Sistemas Inteligentes

Escuela Técnica Superior de Informática

Universitat Politècnica de València

Tema B2T3
Árboles de Clasificación

SIN

Índice

Árboles de decisión y clasificación (ADC)

Los árboles de clasificación (también llamados de decisión o de identificación) se enmarcan en la aproximación no paramétrica al Reconocimiento de Formas. Constituyen un forma de representación del conocimiento especialmente simple y efectiva.

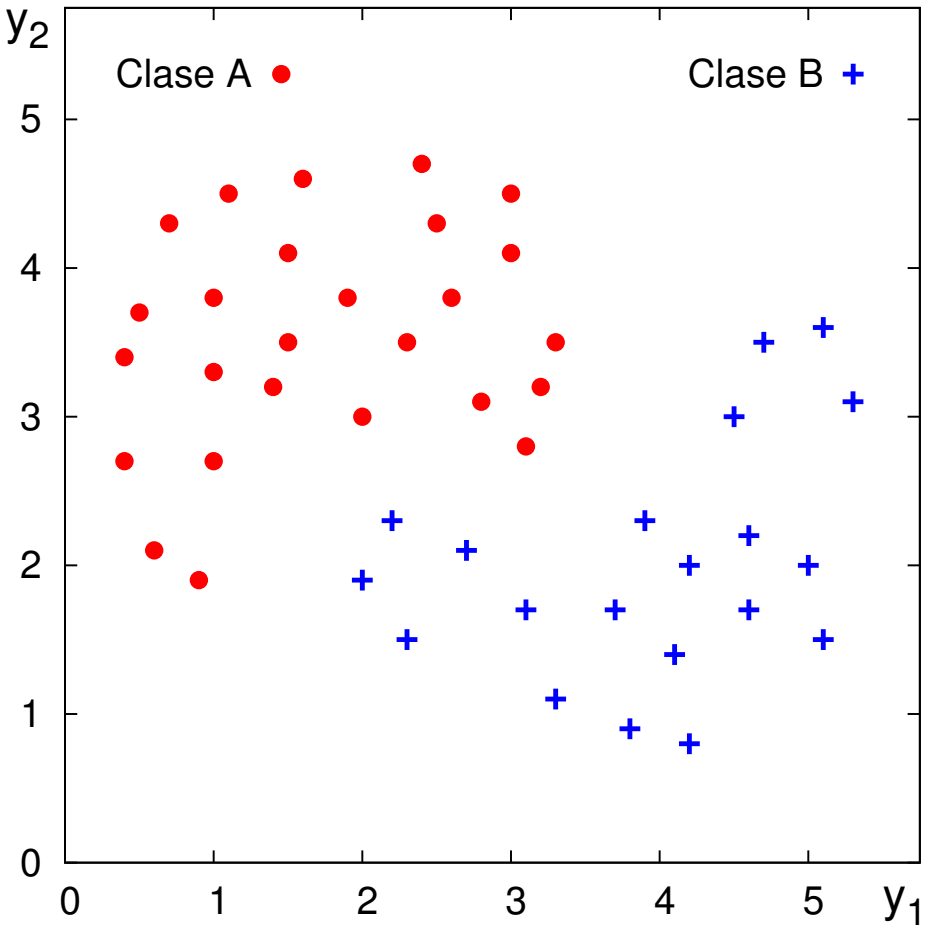
Un árbol de clasificación es la estructura resultante de la *partición recursiva* del *espacio de representación* a partir de una *muestra de aprendizaje*.

Cada nodo interno contiene una pregunta sobre un atributo concreto (con un hijo por cada posible respuesta) y cada nodo hoja o terminal contine una etiqueta de clase y corresponde a una decisión final (clasificación).

Una dato de test se clasifica mediante una serie de preguntas sobre los valores de sus atributos, empezado por el nodo raiz y siguiendo el camino determinado por las respuestas a las preguntas de los nodos internos, hasta llegar a un nodo hoja. La etiqueta de esta hoja es la que se asignará a la muestra a clasificar.

Entre los posibles clasificadores basados en árboles (ID3, C4, C4.5, árboles Bayesianos, etc.) estudiaremos CART (*"Classification And Regression Trees"* o *árboles de clasificación y regresión*) [3]. CART se caracteriaza por adoptar una partición de nodos exclusivamente binaria basada en criterios estadísticos sólidos.

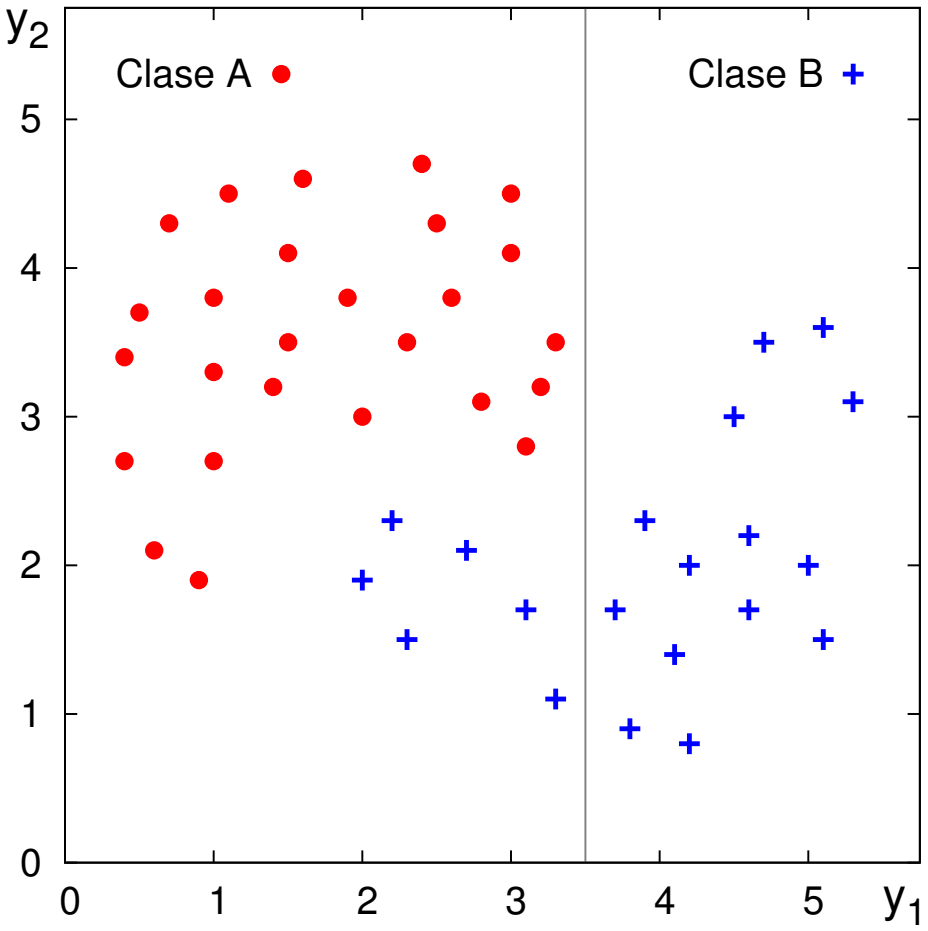
Ejemplo

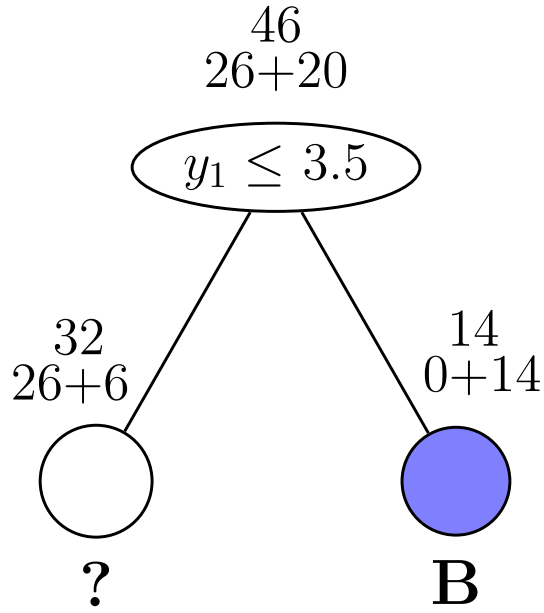


Tarea simple ilustrativa:

- representación en dos dimensiones ($E = \mathbb{R}^2$)
- clasificación en 2 clases *no* separables linealmente
- 46 datos (vectores)
  - 26 de clase A
  - 20 de clase B

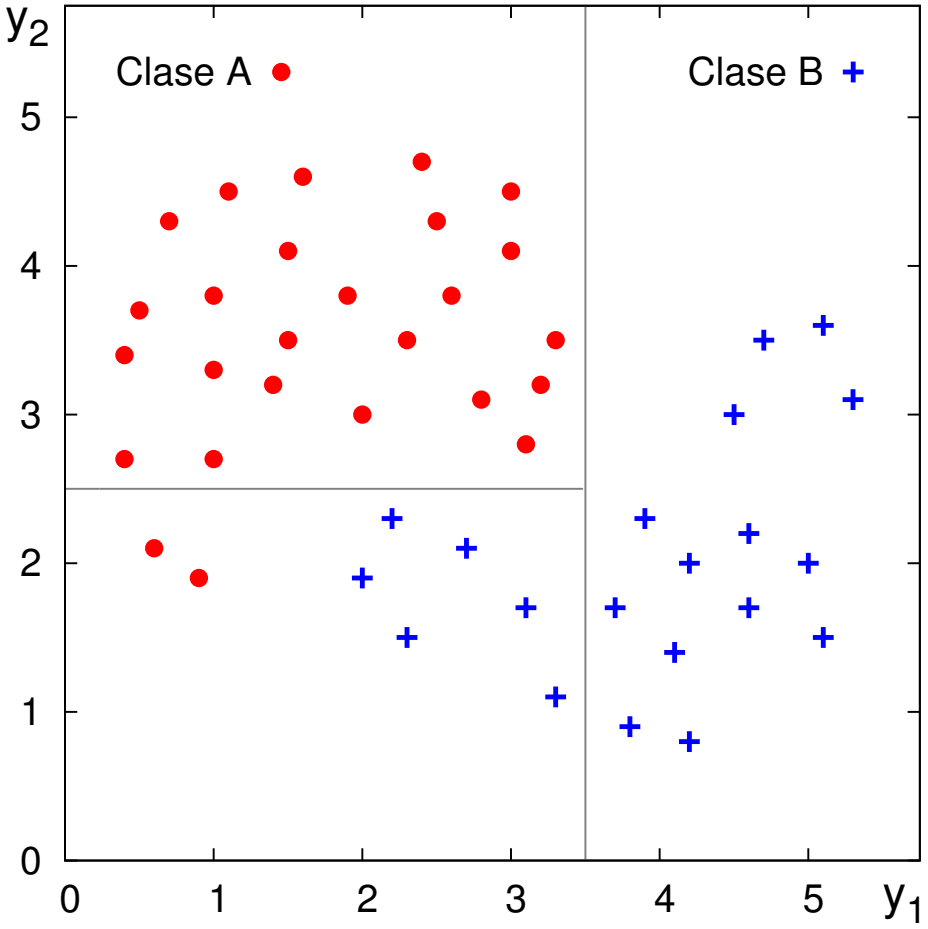
Ejemplo: Primera partición

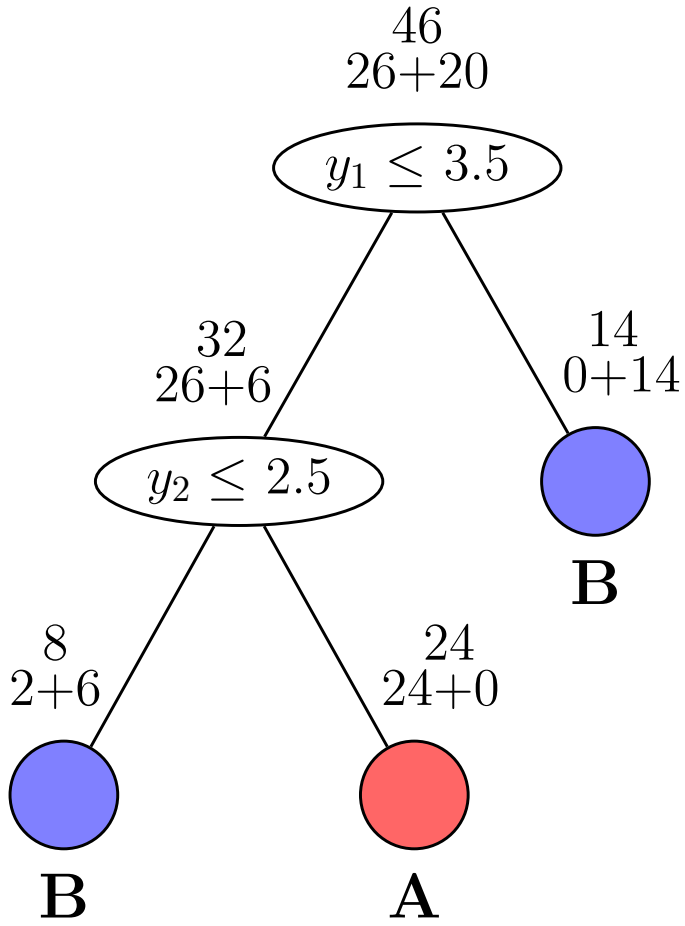




El nodo raíz tiene asociado el conjunto completo de datos. La primera partición se establece en base a la pregunta: ¿$y_1 \leq 3.5$?. El nodo de la derecha contiene 14 datos. Como todos son de la clase B se dice que es “puro” por lo que puede declararse nodo *terminal* y etiquetarse como de clase B.

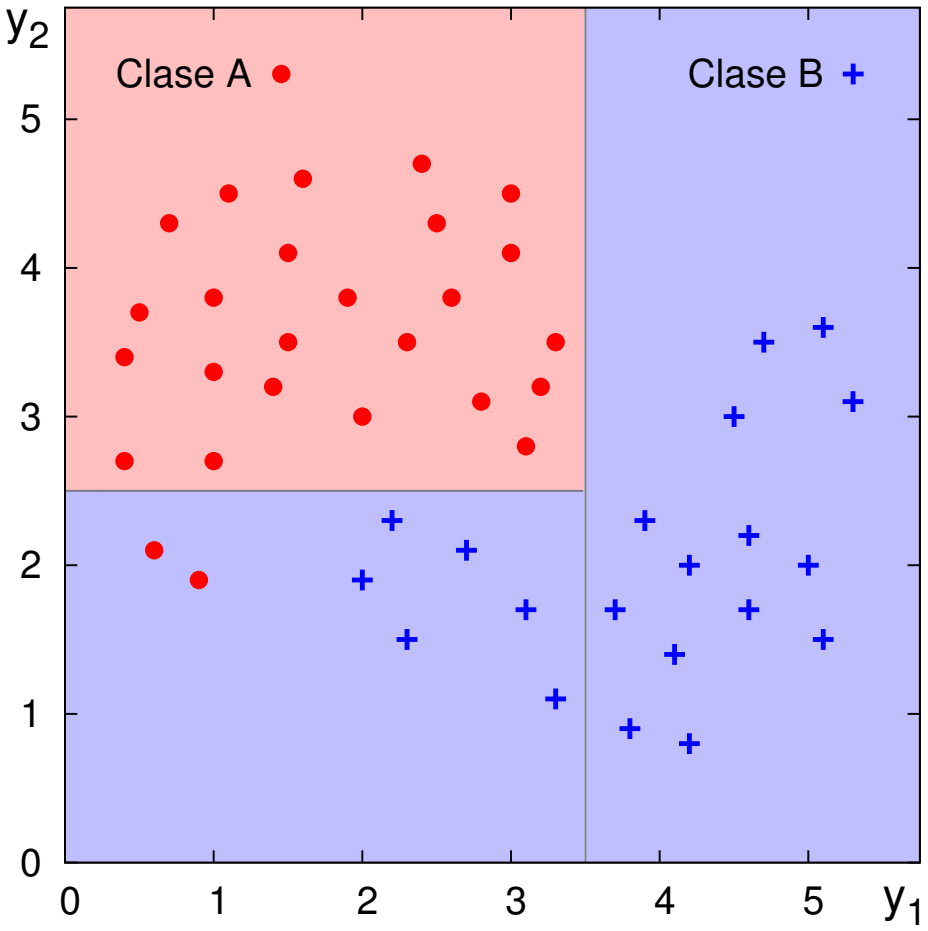
Ejemplo: segunda partición y fronteras de decisión

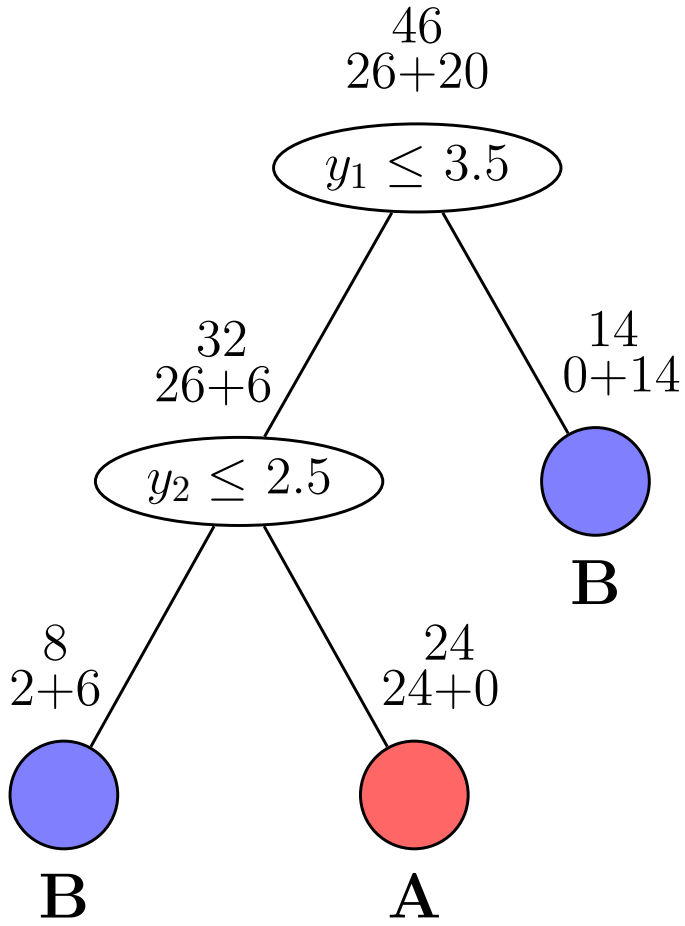




En el nodo de la izquierda se procede a una segunda partición con la pregunta: ¿$y_2 \leq 2.5$?. El nodo derecho es puro y se etiqueta de clase A. El izquierdo aún se podría partir hasta lograr nodos puros, pero se decide terminar y etiquetar este nodo con la clase mayoritariamente representada, la clase B.

Ejemplo: regiones de decisión

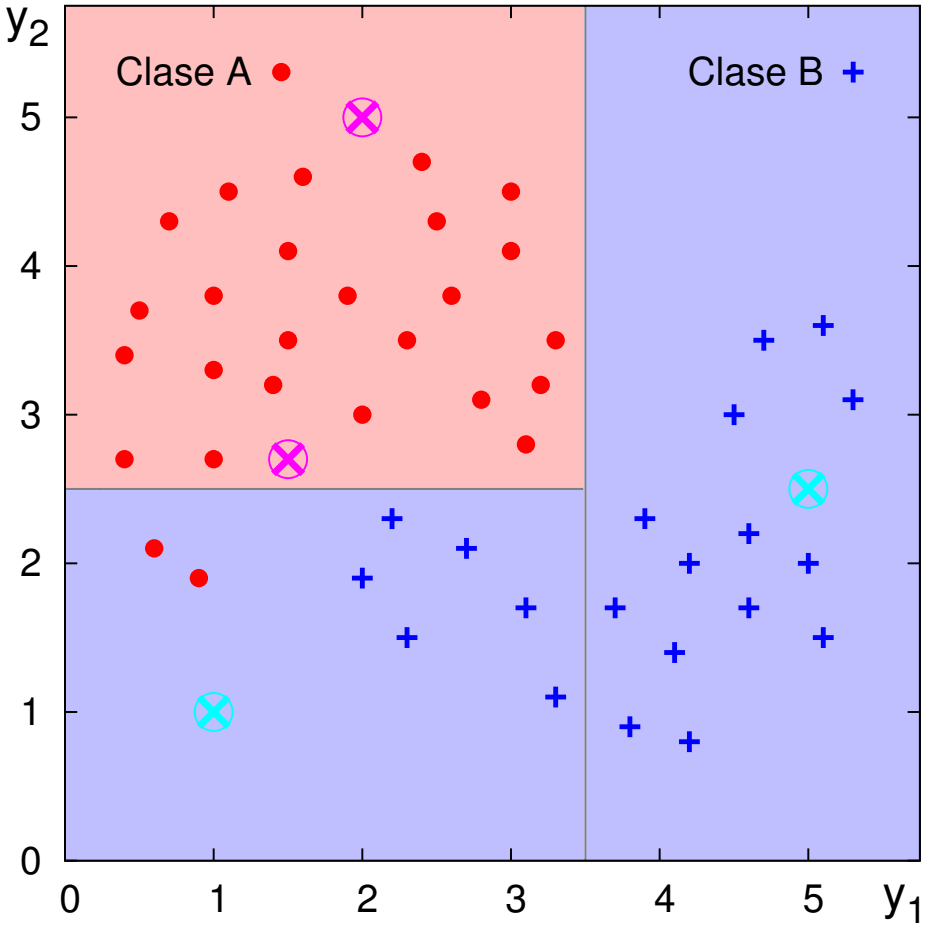


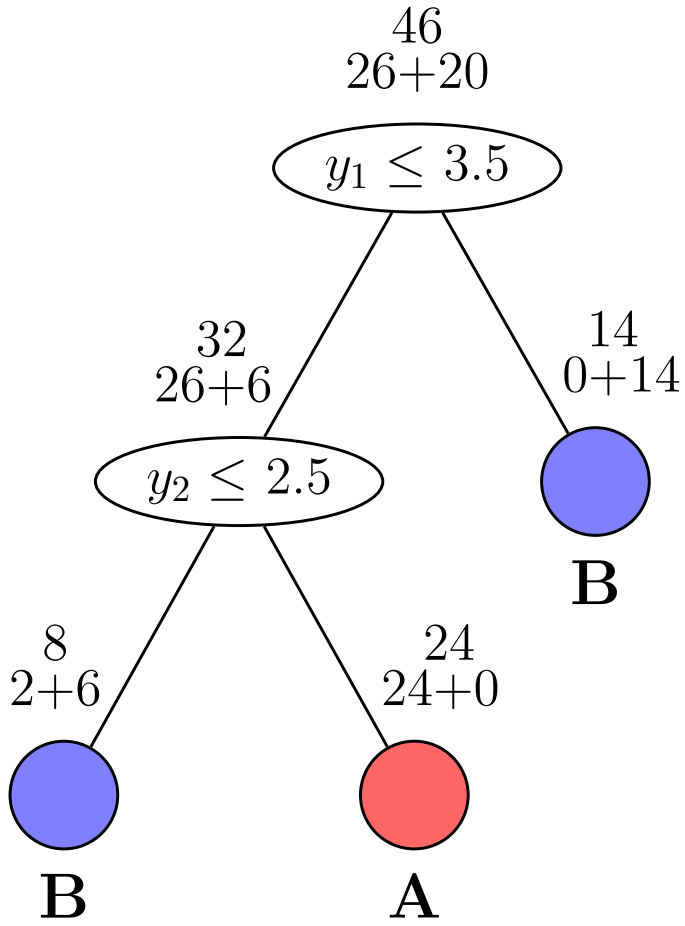


Las regiones de decisión están formadas por bloques de forma rectangular, ya que las fronteras de decisión son siempre paralelas a los ejes.

La probabilidad de error estimada por resustitución es $2/46 = 0.0435 \rightarrow 4.35\,\%$

Ejemplo: clasificación de nuevos datos





El árbol de decisión obtenido permite clasificar nuevos datos:

$(1.0, 1.0)^t$: $y_1 \leq 3.5, y_2 \leq 2.5 \rightarrow$ clase B

$(5.0, 2.5)^t$: $y_1 > 3.5 \rightarrow$ clase B

...

$(1.5, 2.7)^t$: $y_1 \leq 3.5, y_2 > 2.5 \rightarrow$ clase A

$(2.0, 5.0)^t$: $y_1 \leq 3.5, y_2 > 2.5 \rightarrow$ clase A

...

Notación

- Espacio de representación: $E \equiv \mathbb{R}^D$; $\boldsymbol{y} = (y_1, y_2 \ldots, y_D)^t \in E$
- Muestra de aprendizaje: N vectores, con su correcta clasificación: $(\boldsymbol{y}_1, c_1), \ldots, (\boldsymbol{y}_N, c_N), \quad \boldsymbol{y}_i \in E,\ c_i \in \mathbb{C} = \{1, 2, \ldots, C\},\ 1 \leq i \leq N$
- Un árbol se denota por T (*"Tree"*), un nodo por t, sus hijos izquierdo y derecho por t_L, t_R, respectivamente y el conjunto de nodos-hoja o terminales por $\tilde{T}$
- Una partición binaria (*"split"*) se denota por s y el conjunto de particiones admisibles por S

Estimación de probabilidades asociadas a los nodos de un ADC

Sean: $N, N_c, N(t), N_c(t)$, respectívamente, el número total de datos de la muestra de aprendizaje, el número de estos datos de la clase c, el número de los que están representados en el nodo t, y el número de estos últimos que son de la clase c.

Probabilidad a priori de la clase c : $\hat{P}(c) = \frac{N_c}{N}$

Probabilidad a posteriori de clase en el nodo t : $\hat{P}(c \mid t) = \frac{N_c(t)}{N(t)}$

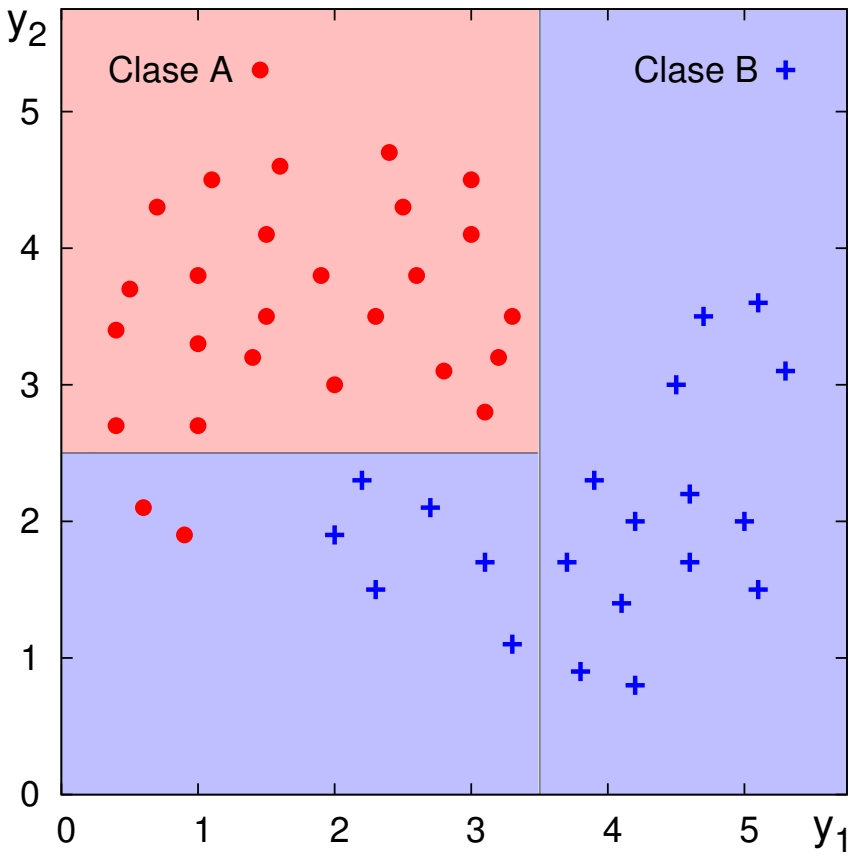
Probabilidad de un nodo *terminal,* $t \in \tilde{T}$: $\hat{P}(t) = \frac{N(t)}{N}$

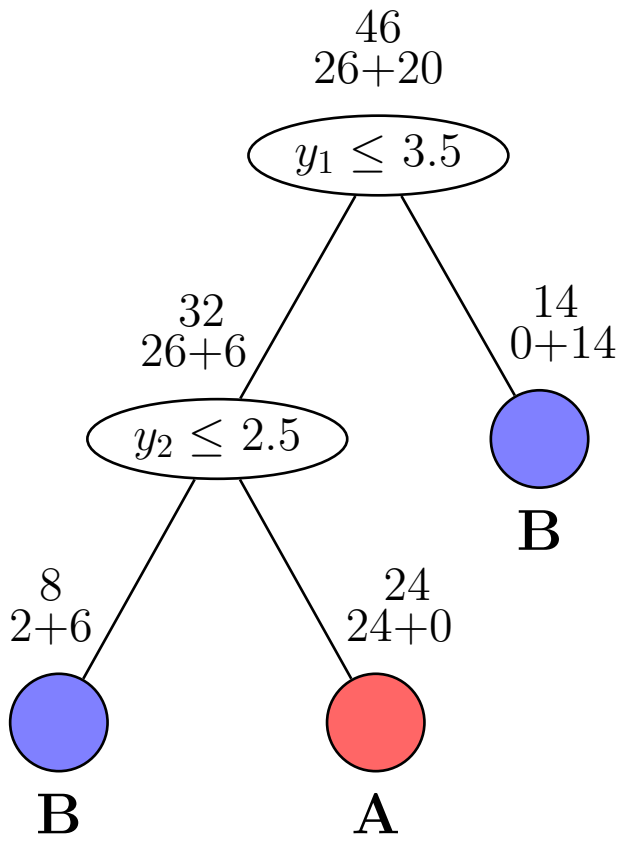
Probabilidad de decisión por el hijo izquierdo de t : $\hat{P}_t(L) = \frac{N(t_L)}{N(t)}$

Probabilidad de decisión por el hijo derecho de t : $\hat{P}_t(R) = \frac{N(t_R)}{N(t)}$

Ejercicio: calcular $\hat{P}(c)$ y $\hat{P}(c \mid t), \hat{P}(t), \hat{P}_t(L), \hat{P}_t(R)\ \forall t$, en el árbol de la página 6.

Solución al ejercicio de la página 9





Nodos:	$\hat{P}(t_i)$	$\hat{P}(A \mid t_i)$	$\hat{P}(B \mid t_i)$	$\hat{P}_{ti}(L)$	$\hat{P}_{ti}(R)$
t_1 (raiz)	–	0.565	0.435	0.696	0.304
t_2 (interno)	–	0.813	0.187	0.250	0.750
t_3 (hoja B)	0.304	0.000	1.000	–	–
t_4 (hoja B)	0.174	0.250	0.750	–	–
t_5 (hoja A)	0.522	1.000	0.000	–	–

Índice

Construcción de un ADC a partir de una muestra de aprendizaje

Elementos necesarios en el proceso de construcción de un árbol de decisión:

1. Método para hacer particiones y para seleccionar la mejor; concretamente:
   - Condiciones o “preguntas” (“splits”) admisibles para formar particiones. Sin pérdida de generalidad, serán de la forma: $\textit{¿}\boldsymbol{y} \in B\textit{?}$, $B \subseteq E$
   - Evaluación y optimización de la calidad de una partición
2. Criterio para considerar que un nodo es suficientemente “puro” (homogéneo) como para declararlo *terminal*
3. Criterio para asignar una etiqueta a un nodo *terminal*

Conjunto de preguntas admisibles para formar particiones

- Cada partición involucra a una única componente j de E, $1 \leq j \leq D$
- Las preguntas ¿$\boldsymbol{y} \in B$? son de la forma: ¿$y_j \leq r$?

  Es decir, un "split" es un par $s = (j, r)$ formado por una componente, $j \in \{1, \ldots, D\}$ y su correspondiente umbral, $r \in \mathbb{R}$.
- Como los "splits" definen hiperplanos paralelos a los ejes de E, las particiones resultantes están formadas por bloques (B) hiper-paralepipédicos (rectangulares en el caso $E = \mathbb{R}^2$),
- Como la muestra de aprendizaje es finita, solo hay un número finito de particiones posibles. Para un nodo t con $N(t)$ elementos:
  - Hay que explorar cada una de las componentes j, $1 \leq j \leq D$, de E
  - Para cada j, hay que explorar (al menos) $N(t)$ posibles valores de r

  Por tanto, para cada nodo t, hay que explorar al menos $\mathcal{O}(D\, N(t))$ "splits"

Evaluación de la calidad de una partición

- Para evaluar las particiones posibles se usa el concepto de *"impureza"*
- La impureza de un nodo t, $\mathcal{I}(t)$, se mide en función de las probabilidades estimadas de las clases en t, para lo cual existen varias aproximaciones.

  Una de las más interesantes se basa en el concepto de *entropía* (pág. 15):

$$\mathcal{I}(t) \;=\; -\sum_{c=1}^{C} \hat{P}(c \mid t) \log_2 \hat{P}(c \mid t) \;=\; -\sum_{c=1}^{C} \frac{N_c(t)}{N(t)} \log_2 \frac{N_c(t)}{N(t)} \qquad (1)$$

  Otras definiciones de $\mathcal{I}(t)$: *índice Gini* y *probabilidad de error* (ver [1,2,3])

- La calidad de una partición del nodo t mediante un *"split"* $s = (j, r)$, se mide mediante el *decremento de impureza*:

$$\Delta\mathcal{I}(j, r, t) \;\stackrel{\text{def}}{=}\; \mathcal{I}(t) - \hat{P}_t(L)\mathcal{I}(t_L) - \hat{P}_t(R)\mathcal{I}(t_R) \qquad (2)$$

- La mejor partición es aquella que prduce mayor decremento de impureza:

$$(j^\star, r^\star) \;=\; \underset{\substack{1 \le j \le D \\ -\infty < r < +\infty}}{\operatorname{argmax}} \; \Delta\mathcal{I}(j, r, t) \qquad (3)$$

Entropía

- Mide la cantidad de información asociada a una decisión $k-$aria:

$$H \;=\; -\sum_{i=1}^{k} P_i \log_2 P_i \qquad (0\,\log 0 \stackrel{\text{def}}{=} 0)$$

  - La unidad es el *bit*: información asociada a tomar una decisión *binaria* en la que las dos alternativas son equiprobables.
  - El valor mínimo es 0 y corresponde a una decisión en la que solo hay una alternativa posible.
  - El valor máximo es $+\infty$ que se da para decisiones k-arias equiprobables con $k \to \infty$:

- Ejemplos:

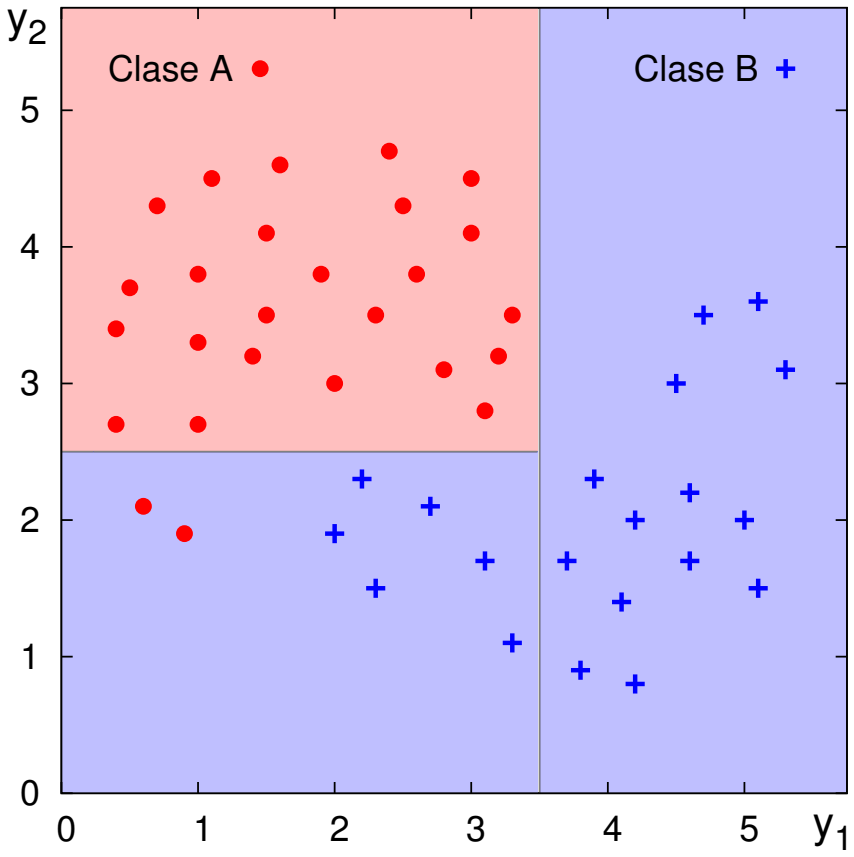
  Si $P_1 = P_2 = 1/2$, $\quad H \;=\; -\left(0.5(0-1)+0.5(0-1)\right) \;=\; 1$ bit

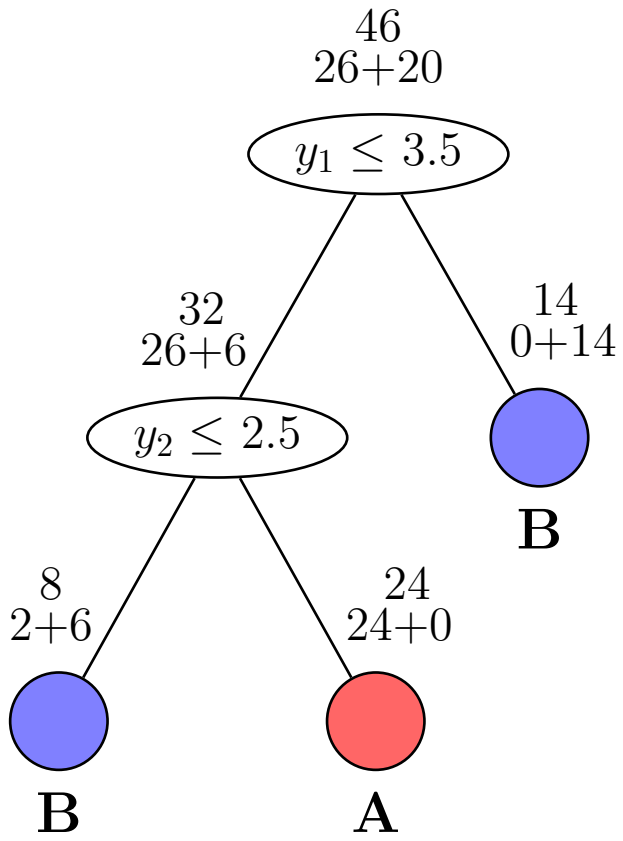
  Si $P_1 = 1,\ P_2 = 0$, $\quad H \;=\; -\,1\cdot 0 + 0 \;=\; 0$ bits

  Si $P_i = 1/k,\ 1 \le i \le k$, $\quad H \;=\; \log_2 k;\ \ H \;\to\; \infty$ si $k \to \infty$

Ejercicio: según Eq.(1), calcular $\mathcal{I}(t)\ \forall t$ en el árbol de la página 6.

Solución al ejercicio de la página 15





Nodos:	$\hat{P}(t_i)$	$\hat{P}(A \mid t_i)$	$\hat{P}(B \mid t_i)$	$\hat{P}_{ti}(L)$	$\hat{P}_{ti}(R)$	$\mathcal{I}(t_i)$
t_1 (raiz)	–	0.565	0.435	0.696	0.304	**0.988**
t_2 (interno)	–	0.813	0.187	0.250	0.750	**0.695**
t_3 (hoja B)	0.304	0.000	1.000	–	–	**0.000**
t_4 (hoja B)	0.174	0.250	0.750	–	–	**0.811**
t_5 (hoja A)	0.522	1.000	0.000	–	–	**0.000**

Criterios de suficiente "pureza" en nodos terminales

Uno de los criterios más simple es considerar que un nodo t es *terminal* si el máximo decremento de impureza posible es demasiado pequeño:

$$\max_{\substack{1\leq j\leq D \\ -\infty<r<+\infty}} \Delta\mathcal{I}(j, r, t) \; < \; \epsilon \tag{4}$$

donde ϵ es una constante pequeña a determinar empíricamente.

Otro posible criterio es exigir que los nodos terminales sean totalmente puros. Este criterio parece preferible, pero tiene el inconveniente que los árboles resultantes suelen ser grandes y con escasa capacidad de generalización.

En este caso se recurre a métodos de poda a posteriori que sacrifican pureza deshaciendo algunas de las particiones realizadas.

Asignación de etiquetas de clase a nodos terminales

Un criterio simple y eficaz: asignar a cada nodo terminal la clase de la mayoría de sus elementos:

$$c^\star(t) \; = \; \operatorname*{argmax}_{1\leq c\leq C} \hat{P}(c \mid t), \qquad \forall t \in \tilde{T} \tag{5}$$

Ejercicio (para hacer en clase)

Con respecto al ejemplo de la página 6:

- Calcular el decremento de impureza que se produce al dividir cada uno de los 2 nodos *no* terminales según Eq (2)

- Las dos particiones que se muestran en este ejemplo son solo ejemplos ilustrativos basados en pura intuición geométrica (es decir, *no* son el resultado de la optimización de ninguna expresión de impureza).

  Según la Eq. (3), analizar la segunda partición (la que en el ejemplo se resuelve con $s = (2, 2.5)$; es decir, $y_2 \leq 2.5$), y determinar si alguna de las siguientes particiones es mejor para ese nodo: $(y_1 \leq 1.95)$, $(y_2 \leq 1.8)$

- Entre los nodos *terminales* hay uno que no es *puro*. Según la Eq. (4), ¿cual sería el mínimo valor de ϵ para el que la decisión de considerar este nodo *terminal* sería correcta?

Solución al ejercicio de la página 18

- Decrementos de impureza para t_1 y t_2:

Nodos:	$\hat{P}(t_i)$	$\hat{P}(A \mid t_i)$	$\hat{P}(B \mid t_i)$	$\hat{P}_{ti}(L)$	$\hat{P}_{ti}(R)$	$\mathcal{I}(t_i)$	$\Delta\mathcal{I}(t_i)$
t_1 (raiz)	–	0.565	0.435	0.696	0.304	0.988	**0.504**
t_2 (interno)	–	0.813	0.187	0.250	0.750	0.695	**0.492**
t_3 (hoja B)	0.304	0.000	1.000	–	–	0.000	–
t_4 (hoja B)	0.174	0.250	0.750	–	–	0.811	–
t_5 (hoja A)	0.522	1.000	0.000	–	–	0.000	–

- *Splits* alternativos en t_2:

$$(y_1 \leq 1.95) : \mathcal{I}(t_L) = 0,\ \mathcal{I}(t_R) = -(11/17)\log(11/17) - (6/17)\log(6/17) = 0.937$$
$$(y_2 \leq 1.80) : \mathcal{I}(t_L) = 0,\ \mathcal{I}(t_R) = -(26/29)\log(26/29) - (3/29)\log(3/29) = 0.480$$

$$\Delta\mathcal{I}(1, 1.95, t_2) = 0.695 - 0 - (17/32) \cdot 0.937 = 0.197 < 0.492$$
$$\Delta\mathcal{I}(2, 1.80, t_2) = 0.695 - 0 - (29/32) \cdot 0.480 = 0.260 < 0.492$$

Por tanto, ninguno de estos *splits* habría sido mejor que $(y_2 \leq 2.5)$.

- La impureza de t_4 es $\mathcal{I}(t_4) = 0.811$. El máximo $\Delta\mathcal{I}(t_4)$ se conseguiría con un *split* que produjera dos nodos puros; por ejemplo, $s = (1, 1.5)$ $(y_1 \leq 1.5)$. En este caso, $\mathcal{I}(t_{4L}) = \mathcal{I}(t_{4R}) = 0$) y el máximo decremento de impureza para t_4 sería $0.811 - 0 - 0 = 0.811$. Por tanto, para que t_4 se considerara terminal ϵ debería ser mayor que 0.811. Evidentemente el ejemplo no corresponde a un resultado real de aprendizaje de ADC, ya que con este valor de ϵ tanto el nodo interno como el mismo nodo raiz se habrían considerado terminales).

Algoritmo ADC

Árbol CreaNodo(*clase* c, *componente* j, *umbral* u, *nodos* t_L, t_R) //crea un árbol (↑nodo)

Árbol ADC(*muestra* $\mathcal{Y} \equiv (\boldsymbol{y}_1, c_1), \ldots, (\boldsymbol{y}_n, c_n)$) { // aprende un Árbol de Clasificación

$(j^\star, r^\star, \delta) =$ MejorDivisión$(\mathcal{Y})$ // según Eq. (1,2,3); δ es el decremento de impureza

if $(\delta < \epsilon)$ { // si δ es demasiado pequeño – según Eq. (4)

$c =$ ClaseDominante$(\mathcal{Y})$ // según Eq. (5)

return CreaNodo$(c, -, -,$ NULL, NULL) // crea nodo terminal y le asigna la clase c

} ***else*** {

$\mathcal{Y}_L = \mathcal{Y}_R = \emptyset$ // $\emptyset$ es el conjunto vacío

$\forall (\boldsymbol{y}, c) \in \mathcal{Y}$ { // realiza la partición en función de $j^\star, r^\star$

if $(y_{j^\star} \leq r^\star)$ $\mathcal{Y}_L = \mathcal{Y}_L \cup \{(\boldsymbol{y}, c)\}$

else /*$(y_{j^\star} > r^\star)$*/ $\mathcal{Y}_R = \mathcal{Y}_R \cup \{(\boldsymbol{y}, c)\}$

}

$t_L =$ ADC$(\mathcal{Y}_L)$ // crea recursivamente el subárbol izquierdo

$t_R =$ ADC$(\mathcal{Y}_R)$ // crea recursivamente el subárbol derecho

return CreaNodo$(0, j^\star, r^\star, t_L, t_R)$

}

}

Complejidad

- *Complejidad temporal*: En un nodo t, *MejorDivisión()* ha de explorar $m = N(t)$ valores del umbral (r), para cada una de las D componentes (j). Y para cada j y r, ha de calcular las impurezas (entropías) de t_L y t_R.

  Para ello ha de contabilizar $N(t_L), N_c(t_L), N(t_L)$ y $N_c(t_L)$, lo que puede hacerse fácilmente de forma incremental si antes se ordenan los m elementos de $\mathcal{Y}$ según la componente j (lo que puede hacerse en $\Theta(m \log m)$ pasos).

  Por tanto, el coste de *MejorDivisión()* es $\Theta(D\,(m+m\log m)) = \Theta(D\,m\log m)$.

  Suponiendo que el árbol queda aproximadamente equilibrado, su profundidad es $\log N$ y cada nodo a profundidad h tiene $m = N/2^h$ elementos. Entonces, el coste temporal total es (ver [1]):

$$\mathcal{O}(\sum_{h=0}^{logN} 2^h\, D\, \frac{N}{2^h} \log \frac{N}{2^h}) \quad = \quad \mathcal{O}(D\,N\,(logN)^2\,)$$

- *Complejidad espacial*: Un árbol tiene menos de $2N$ nodos y cada uno usa una cantidad fija (y pequeña) de memoria. Por tanto, el coste espacial es $\mathcal{O}(N)$

Estimación por resustitución del error de clasificación

- Según la *teoría de la decisión estadística*, la probabilidad de error de un nodo *terminal* t, estimada por *resustitución*, es:

$$\hat{P}_t(\text{error}) \;=\; 1 - \max_{1 \le c \le C} \hat{P}(c \mid t)$$

  Y para un árbol T:

$$\hat{P}_T(\text{error}) \;=\; \sum_{t \in \tilde{T}} \hat{P}(t) \hat{P}_t(\text{error})$$

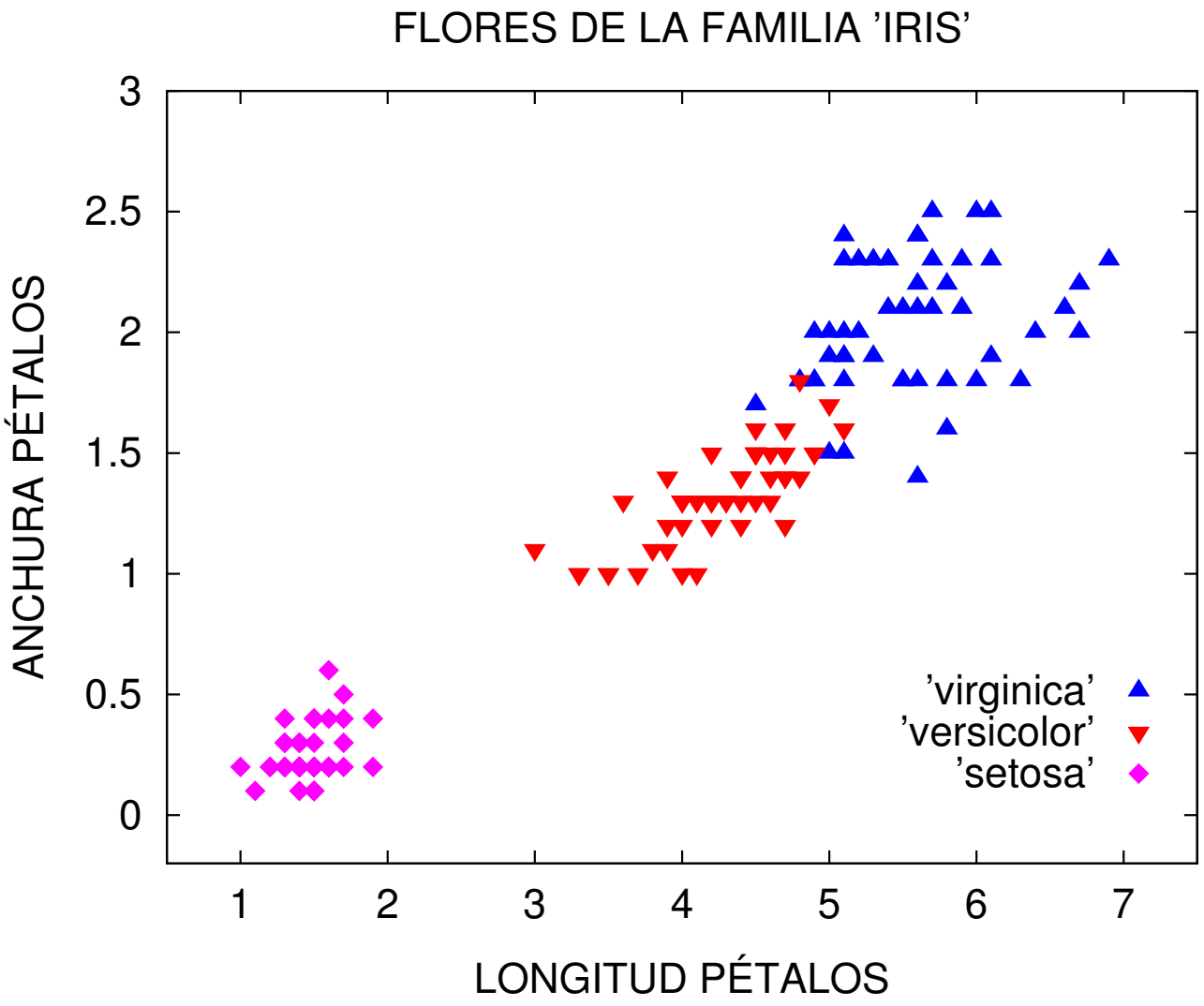
  En el ejemplo de la página 6, (3 nodos terminales, 2 totalmente puros):

$$\hat{P}_T(\text{error}) \;=\; \frac{14}{46} \cdot 0 + \frac{8}{46} \cdot \frac{2}{8} + \frac{24}{46} \cdot 0 \;=\; \frac{2}{46} \;\approx\; 0.0435 \;\rightarrow\; 4.35\,\%$$

- Si hace crecer el árbol hasta que los nodos terminales sean totalmente puros, el error estimado será nulo, ya que en este caso $\hat{P}_t(\text{error}) = 0 \; \forall t \in \tilde{T}$.

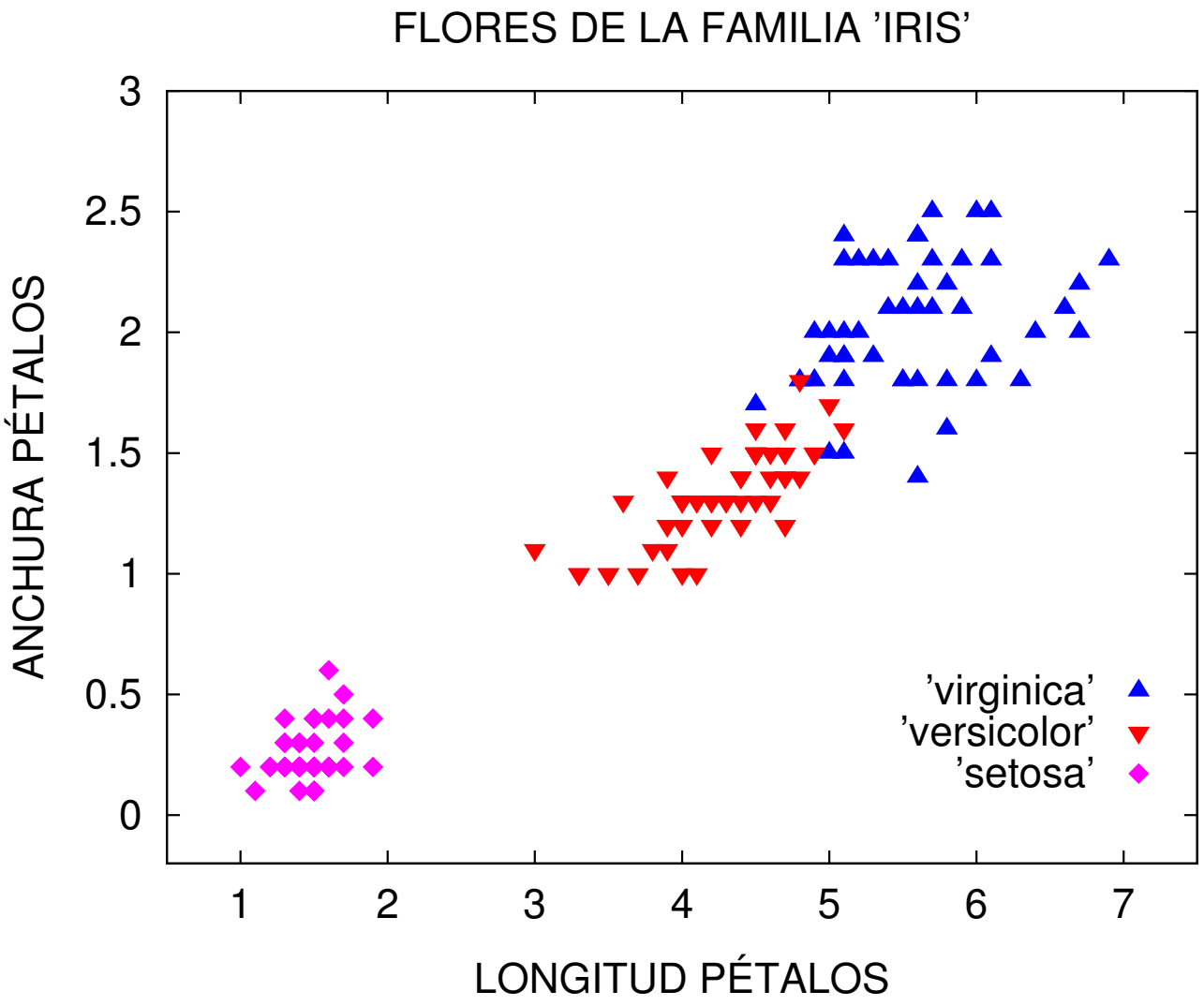
  Esto conlleva un *sobreaprendizaje* que generalmente no es desable $\Rightarrow$ esencialmente el árbol se convierte en mero almacén de la muestra de aprendizaje, sin capacidad de generalización ante nuevos datos

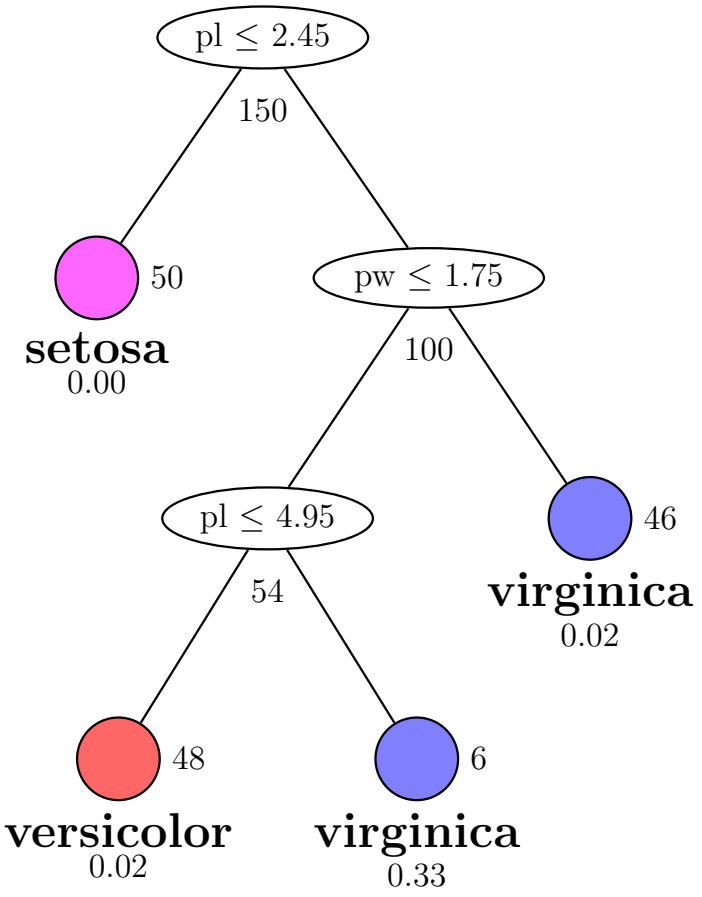
Ejemplo: aprendizaje de un ADC para clasificar flores Iris



- 3 clases,
  una fácilmente separable
- 4 dimensiones ($E = \mathbb{R}^4$)
- 150 vectores,
  50 de cada clase

Ejemplo: aprendizaje de un ADC para clasificar flores Iris

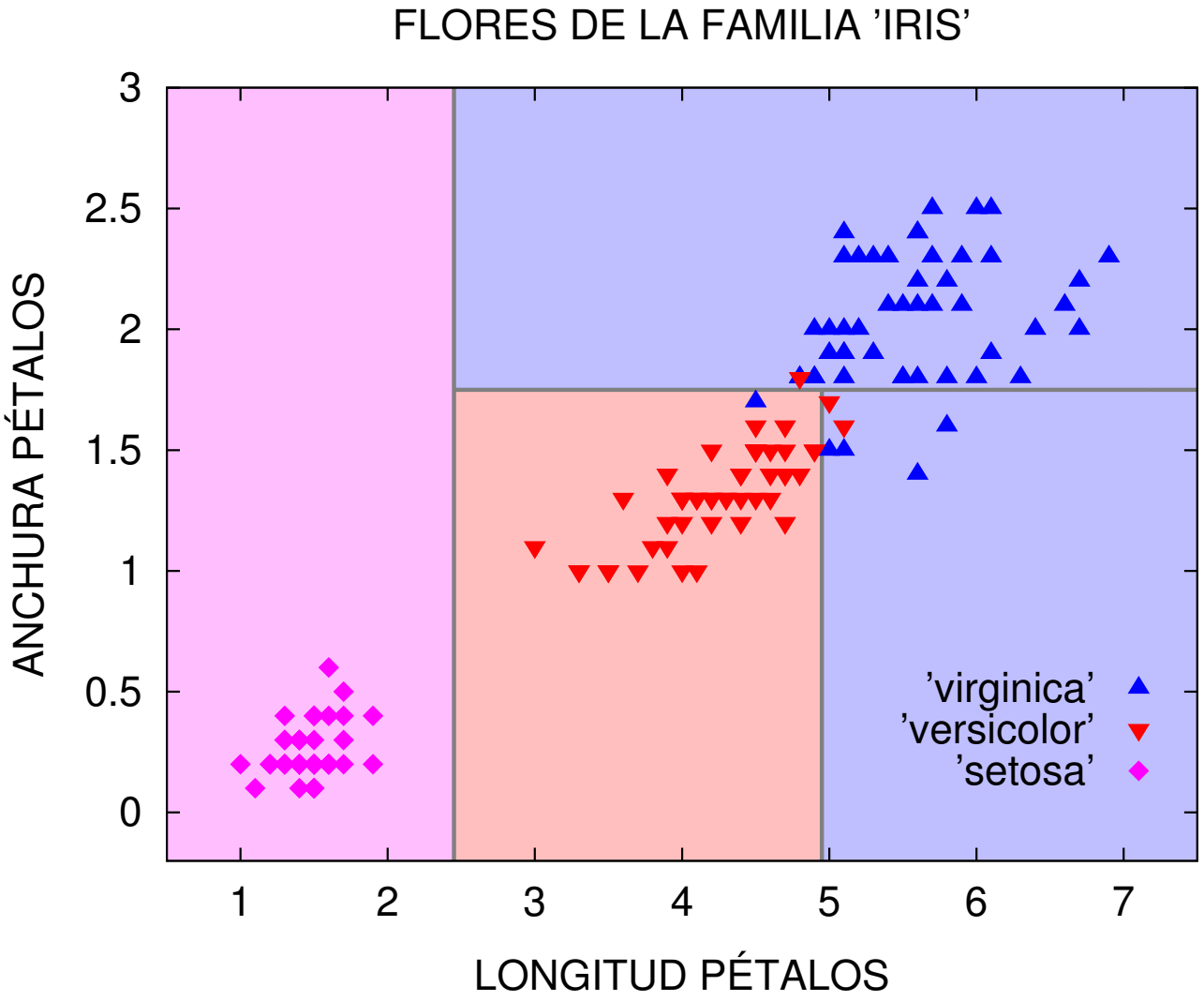


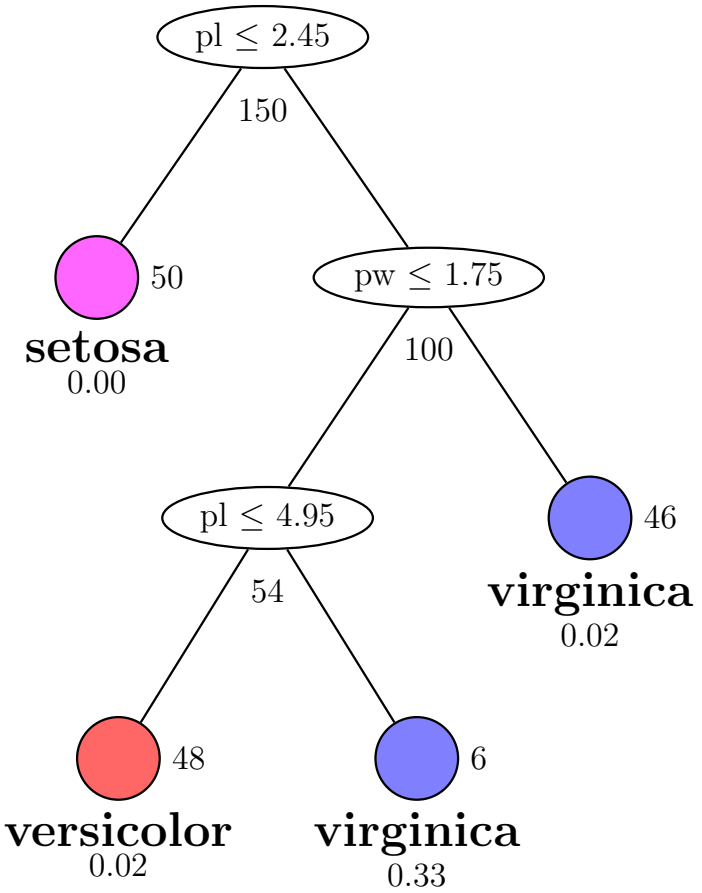


ADC aprendido:

Solo se usan 2 de las 4 dimensiones

Ejemplo: aprendizaje de un ADC para clasificar flores Iris





Fronteras y regiones de clasificación. Error estimado por resustitución:

$$(50/150)0.0 + (46/150)0.02 + (48/150)0.02 + (6/150)0.33 = 0.02 \rightarrow 2.6\,\%$$

Otros planteamientos y criterios en Árboles de Decisión

Ver detalles en [1,2,3]:

- Atributos discretos y categóricos
- Nodos terminales puros y poda explícita a posteriori
- Otras medidas de impureza: *índice Gini* y *probabilidad de error*
- Particiones no paralelas a los ejes: árboles “oblícuos” o multivariados
- Particiones no binarias
- Mejora del comportamiento en problemas multiclase: criterio de *twoing*
- Mejora de la estimación del error: validación cruzada
- Tratamiento de “valores perdidos” para algunas componentes
- Árboles de Regresión (en vez de Clasificación)

Índice

Bibliografía

[1] R.O. Duda, D.G. Stork, P.E. Hart. Pattern Classification. Wiley, 2001.

[2] A. R. Webb, K. D. Copsey. Statistical Pattern Recognition. Wiley, tercera ed., 2011.

[3] Classification and Regression Trees by L. Breiman, J.H. Friedman, R.A. Olshen y C.J. Stone. Chapman & Hall, 1984.

El material de este tema se basa principalmente en [1] y [2].