(c_t) = \sum t_i \log_2 \frac{t}{t_i} + \frac{m-1}{2} \log_2 \frac{t}{2} + \log_2 \frac{\pi^{m/2}}{\Gamma(m/2)}$$

donde Γ es la función gama y t_i representa el número de veces que un valor aparece dentro de la secuencia de clase c_t. Esta longitud de codificación, llamada *complejidad estocástica*, tiene propiedades de optimización bien probadas.

La decisión de podar el subárbol que cuelga de un nodo determinado se toma, en este método, utilizando como criterio la longitud de descripción de los diferentes subárboles considerados. Si para un nodo determinado, a, la longitud de descripción es l_a y la longitud ponderada de sus k subárboles a_1, ..., a_k es:

$$l_{hijos} = \frac{1}{n} \sum_{i=1}^{\kappa} l_i$$

entonces se podarán los subárboles si l_{hijos}, es menor que l_a.

Lecturas complementarias

Para una discusión más detallada y la comparación de resultados con relación al método de mínima descripción de longitud, podéis consultar las obras siguientes:

R.E. Krichevsky; V.K. Trofimov (1983). "The Performance of Universal Coding". *IEEE Transactions on Information Theory* (vol. IT-27, núm. 2).

M. Rissanen (1989). *Stochastic Complexity in Statistical Inquiry*. World Scientific Publishers Company.

3. Métodos de construcción de árboles de decisión para regresión y clasificación (CART)

El método CART es un algoritmo de construcción de árboles de decisión propuesto en 1984 por Breiman y sus colaboradores. CART es el acrónimo de *Classification and Regression Trees* (que quiere decir 'árboles de clasificación y regresión'). La mayoría de los paquetes comerciales de construcción de árboles de decisión o de construcción de reglas de decisión utilizan de una manera más o menos explícita este algoritmo.

Lectura complementaria

Encontraréis la presentación del algoritmo CART en la obra siguiente:

L. Breiman; H. Friedman; R. Olshen; C. Stone (1984). *Classification and Regression Trees*. Belmont: Wadsworth.

Las diferencias principales que presente este método con respecto al método ID3 se concentran en los aspectos siguientes:

a) El tipo de árbol que se construye. En el algoritmo CART los árboles que se construyen son binarios; en cada nodo hay un punto de corte (por un procedimiento parecido al que hemos explicado para encontrar puntos de corte en la discretización de atributos continuos) que separa en dos el conjunto de observaciones.

b) Los tipos de atributos. En principio, el algoritmo CART puede trabajar con atributos continuos (aunque las modificaciones de ID3 también pueden hacerlo).

c) Puede hacer tanto clasificación como regresión. En el primer caso, la variable para predecir tiene que ser categórica con un valor para cada clase posible.

d) Medida de homogeneidad y criterio de paro en el proceso de partición y división del árbol. En este método es la **medida de Gini**, aunque hay variantes que eligen otras.

En el CART, el procedimiento de construcción del árbol pasa por considerar en cada momento el hecho de encontrar el atributo que actúa como mejor separador* o punto de corte.

*** En inglés, *splitter*.**

Pues bien, para hacerlo utiliza la **medida de Gini** como índice de diversidad de cada partición posible. La medida de Gini establece que el mejor separador es aquel atributo que reduce la diversidad de las diferentes particiones. Por lo tanto, lo que hace CART es maximizar su diferencia:

Medida de Gini = diversidad antes de la partición –
– diversidad en el subárbol izquierdo + diversidad en el subárbol derecho.

Se elige, pues, el mejor separador y se convierte en la raíz del árbol; a partir de esto se obtendrán dos particiones, y se vuelve a proceder con cada una de éstas del mismo modo. Si en este punto algún atributo toma un solo valor con respecto a todos los elementos de la partición, entonces dejamos de considerarlo. Cuando ya no se pueden encontrar más separadores, el nodo se etiqueta como una hoja terminal. Cuando todas las particiones corresponden a hojas terminales, entonces podemos decir que se ha acabado de construir el árbol.

La poda en CART se efectúa mediante tres pasos:

1) Generación de varios subárboles podados "interesantes".
2) Obtención de estimaciones del error de cada uno de estos subárboles.
3) Elección del subárbol que presente la mejor estimación.

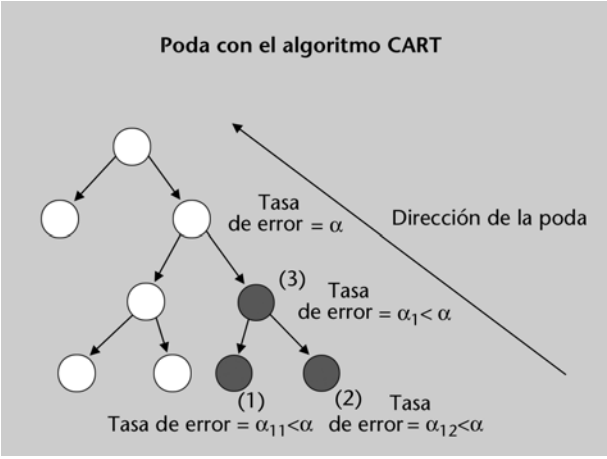
El primer paso consiste en generar secuencias de subárboles anidadas:

$$\acute{a}rbol_1, \ldots, \acute{a}rbol_n$$

En cada paso *i,* el nodo para podar, se obtiene del árbol predecesor en la secuencia $\acute{a}rbol_{i\text{-}1}$. Para elegirlo, se utiliza la tasa de error ajustada, *e'*, del subárbol correspondiente:

$$e' = e_{arbol_i} + \alpha \cdot nhojas(arbol_i)$$

donde e_{arbol_i} es la proporción de error observada en el árbol *i*, *nhojas*($\acute{a}rbol_i$) es el número de hojas de $\acute{a}rbol_i$ y α es un coeficiente que presenta un comportamiento interesante. Para encontrar el primer subárbol, se calculan las tasas de error ajustadas de todos los subárboles que contienen la raíz incrementando gradualmente α. Cuando la tasa de error ajustada de algunos de los subárboles es menor o igual que la de todo el árbol, ya tenemos un primer subárbol candidato, $\acute{a}rbol_1$, para ser podado. Entonces se pueden todas las ramas que no son parte de $\acute{a}rbol_1$ y el proceso se repite. De esta manera, empezando por las hojas, se va procediendo hacia la raíz del árbol. Gráficamente, el proceso es el siguiente:



Los números indican en qué orden se consideran los nodos. Los que salen de color negro son candidatos a ser podados. Notad que los dos nodos del subárbol de la derecha no son candidatos a ser podados hasta que su padre muestra una tasa inferior a α.

Ahora aplicaremos CART a otro conjunto de datos lo suficientemente conocido, también procedente del repositorio de la UCI en Irvine. Es el conjunto IRIS, que guarda datos sobre la clasificación de las distintas especies de iris (lirios) existentes en función de los parámetros siguientes: la anchura del pétalo (*Petal width*), la longitud del pétalo (*Petal length*), la anchura del sépalo (*Sepal width*) y la longitud del sépalo (*Sepal length*).

Encontraréis los datos utilizados en este ejemplo en la dirección siguiente: *http://www.ics.uci.edu/~mlearn/MLRepository.html.*

La variable de clasificación es la clase de la planta, evidentemente. Hay tres: *Iris versicolor*, *Iris setosa* e *Iris virginica*.

Clase	Longitud del sépalo	Anchura del sépalo	Longitud del pétalo	Anchura del pétalo
Iris setosa	5,1	3,5	1,4	0,2
Iris setosa	4,9	3	1,4	0,2
Iris setosa	4,7	3,2	1,3	0,2
Iris setosa	4,6	3,1	1,5	0,2
Iris versicolor	7	3,2	4,7	1,4
Iris versicolor	6,4	3,2	4,5	1,5
Iris virginica	5,8	2,7	5,1	1,9
Iris virginica	7,1	3	5,9	2,1
Iris virginica	6,3	2,9	5,6	1,8
Iris versicolor	6,9	3,1	4,9	1,5

En este problema todos los datos son numéricos, y CART encuentra en cada paso lo siguiente:

- el atributo más discriminante,
- el punto de corte que garantice particiones uniformes para el atributo.

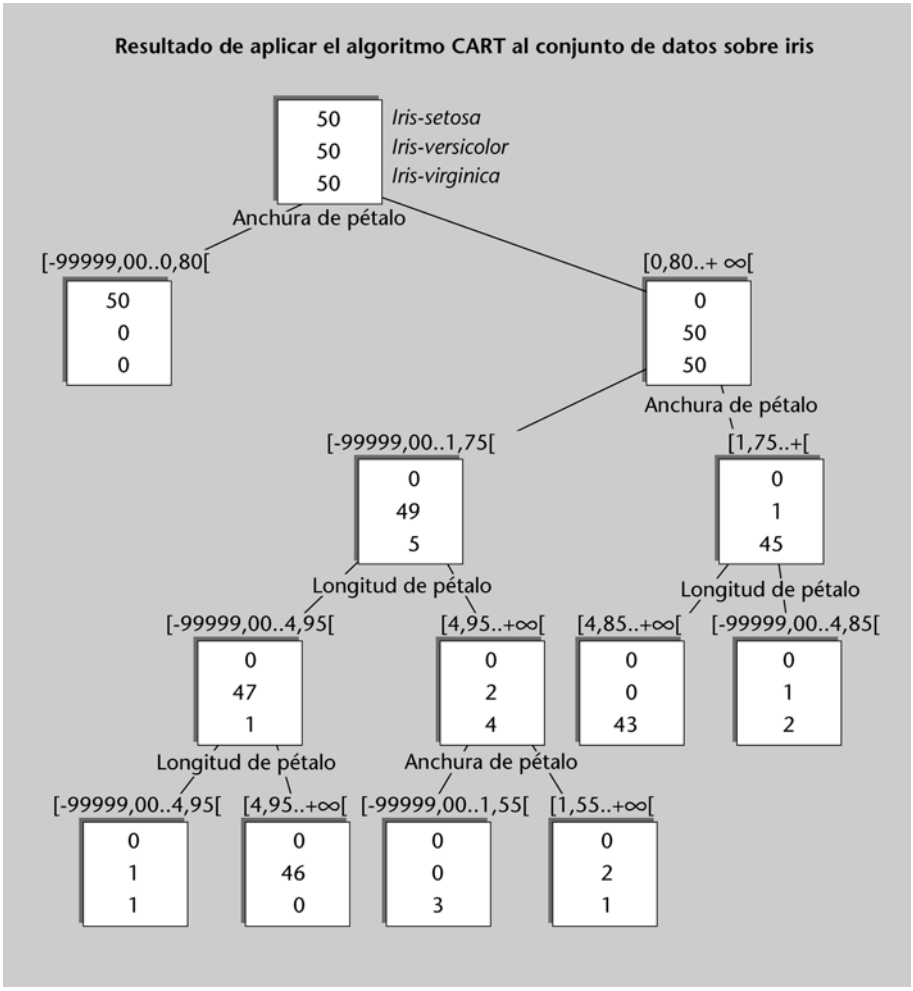
Con todo, en cada nodo se efectúa un test que compara los valores de la variable de que se trate con el punto de corte, de manera que se separan los datos en dos particiones: los que tienen un valor inferior al punto de corte y los que tienen uno superior.

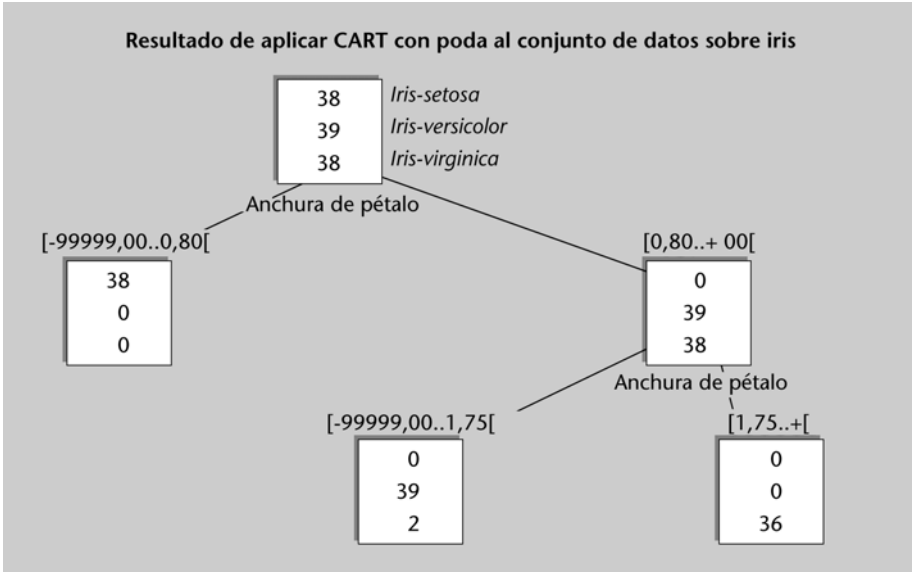
Lectura recomendada

Para una comparación de métodos de poda, podéis consultar la obra siguiente:

F. Esposito; D. Malerba; G. Semeraro (1997). "A Comparative Analysis of Methods for Pruning Decision Trees". *IEEE Transactions on Pattern Analysis and Machine Intelligence* (vol. 19, pág. 476-493).

En la figura de la página siguiente podéis ver el resultado de aplicar el algoritmo CART al conjunto de datos IRIS. Observad en este ejemplo que cada subárbol es binario:


Resultado de aplicar el algoritmo CART al conjunto de datos sobre iris
50
50
50
Iris-setosa
Iris-versicolor
Iris-virginica
Anchura de pétalo
[-99999,00..0,80[
50
0
0
[0,80..+ ∞[
0
50
50
Anchura de pétalo
[-99999,00..1,75[
0
49
5
[1,75..+[
0
1
45
Longitud de pétalo
Longitud de pétalo
[-99999,00..4,95[
0
47
1
[4,95..+∞[
0
2
4
[4,85..+∞[
0
0
43
[-99999,00..4,85[
0
1
2
Longitud de pétalo
Anchura de pétalo
[-99999,00..4,95[
0
1
1
[4,95..+∞[
0
46
0
[-99999,00..1,55[
0
0
3
[1,55..+∞[
0
2
1


Resultado de aplicar CART con poda al conjunto de datos sobre iris
38
39
38
Iris-setosa
Iris-versicolor
Iris-virginica
Anchura de pétalo
[-99999,00..0,80[
38
0
0
[0,80..+ 00[
0
39
38
Anchura de pétalo
[-99999,00..1,75[
0
39
2
[1,75..+[
0
0
36

4. Métodos de construcción de árboles de decisión para predicción numérica (CHAID)

El método CHAID de construcción de árboles de decisión es un algoritmo propuesto por Hartigan en 1975. Procede de un método anterior, AID, que intentaba encontrar relaciones estadísticas significativas entre variables (AID es la sigla de la expresión inglesa *Automatic Interaction Detection*, y significa 'detección automática de interacciones'). La manera de detectar las interrelaciones entre las variables pasa por construir un árbol de decisión.

Lectura complementaria

Podréis consultar la presentación del método CHAID en la obra siguiente:

J.A. Hartigan (1975). *Clustering Algorithms*. Nueva York: John Wiley.

Las diferencias principales del método CHAID con respecto a los otros métodos residen en los aspectos siguientes:

- El método CHAID se encuentra restringido a variables categóricas.

- El método detiene la construcción del árbol en el momento en que detecta que puede producirse una sobreespecialización.

El criterio de establecimiento del punto de corte en cada nodo es el mismo que se utiliza en el método *chi-merge* para efectuar la discretización de variables continuas. La poda se realiza efectuando una prueba de significación a los errores esperados de cada subárbol.

5. Métodos de construcción de árboles de decisión multivariantes (LMDT)

Hemos comentado antes que uno de los problemas que genera ciertos métodos de construcción de árboles es la complejidad de la estructura resultante, que provoca una transformación que lleva a reglas poco comprensibles. Ya hemos visto que la poda puede resolver este tipo de problemas, introduciendo ciertos aumentos de coste computacional. Otra manera de reducir el problema es preguntarse en cada nodo no sólo por el valor de un único atributo, sino por los valores de más de un atributo al mismo tiempo. Lo que acabamos de plantear, además, posee una gran ventaja añadida, y es que se pueden resolver problemas de clasificación a los cuales los árboles basados en particiones binarias o en un único valor no pueden dar solución.

El objetivo de los métodos LMDT...

... es construir árboles más sencillos, igualmente predictivos y que puedan resolver problemas más complejos.

El problema de la complejidad excesiva del modelo surge cuando se dispone de conjuntos de datos en los que las diferentes clases no generan regiones con límites que se puedan definir con líneas rectas o zonas del espacio que no se pueden describir con ecuaciones lineales. Este problema se conoce como **problema de las regiones de clasificación no linealmente separables** y afecta a todos los métodos de clasificación.

Podéis consultar el módulo "Redes neuronales" de esta asignatura.

La diferencia principal de este método con respecto a los demás reside en que, en lugar de considerar un único atributo cada vez para efectuar la separación, se consideran n atributos. Dado un nodo de decisión, en lugar de aplicar la prueba tradicional al grado de separabilidad que induce un único atributo, el algoritmo aplica el test a más de un atributo al mismo tiempo. Y esto se hace para resolver un problema típico de la clasificación con la dificultad o imposibilidad de representar particiones que no son ortogonales a los ejes del espacio de que se trate.

Hemos de entender que el conjunto de atributos que representa el dominio define un espacio de tantas dimensiones como atributos haya. Para cada atributo, X_1, ..., X_n, definimos un eje. Una observación se convierte en un punto en ese espacio.

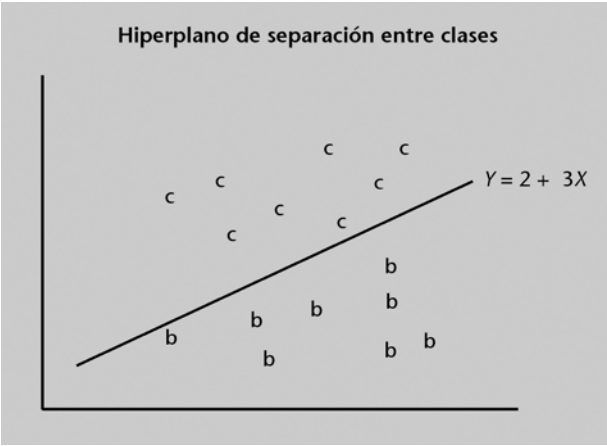
Ejemplo de espacio de dos dimensiones

Supongamos que sólo representemos la renta y la edad de los socios de Hyper-Gym. Bien, con esto tendremos definido un espacio de dos dimensiones. La observación (5.000.000,40) correspondiente a una renta de cinco millones y a una edad de cuarenta años es un punto representado en un espacio donde, por ejemplo, el eje X corresponde al atributo *Renta*, y el eje Y, al atributo *Edad*. Podéis extender este razonamiento a más atributos y, por tanto, a más dimensiones. Ésta es una idea bastante potente que nos iremos encontrando continuamente. Le dedicaremos más espacio cuando hablemos de los métodos de agregación.

El **problema de la separabilidad** consiste en que, para ciertos conjuntos de observaciones, es imposible definir líneas (o hiperplanos) en el espacio correspondiente que distingan perfectamente las observaciones que corresponden a una clase y a otra.

5.1. Tratamiento de los conjuntos linealmente separables

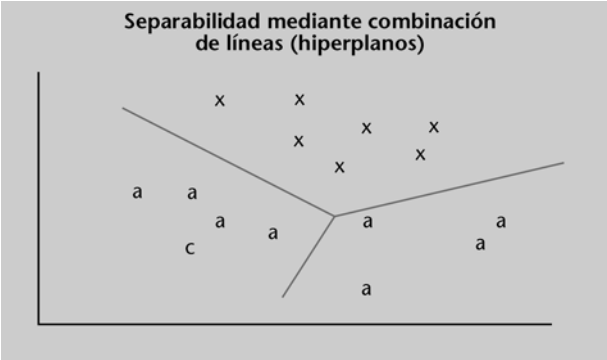
En la figura siguiente tenemos observaciones de dos clases que se pueden separar describiendo la línea que las separa mediante una ecuación lineal:



Si utilizamos sólo un test referente a los valores de un único atributo, entonces no podremos separar las clases correctamente. Lo que sí podemos hacer mediante los tests tradicionales es encontrar condiciones para separar las particiones cuando éstas son ortogonales a los ejes (en dos dimensiones, cuando los dominios de las clases son cuadrados).

En el caso de estas particiones, las líneas que las describen (sobre el plano *X*-*Y*) tienen una expresión del tipo $y \leq N$. Cuando establecemos una comparación en un nodo de decisión, estamos imponiendo o descubriendo qué valor de la variable que nos interesa (el atributo del nodo de decisión) permite englobar todos los casos que pertenecen de la misma clase bajo la recta que se determine. En la figura anterior podemos ver que la partición que identifica los objetos etiquetados con una *x* corresponde a la región $Y \geq 2$ y $X \leq 2$, que es una región cuadrangular. No es posible encontrar líneas tan sencillas como éstas para aislar la región del espacio bidimensional.

Cuando, en lugar de un espacio de dimensión 2, nos encontramos con espacios de dimensionalidad más alta, tenemos que definir hiperplanos ortogonales en los distintos ejes. Evidentemente, si el conjunto de casos no ocupa una región que pueda caracterizarse mediante hiperplanos ortogonales, entonces es necesario aproximar la región por medio de otro tipo de planos.



En este caso, la línea que puede dividir las dos regiones, las de los objetos etiquetados como *a* y la de los etiquetados como *b*, se puede separar mediante una línea diagonal que es una "combinación lineal" de *Y* y de *X*.

Si tenemos espacios de dimensión mayor, entonces será necesario realizar una aproximación a la forma en el espacio de cada partición, encontrando las combinaciones lineales de varios atributos:

$$Y = a_1x_1 + a_2x_2 + \ldots + a_nx_n$$

El problema, claro, reside en saber cuáles son los valores a_i, los pesos, que nos permitirán aislar adecuadamente cada una de las particiones. Lo que está claro, no obstante, es que hay que utilizar simultáneamente los valores de más de un atributo.

Una propuesta de mejora para este problema es la que aporta el sistema LMDT, propuesta por Utgoff en 1986.

***LMDT* es la sigla de la expresión inglesa *Linear Machine Decision Trees*.**

El **método LMDT de construcción de árboles de decisión multivariantes** asigna una clase a cada nodo como resultado del test referente a diferentes atributos. En lugar de preguntar por un único valor en cada nodo, se hace un test con respecto al conjunto de atributos.

El algoritmo que efectúa este test construye un conjunto de funciones linealmente discriminantes, de manera que cada una de estas funciones se utiliza más tarde para asignar cada observación a la clase correspondiente. Para todo nodo que no es una hoja, se definen tantas funciones discriminantes como clases haya. Si hay *n* clases, se tienen que definir *n* funciones discriminantes lineales para todo nodo del subárbol.

Supongamos que se realiza una observación *Y* definida en términos de los atributos que definen el dominio $X_1, \ldots, X_n$: $Y = x_1, \ldots, x_n$, donde las letras en minúscula representan los valores que toman las variables correspondientes, el método LMDT actúa sobre valores continuos y normalizados.

Una función discriminante se define como: $g_{i(Y)}W_i^T Y$, donde W es un vector de coeficientes ajustables que llamaremos **pesos**.

Si tenemos un nodo (que no sea una hoja) que contiene un conjunto de funciones discriminantes, g_1, ..., g_n, y dado un conjunto de clases posibles, c_1, ..., c_n, entonces:

$$Y \in c_i \Leftrightarrow g_i(Y) > g_j(Y),\ \forall i,\ 1 \le i \le n,\ i \ne j$$

Es decir, una observación pertenece a una clase c_i cuando el producto interior con el vector de pesos W_i es máximo con respecto al resto de las funciones discriminantes (recordamos que hay una para cada clase).

Así pues, es importante que podamos extraer a partir del conjunto de datos originales los pesos W_i que aseguren que obtendremos el mejor árbol posible.

El aprendizaje propiamente dicho consiste en ajustar los vectores W correspondientes a las funciones discriminantes, g_i, aumentando los pesos de W_i, donde i es la clase a la cual pertenece una observación, y disminuyendo los pesos de W_j, donde j sería una clase errónea para la observación que tratamos en un momento determinado.

Dado un peso W_i, al ver una nueva observación Y, tenemos que efectuar un ajuste que formalizamos como se sigue:

$$W_i \leftarrow W_i + cY$$
$$W_j \leftarrow W_j - cY,\ \forall i,\ 1 \le i \le n,\ \mathrm{i} \ne j$$

donde j es el índice que representa las clases erróneas para la observación que consideramos y c, un factor de corrección que se calcula de manera que toda observación que se haya clasificado erróneamente ahora sea clasificada de forma correcta.

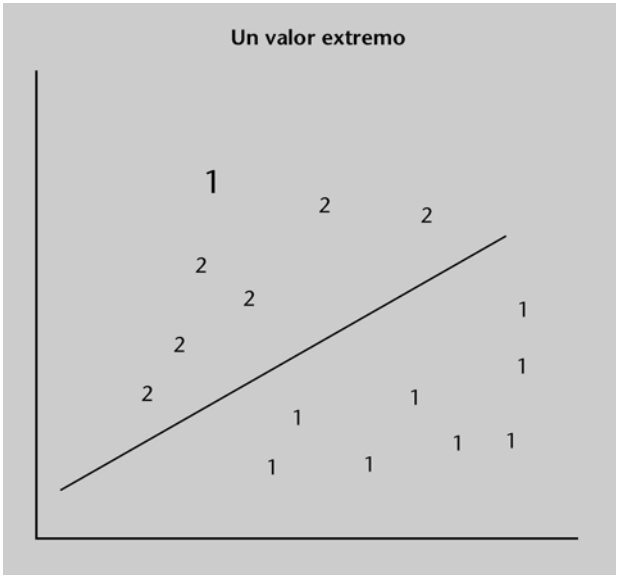
5.2. Tratamiento de los conjuntos no linealmente separables

Para el caso de los **conjuntos de datos no linealmente separables** el sistema LMDT introduce correcciones continuas en los errores de clasificación, de manera que el proceso no se estabiliza o hace que la clasificación sea imprevisible. Hay que asegurar un comportamiento estable en estas situaciones. La solución consiste en introducir una modificación al parámetro de corrección c.

La idea es prestar más atención a los errores de clasificación mayores (se ve una idea parecida en el caso de la retropropagación en redes neuronales).

Podéis ver la retropropagación en el módulo "Clasificación: redes neuronales" de esta asignatura.

La figura de la página siguiente intenta explicar la situación que acabamos de presentar:



La observación destacada en la esquina superior izquierda requeriría un ajuste bastante importante en la línea que establece la decisión para, de ese modo, poder clasificarla correctamente. Pero esto sería negativo con respecto a la situación actual, si tenemos en cuenta que la línea definida actualmente ya se puede considerar una línea de separación bastante buena.

La traducción de esta intuición a términos formales introduce el factor de corrección siguiente:

$$c = \frac{1}{1+k}$$

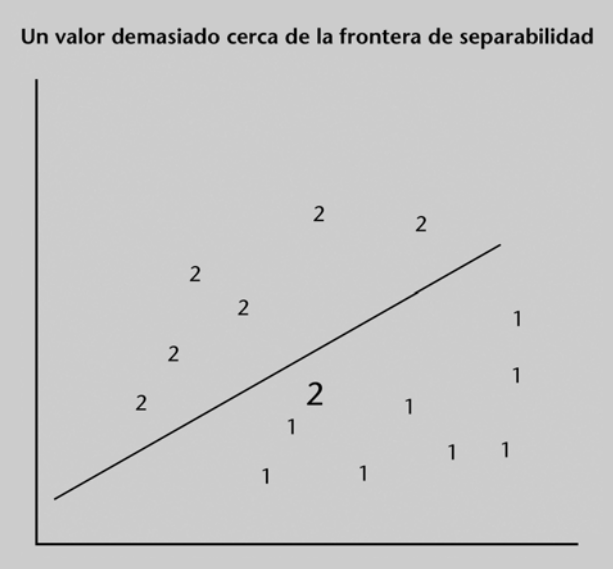
donde k es el factor de la corrección necesaria para ajustar los vectores de pesos de manera que una observación clasificada incorrectamente pase a estarlo correctamente. Para alcanzar la estabilidad en el comportamiento del clasificador, hay que ir reduciendo gradualmente la atención en lo que concierne a los errores importantes con el factor:

$$c = \frac{\beta}{\beta+k}$$

y ir atenuando el factor β durante el entrenamiento. Este valor está fijado por defecto en 0,995, pero el usuario del sistema puede modificarlo. Eso es equivalente a un proceso de *simulated annealing,* que busca la estabilidad del proceso haciendo que converja hacia una solución estable.

Un segundo problema es el de las observaciones que están demasiado cerca de la frontera entre clases. Estas observaciones hacen que el problema se convier-

ta en no separable. En la figura de la página siguiente tenemos la situación expresada de forma gráfica:



En este caso, k se aproxima a 0, lo cual hace que c se aproxime a 1, independientemente del valor que adopte β. En ese caso hay que atenuar el factor de corrección c independientemente de k, algo que se consigue multiplicando el coeficiente anterior por β. Entonces, el nuevo coeficiente c queda de la manera siguiente:

$$c = \frac{\beta^2}{\beta + k}$$

El algoritmo que sigue el método LMDT introduce la atenuación de β sólo cuando la magnitud de los vectores W_i se ha reducido en el paso actual de ajuste de pesos, pero se había incrementado en el ajuste de pesos que se hizo en la iteración anterior.

La **magnitud de los vectores** se define como la suma de las magnitudes de cada uno de éstos.

Este criterio ha sido adoptado ante la observación de que la magnitud de los vectores se incrementa rápidamente durante el comienzo del entrenamiento y se estabiliza cuando la línea de decisión se acerca a su ubicación final.

El proceso LMDT, pues, comprende los pasos siguientes:

1) Asignación inicial de valores a los vectores de pesos de las 1, ..., n clases para el nodo inicial.

2) Ajuste de valores de los pesos según las clasificaciones correctas.

3) Detección de las variables que presentan una menor contribución a la discriminación de los conjuntos de observaciones correspondientes a cada rama.

4) Eliminación de las variables menos discriminantes.

El proceso se repite para los nodos que se abren en el extremo de cada rama.

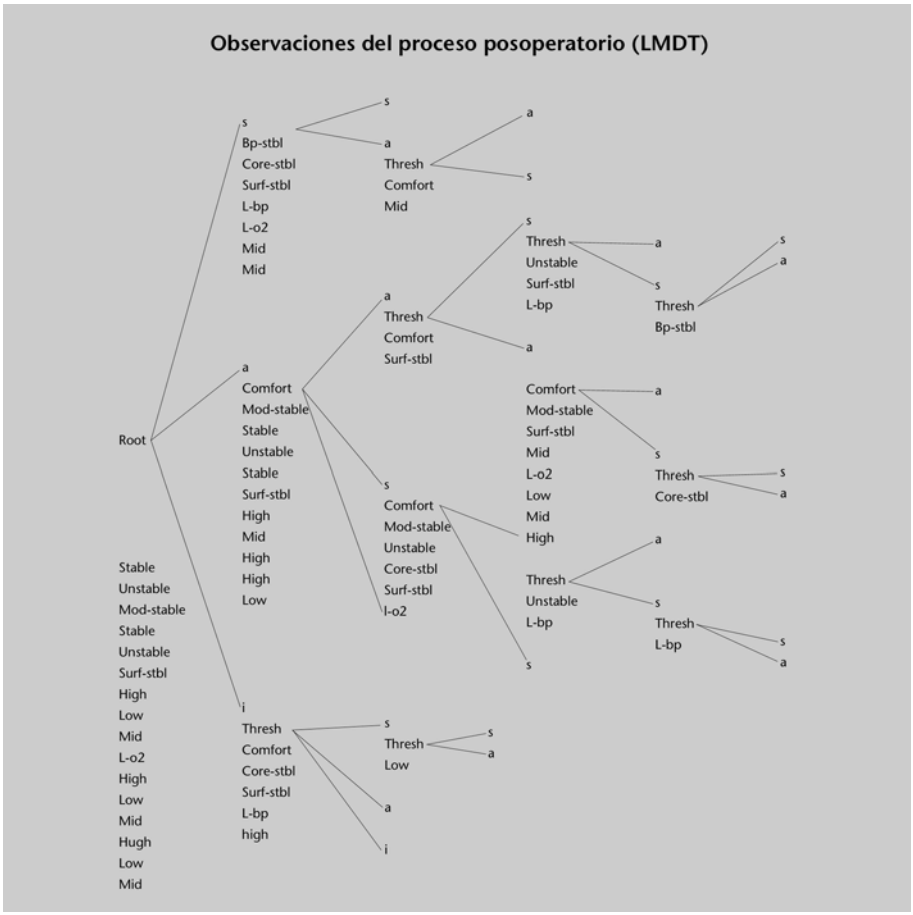
La **eliminación** consiste en tratar de eliminar los atributos uno por uno hasta que uno reduce la precisión de clasificación del nodo (medida como proporción de observaciones clasificadas de forma correcta).

La **medida de la contribución de un atributo** se calcula en función de la dispersión que genera en los pesos de cada una de las clases. Un peso importante contribuye positivamente a la función de discriminación para el atributo correspondiente y, por tanto, a su capacidad de discriminación. Un atributo que tiene sus pesos distribuidos de manera uniforme entre las clases posibles es poco discriminante y, en consecuencia, contribuye poco a las funciones discriminantes de cada clase.

El **criterio de eliminación de atributos** consiste, pues, en eliminar de la consideración aquellas variables que tienen sus pesos distribuidos de manera uniforme entre las clases.

La **medida de la dispersión de los pesos** se calcula mediante la distancia media cuadrática entre los pesos de cada par de clases. El atributo con la dispersión más baja se elimina.

Las distancias se presentan con más detalle en el módulo "Agregación (*clustering*)" de la presente asignatura.



Ejemplos de la aplicación de los métodos ID3 y LMDT

En este gráfico podemos ver los árboles resultantes de aplicar LMDT. Utilizamos el conjunto de valores sobre la evolución posoperatoria de pacientes de un hospital.

Alguno de los problemas que hemos apuntado ahora residen en la motivación práctica de los métodos de clasificación propios de las redes neuronales que veremos en el módulo correspondiente, y en otras técnicas como las *Support Vector Machines* (Burges, 1998).

6. Interpretación de los resultados obtenidos con árboles de decisión

El resultado de un proceso de construcción de un árbol de decisión es un clasificador. El clasificador ha inducido una partición en el conjunto de datos originales. Habrá que ver cuál es el significado de las particiones.

En ese sentido, y esto es válido también para otros modelos de clasificación como las redes neuronales o las reglas de asociación, hay que saber cómo se dividen los datos.

En general, las diferentes herramientas de construcción de árboles permiten visualizar algunos de los aspectos siguientes:

a) Número de observaciones de cada clase por cada nodo y condición sobre valores. Esto da una idea del grado de pureza que hay en cada nivel del árbol.

b) Atributo más discriminante según varios criterios. Por norma general, se valora según la misma función de construcción del árbol, pero también acostumbran a presentarse ordenados según el porcentaje de separación de clases.

c) Resto de atributos ordenados por valor de discriminación en aquel nivel. Esta operación nos permite hacernos una idea de la calidad de cada atributo como separador.

Lectura complementaria

Podéis consultar la visualización de las trayectorias de una observación con el *software* Sipina en la obra siguiente:

A. Zighed; J.P. Auray; G. Duru (1992). *Sipina: Méthode et Logiciel*. Lyon: Editions A. Lacassagne.

d) Tasa de error en cada nivel del árbol.

e) Trayectoria de cada observación (como hace, por ejemplo, Sipina). Esto permite conocer la estabilidad del proceso de clasificación, ya que podemos comparar cómo ha ido cambiando de partición una observación.

f) Matriz de confusión. Medida típica para expresar la calidad de las clases obtenidas con un modelo (podéis consultar el módulo sobre evaluación de modelos).

Podéis ver la matriz de confusión en el módulo "Evaluación de modelos" de esta asignatura.

7. Ponderación final de los árboles de decisión

En este apartado haremos algunos comentarios finales referentes a los distintos temas que hemos tratado en este módulo dedicado a los árboles de decisión.

Preparación de datos

No todos los métodos de construcción de árboles admiten atributos con valores continuos. De hecho, ID3 estaba pensado para funcionar con atributos categóricos. Si se dispone de datos continuos, hay que efectuar una discretización detallada que asegure una pérdida mínima de información. En caso de que sea necesario utilizar un método que permita la predicción de valores numéricos, entonces habrá que recurrir a los árboles que implementan regresión.

Evaluación de árboles de decisión

El método típico para evaluar un árbol de decisión consiste en aplicarlo a un conjunto de datos de prueba y evaluar su porcentaje de casos clasificados de foma correcta, o alguna otra medida de error de clasificación de las que hemos comentado. Ya hemos expresado cómo hay que ponderar la contribución de cada rama al error final de todo el modelo. Con respecto a esta medida básica, se puede seguir el método típico de validación cruzada.

Podéis ver la validación cruzada en el módulo "Evaluación de modelos" de esta asignatura.

Poda

La poda simplifica el modelo sin que éste pierda demasiada capacidad predictiva, pero introduce un coste adicional de cálculo.

Ventajas e inconvenientes

Comentamos algunas de las características principales de los árboles de decisión como modelos.

1) Eficiencia en clasificación: el número de tests que hay que realizar para decidir si es necesario o no efectuar una partición es, en el peor de los casos, tan alto como el número de atributos del dominio.

2) Comprensibilidad: cuanto más reducido es el árbol resultante, más sencilla es su interpretación. Con **árboles** que consideran un gran número de atributos y que generan un número de niveles elevado, surgen dificultades de interpretación. En principio, pues, son preferibles aquellos métodos que dan

árboles más "planos", bien porque acumulan varios tests en un mismo nodo (es decir, tienen en cuenta más de un atributo al mismo tiempo), bien porque aplican la poda. Por el contrario, los árboles multivariantes acostumbran a presentar condiciones más complejas en cada nodo.

3) Importancia relativa de los diferentes atributos para la clasificación: el nivel en que aparece un nodo dentro del árbol es una indicación de la relevancia del atributo correspondiente para la tarea de clasificación y, por tanto, nos da una interpretación adicional del dominio.

4) Inconvenientes: el inconveniente principal puede residir en la dificultad de comprensión de algunos **métodos** que generan árboles demasiado complejos. Por otra parte, si el conjunto de observaciones define una serie de particiones que no se pueden separar linealmente, los árboles que efectúan una división binaria pueden dar un mal rendimiento. Lo mismo sucede con los árboles multivariantes que sólo generan superficies (hiperplanos) rectangulares.

Resumen

Los árboles de decisión son un modelo de clasificación que se basa en encontrar atributos discriminantes en el sentido de que generan particiones de clasificación bastante uniformes.

La formulación original de los árboles de decisión puede remontarse hasta los árboles basados en clasificación y regresión propuestos por Breiman. Básicamente, este tipo de árboles proceden de manera iterativa, seleccionando en cada paso el atributo más discriminante y dividiendo el conjunto de datos en dos particiones según el valor de corte elegido para este atributo. La formulación más conocida para clasificación es la de Quinlan y sus métodos de inducción de árboles de decisión descendente (*top down*) e iterativa.

Una mejora posible de los árboles de decisión consiste en incorporar métodos de poda para impedir o recortar la generación de subárboles a los que les corresponden particiones con un porcentaje de error en clasificación no admisible.

Como sucede con otros sistemas de clasificación, los árboles de decisión tienen problemas para tratar dominios en los que no se puede asegurar la linealidad de las hipersuperficies de separación entre las diferentes clases. Una posible extensión y mejora para este tipo de problemas es la que presentan los árboles de decisión multivariantes lineales de Utgoff (LMDT, *Linear Multivariant Decision Trees*).

Finalmente, los árboles de decisión admiten una traducción fácil a reglas de decisión.

Actividades

1. Acceded a la dirección de Internet que podéis ver al margen y comparad las especificaciones de los diferentes *software* dirigidos a construir árboles.

Para hacer la actividad 1, acceded a la dirección *http://www.kdnuggets.com.*

2. Para el problema que os habíais propuesto inicialmente, ¿os sirven los árboles de decisión? ¿Qué método creéis que os resultaría más conveniente? Ved la actividad 1 del módulo "Extracción de conocimiento a partir de datos" de esta asignatura.

3. Utilizad como criterio de poda la tasa de error esperada de un árbol para indicar qué nodos habría que podar con el *software* Sipina para el conjunto de datos y el árbol que resulta de aplicar el método ID3. Se trata de una base de datos que podéis explorar más detalladamente con el *software* Sipina. Recoge datos de más de dos mil pasajeros del *Titanic* que indican qué clase de pasaje tenían (primera, segunda, tercera clase y tripulación, 1, 2, 3 y 4, respectivamente), la edad (1 = 'Mayor', 2 = 'Joven'), el sexo (1 = 'Hombre', 2 = 'Mujer') y si sobrevivieron o no (1 = 'Sí', 2 = 'No').

Pasajero	Clase	Edad	Sexo	Sobreviviente
1	1	1	1	1
2	1	1	1	1
3	1	1	1	1
4	1	1	1	1
5	1	1	1	1
6	1	1	1	1
7	1	1	1	1
8	1	1	1	1
324	1	2	1	1
325	1	2	2	1
326	2	1	1	1
327	2	1	1	1
328	2	1	1	1
329	2	1	1	1
330	2	1	1	1
331	2	1	1	1
332	2	1	1	1
1.998	4	1	1	2
1.999	4	1	1	2
2.000	4	1	1	2
2.001	4	1	1	2
2.002	4	1	1	2
2.003	4	1	1	2
2.004	4	1	1	2
2.005	4	1	1	2
2.006	4	1	1	2
2.007	4	1	1	2
2.201	4	1	2	2

4. Haced el mismo con el método CART.

Ejercicios de autoevaluación

Dado el siguiente **conjunto** de datos, si el atributo de clasificación es *Riesgo*:

Sexo	Edad	Velocidad	Color coche	Riesgo
Mujer	Mayor	Rápida	Rojo	No
Mujer	Joven	Rápida	Gris	Sí
Mujer	Mayor	Rápida	Gris	No
Mujer	Joven	Lenta	Rojo	Sí
Hombre	Joven	Lenta	Gris	No
Hombre	Joven	Lenta	Gris	No
Mujer	Joven	Rápida	Rojo	No
Hombre	Mayor	Lenta	Rojo	No
Mujer	Mayor	Lenta	Rojo	No
Mujer	Joven	Rápida	Rojo	No
Hombre	Mayor	Lenta	Rojo	No
Mujer	Mayor	Lenta	Rojo	No
Mujer	Joven	Rápida	Rojo	No
Mujer	Joven	Lenta	Rojo	No
Mujer	Mayor	Rápida	Gris	Sí
Hombre	Joven	Rápida	Gris	Sí
Mujer	Joven	Rápida	Rojo	Sí
Mujer	Mayor	Rápida	Rojo	No
Mujer	Joven	Rápida	Gris	Sí
Hombre	Joven	Lenta	Rojo	Sí
Mujer	Mayor	Rápida	Rojo	No
Mujer	Joven	Rápida	Rojo	Sí
Mujer	Joven	Rápida	Rojo	No
Mujer	Joven	Rápida	Rojo	No
Hombre	Mayor	Lenta	Rojo	No
Mujer	Mayor	Rápida	Rojo	No
Mujer	Joven	Rápida	Rojo	No

a) Listad las particiones que se pueden obtener atendiendo únicamente al valor *Riesgo*.
b) Calculad el valor de información (entropía) de cada una de las particiones.

2. Aplicad ID3 al conjunto anterior de datos suponiendo que el atributo de clasificación es *Riesgo*.

3. Encontrad la tasa de error del árbol que habéis obtenido con respecto al conjunto de datos de prueba siguiente.

Sexo	Edad	Velocidad	Color del coche	Riesgo
Mujer	Joven	Rápida	Gris	No
Hombre	Joven	Rápida	Rojo	Sí
Hombre	Joven	Rápida	Rojo	Sí
Mujer	Mayor	Rápida	Rojo	No
Hombre	Mayor	Rápida	Gris	Sí
Mujer	Joven	Rápida	Rojo	Sí
Hombre	Mayor	Lenta	Rojo	No
Hombre	Joven	Rápida	Gris	Sí
Mujer	Mayor	Rápida	Rojo	Sí
Hombre	Mayor	Lenta	Gris	No
Mujer	Joven	Rápida	Gris	No
Hombre	Joven	Rápida	Rojo	No
Hombre	Mayor	Lenta	Gris	Sí

Bibliografía

Breiman, L.; Friedman, H.; Olshen, R.; Stone, C. (1984). *Classification and Regression Trees*. Belmont: Wadsworth.

Brodley, C.E.; Utgoff, P. (1992). "Multivariate Decision Trees". *Reporte Técnico, MASSCS-92-93*. Amherst: University of Massachusetts.

Burges, C.J. (1998). "A Tutorial on Support Vector Machines for Pattern Recognition". *Data Mining and Knowledge Discovery* (vol. 2, núm. 2).

Cendrowska, J. (1987). "PRISM: an Algorithm for Inducing Modular Rules". *International Journal of Man-Machine Studies* (vol. 4, núm. 27, págs. 349-370).

Esposito, F.; Malerba, D.; Semeraro, G. (1997). "A Comparative Analysis of Methods for Pruning Decision Trees". *IEEE Transactions on Pattern Analysis and Machine Intelligence* (vol. 19, págs. 476-493).

Hartigan, J.A. (1975). *Clustering Algorithms*. Nueva York: John Wiley.

Ileath, D.; Kasif, S.; Salzberg, S. (1993). "Induction of Oblique Decision Trees". *Proceedings of the 13th International Conference in Artificial Intelligence* (págs. 1002-1007). IJCAI-93. San Francisco: Morgan Kaufmann.

Krichevsky, R.E.; Trofimov, V.K. (1983). "The Performance of Universal Coding". *IEEE Transactions on Information Theory* (vol. IT-27, núm. 2).

Lim, T.S.; Loh, W.Y.; Shih, Y.S. (1997). *An Empirical Comparison of Decision Trees and Other Classification Methods*. Reporte Técnico TR-979. Madison: Department of Statistics, University of Wisconsin-Madison.

Lim, T.S.; Shih, Y.S. (1997). "Split Selection Methods for Classification Trees". *Statistica Sinica*.

López de Màntaras, R. (1991). "A Distance-Based Attribute Selection Measure for Decision Tree Induction". *Machine Learning* (vol. 6, núm. 1, págs. 81-92).

Mingers, J. (1989). "An Empirical Comparison of Pruning Methods for Decision Trees Induction". *Machine Learning* (núm. 4, págs. 227-243).

Quinlan, J.R. (1987). "Generating Production Rules from Induction Trees". *Proceedings of the Fourth International Machine Learning Workshop* (págs. 304-307). San Francisco: Morgan Kaufmann.

Quinlan, J.R. (1987). "Simplifying Decisions Trees". *International Journal of Man-Machine Studies* (núm. 27, págs. 221-234).

Quinlan, J.R. (1993). *C4.5: Programs for Machine Learning*. Morgan Kaufmann.

Rissanen, M. (1985). "The Minimum Description Length Principle". En: Kotz, S.; Johnson, N.L. (eds.). *Encyclopedia of Statistical Sciences* (vol. 5). Nueva York: John Wiley & Sons.

Rissanen, M. (1989). *Stochastic Complexity in Statistical Inquiry*. World Scientific Publishers Company.

Zighed, A.; Auray, J.P.; Duru, G. (1992). *Sipina: Méthode et Logiciel*. Lyon: Editions A. Lacassagne.