


UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO
FACULTAD DE CIENCIAS

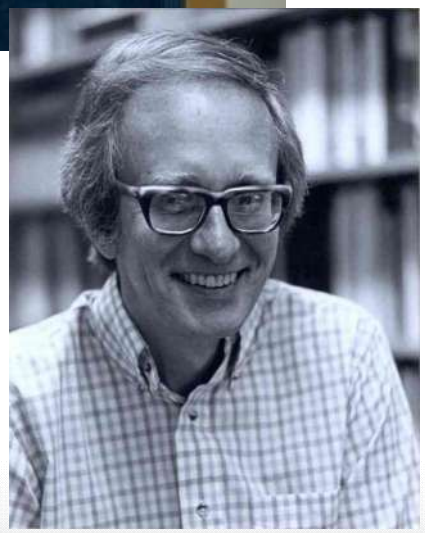
**ALMACENES Y MINERÍA DE DATOS**

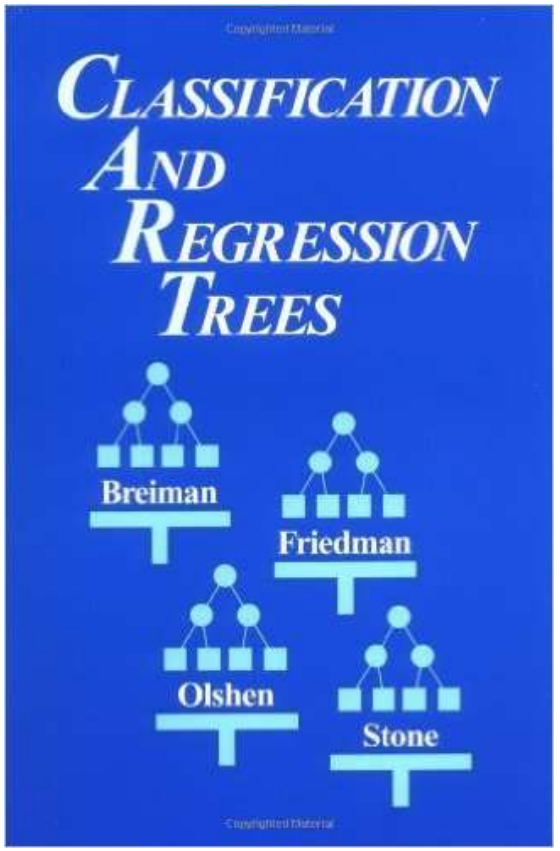
# Árboles de decisión: CART

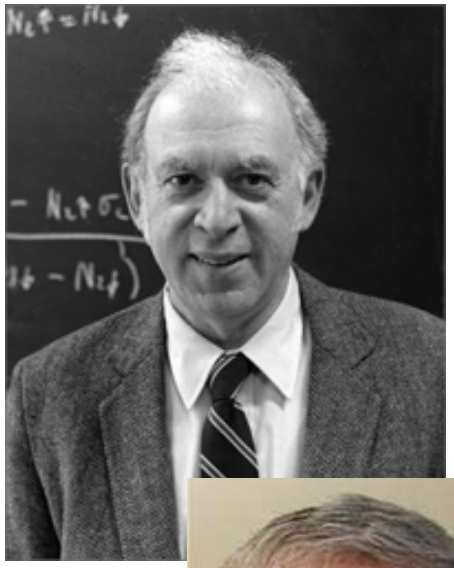
**Gerardo Avilés Rosas**
gar@ciencias.unam.mx

# Introducción

- En 1984 (***L. Breiman, J. Friedman, R.Olshen y C.Stone***) publicaron el libro **Árboles de Clasificación y Regresión** (CART), el cual describe la generación de un árbol de decisión binario.

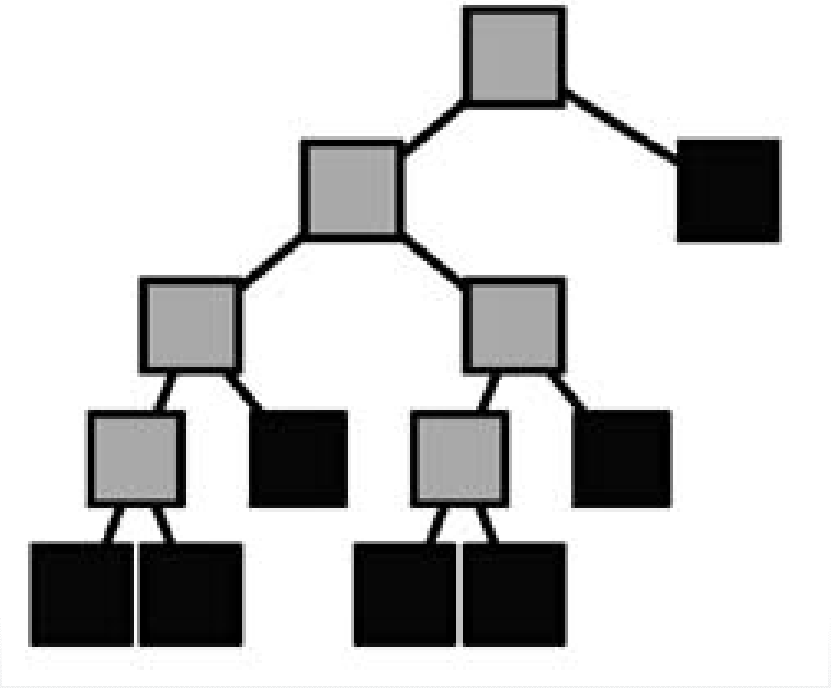




# …Introducción

- Este algoritmo se caracteriza fundamentalmente, por realizar particiones binarias utilizando una estrategia de poda basada en el criterio de costo-complejidad.
- Las particiones se realizan de modo que "la impureza" de los subconjunto hijos sea menor que la partición original.
- El objetivo es dividir la respuesta en grupos homogéneos y a la vez mantener el árbol razonablemente pequeño.



# …Introducción

- La metodología **CART** utiliza datos históricos para construir el árbol de clasificación o de regresión.
- Estos árboles pueden manipular fácilmente variables numéricas y/o categóricas.
- Entre otras ventajas está su robustez a **outliers**, la invarianza en la estructura de sus árboles de clasificación a transformaciones monótonas de las variables independientes, y sobre todo, su interpretabilidad.
- Busca minimizar el **error de resustitución** (*probabilidad de equivocarse en la clasificación de una muestra*).

# CART: GINI Index

- Esta medida es utilizada en el algoritmo **CART** y su objetivo es **medir la impureza** de **D** que puede ser una partición de datos o bien un conjunto de tuplas de entrenamiento.

$$Gini(D) = 1 - \sum_{i=1}^{m} p_i^2$$

Donde:

- $\mathbf{p_i}$ es la probabilidad de que una tupla en **D** pertenezca a una clase $\mathbf{C_i}$, se estima a partir de $|\mathbf{C_{i,D}}|/|\mathbf{D}|$

- La suma se calcula sobre **m clases**
- Esta medida considera solo **particiones binarias** para cada atributo.

# …CART: GINI Index

- Vamos a considerar el caso cuando **A** es un atributo que tiene valores discretos, teniendo **n** distintos valores **$\{a_1, a_2, \ldots, a_n\}$.**
- Para determinar la **mejor partición** binaria sobre **A**, es necesario examinar **todos los posible subconjuntos** que pueden formarse usando los valores conocidos de **A**.
- Cada subconjunto **$S_A$**, puede ser considerado como una prueba binaria sobre el atributo **A**, tomando la forma:

  **"¿A ∈ $S_A$?".**
- Si **A** tiene **n posibles valores**, tendríamos entonces **$2^n$** posibles subconjuntos, lo cual generaría en principio un subconjunto con **todos** los atributos y un subconjunto **sin ningún** atributo; los cuales se eliminan debido a que **conceptualmente** ninguno de los dos **representa una partición**.
- De esta forma tendríamos **$2^n - 2$** formas de crear particiones binarias.

# …CART: GINI Index

- Cuando se considera una partición binaria, es necesario calcular una **suma ponderada** de la impureza de cada partición resultante.
- Por ejemplo, sin una partición binaria sobre **A** divide a **D** en $\mathbf{D_1}$ y $\mathbf{D_2}$, el **GNI Index** de cada partición está dado por:

$$Gini_A(D) = \frac{|D_1|}{|D|} Gini(D_1) + \frac{|D_2|}{|D|} Gini(D_2)$$

- El cálculo se hace para cada atributo y para el caso de valores discretos, el subconjunto que proporcione el **menor GINI Index** se selecciona como atributo de partición.
- Para atributos que tienen valores continuos, cada posible punto de partición debe considerarse y se utiliza la **misma estrategia** que para la ganancia de información.

# …CART: GINI Index

- La **reducción de impureza** que se podría tener al realizar particiones binarias en atributos con valores continuos o discretos esta dada por:

$$\Delta Gini(A) = Gini(D) - Gini_A(D)$$

- De esta forma, el atributo que **maximice la reducción de impureza** se selecciona como atributo de partición.

# Ejemplo: GINI Index

Regresando al ejemplo que se analizó para el árbol C4.5:

| ID | edad | ingreso | estudiante | calificacion_credito | comprar_computadora |
|---|---|---|---|---|---|
| 1 | youth | high | no | fair | no |
| 2 | youth | high | no | excellent | no |
| 3 | middle_aged | high | no | fair | yes |
| 4 | senior | medium | no | fair | yes |
| 5 | senior | low | yes | fair | yes |
| 6 | senior | low | yes | excellent | no |
| 7 | middle_aged | low | yes | excellent | yes |
| 8 | youth | medium | no | fair | no |
| 9 | youth | low | yes | fair | yes |
| 10 | senior | medium | yes | fair | yes |
| 11 | youth | medium | yes | excellent | yes |
| 12 | middle_aged | medium | no | excellent | yes |
| 13 | middle_aged | high | yes | fair | yes |
| 14 | senior | medium | no | excellent | no |

# …Ejemplo: GINI Index

- El calculo de impureza de D es:

$$Gini(D) = 1 - \left(\frac{9}{14}\right)^2 - \left(\frac{5}{14}\right)^2 = 0.459$$

- Para encontrar el criterio de partición en D se necesita calcular el **GINI Index** de cada atributo:
  - Si tomamos el atributo **ingreso**, es necesario considerar todos sus posibles subconjuntos de partición:
    - ✓ **{low,medium,high}**
    - ✓ **{low,medium}**
    - ✓ **{low,high}**
    - ✓ **{medium,high}**
    - ✓ **{low}**
    - ✓ **{medium}**
    - ✓ **{high}**
    - ✓ **{ }**

# …Ejemplo: GINI Index

- Si consideramos el subconjunto **{low,medium}**, resulta que se tienen **10 tuplas** en $D_1$ que satisfacen la condición:

**¿ingreso ∈ {low,medium}?**

| ID | edad | ingreso | estudiante | calificacion_credito | comprar_computadora |
|---|---|---|---|---|---|
| 1 | youth | high | no | fair | no |
| 2 | youth | high | no | excellent | no |
| 3 | middle_aged | high | no | fair | yes |
| 4 | senior | medium | no | fair | yes |
| 5 | senior | low | yes | fair | yes |
| 6 | senior | low | yes | excellent | no |
| 7 | middle_aged | low | yes | excellent | yes |
| 8 | youth | medium | no | fair | no |
| 9 | youth | low | yes | fair | yes |
| 10 | senior | medium | yes | fair | yes |
| 11 | youth | medium | yes | excellent | yes |
| 12 | middle_aged | medium | no | excellent | yes |
| 13 | middle_aged | high | yes | fair | yes |
| 14 | senior | medium | no | excellent | no |

- Las 4 tuplas restantes se asignan a la partición $D_2$.

# CART: GINI Index

- De esta forma:

$$Gini_{ingreso \in \{low,medium\}}(D) = \frac{10}{14} Gini(D_1) + \frac{4}{14} Gini(D_2)$$

- La distribución entre las personas que sí comprarían y las que no es:

| ID | edad | ingreso | estudiante | calificacion_credito | comprar_computadora |
|---|---|---|---|---|---|
| 1 | youth | high | no | fair | no |
| 2 | youth | high | no | excellent | no |
| 3 | middle_aged | high | no | fair | yes |
| 4 | senior | medium | no | fair | yes |
| 5 | senior | low | yes | fair | yes |
| 6 | senior | low | yes | excellent | no |
| 7 | middle_aged | low | yes | excellent | yes |
| 8 | youth | medium | no | fair | no |
| 9 | youth | low | yes | fair | yes |
| 10 | senior | medium | yes | fair | yes |
| 11 | youth | medium | yes | excellent | yes |
| 12 | middle_aged | medium | no | excellent | yes |
| 13 | middle_aged | high | yes | fair | yes |
| 14 | senior | medium | no | excellent | no |

# CART: GINI Index

- Entonces:

$$Gini_{ingreso \in \{low,medium\}}(D) = \frac{10}{14}\left(1-\left(\frac{7}{10}\right)^2-\left(\frac{3}{10}\right)^2\right)+\frac{4}{14}\left(1-\left(\frac{2}{4}\right)^2-\left(\frac{2}{4}\right)^2\right)$$

$$Gini_{ingreso \in \{low,medium\}}(D) = 0.443$$

- Por otro lado, es fácil notar que:

$$Gini_{ingreso \in \{low,medium\}}(D) = Gini_{ingreso \in \{high\}}(D)$$

# CART: GINI Index

$$Gini_{ingreso \in \{low,high\}}(D) =$$

$$\frac{8}{14}\left(1-\left(\frac{5}{8}\right)^2-\left(\frac{3}{8}\right)^2\right)+\frac{6}{14}\left(1-\left(\frac{4}{6}\right)^2-\left(\frac{2}{6}\right)^2\right)$$

$$Gini_{ingreso \in \{low,high\}}(D) = 0.458 = Gini_{ingreso \in \{medium\}}(D)$$

$$Gini_{ingreso \in \{medium,high\}}(D) =$$

$$\frac{10}{14}\left(1-\left(\frac{6}{10}\right)^2-\left(\frac{4}{10}\right)^2\right)+\frac{4}{14}\left(1-\left(\frac{3}{4}\right)^2-\left(\frac{1}{4}\right)^2\right)$$

$$Gini_{ingreso \in \{medium,high\}}(D) = Gini_{ingreso \in \{low\}}(D) = 0.450$$

- Por lo tanto, la mejor partición binaria para el atributo **ingreso** sería **{low,medium} (o {high})**

# CART: GINI Index

Realizando las operaciones para los demás atributos:

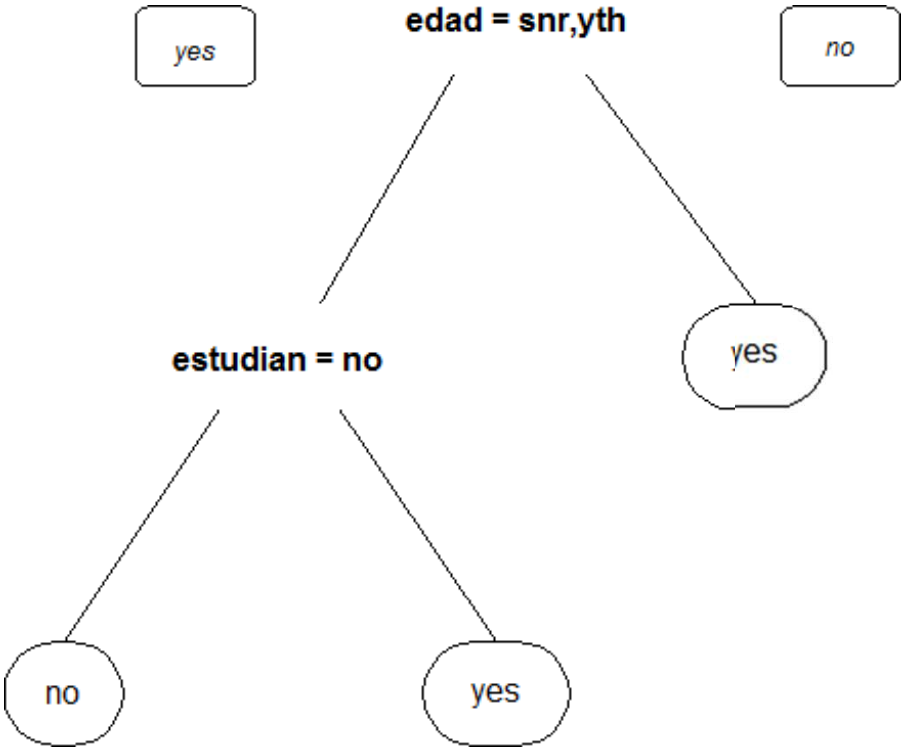
| Atributo | Combinación | gini | giniA | delta |
|---|---|---|---|---|
| Ingreso | {low,medium} | 0.459 | 0.443 | 0.016 |
| | {low,high} | 0.459 | 0.458 | 0.001 |
| | {medium,high} | 0.459 | 0.450 | 0.009 |
| | {low} | 0.459 | 0.450 | 0.009 |
| | {medium} | 0.459 | 0.458 | 0.001 |
| | {high} | 0.459 | 0.443 | 0.016 |

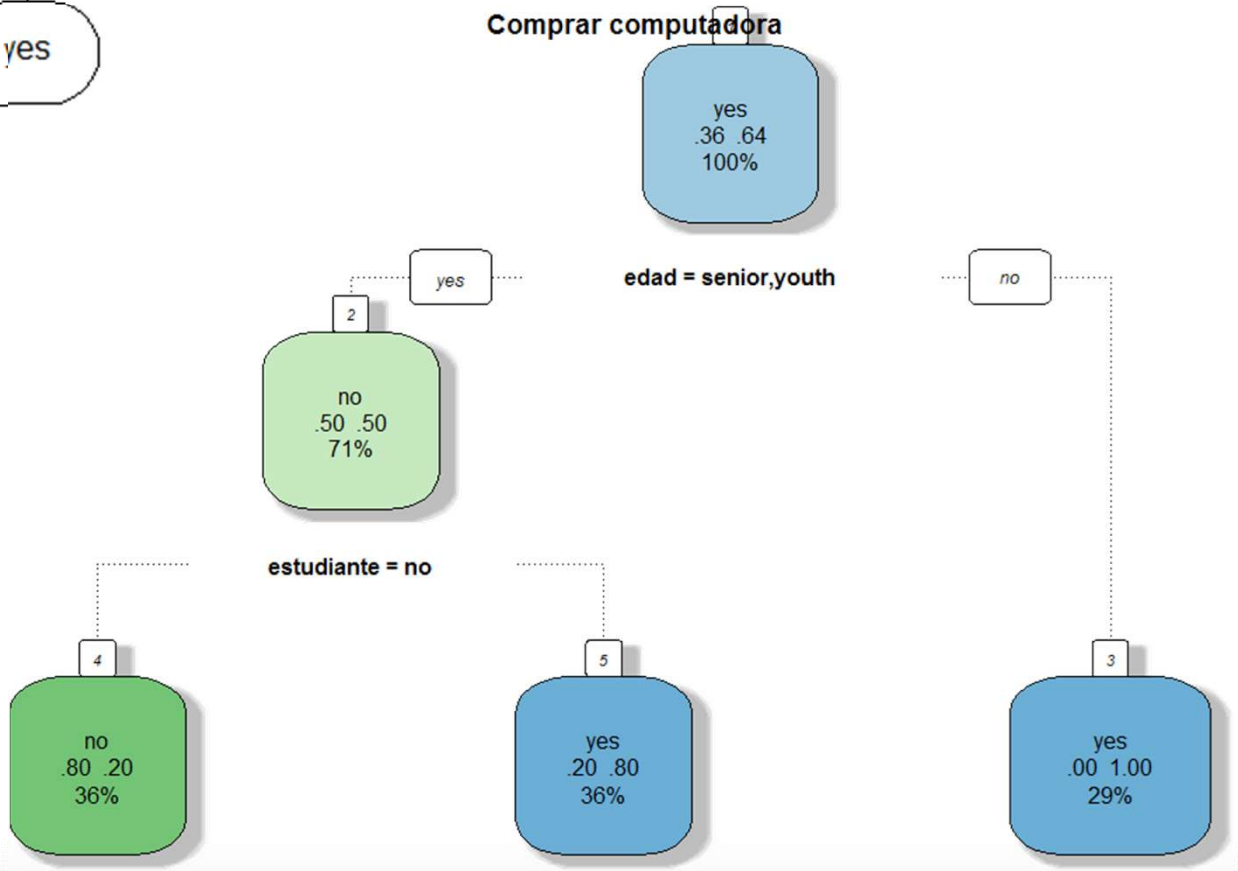
| Atributo | Combinación | gini | giniA | delta |
|---|---|---|---|---|
| Edad | {youth,middle} | 0.459 | 0.457 | 0.002 |
| | {youth,senior} | 0.459 | 0.357 | 0.102 |
| | {middle,senior} | 0.459 | 0.394 | 0.066 |
| | {youth} | 0.459 | 0.394 | 0.066 |
| | {middle} | 0.459 | 0.357 | 0.102 |
| | {senior} | 0.459 | 0.457 | 0.002 |

| Atributo | Combinación | gini | giniA | delta |
|---|---|---|---|---|
| Califi_cred | {fair} | 0.459 | 0.429 | 0.031 |
| | {excellent} | 0.459 | 0.429 | 0.031 |

| Atributo | Combinación | gini | giniA | delta |
|---|---|---|---|---|
| estudiante | {si} | 0.459 | 0.367 | 0.092 |
| | {no} | 0.459 | 0.367 | 0.092 |

# CART: GINI Index



# Poda de árboles

- Al construir árboles de decisión, muchas de las ramas podrían reflejar **anomalías** debidas a la presencia de ***ruido u outliers*** en los datos de entrenamiento.
- La **poda de árboles** es una metodología que permite enfrentar el problema de **sobreajuste** de los datos:
  - Estos métodos típicamente utilizan **medidas estadísticas** para remover las ramas menos fiables.
  - Los **árboles podados** tienden a ser **más pequeños**, **menos complejos** → ***más fáciles de comprender***.
  - Suelen ser **más rápidos** y mejores para hacer clasificaciones independientemente de los datos de prueba.
- Existen dos enfoques para podar árboles de decisión: **pre-poda** o **post-poda**

# Pre-poda

- En este enfoque, el árbol se poda **deteniendo su construcción** desde el inicio (*p.e. decidir no particionar más el subconjunto de tuplas de entrenamiento en un nodo dado*).
- Al detener la construcción, el nodo se convierte en hoja (*la cual puede contener las clase que con más frecuencia se presenta entre el subconjunto de tuplas o bien una distribución de probabilidad de esas tuplas*).
- Los algoritmos de **pre-poda** no realizan literalmente **"poda"** porque nunca podan las ramas existentes de un árbol de decisión:

  ***"podar"*** *significa suprimir el crecimiento de una rama si no se espera una estructura adicional para aumentar la precisión.*

- Este enfoque es referible debido a **los efectos de interacción**, ya que estos efectos son visibles en el árbol completamente crecido (efecto horizonte).

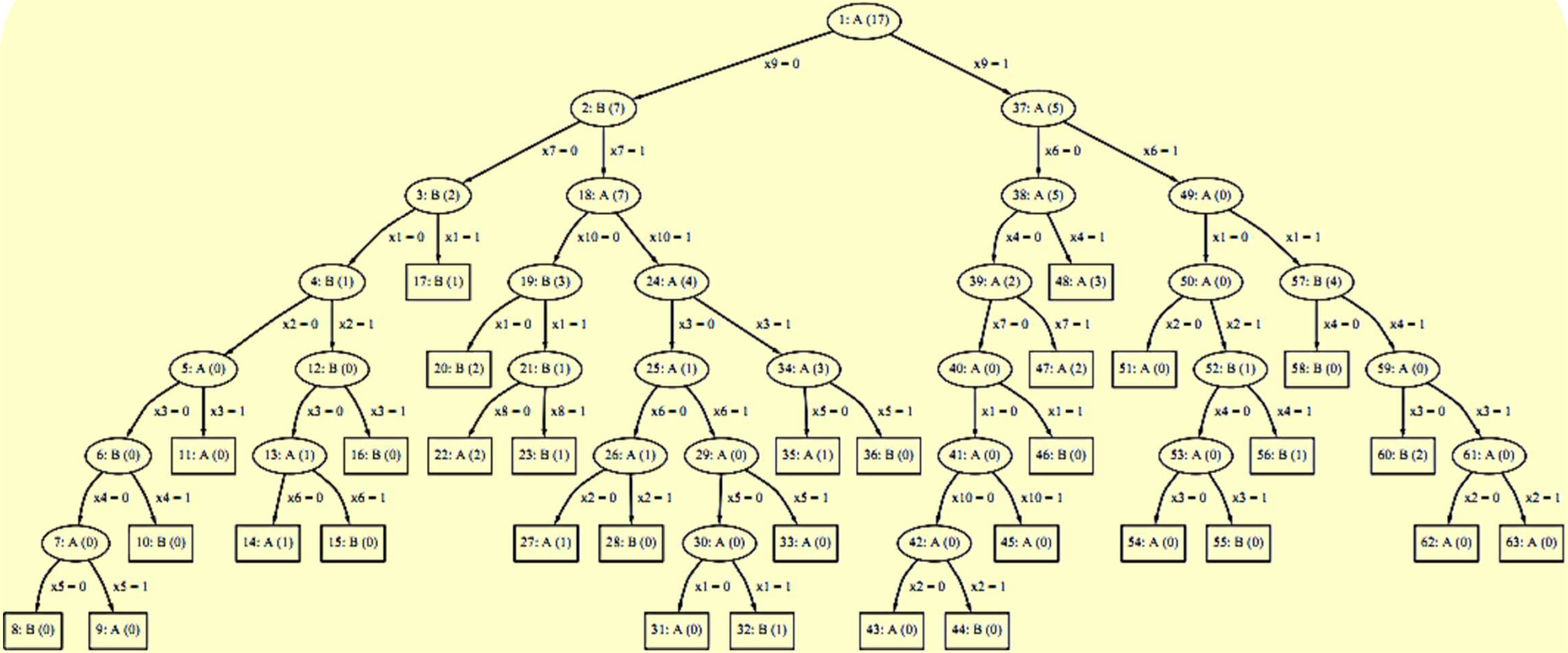
# Pre-poda

- Medidas como la ***significancia estadística, ganancia de información, GINI index***, se utilizan para asegurar la correctud de una partición.
- Si una partición de tuplas en un nodo pudiera resultar en una partición que cae por debajo de un umbral especificado previamente, entonces las particiones adicionales se detienen:
- Es difícil sin embargo, elegir umbrales adecuados:
  - **Un umbral alto podría resultar en un árbol simplificado.**
  - **Umbrales bajos podrían tener poca simplificación**.

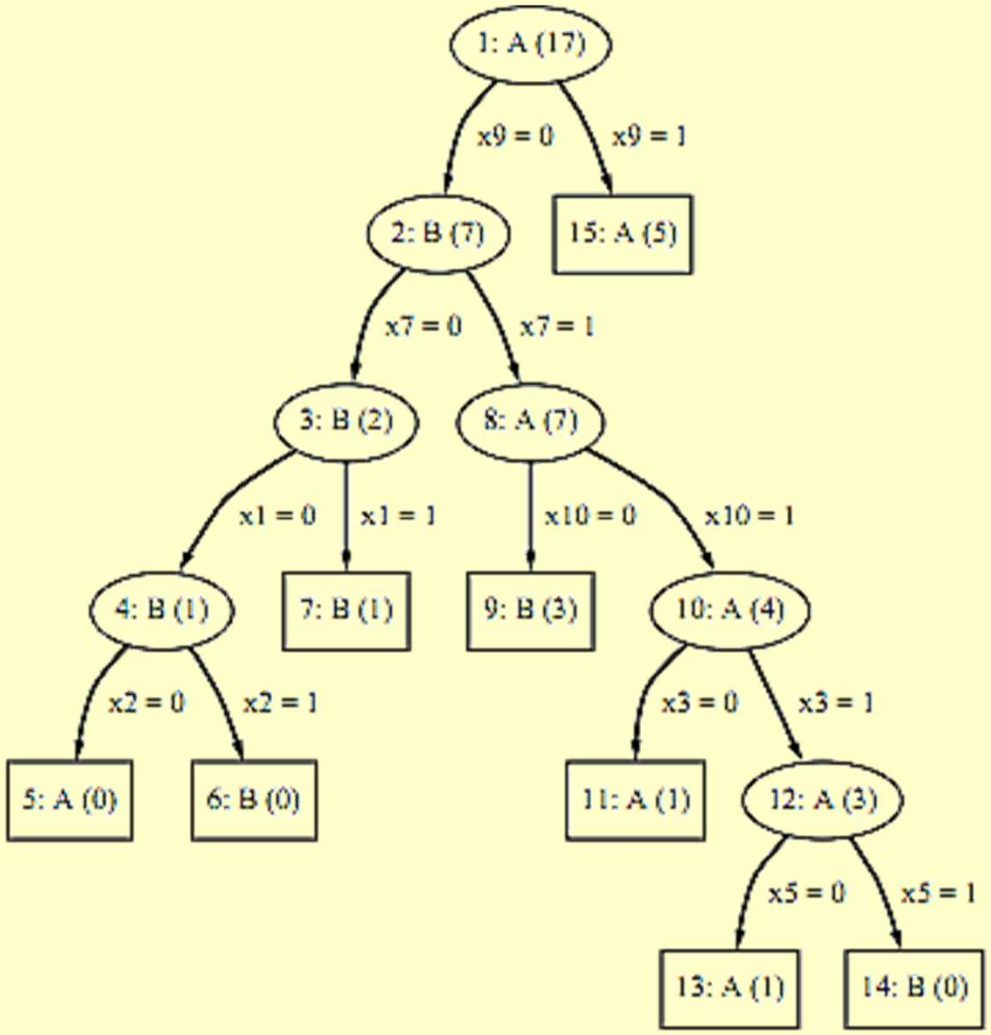
# Post-poda

- Es el enfoque que se utiliza con mayor frecuencia.
- Su objetivo es remover sub-árboles de un árbol que ha crecido mucho:
  - Permite que los datos se **sobreajusten** y después se poda reemplazando sub-árboles por una hoja.
  - Para podar un sub-árbol en un nodo dado, se retiran todas sus ramas y se sustituye por un nodo hoja.
  - La hoja se etiqueta con la clase que con más frecuencia se presentó entre las clases del sub-árbol que fue reemplazado.
  - Se poda solo si el árbol podado resultante mejora o iguala el rendimiento del árbol original sobre el conjunto de prueba.
- El proceso es iterativo, escogiendo siempre el nodo a podar que mejore la precisión en el conjunto de prueba hasta que ya no convenga (momento en que la precisión disminuye).

# Post-poda

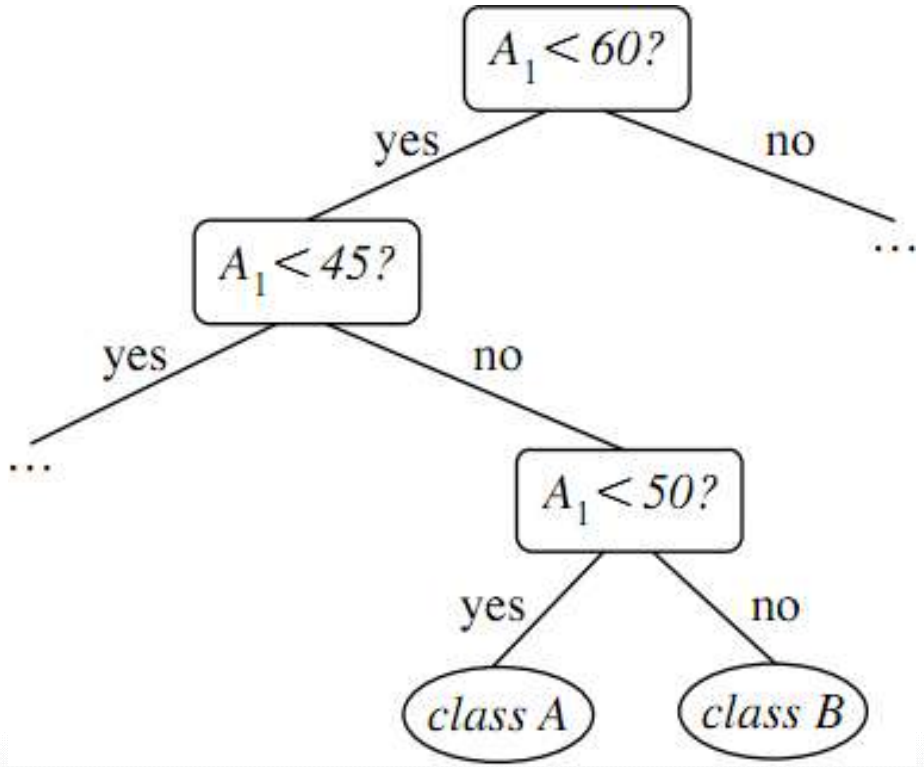

1: A (17)
x9 = 0
x9 = 1
2: B (7)
37: A (5)
x7 = 0
x7 = 1
x6 = 0
x6 = 1
3: B (2)
18: A (7)
38: A (5)
49: A (0)
x1 = 0
x1 = 1
x10 = 0
x10 = 1
x4 = 0
x4 = 1
x1 = 0
x1 = 1
4: B (1)
17: B (1)
19: B (3)
24: A (4)
39: A (2)
48: A (3)
50: A (0)
57: B (4)
x2 = 0
x2 = 1
x1 = 0
x1 = 1
x3 = 0
x3 = 1
x7 = 0
x7 = 1
x2 = 0
x2 = 1
x4 = 0
x4 = 1
5: A (0)
12: B (0)
20: B (2)
21: B (1)
25: A (1)
34: A (3)
40: A (0)
47: A (2)
51: A (0)
52: B (1)
58: B (0)
59: A (0)
x3 = 0
x3 = 1
x3 = 0
x3 = 1
x8 = 0
x8 = 1
x6 = 0
x6 = 1
x5 = 0
x5 = 1
x1 = 0
x1 = 1
x4 = 0
x4 = 1
x3 = 0
x3 = 1
6: B (0)
11: A (0)
13: A (1)
16: B (0)
22: A (2)
23: B (1)
26: A (1)
29: A (0)
35: A (1)
36: B (0)
41: A (0)
46: B (0)
53: A (0)
56: B (1)
60: B (2)
61: A (0)
x4 = 0
x4 = 1
x6 = 0
x6 = 1
x2 = 0
x2 = 1
x5 = 0
x5 = 1
x10 = 0
x10 = 1
x3 = 0
x3 = 1
x2 = 0
x2 = 1
7: A (0)
10: B (0)
14: A (1)
15: B (0)
27: A (1)
28: B (0)
30: A (0)
33: A (0)
42: A (0)
45: A (0)
54: A (0)
55: B (0)
62: A (0)
63: A (0)
x5 = 0
x5 = 1
x1 = 0
x1 = 1
x2 = 0
x2 = 1
8: B (0)
9: A (0)
31: A (0)
32: B (1)
43: A (0)
44: B (0)

# Post-poda


1: A (17)
x9 = 0
x9 = 1
2: B (7)
15: A (5)
x7 = 0
x7 = 1
3: B (2)
8: A (7)
x1 = 0
x1 = 1
x10 = 0
x10 = 1
4: B (1)
7: B (1)
9: B (3)
10: A (4)
x2 = 0
x2 = 1
x3 = 0
x3 = 1
5: A (0)
6: B (0)
11: A (1)
12: A (3)
x5 = 0
x5 = 1
13: A (1)
14: B (0)

# Repetición y duplicación

- Aunque los árboles podados tienden a ser más compactos que sus contrapartes no podadas, éstos todavía pueden ser bastante grandes y complejos.
- Los árboles de decisión pueden sufrir de efectos de repetición y la duplicación.



# Repetición y duplicación