Arboles de decisión [1]

J. Minguillón y J. Pujol

Depto. de Informática
Universidad Autónoma de Barcelona.
e-mail: {Julia.Minguillon,Jaume.Pujol}@uab.es

Resumen

En este artículo presentamos el trabajo realizado por los autores dentro del marco general de clasificación supervisada mediante el uso de árboles de decisión. Entre los sistemas de clasificación, los árboles de decisión constituyen uno de los métodos más utilizados por su simplicidad y por su facilidad de construcción. No obstante, usualmente han sido substituidos por otros métodos (redes neuronales, *support vector machines*,...) debido en parte a su dependencia de los conjuntos de datos utilizados en el entrenamiento y debido también a su limitada capacidad para predecir resultados.

No obstante, cuando los árboles de decisión son utilizados dentro del marco general de la combinación de clasificadores pueden ayudar a la toma de decisiones. Además hemos comprobado que, mediante la introducción de una nueva clase de árboles de decisión, los árboles de decisión progresivos, es posible alcanzar resultados tan buenos (y en algunos casos mejores) como con otros métodos pero con un coste mucho menor.

Los resultados obtenidos han podido verificarse en diversas aplicaciones (reconocimiento de documentos, clasificación de imágenes hiperespectrales, diagnosis médica,...) que han permitido corroborar que la propuesta presentada aporta avances significativos en el uso de los árboles de decisión en los sistemas de clasificación supervisados.

1 Arboles de decisión

Un árbol de decisión T es un árbol m-ario con un conjunto de hojas que denotaremos $\tilde{T}$. Si $m = 2$ será un árbol de decisión binario. En todo este trabajo sólo trataremos con árboles de decisión binarios por lo que obviaremos el término binario. La Figura 1 muestra un ejemplo de un árbol de decisión tomado de Breiman et alt.[2] y que fue desarrollado para una aplicación de diagnosis médica.

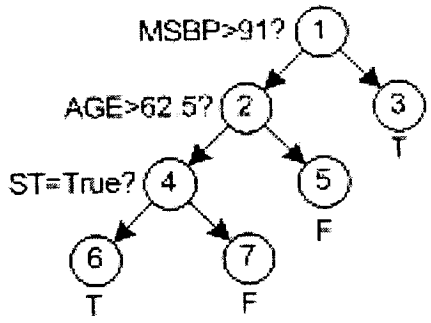

Figura 1: Ejemplo de árbol de decisión.

Los árboles de decisión son el resultado de un método de aprendizaje supervisado, esto es, se necesita un conjunto de entrenamiento correctamente etiquetado para crear el clasificador. Sea $D_n = \{(X_i, Y_i) \mid X_i \in \mathbf{R}^d, Y_i \in \{0, 1 \ldots, K-1\}, i = 1, \ldots, n\}$ el conjunto de entrenamiento etiquetado. Está formado por una secuencia de n pares (X_i, Y_i) donde X_i es un vector d-dimensional y Y_i representa la clase asociada a X_i. Puesto que la distribución real de todos los pares (X, Y) es desconocida, el conjunto (X_i, Y_i) puede verse como una muestra de la distribución conjunta (X, Y) que se desea reconocer. Cuanto más representativa sea la secuencia (X_i, Y_i) mejor será el resultado del proceso de clasificación.

La construcción de los árboles de decisión se basa en un algoritmo que permite obtener una partición finita del espacio $\mathbf{R}^d$:

[1] Trabajo parcialmente subvencionado por el proyecto CICYT de referencia TIC2000-0739-C04-01

Algoritmo de Crecimiento
T=(conjunto de datos, D_n) /* inicialmente el árbol contiene una sola hoja */
while CriterioDeParada(T) es falso
seleccionar t de T que maximiza CriterioDeSelección(t)
partir $t = (t_L, t_R)$ que maximiza CriterioDePartición(t,t_L,t_R)
reemplazar t en T con (t_L, t_R)
end

podar T usando el algoritmo BFOS
elegir el subárbol T' de T que minimiza el error de clasificación en un segundo conjunto de datos D'_n

La partición obtenida puede ser descrita mediante un árbol: los nodos internos son preguntas, $t : \mathbf{R}^d \rightarrow \{1, \ldots, s\}$, y las hojas son conjuntos de datos etiquetados según una regla de etiquetado fija. Normalmente, las preguntas representan hiperplanos, $H\mathbf{x} \leq h$. Las hojas se etiquetan atendiendo a la siguiente regla de etiquetado: dada una hoja t,

$$l(t) = \arg_j \min\{r(t) = \sum_{k=0}^{K-1} C(j,k)p(k|t)\}$$
$$j = 0, \ldots, K-1$$

donde K es el número de clases, $C(j,k)$ es el coste de clasificar la clase j como clase k, y $p(k|t)$ es la probabilidad estimada de la clase k en t.

El algoritmo de crecimiento depende, básicamente, de tres parámetros:

a) *Condición de parada.* Es la condición usada para detener el crecimiento del árbol. Puede depender de la estructura del árbol, número máximo de iteraciones o profundidad del árbol, o depender del error de clasificación.

b) *Selección del nodo.* Para cada hoja $t \in \tilde{T}$ una función de selección es calculada dependiendo de características locales o globales: pureza del nodo, nodo más poblado,...

c) *Partición del nodo.* Esta es la característica más crítica en el crecimiento de un árbol de decisión puesto que una elección errónea se traslada a los subárboles de este nodo. Los métodos más utilizados y estudiados dependen de un criterio de impureza, tratando de hallar la partición que maximiza una disminución de la impureza. Ejemplos de funciones de impureza pueden ser el error de Bayes $\Phi(p) = 1 - \max\{p_i\}$, la función Gini $\Phi(p) = 1 - \sum_{i=0}^{k-1} p_i^2$ [2], la entropía $\Phi(p) - \sum_{i=0}^{K-1} p_i \log p_i$, [3], la R-norma $\Phi(p) = \frac{R}{R-1}[1 - (\sum_{i=0}^{K-1} p_i^R)^{1/R}]$ [1].

La impureza de un nodo se define como $i(t) = \Phi(p(0|t), \ldots, p(K-1|t))$, donde $p(j|t)$ es la proporción de elementos que pertenecen a la clase j en el nodo t.

Computacionalmente, el problema de la construcción de árboles de decisión es un problema difícil. El número de posibles árboles generados por el algoritmo de crecimiento crece exponencialmente con el número de iteraciones y con el número de datos disponibles. La secuencia de número de hojas disponibles para partir en el caso binario es 1, 2, 5, 14,... que corresponde a la secuencia de los números de Catalan que crece exponencialmente. Además, el problema de la construcción de un árbol de decisión óptimo es un problema NP-completo [7]. Finalmente, el problema de hallar la partición del nodo que minimiza el número de puntos mal clasificados, dados dos conjuntos de puntos mutuamente excluyentes, es también NP-completo [6].

El rendimiento de un árbol de decisión se mide mediante un estimador definido por:

$$\hat{L}_n(T) = \frac{1}{n} \sum_{i=1}^{n} 1_{T(X_i) \neq Y_i}$$

donde 1 es una función que se evalúa como 1, cuando la condición de evaluación es cierta, o 0, en caso contrario; y $T(X_i)$ es la etiqueta asignada por el árbol T al vector X_i.

Por tanto, $\hat{L}_n(T)$ cuenta el número de errores que T comete clasificando D_n. Aunque el árbol obtenido es adecuado para el conjunto de entrenamiento D_n, suele dar pobres resultado para otros

conjuntos de datos. Este fenómeno, llamado *overfitting*, se produce, por ejemplo, cuando el conjunto de entrenamiento D_n es particionado hasta que todas las hojas de T son puras, es decir, cuando $\hat{L}_n(T) = 0$.

Para minimizar este problema se utiliza un proceso de poda (*pruning algorithm*, también llamado BFOS) para hallar un subárbol de T que minimiza el error de clasificación para un segundo conjunto de datos llamado el *corpus set*.

Incluso, después del algoritmo de poda, el árbol de decisión resultante puede ser grande y muy específico. Nuestro objetivo es la construcción de un sistema de clasificación basado en árboles de decisión que no sufra de *overfitting* (o al menos, que el sistema intente minimizarlo), con un coste de clasificación reducido, use un conjunto reducido de características de clasificación y tenga un buen rendimiento cuando el número de clases presentes en el conjunto de datos sea elevado.

2 Arboles de decisión progresivos

En esta sección presentamos un nuevo sistema de clasificación que intenta resolver algunos de los problemas clásicos que presentan los árboles de decisión. Trataremos de construir un sistema de clasificación que permita:

a) Mejorar el error de clasificación reduciendo el *overfitting*.

b) Reducir la complejidad de los algoritmos de crecimiento y poda.

c) Agrupar elementos de clases diferentes en una sola superclase que podría ser clasificada posteriormente por otro clasificador.

Este nuevo sistema se basa en el concepto de *árbol de decisión progresivo*. Intuitivamente, un árbol de decisión progresivo es un conjunto de árboles de decisión siendo cada uno de ellos un clasificador parcial.

En el proceso de creación de un árbol de decisión introducimos una nueva clase que llamaremos clase *mixta* y que denotaremos por M. La nueva regla de etiquetado pasaría a ser:

$$l'(t) = \begin{cases} l(t) & \text{si } r(t) \leq \epsilon \\ M & \text{en caso contrario.} \end{cases}$$

donde $l(t)$ y $r(t)$ están definidos en la sección anterior; y ϵ es un límite que determina la impureza máxima admisible de una hoja. ϵ está relacionado con el concepto de margen [12].

En esta nueva situación, el algoritmo de crecimiento debe ser modificado de tal manera que cuando una hoja t es dividida, un nuevo subárbol es creado. Este subárbol es entrenado usando sólo aquellos vectores de D_n que estaban en t, es decir, la región del espacio representada por t.

En el proceso de poda, cuando varias hojas son clasificadas en la clase mixta por ser demasiado "impuras", todas estas hojas son agrupadas en un nuevo conjunto que representa todo el espacio inicial con un número finito de "agujeros". Esta nueva hoja es etiquetada como mixta, así el proceso es repetido recursivamente, en un segundo proceso de poda.

La figura 2 muestra un ejemplo bidimensional de árbol de decisión progresivo.

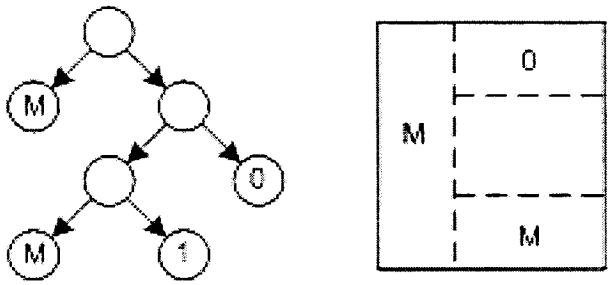

Figura 2: Regiones en un árbol de decisión progresivo.

Cuando ϵ es pequeño muchas hojas son etiquetadas como mixtas y agrupadas para crear árboles de dimensión pequeña. Si ϵ es demasiado pequeño, entonces se obtiene un árbol de decisión degenerado con una sola hoja etiquetada como mixta. Si ϵ es grande, pueden obtenerse árboles grandes pero más específicos. Los experimentos demuestran que es conveniente que los primeros árboles sean pequeños (tomando ϵ pequeño) mientras que a medida que el sistema va clasificando elementos es conveniente la creación de árboles más específicos (tomando ϵ de mayor tamaño).

Cuando dos o más hojas son agrupadas para crear una nueva hoja que será recursivamente particionada, los árboles de decisión se convierten en

grafos de decisión acíclicos (DAG, [11]).

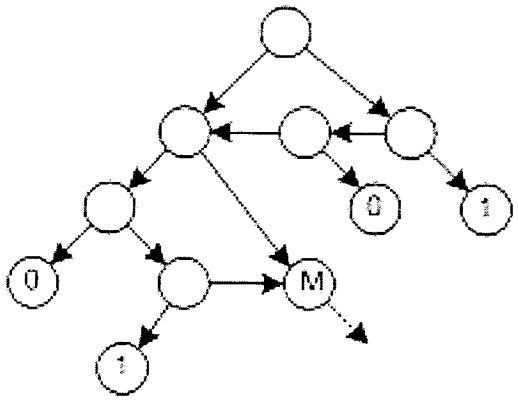

Figura 3: Grafo de decisión progresivo.

3 Combinación de clasificadores

La principal idea subyacente en los árboles de decisión progresivos es la clasificación parcial: un árbol inicial, de reducido tamaño, clasifica un porcentaje de los datos de entrada con un error pequeño y a un coste reducido. Posteriormente, un conjunto de árboles de decisión progresivos de pequeño tamaño son combinados para mejorar la clasificación a un coste reducido. En general, uno o más modelos de primer nivel son combinados usando un segundo nivel para mejorar el rendimiento del sistema de clasificación. Existen, básicamente, tres métodos de combinar clasificadores: *voting*, *stacking* y *cascading*. Los dos primeros pueden considerarse *sistemas multiexpertos*, el tercero es un *sistema multietapa*.

En un esquema de votación (*voting*) cada árbol de decisión hace una predicción de cada vector de entrada. La predicción que recibe más votos es elegida como la predicción final. En un esquema de *stacking* se usa un sistema de aprendizaje para crear un árbol de árboles que combina las predicciones de los árboles de decisión. Finalmente, *cascading* es un proceso iterativo de combinación de clasificadores: en cada iteración, el conjunto de entrenamiento es extendido con las predicciones obtenidas en la iteración anterior.

La combinación de clasificadores trata de extraer lo mejor de cada uno de ellos: Gama y Brazdil [5] prueban que el error combinado depende del error de los clasificadores individuales y de la correlación entre ellos. La descomposición bias-varianza [4] es una herramienta muy útil para medir la eficiencia y las diferencias de comportamiento de diferentes clasificadores para un mismo conjunto de entrenamiento.

Nuestra propuesta puede ser vista como un caso particular de *cascading*: un árbol inicial trata de clasificar aquellos vectores que parecen más fáciles de clasificar. Posteriormente, en etapas sucesivas se clasifican el resto de vectores.

Siguiendo Gama et alt. [5] proponemos tres tipos distintos de combinar árboles de decisión dentro del paradigma de *cascading*, según se use, o no, información adicional:

Tipo A: No se hace uso de información adicional y sólo los vectores clasificados como clase mixta son usados en la siguiente iteración.

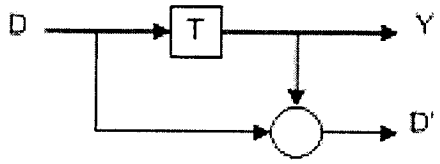

Figura 4: Arquitectura combinación Tipo A.

Tipo B: En la iteración siguiente se hace uso de información adicional (la probabilidad estimada de las clases y el margen).

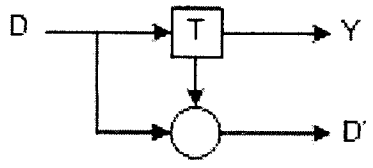

Figura 5: Arquitectura combinación Tipo B.

Tipo C: Este tercer modelo es la combinación de los dos anteriores. Se hace uso de información adicional para todos aquellos vectores que no han sido clasificados en la iteración precedente.

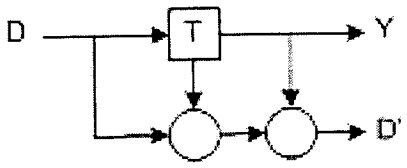

Figura 6: Arquitectura combinación Tipo C.

4 Resultados experimentales y aplicaciones

En este capítulo presentamos algunos de los resultados más relevantes obtenidos con distintos conjuntos de datos usando los árboles de decisión progresivos junto con los esquemas de *cascading* propuestos en la sección anterior.

Los datos provienen tanto de una base de datos aceptada comúnmente como estándar por la comunidad internacional (*UCI collection*), como de conjunto de datos reales obtenidos de tres proyectos de clasificación: un sistema de reconocimiento de documentos [8], un sistema de clasificación de imágenes hiperespectrales [9] y un sistema de diagnosis de tumores cerebrales [10].

4.1 Sistema de reconocimiento de documentos

El objetivo consistía en el reconocimiento de las distintas partes de un documento. Para reducir los costes de almacenamiento era necesario automatizar el reconocimiento de los distintos tipos de información del documento (fondo, texto, imágenes,...) para poder aplicar distintos esquemas de compresión (con pérdida y sin pérdida). Básicamente, debían reconocerse cuatro clases: el fondo, el texto, las líneas de demarcación y gráficos, y las imágenes.

La imagen se segmenta en bloques de 8×8 pixels y cada bloque intenta clasificarse utilizando valores calculados para cada bloque. La tabla siguiente muestra los resultados obtenidos utilizando un único árbol de decisión:

$N \times N$	Núm. Bloques	$\|T\|$	R	d_m	Error
8×8	211200/211200	721	8.56	38	0.078

En esta tabla $|T|$ representa el número de nodos del árbol, R es la altura media del árbol y d_m la profundidad máxima del árbol construido.

La tabla siguiente muestra los resultados obtenidos utilizando una aproximación progresiva (de hecho jerárquica):

$N \times N$	Núm. Bloques	$\|T\|$	R	d_m	Error
64×64	3360/3360	6	2.77	4	0.089
32×32	7856/13440	14	4.17	6	0.047
16×16	21052/53760	11	3.72	6	0.042
8×8	27892/215040	18	4.73	8	0.065

El resultado más interesante es que sólo 60160 bloques necesitan ser clasificados en un esquema de clasificación progresivo mientras que un único árbol de decisión debe clasificar 211200 bloques, así que el coste de clasificación se ve reducido, así como igualmente se reduce el coste de entrenamiento del sistema. Además el sistema de clasificación multietapa genera zonas homogéneas grandes que facilitan la interpretación en un sistema de reconocimiento de documentos.

4.2 Clasificación de imágenes hiperespectrales

El objetivo consistía en construir un sistema de clasificación parcial que permitiera identificar un gran número de clases (19 clases) y las bandas (de un total de 14) que aportaban la mayor información para cada clase, detectando las áreas del conjunto de entrenamiento posiblemente mal etiquetadas de origen.

Usando un árbol de decisión clásico obtenemos los resultados siguientes:

Árbol	$\|T\|$	R	P_T	Error
T_1	836	9.83	1.0	0.163

En esta tabla, P_T es el porcentaje de vectores de entrada procesados por el árbol T. Usando árboles de decisión progresivos con sólo dos etapas obtenemos:

Árbol	$\|T\|$	R	P_T	Error
T_{2A}	9	3.02	0.523	0.056
T_{2B}	8	2.14	0.383	0.199
T_2	44	4.84	0.706	0.094

En este caso observamos la enorme diferencia entre la profundidad del árbol T_1 y la profundidad del árbol progresivo T_2.

4.3 Clasificación de tumores cerebrales

En este caso el objetivo es la creación de un sistema de clasificación de tumores cerebrales para la ayuda al diagnóstico clínico. En este caso, el principal problema para la construcción de un buen clasificador se encuentra en el gran número de clases que deben ser reconocidas y el reducido número de muestras que se disponen. En este caso se ha utilizado un sistema mixto que combina distintos clasificadores: un sistema de votación y un sistema de *cascading*. La utilización de k-NN, LDA y árboles de decisión obedece a criterios de diferencia según la descomposición bias-varianza. Cuando los clasificadores no coinciden en su predicción, se utiliza una clase especial "mixta" para indicarlo, reduciendo el número de errores cometidos.

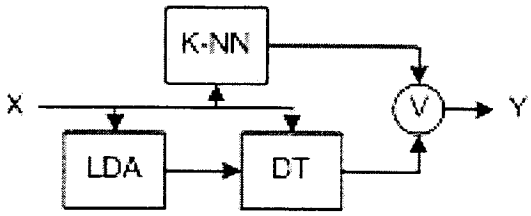

Figura 7: Combinación de clasificadores en la clasificación de tumores cerebrales.

Con esta arquitectura se ha procedido a la clasificación del conjunto de muestras. La Figura 8 resume los resultados obtenidos. El porcentaje de cada recuadro representa el porcentaje de muestras correctamente clasificadas para cada categoría, mientras que en cada bifurcación se muestra el porcentaje de muestras clasificadas.

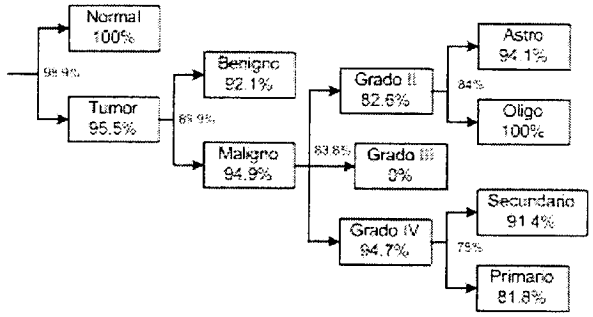

Figura 8: Clasificación de tumores.

Referencias

[1] Boekee, D. E. and der Lubbe, J. C. A. V. *The R-norm information measure.* Information and Control 45: 136155. 1980.

[2] Breiman, L. et al. *Classification and Regression Trees.* Wadsworth International Group. 1984.

[3] Cover, T. M. and Thomas, J. A. *Elements of Information Theory.* John Wiley & Sons. 1991.

[4] Domingos, P. *A unified bias-variance decomposition and its applications.* Proceedings of the 17th Int. Conf. on Machine Learning, pp. 231–238. 2000.

[5] Gama, J. et al. *Cascade Generalization.* Machine Learning. 41(3):315–343.

[6] Heath, D. *A geometric framewwork for machine learning.* PhD thesis, The Johns Hopkins University, Baltimore, MD, USA. 1992.

[7] Hyafil, L. and Rivest, R. L. *Constructing optimal binary decision trees is NP-complete.* Information Processing Letters 5(1): 15-17. 1976.

[8] Minguillón, J., Pujol, J. and Zeger, K. *Progressive classification scheme for document layout recognition.* SPIE Proceedings, Mathematical Modeling, Bayesian Estimation, and Inverse Problems, Vol. 3816, Denver, CO, pp. 241–250. 1999.

[9] Minguillón, J., Pujol, J., Serra, J., Ortuño, I. and Guitart, P. *Adaptive lossy compression and classification of hyperspectral images.* Proceedings of Image and Signal Processing for Remote Sensing VI, Vol. 4170, Barcelona, Spain, pp. 214–225. 2002.

[10] Minguillón, J., Tate, A. R., Arús, C. and Griffiths, J. R. *Classifier combination for in vivo magnetic resonance spectra of brain tumours.* in F. Roli (ed.), Multiple Classifier Systems, Lecture Notes in Computer Science, Springer. 2002.

[11] Oliver, J. J. *Decision graphs - an extension of decision trees.* Proceedings of the Fourth International Workshop on Artificial Intelligence and Statistics, pp. 343-350. Extended version available as TR 173, Dept. of Computer Science, Monash University, Clayton, Victoria 3168, Australia. 1993.

[12] Schapire, R. E., Freund, Y., Bartlett, P. and Lee, W. S. *Boosting the margin: a new explanation for the effectiveness of voting methods.* Annals of Statistics, **26**(5): 1651–1686.