UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO
FACULTAD DE CIENCIAS
DIPLOMADO EN MINERÍA DE DATOS

Módulo 5. *Minería de Datos*

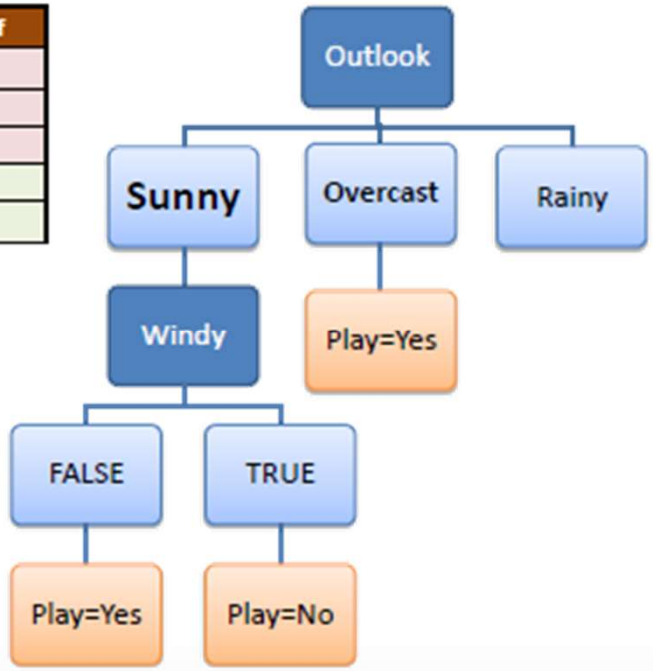
Árboles de decisión: ID3, C4.5 y CART

Gerardo Avilés Rosas
gar@ciencias.unam.mx

Introducción

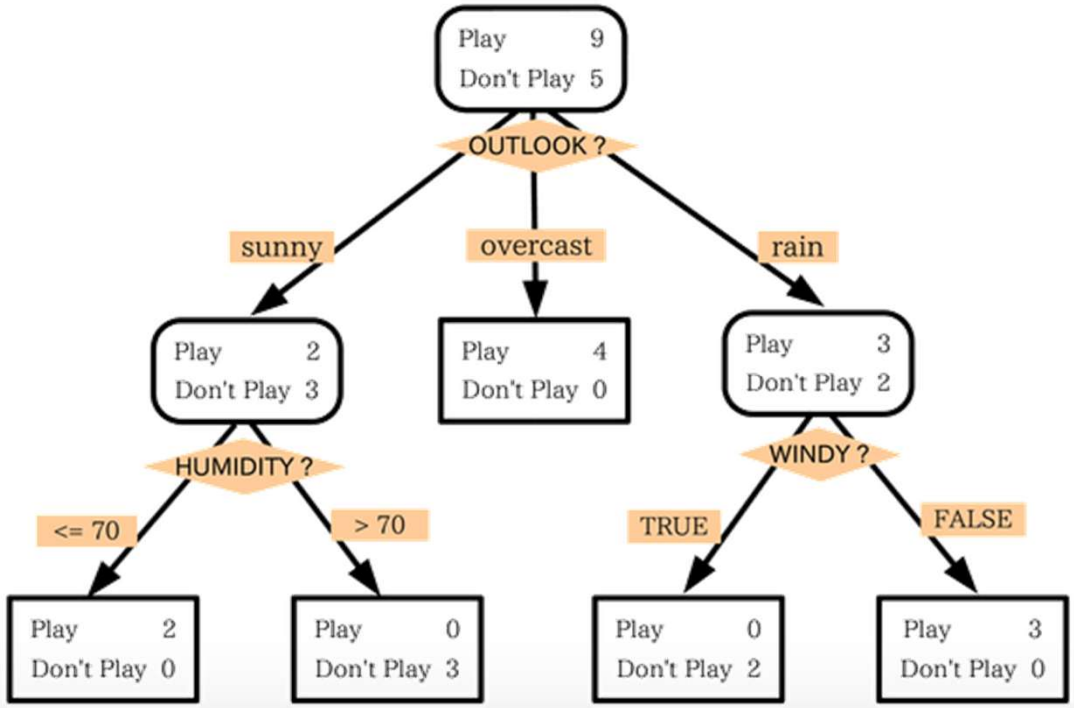
- Los **árboles de decisión** son una de las opciones **más populares** para aprender sobre características basadas en ejemplos y han sido objeto de modificaciones para hacer frente a consideraciones lingüísticas, requisitos de memoria y/o de eficiencia.
- Este esquema de clasificación genera un **árbol** y **un conjunto de reglas**, que representa el modelo para predecir etiquetas de clase, a partir de un conjunto de datos conocido.

Temp.	Humidity	Windy	Play Golf
Mild	High	FALSE	Yes
Cool	Normal	FALSE	Yes
Mild	Normal	FALSE	Yes
Cool	Normal	TRUE	No
Mild	High	TRUE	No



Inducción de Árboles de Decisión

- Se trata de una técnica que permite que los **árboles de decisión aprendan** de un conjunto de **tuplas de entrenamiento** con **etiquetas de clase**.
- Genera un "*diagrama de flujo*" con la estructura de un árbol. Cada cada ***nodo interno*** denota una **prueba** sobre un atributo y ***cada rama*** representa el **resultado** de dicha prueba.
- Cada ***nodo hoja*** almacena una **etiqueta de clase**.

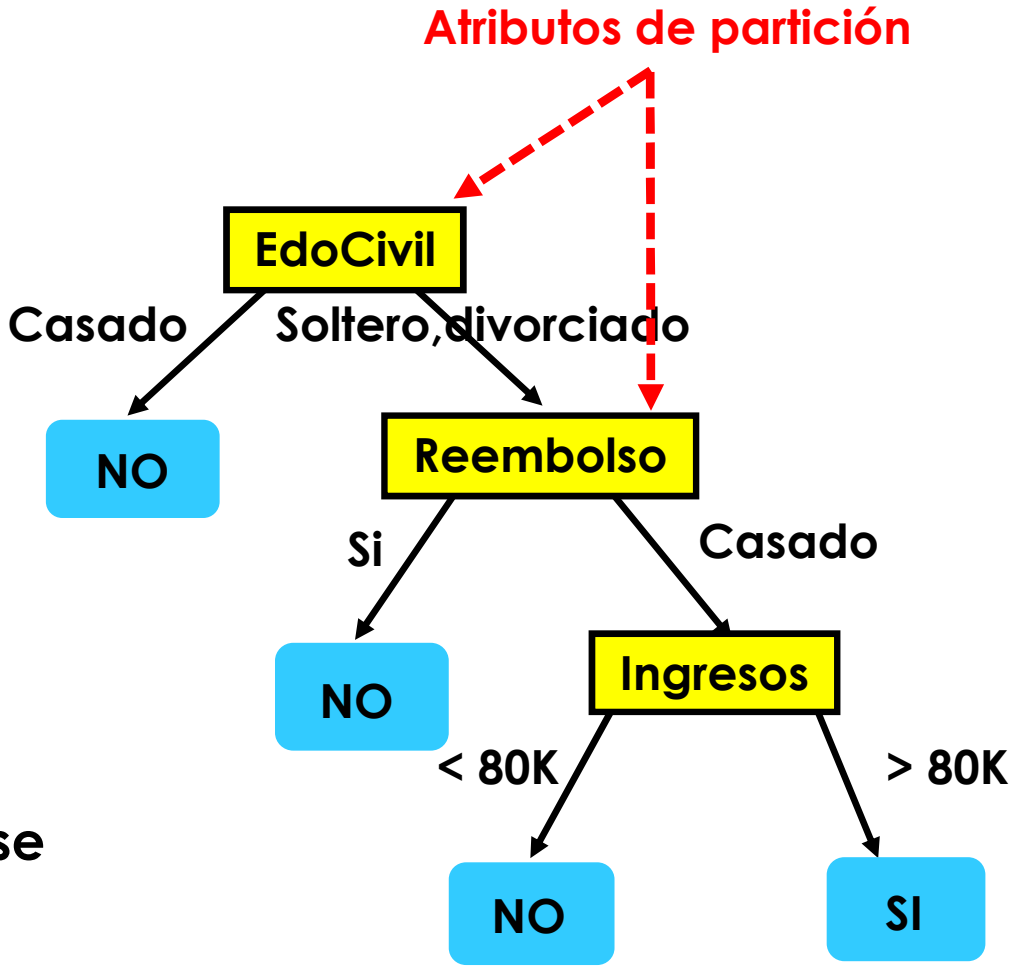


…Inducción de Árboles de Decisión

id	Reembolso	Estado civil	Ingresos	Fraude
	categórica	categórica	continua	clase
1	Si	Soltero	125K	No
2	No	Casado	100K	No
3	No	Soltero	70K	No
4	Si	Casado	120K	No
5	No	Divorciado	95K	Si
6	No	Casado	60K	No
7	Si	Divorciado	220K	No
8	No	Soltero	85K	Si
9	No	Casado	75K	No
10	No	Soltero	90K	Si

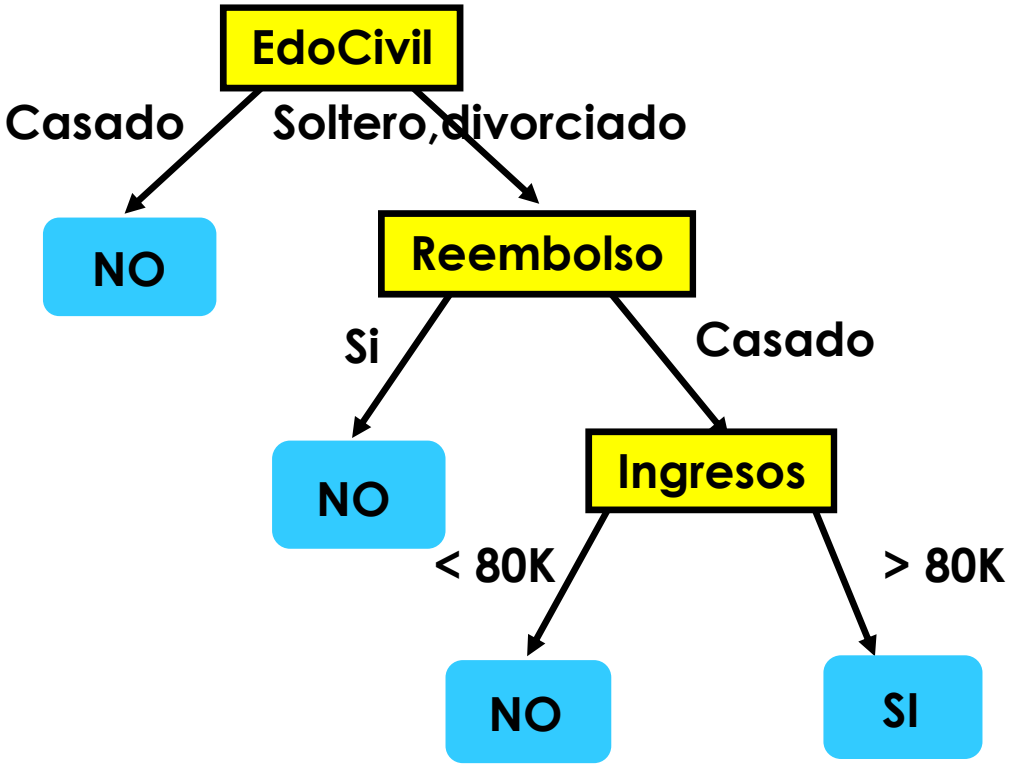
Tuplas de entrenamiento

¿Podría haber más de un árbol que se ajuste a los mismos datos?

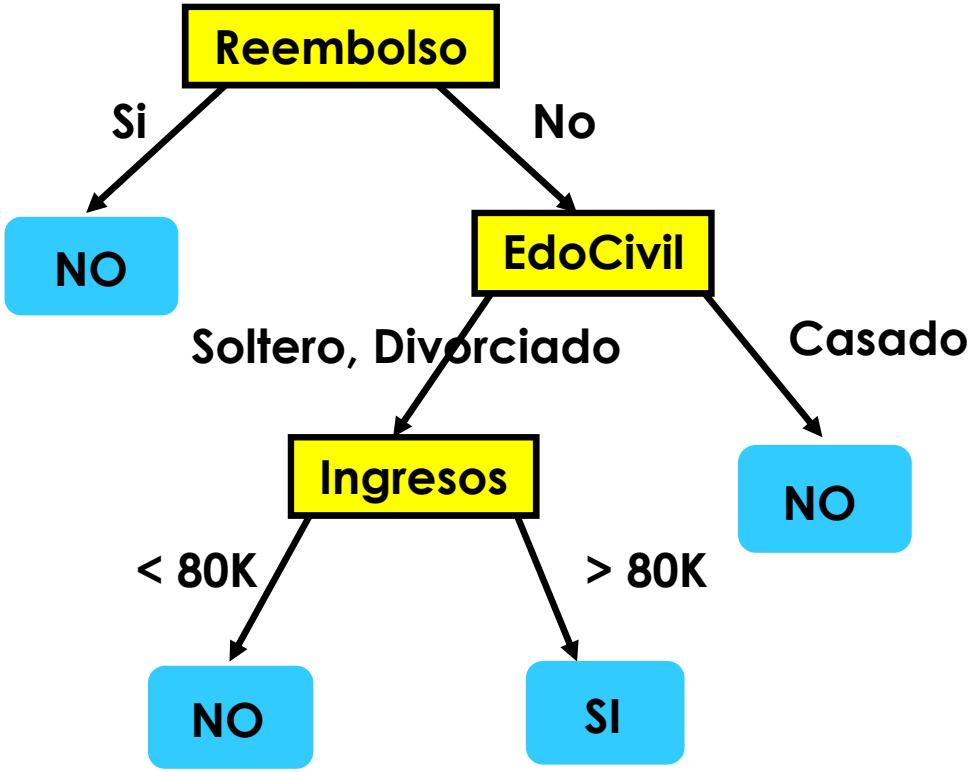


Modelo: Árbol de decisión

…Inducción de Árboles de Decisión



Modelo 1



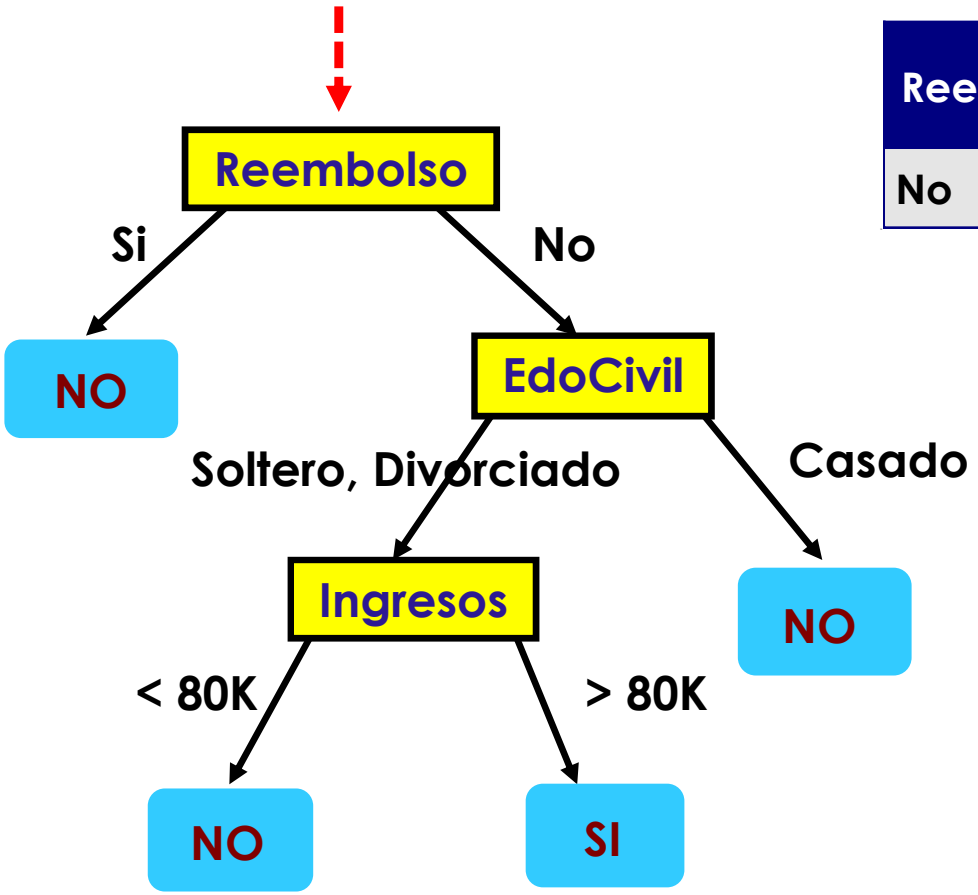
Modelo 2

Objetivo: Buscar el "mejor árbol"

…Inducción de Árboles de Decisión

La forma en que se hace la clasificación es la siguiente:

Comienza con la raíz del árbol

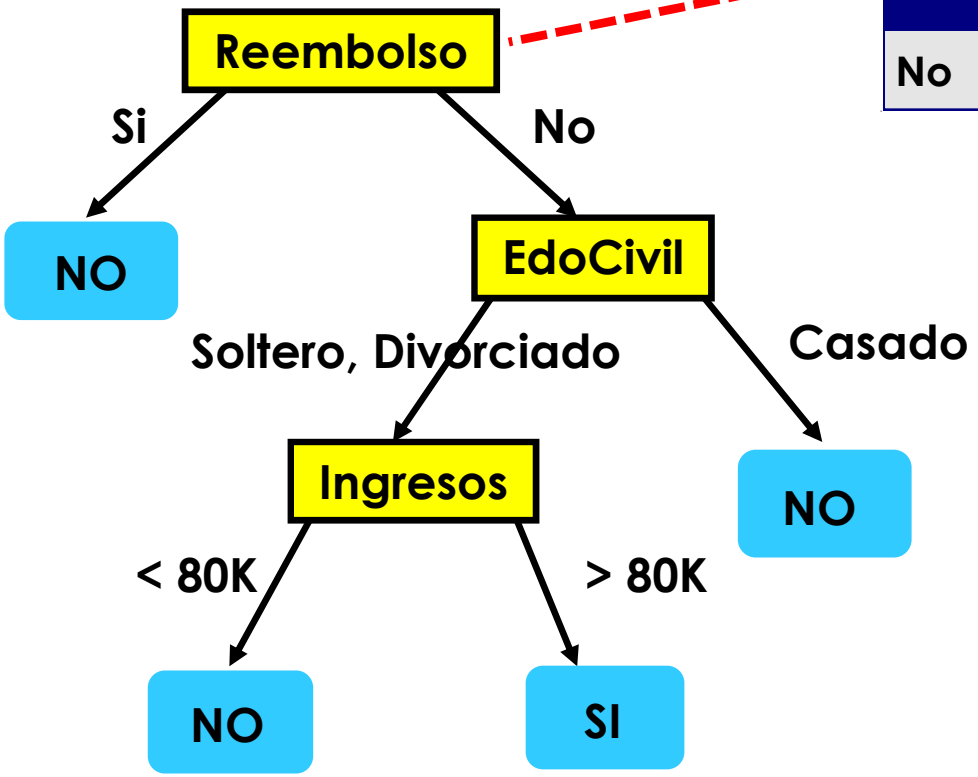


Datos de prueba

Reembolso	Estado Civil	Ingresos	Fraude
No	Casado	80K	?

…Inducción de Árboles de Decisión

La forma en que se hace la clasificación es la siguiente:

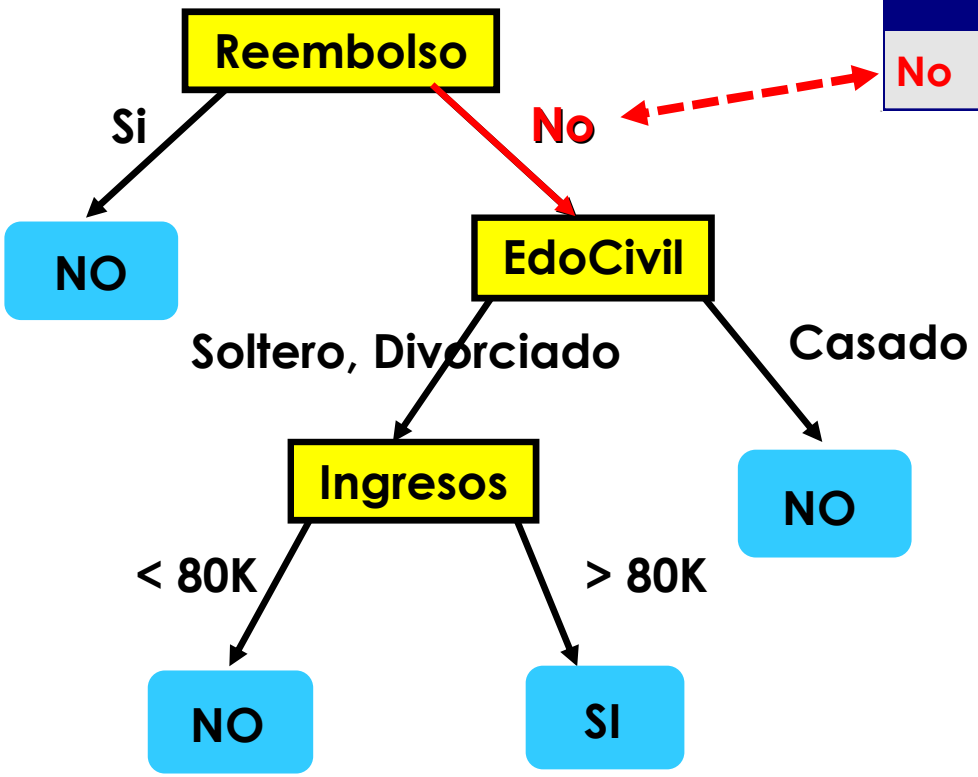


Datos de prueba

Reembolso	Estado Civil	Ingresos	Fraude
No	Casado	80K	?

...Inducción de Árboles de Decisión

La forma en que se hace la clasificación es la siguiente:

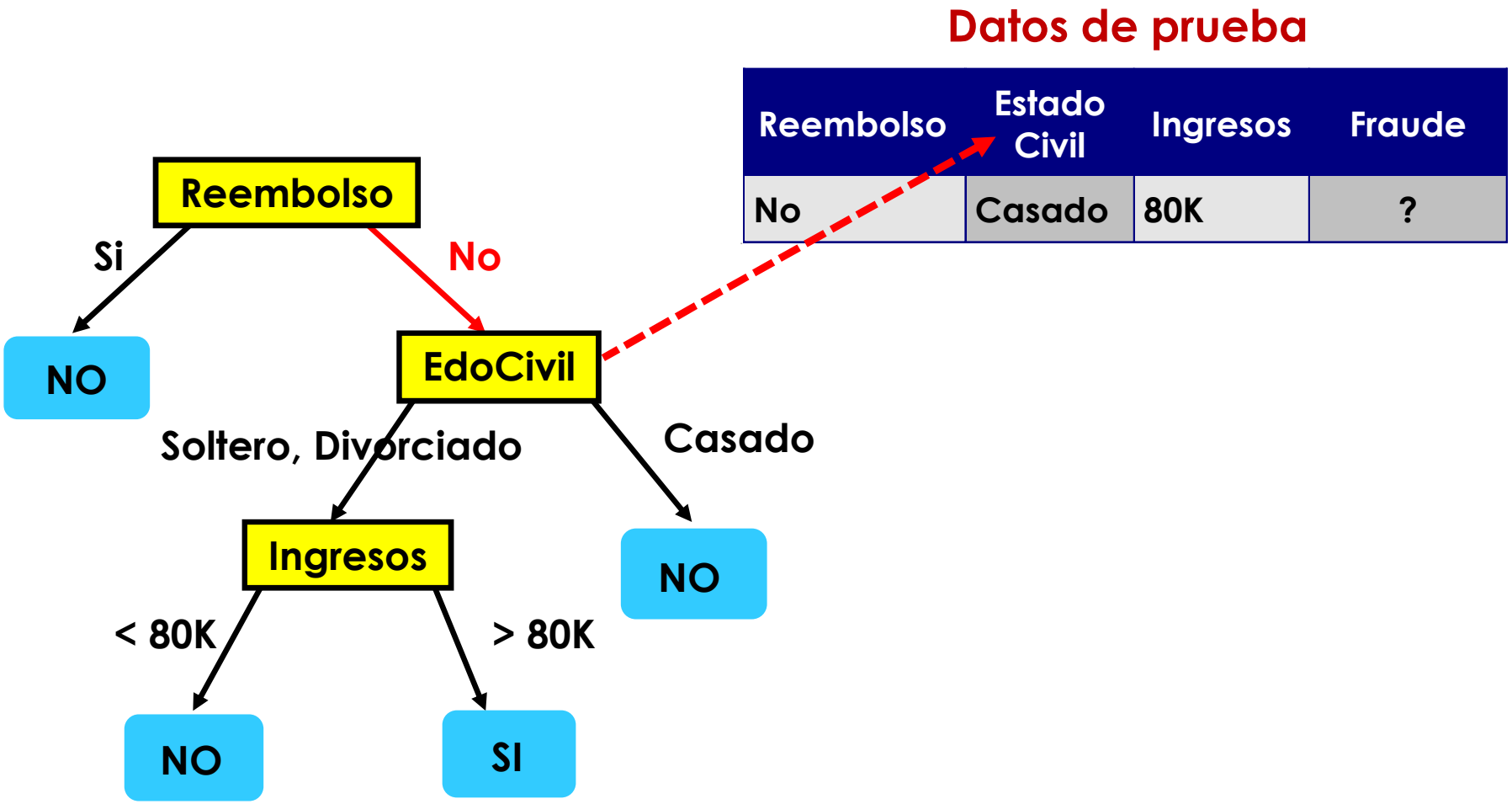


Datos de prueba

Reembolso	Estado Civil	Ingresos	Fraude
No	Casado	80K	?

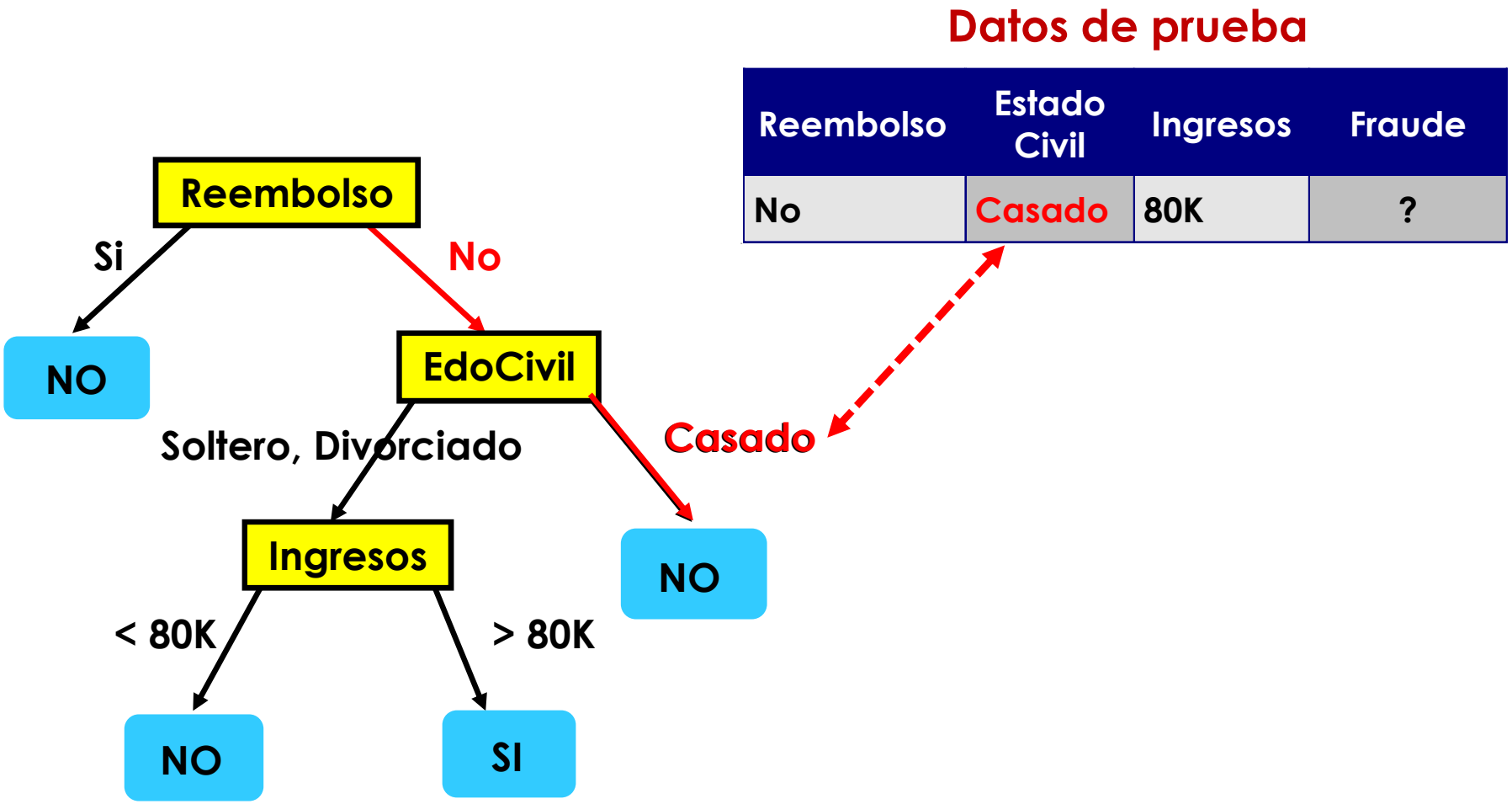
…Inducción de Árboles de Decisión

La forma en que se hace la clasificación es la siguiente:



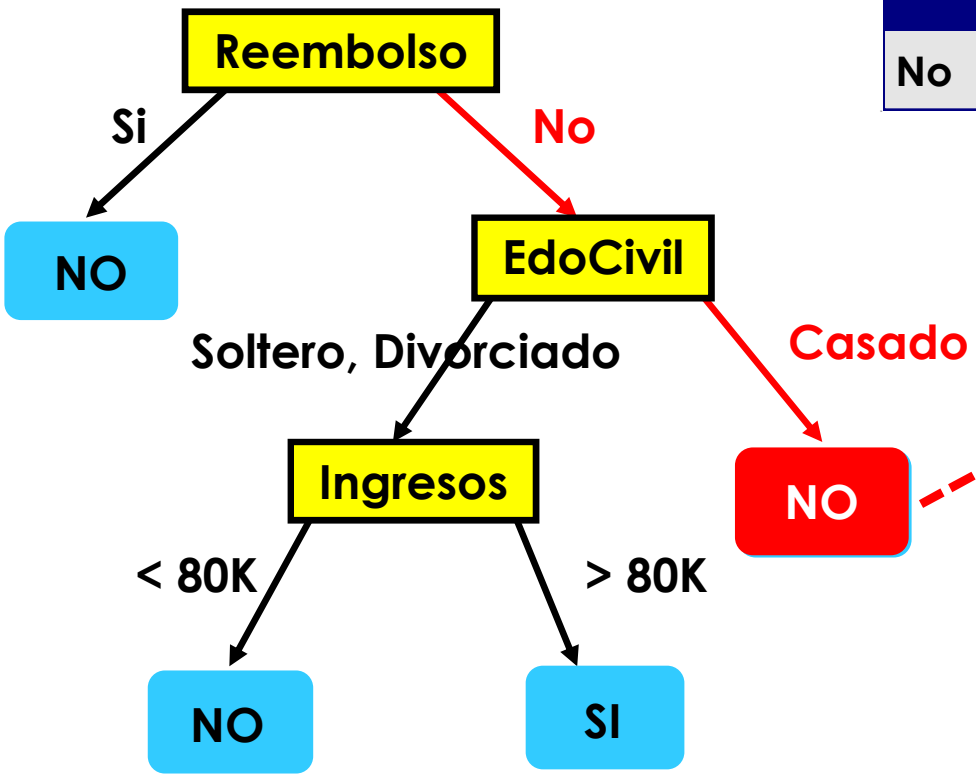
…Inducción de Árboles de Decisión

La forma en que se hace la clasificación es la siguiente:



…Inducción de Árboles de Decisión

La forma en que se hace la clasificación es la siguiente:



Datos de prueba

Reembolso	Estado Civil	Ingresos	Fraude
No	Casado	80K	NO

Asigna **Fraude** como **"NO"**

…Inducción de Árboles de Decisión

- Los **árboles de decisión** son muy populares ya que para su construcción no se requiere ningún conocimiento de dominio o establecimiento de parámetros, por lo que son recomendados para hacer **descubrimiento de conocimiento**.
- Debido a su forma de construcción son **fáciles de asimilar**, los pasos de aprendizaje son simples y rápidos, en general tienen buena **exactitud**.
- Su éxito depende de los datos sobre los que se aplique.
- Son utilizados en varias áreas de aplicación: **medicina, manufactura y producción, análisis financiero, astronomía, biología molecular, etc**.

¿Cómo construir un árbol de decisión a partir de un conjunto de entrenamiento?

Algoritmo de Hunt (1950)

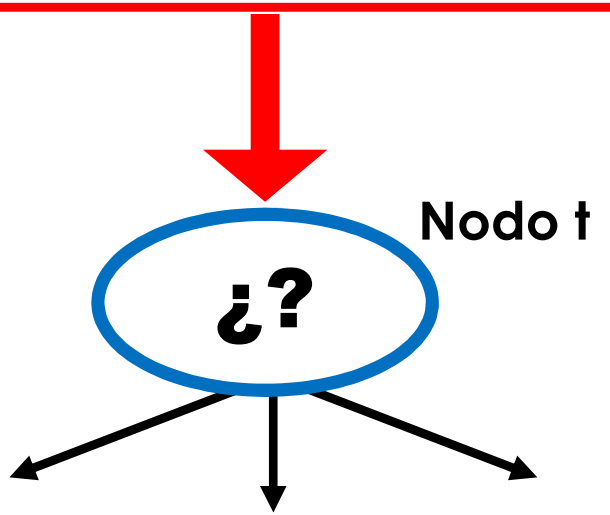
Dado un conjunto D_t (*tuplas de entrenamiento*) que llega a un nodo ***t***, el procedimiento es el siguiente:

- Si D_t contiene registros que pertenecen a una misma clase y_t, entonces ***t*** es un nodo hoja etiquetado con y_t .
- Si D_t es un conjunto vacío, entonces ***t*** es una nodo hoja etiquetado con la clase default y_d.
- Si D_t contiene registros que pertenecen a más de una clase, utilizar un atributo de prueba para **dividir** los datos en subconjuntos más pequeños.
- Aplicar el procedimiento de forma recursiva a cada subconjunto.

¿Cuál atributo debiera probarse en cada división?

D_t

id	Reemb.	Estado civil	Ingresos	Fraude
1	Si	Soltero	125K	No
2	No	Casado	100K	No
3	No	Soltero	70K	No
4	Si	Casado	120K	No
5	No	Divorciado	95K	Si
6	No	Casado	60K	No
7	Si	Divorciado	220K	No
8	No	Soltero	85K	Si
9	No	Casado	75K	No
10	No	Soltero	90K	Si



Generación de un árbol de decisión

Algoritmo:

- **Objetivo:**

  Generar un árbol de decisión a partir de un conjunto de tuplas de entrenamiento de una partición D.

- **Entrada:**
  - **Una partición de datos D**, la cual es un conjunto de tuplas de entrenamiento y sus etiquetas de clase asociadas;
  - **Una *lista_de_atributos***, la cual es el conjunto de atributos de partición candidatos;

- **Método_selección_atributos:**

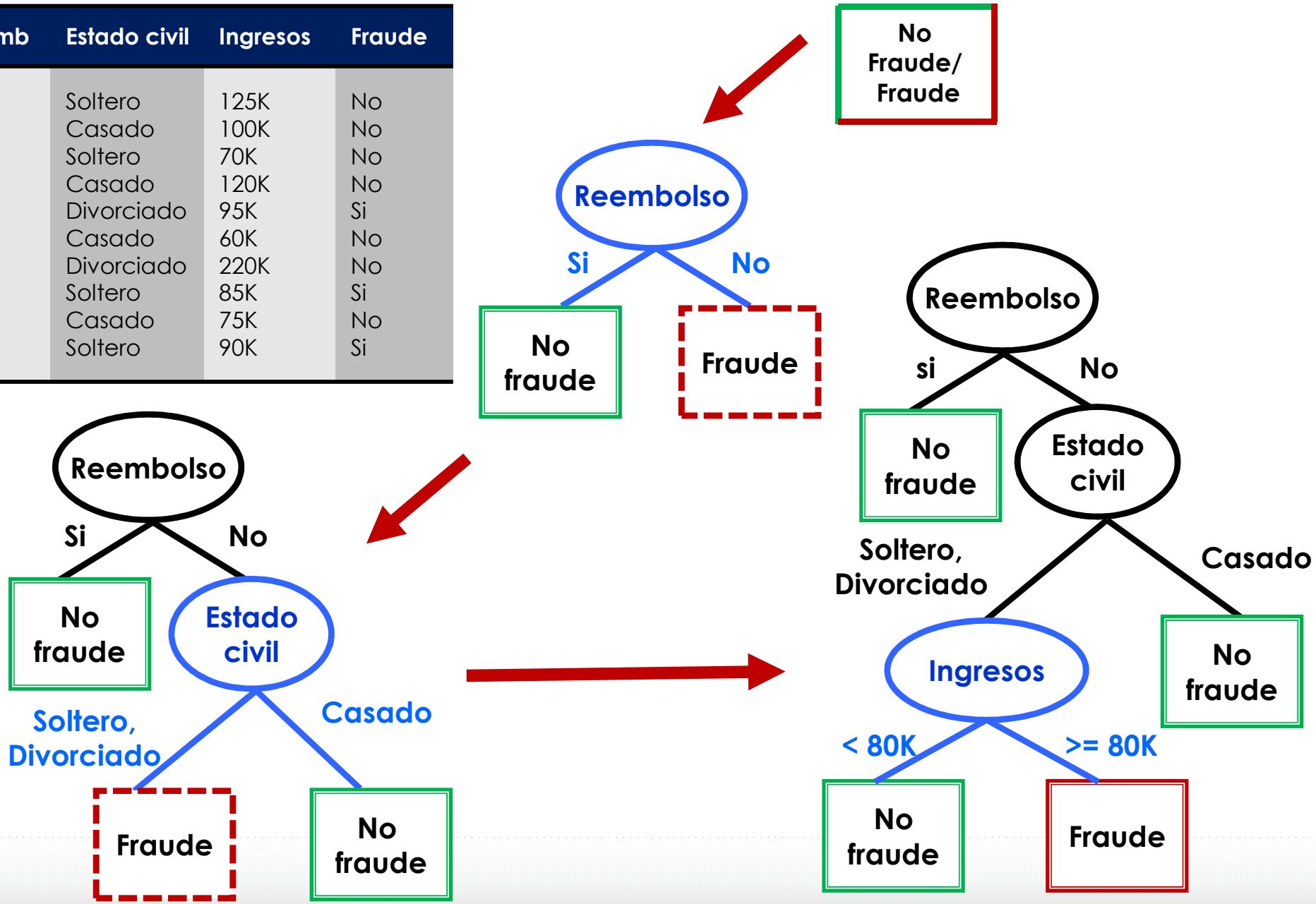
  Un procedimiento para determinar el **criterio de partición** que **mejor divida** las tuplas en **clases individuales**. Este criterio consiste de un **atributo de partición** y posiblemente de un punto de partición o un subconjunto de partición.

- **Salida:**

  Un árbol de decisión.

...Generación de un árbol de decisión

id	Reemb	Estado civil	Ingresos	Fraude
1	Si	Soltero	125K	No
2	No	Casado	100K	No
3	No	Soltero	70K	No
4	Si	Casado	120K	No
5	No	Divorciado	95K	Si
6	No	Casado	60K	No
7	Si	Divorciado	220K	No
8	No	Soltero	85K	Si
9	No	Casado	75K	No
10	No	Soltero	90K	Si



Aspectos a considerar

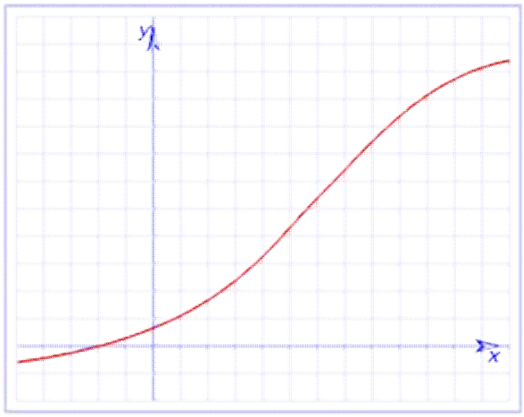
- Utiliza una estrategia **greedy top-down**:

  Dividir los registros en función de una prueba de atributo que optimiza cierto criterio.

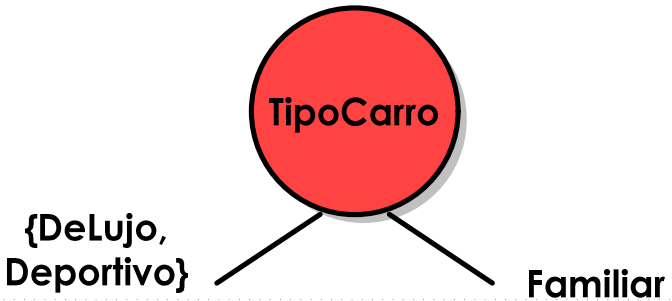
- Cuestiones:
  - ❑ Determinar cómo particionar los registros:
    - → **¿Cómo especificar la condición de prueba?**
    - → **¿Cómo determinar la mejor partición?**
  - ❑ Determinar cuándo detener el particionado
  - ❑ ¿Se debe utilizar una partición binaria o múltiple?

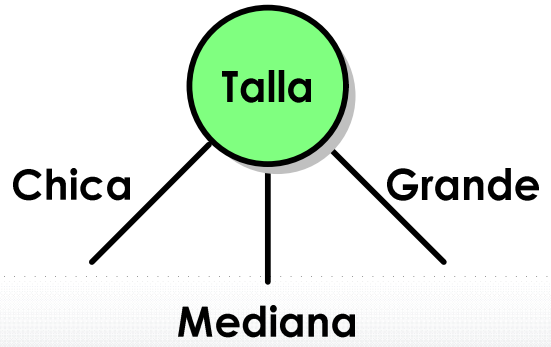
Condición de prueba

- Depende del tipo de atributo
  - ***Nominal***
  - ***Ordinal***
  - ***Continuo***

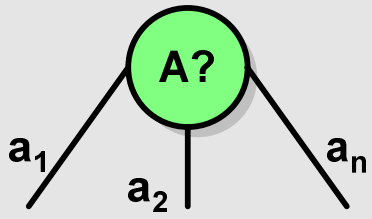
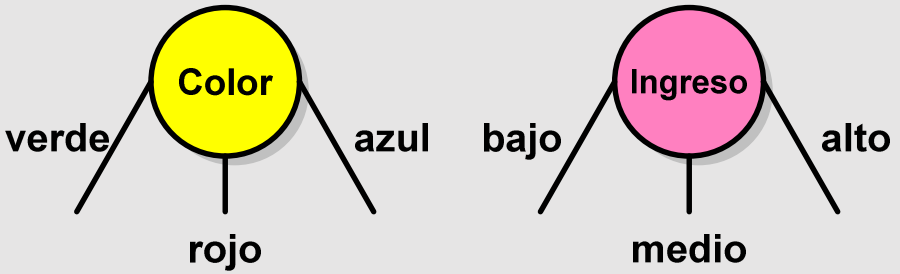
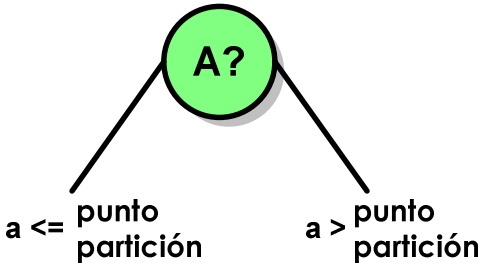
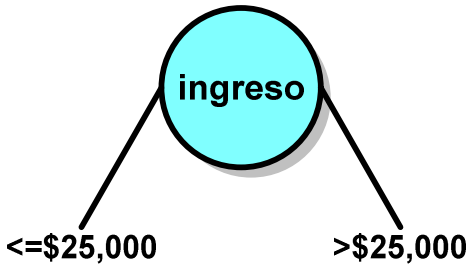
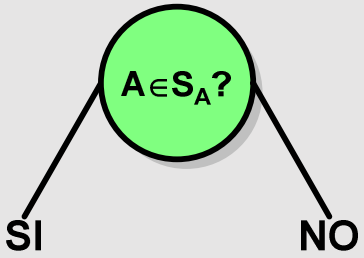
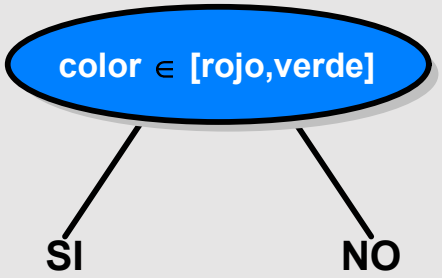


- Depende del número de formas de dividir:



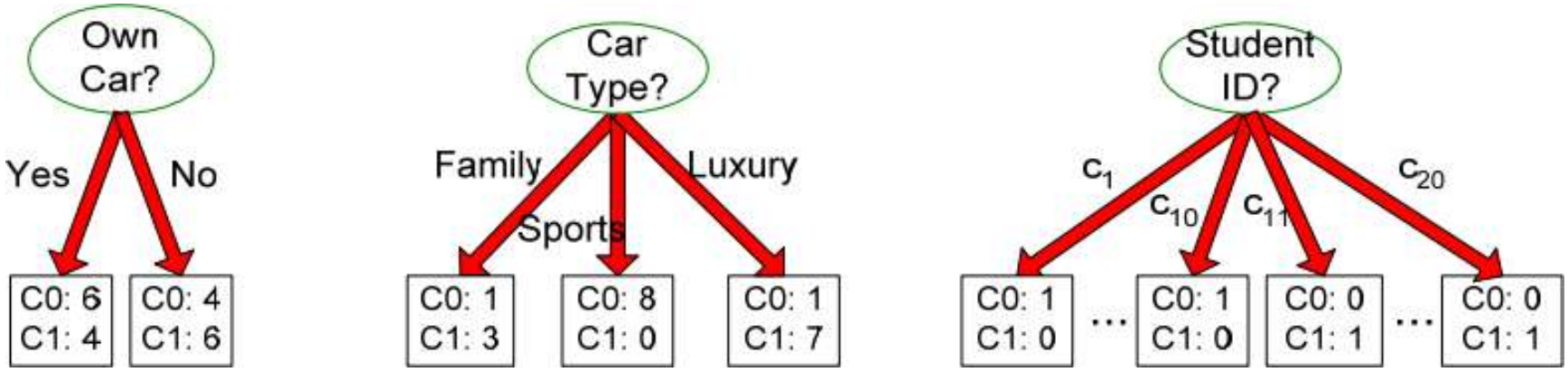


Particiones

Tipo de partición	Ejemplo
Valor discreto	 
Valor continuo	 
Valor discreto para árbol binario	 

¿Cuál es la mejor partición?

- Se tiene un conjunto de **20 tuplas**, 10 de ellas etiquetadas con la **clase 0** y 10 etiquetados con la **clase 1**.



- **¿Cuál condición de prueba es la mejor?**
  - Se van a preferir nodos con distribución de clases homogéneos.
  - Necesitamos por ende, una medida de la impureza del nodo.

C0: 5 C1: 5	C0: 9 C1: 1
Non-homogeneous, **High degree of impurity**	**Homogeneous,** **Low degree of impurity**

Medidas de selección de atributos

- Se trata de un conjunto de **heurísticas** para determinar el **criterio de partición** que **mejor** divida un conjunto de datos **D** (*contiene etiquetas de clase y tuplas de entrenamiento*) en **clases individuales**.
- Si se desea dividir **D** en particiones **más pequeñas** de acuerdo a los resultados del **criterio de partición**, idealmente cada partición debería ser **pura** (*las tuplas que pertenecen a una partición determinada son de la misma clase*).
- Estas medidas también son conocidas como **reglas de partición** (*determinan cómo las tuplas en un nodo dado se deben dividir*)
- Estas medidas proporcionan un **ranking** por cada atributo descrito en las tuplas de entrenamiento que se proporcionan.
- El atributo que tiene la **mejor puntuación** para la medida es el que se elige como atributo de partición para las tuplas dadas.

ID3: Ganancia de información

- A finales de los **70s** y principios de los **80s J. Ross Quinlan** (*investigador en máquinas de aprendizaje*) desarrolló el algoritmo ID3 (***Iterative Dichotomiser 3***).
- Utiliza un enfoque **greedy** apoyándose en el enfoque ***top-down***.

Machine Learning 1: 81–106, 1986

Induction of Decision Trees

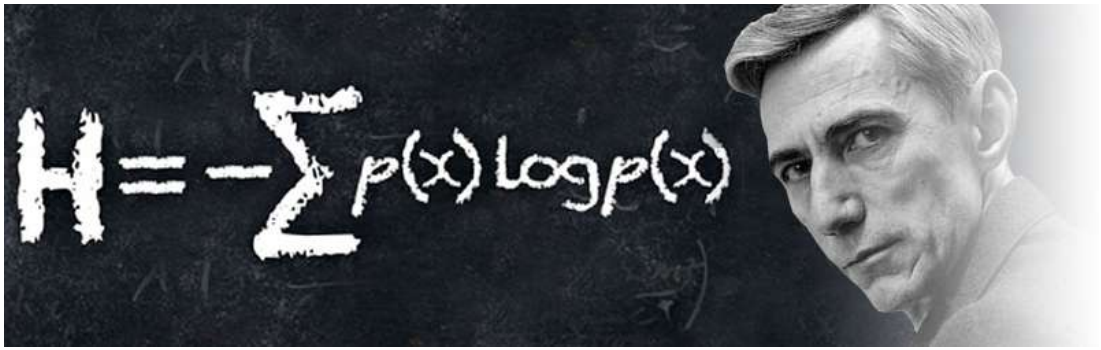
J.R. QUINLAN (munnari!nswitgould.oz!quinlan@seismo.css.gov)
Centre for Advanced Computing Sciences, New South Wales Institute of Technology, Sydney 2007, Australia

(Received August 1, 1985)

Key words: classification, induction, decision trees, information theory, knowledge acquisition, expert systems

…ID3: Ganancia de información

- Este algoritmo se basa en los estudios de **Claude Shannon** (*pionero de la Teoría de la Información*), que estudiaba el **valor o contenido** de la información de los mensajes:



Reprinted with corrections from *The Bell System Technical Journal*,
Vol. 27, pp. 379–423, 623–656, July, October, 1948.

A Mathematical Theory of Communication

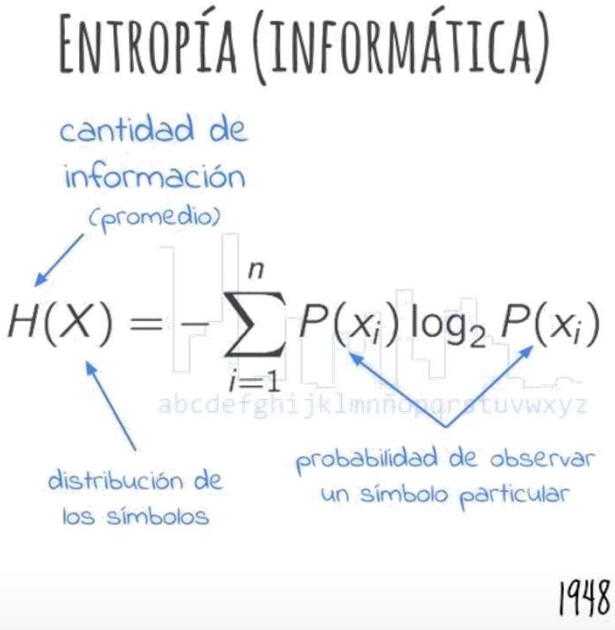
By C. E. SHANNON

INTRODUCTION

THE recent development of various methods of modulation such as PCM and PPM which exchange bandwidth for signal-to-noise ratio has intensified the interest in a general theory of communication. A basis for such a theory is contained in the important papers of Nyquist[1] and Hartley[2] on this subject. In the present paper we will extend the theory to include a number of new factors, in particular the effect of noise in the channel, and the savings possible due to the statistical structure of the original message and due to the nature of the final destination of the information.

…ID3: Ganancia de información

- Estableció el concepto de **Entropía de la información,** la cual *mide* la incertidumbre de una fuente de información. Se puede considerar como la ***cantidad de información promedio*** que contienen los símbolos usados:
  - Los símbolos con ***menor probabilidad*** son los que aportan mayor información; por ejemplo, si se considera como sistema de símbolos a las palabras en un texto, palabras frecuentes como **"*que*", "*el*", "*a*"** aportan poca información, mientras que palabras menos frecuentes como **"*corren*", "*niño*", "*perro*"** aportan más información.



…ID3: Ganancia de información

- Dado un nodo **N** que representa a las tuplas de la partición **D**, el atributo que tenga la **mayor ganancia de información** se elige como el atributo de partición para el nodo **N**.
- Este atributo debe **reducir al mínimo** la **información necesaria** para clasificar las tuplas en las particiones resultantes y **reflejar menos aleatoriedad** (impureza) en estas particiones.
- Este enfoque **minimiza** el número esperado de ensayos necesarios para **clasificar una tupla dada** y **garantiza encontrar un árbol de forma simple** (*pero no necesariamente el árbol más simple*).

…ID3: Ganancia de información

La **información esperada**, necesaria para clasificar una tupla en D está dada por:

$$Info(D) = -\sum_{i=1}^{m} p_i \log_2 (p_i)$$

Donde:

- $\mathbf{p_i}$ es la probabilidad de que una tupla arbitraria en **D** pertenezca a una clase $\mathbf{C_i}$, se estima a partir de $|\mathbf{C_{i,D}}|/|\mathbf{D}|$
- $|\mathbf{C_{i,D}}|$ número de tuplas de la clase $\mathbf{C_i}$ en la partición **D**
- $|\mathbf{D}|$ es el número de tuplas en la partición **D**.
- Se utiliza $\mathbf{log_2}$ debido a que la información se codifica en **bits**.
- **Info(D)** es la cantidad promedio de información necesaria para identificar la etiqueta de clase de una tupla en **D.**

A **Info(D)** también se le conoce como **Entropía**.

…ID3: Ganancia de información

- Ahora supongamos que queremos dividir la tuplas en **D** en algunos atributos de **A**, que tiene **n** valores distintos, **$\{a_1, a_2, \ldots, a_n\}$**.
- Si **A** tiene valores discretos, estos valores corresponden directamente a **n resultados** de una prueba en **A**, entonces, el atributo **A** puede utilizarse para dividir **D** en **n particiones** o subconjuntos, **$\{D_1, D_2, \ldots, D_n\}$**, donde **$D_j$** contiene aquellas tuplas en **D** que tiene el resultado **a_j** de **A**:

*Idealmente, deseamos que estas particiones produzcan clasificaciones exactas de tuplas (**particiones puras**).*

- Sin embargo, es mucho más probable que se obtengan particiones impuras.

*¿**Cuánta información adicional** necesitamos (después de la partición) para obtener una **partición exacta**?*

…ID3: Ganancia de información

$$Info_A(D) = \sum_{j=1}^{n} \frac{|D_j|}{|D|} \bullet Info(D_j) = \frac{|D_j|}{|D|} \bullet \sum_{j=1}^{n} Info(D_j)$$

Donde:

- El término $|\mathbf{D_j}|/|\mathbf{D}|$ actúa como el peso de la partición de **orden j**.
- **$Info_A(D)$** es la información esperada necesaria para clasificar una tupla de D **basada en la partición hecha por el atributo A**.
- **Cuanto menor** sea la información esperada requerida, **mayor es la pureza** de las particiones.

…ID3: Ganancia de información

- Finalmente, la **ganancia de información** se define como la **diferencia** entre el **requerimiento de información original** (sobre la base de la proporción completa de clases) y el **nuevo requerimiento** (obtenida después de la partición en A):

$$Gain(A) = Info(D) - Info_A(D)$$

- La ganancia nos dice qué tanto ganaríamos si partimos un nodo **N** en el atributo **A.**
- El atributo **A** con la **mayor ganancia de información** se elige como el atributo de partición en el nodo **N**.

ID3: Ejemplo

Supongamos que se tiene el siguiente conjunto de entrenamiento D, con tuplas que tienen etiquetas de clase:

ID	edad	ingreso	estudiante	calificacion_credito	comprar_computadora
1	joven	alto	no	favorable	no
2	joven	alto	no	excelente	no
3	adulto	alto	no	favorable	si
4	a_mayor	medio	no	favorable	si
5	a_mayor	bajo	si	favorable	si
6	a_mayor	bajo	si	excelente	no
7	adulto	bajo	si	excelente	si
8	joven	medio	no	favorable	no
9	joven	bajo	si	favorable	si
10	a_mayor	medio	si	favorable	si
11	joven	medio	si	excelente	si
12	adulto	medio	no	excelente	si
13	adulto	alto	si	favorable	si
14	a_mayor	medio	no	excelente	no

ID3: Ejemplo

Para determinar el criterio de partición, necesitamos calcular la **ganancia de información** de cada atributo:

- Lo primero que debemos hacer es calcular la información necesaria esperada para clasificar una tupla en la partición D:

$$Info(D) = -\frac{9}{14}\log_2\left(\frac{9}{14}\right) - \frac{5}{14}\log_2\left(\frac{5}{14}\right) = 0.940 \text{ bits}$$

- Ahora, vamos a calcular los requerimientos de información esperados para cada atributo. Comencemos con edad:

$$Info_A(D) = \frac{|D_j|}{|D|} \bullet \sum_{j=1}^{n} Info(D_j)$$

$$Info_{edad}(D) = \frac{5}{14} \times \left(-\frac{2}{5}\log_2\left(\frac{2}{5}\right) - \frac{3}{5}\log_2\left(\frac{3}{5}\right) \right) + \cdots$$

Total de jóvenes que SI comprarían

Total de jóvenes que NO comprarían

Total de jóvenes

Total de tuplas

Total de jóvenes

…ID3: Ejemplo

Info*$_{edad}$*(D) quedaría de la siguiente forma:

$$Info_{edad}(D) = \frac{5}{14} \times \left(-\frac{2}{5}\log_2\left(\frac{2}{5}\right) - \frac{3}{5}\log_2\left(\frac{3}{5}\right)\right) +$$

$$\frac{4}{14} \times \left(-\frac{4}{4}\log_2\left(\frac{4}{4}\right) - \frac{0}{4}\log_2\left(\frac{0}{4}\right)\right) +$$

$$\frac{5}{14} \times \left(-\frac{3}{5}\log_2\left(\frac{3}{5}\right) - \frac{2}{5}\log_2\left(\frac{2}{5}\right)\right)$$

$$\boxed{Info_{edad}(D) = 0.3468 + 0 + 0.3468 = 0.6936 \text{ bits}}$$

Info*$_{estudiante}$*(D) quedaría de la siguiente forma:

$$Info_{estudiante}(D) = \frac{7}{14} \times \left(-\frac{3}{7}\log_2\left(\frac{3}{7}\right) - \frac{4}{7}\log_2\left(\frac{4}{7}\right)\right) +$$

$$\frac{7}{14} \times \left(-\frac{6}{7}\log_2\left(\frac{6}{7}\right) - \frac{1}{7}\log_2\left(\frac{1}{7}\right)\right) = \boxed{0.789 \text{ bits}}$$

…ID3: Ejemplo

Info*$_{ingreso}$*(D) quedaría de la siguiente forma:

$$Info_{ingreso}(D) = \frac{4}{14}\times\left(-\frac{2}{4}\log_2\left(\frac{2}{4}\right)-\frac{2}{4}\log_2\left(\frac{2}{4}\right)\right)+$$

$$\frac{6}{14}\times\left(-\frac{4}{6}\log_2\left(\frac{4}{6}\right)-\frac{2}{6}\log_2\left(\frac{2}{6}\right)\right)+$$

$$\frac{4}{14}\times\left(-\frac{3}{4}\log_2\left(\frac{3}{4}\right)-\frac{1}{4}\log_2\left(\frac{1}{4}\right)\right)=\boxed{0.911\,\text{bits}}$$

Info*$_{calif\_crédito}$*(D) quedaría de la siguiente forma:

$$Info_{calif\_crédito}(D) = \frac{8}{14}\times\left(-\frac{6}{8}\log_2\left(\frac{6}{8}\right)-\frac{2}{8}\log_2\left(\frac{2}{8}\right)\right)+$$

$$\frac{6}{14}\times\left(-\frac{3}{6}\log_2\left(\frac{3}{6}\right)-\frac{3}{6}\log_2\left(\frac{3}{6}\right)\right)=0.892\ \text{bits}$$

…ID3: Ejemplo

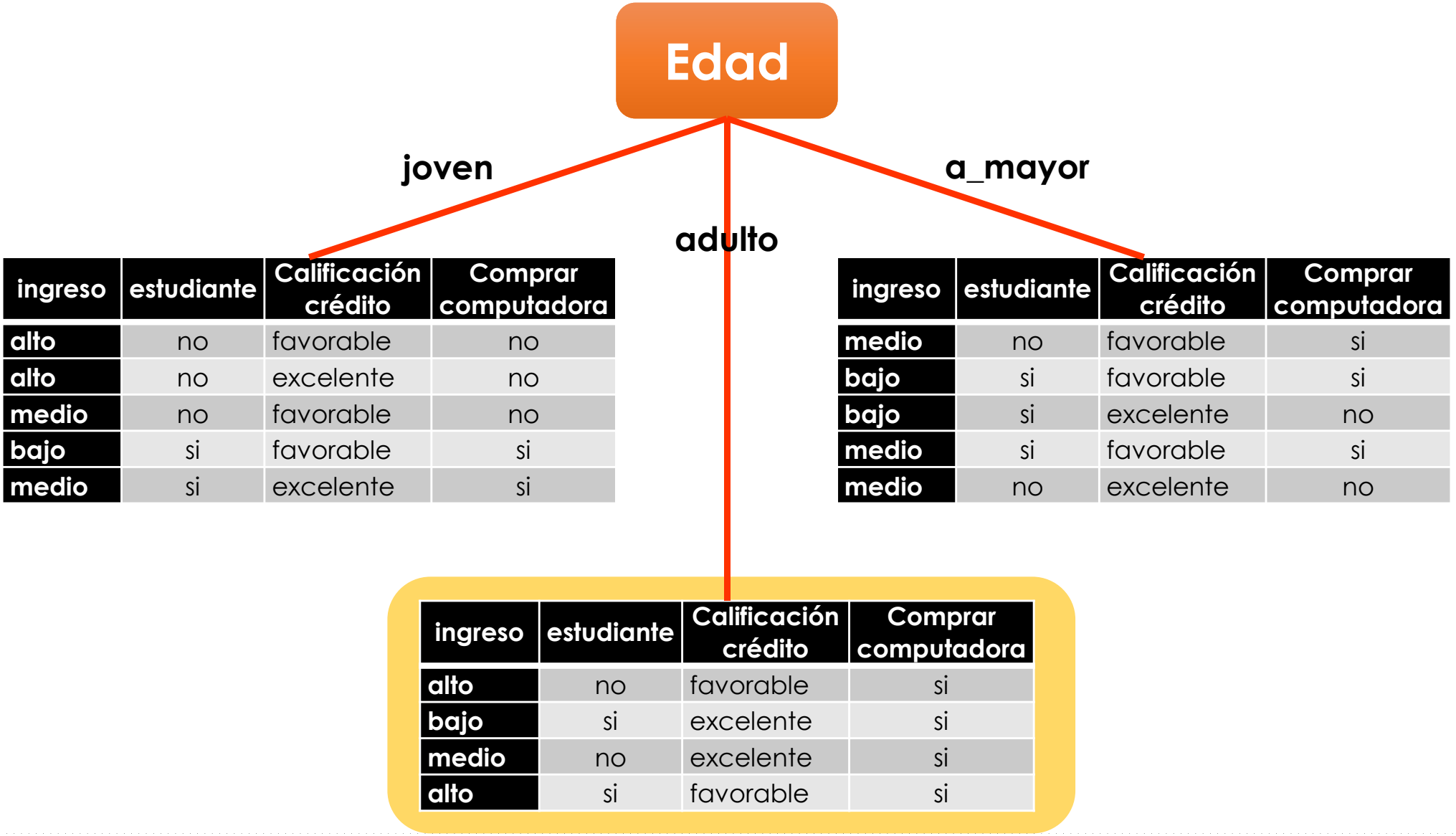
$$Ganancia(edad) = 0.940 - 0.694 = 0.246$$

$$Ganancia(\text{ingreso}) = 0.940 - 0.911 = 0.029$$

$$Ganancia(\text{estudiante}) = 0.940 - 0.789 = 0.151$$

$$Ganancia(\text{calif\_credito}) = 0.940 - 0.892 = 0.048$$

…ID3: Ejemplo



ingreso	estudiante	Calificación crédito	Comprar computadora
alto	no	favorable	no
alto	no	excelente	no
medio	no	favorable	no
bajo	si	favorable	si
medio	si	excelente	si

ingreso	estudiante	Calificación crédito	Comprar computadora
alto	no	favorable	si
bajo	si	excelente	si
medio	no	excelente	si
alto	si	favorable	si

ingreso	estudiante	Calificación crédito	Comprar computadora
medio	no	favorable	si
bajo	si	favorable	si
bajo	si	excelente	no
medio	si	favorable	si
medio	no	excelente	no

Partición pura

ID3: Atributos continuos

- Se debe seleccionar el mejor punto de partición para **A**. La partición se hace en un conjunto discreto de intervalos, por ejemplo: **A < c** y **A ≥ c**.
- **¿Cómo seleccionar c?** Nos gustaría el valor que produzca la **mayor ganancia de información**. Se sigue la siguiente estrategia:
  - Se **ordenan** todos los valores de forma **creciente.**
  - Típicamente, se selecciona el **punto intermedio** que se encuentra entre cada par de valores adyacentes y cada uno se considera como posible punto de división:

$$\frac{a_i + a_{i+1}}{2}$$

  - Para cada posible punto de división se necesita evaluar **$\text{Info}_A(D)$**, donde el numero de particiones es **2**.
  - El punto con los **menor requerimiento de información** se selecciona para hacer la partición.

…ID3: Atributos continuos

- Por ejemplo, pensemos que tenemos los siguientes datos:

Temperatura	40	48	60	72	80	90
Jugar Tenis	No	No	Si	Si	Si	No

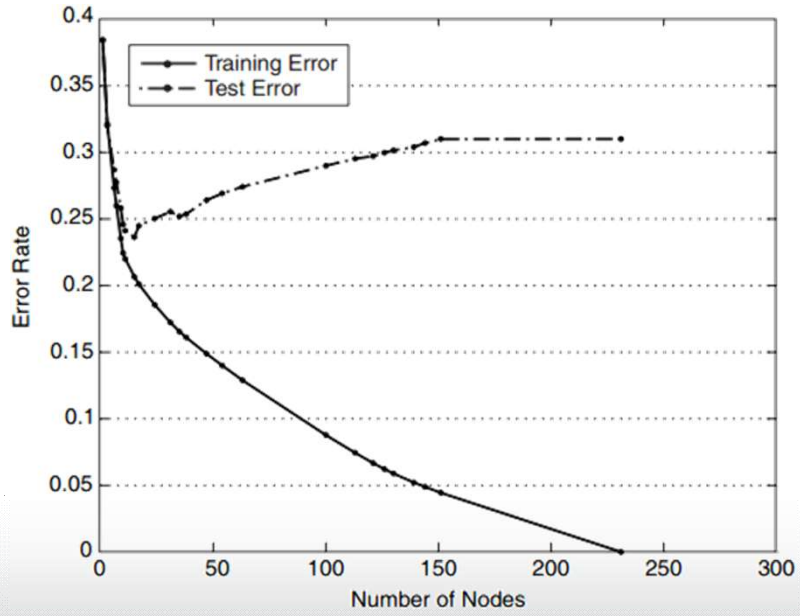
- Candidatos para particionar:

$$c_1 = \frac{40+48}{2} = 44 \qquad c_2 = \frac{48+60}{2} = 54 \qquad c_3 = \frac{60+72}{2} = 66$$

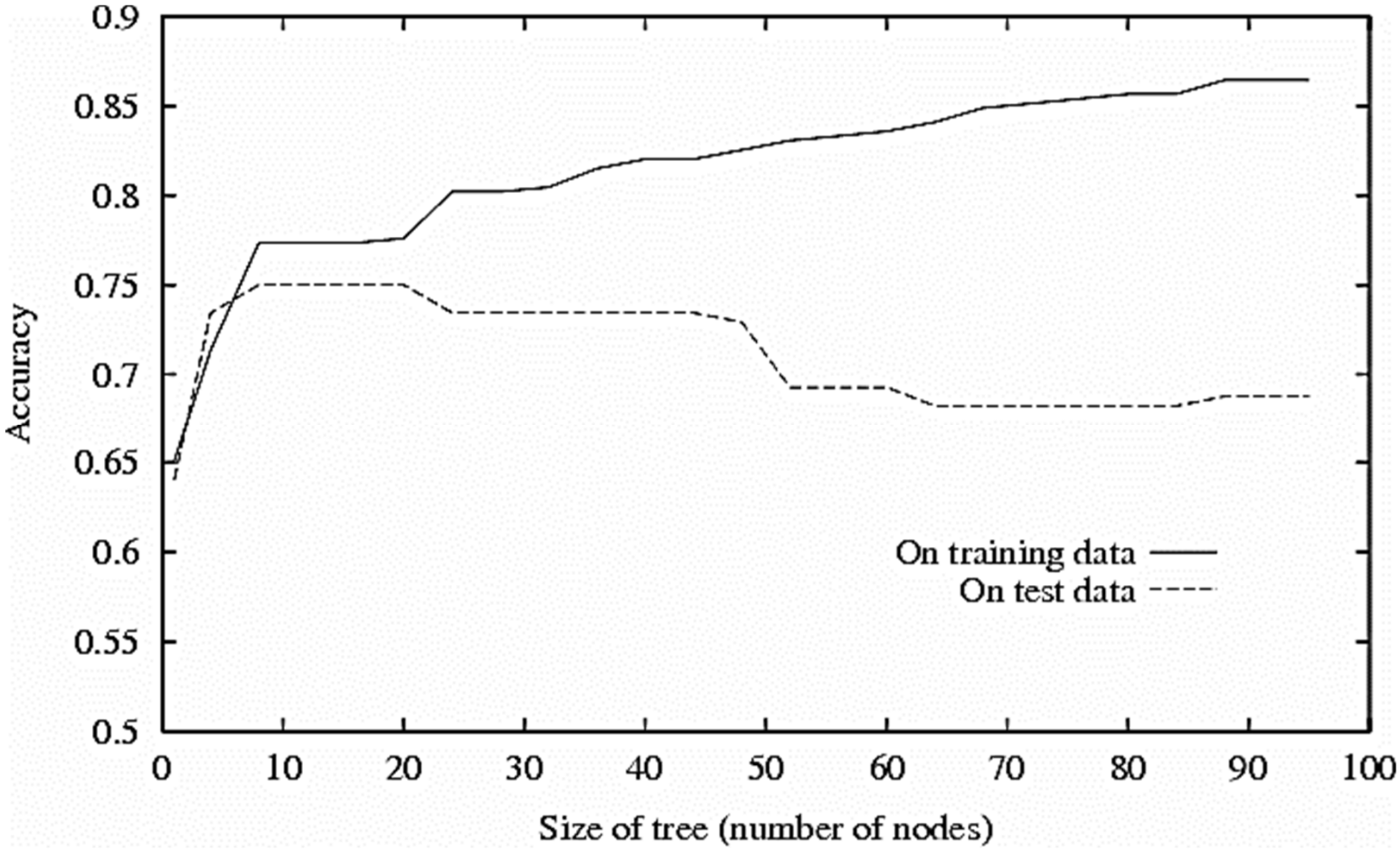
$$c_4 = \frac{72+80}{2} = 76 \qquad c_5 = \frac{80+90}{2} = 85$$

ID3: Problemas

- Su complejidad crece **linealmente** con el número de tuplas de entrenamiento y **exponencialmente** con el número de atributos.
- Favorece la elección de variables con **mayor número** de valores.
- **Problema de sobreajuste:** al hacer crecer el árbol hasta que clasifique correctamente todas la tuplas de entrenamiento:
  - Si hay ruido en las tuplas, el árbol **aprende del ruido.**
  - Si hay pocas tuplas en los nodos hoja, **no son representativos.**
  - **No son capaces de generalizar.**



ID3: Sobreajuste



¿Cómo evitarlo?

- Detener el crecimiento cuando la partición no sea estadísticamente significativa.
- Obtener el árbol completo y hacer una **post-poda**.

Árbol C4.5

- **J. Ross Quinlan** propuso en **1993** al sucesor del algoritmo **ID3**, al cual llamó algoritmo **C4.5** y se convirtió en un ***benchmark*** para los nuevos algoritmos de aprendizaje supervisado.

Machine Learning, 16, 235–240 (1994)

Book Review: *C4.5: Programs for Machine Learning* **by J. Ross Quinlan. Morgan Kaufmann Publishers, Inc., 1993.**

STEVEN L. SALZBERG salzberg@cs.jhu.edu

Department of Computer Science, Johns Hopkins University, Baltimore, MD 21218

…Árbol C4.5

- Al igual que **ID3** adopta un enfoque ***greedy***.
- Genera un árbol de decisión a partir de los datos de entrenamiento mediante **particiones recursivas**.
- Utiliza la estrategia **profundidad-primero** (*depth-first*), considera todas las pruebas posibles que pueden dividir el conjunto de datos y selecciona aquella que resulte con **mayor ganancia de información**.
- Trabaja con valores **discretos y continuos**.
- Se trata de árboles **menos frondosos** ya que cada hoja cubre **una distribución de clases**.
- La última versión libre fue el algoritmo **C4.8** antes de que se publicara su versión comercial: **C5**

C4.5: Información de partición

- La **ganancia de información** es una medida que está **sesgada** hacia pruebas que tienen muchos resultados: *prefiere seleccionar atributos que tengan un gran numero de valores*:
  - Por ejemplo, pensemos en un atributo que funciona como **identificador único** (p.e. un ID), si hiciéramos un partición sobre éste, se encontrarían un **gran número de particiones** (tantas como valores se tengan en el atributo), cada una conteniendo **solo una tupla**.
  - En este caso resultan **particiones puras**, donde la información necesaria para clasificar un conjunto de datos **D** basados en esta partición sería **cero**.
  - La información obtenida mediante la división de este atributo es **máxima**.
  - **Es evidente que una partición de este tipo es inútil para la clasificación.**

…C4.5: Información de partición

- El algoritmo **C4.5** utiliza una heurística llamada **tasa de ganancia** (*gain ratio*), la cual aplica una especie de **normalización** a la ganancia de información usando un valor llamado **información de partición**:

$$SplitInfo_A(D) = -\sum_{i=1}^{v} \frac{|D_j|}{|D|} \times \log_2\left(\frac{|D_j|}{|D|}\right)$$

- Dicha medida representa la **información potencial** que se generaría si se dividiera el conjunto de entrenamiento en ***v particiones*** que corresponden a ***v resultados*** de una prueba sobre un atributo **A**.
- La **tasa de ganancia** se define entonces:

$$GainRatio(A) = \frac{Gain(A)}{SplitInfo(A)}$$

- El atributo con la **máxima tasa de ganancia** es seleccionado como el atributo de partición.

C4.5: Ejemplo

Regresando al ejemplo que se analizó para el árbol **ID3**:

ID	edad	ingreso	estudiante	calificacion_credito	comprar_computadora
1	joven	alto	no	favorable	no
2	joven	alto	no	excelente	no
3	adulto	alto	no	favorable	si
4	a_mayor	medio	no	favorable	si
5	a_mayor	bajo	si	favorable	si
6	a_mayor	bajo	si	excelente	no
7	adulto	bajo	si	excelente	si
8	joven	medio	no	favorable	no
9	joven	bajo	si	favorable	si
10	a_mayor	medio	si	favorable	si
11	joven	medio	si	excelente	si
12	adulto	medio	no	excelente	si
13	adulto	alto	si	favorable	si
14	a_mayor	medio	no	excelente	no

…C4.5: Ejemplo

Vamos por ejemplo a calcular la **tasa de ganancia** para el atributo **ingreso**:

- Una prueba sobre este atributo dividiría los datos en **tres particiones** (***bajo, medio y alto***), las cuales contienen **4, 6 y 4** tuplas respectivamente, por lo tanto:

$$SplitInfo_{ingreso}(D) = -\frac{4}{14}\times\log_2\left(\frac{4}{14}\right) - \frac{6}{14}\times\log_2\left(\frac{6}{14}\right) - \frac{4}{14}\times\log_2\left(\frac{4}{14}\right)$$

$$= 1.557$$

$$GainRatio(ingreso) = \frac{0.029}{1.557} = \boxed{0.0186}$$

…C4.5: Ejemplo

Para los otros tres atributos quedaría de la siguiente forma:

$$SplitInfo_{edad}(D) = -\frac{5}{14}\times\log_2\left(\frac{5}{14}\right) - \frac{4}{14}\times\log_2\left(\frac{4}{14}\right) - \frac{5}{14}\times\log_2\left(\frac{5}{14}\right)$$

$$= 1.577$$

$$GainRatio(edad) = 0.246/1.577 = \boxed{0.156}$$

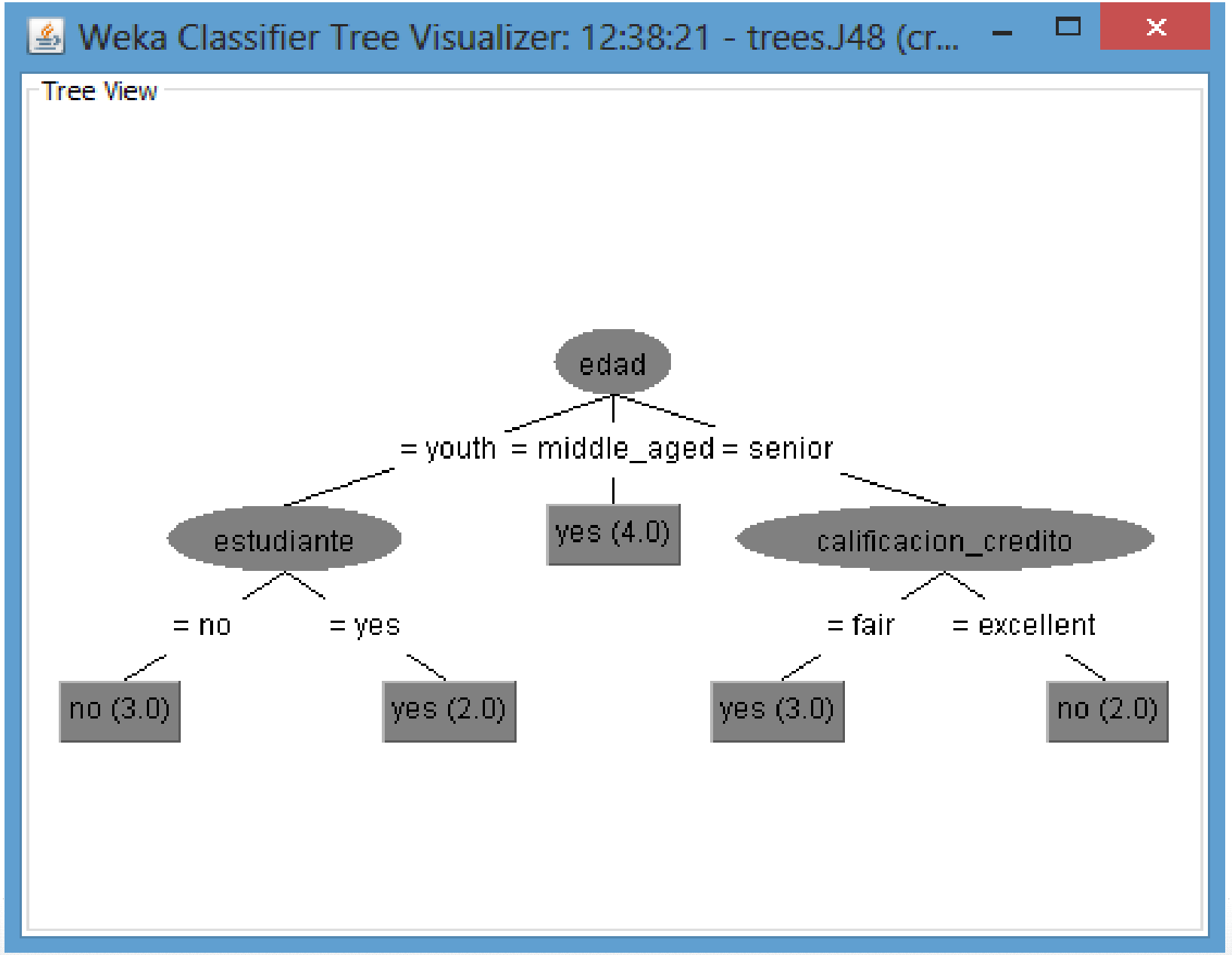
$$SplitInfo_{estudiante}(D) = -\frac{7}{14}\times\log_2\left(\frac{7}{14}\right) - \frac{7}{14}\times\log_2\left(\frac{7}{14}\right) = 1.0$$

$$GainRatio(estudiante) = 0.151/1.0 = \boxed{0.151}$$

$$SplitInfo_{calif\_cred}(D) = -\frac{8}{14}\times\log_2\left(\frac{8}{14}\right) - \frac{6}{14}\times\log_2\left(\frac{6}{14}\right) = 0.985$$

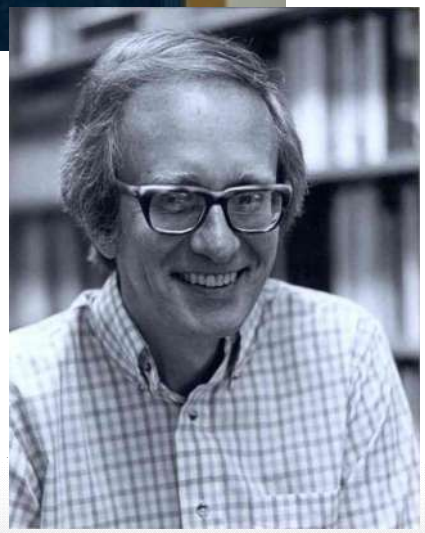
$$GainRatio(calif\_cred) = 0.048/0.985 = \boxed{0.048}$$

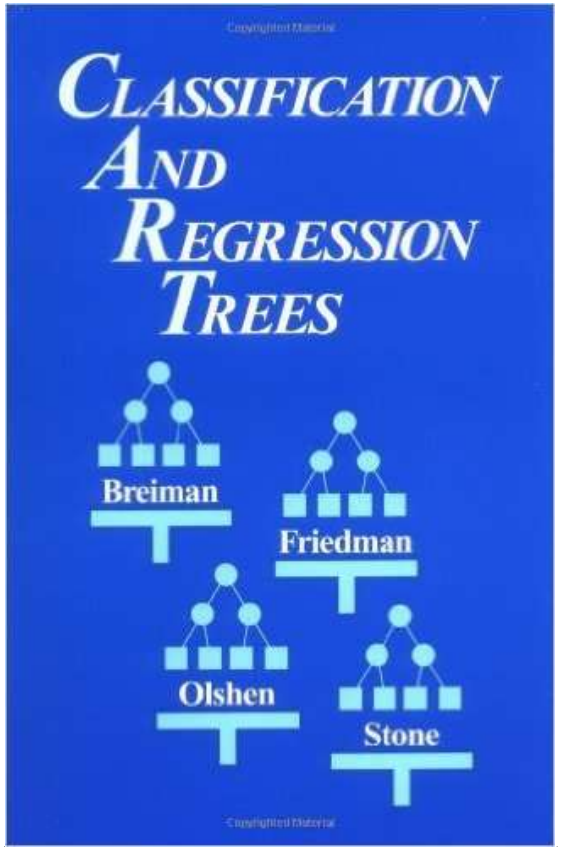
…C4.5: Ejemplo

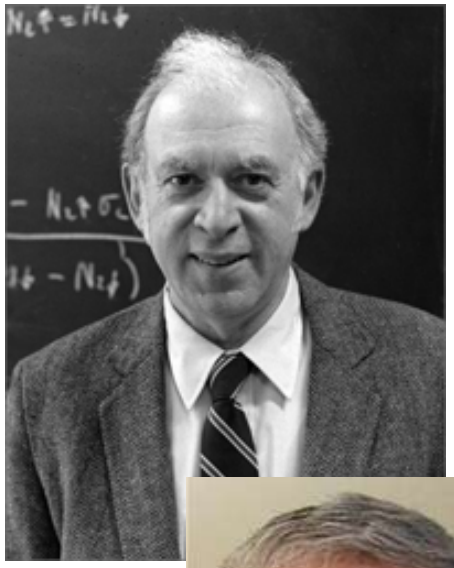

Weka Classifier Tree Visualizer: 12:38:21 - trees.J48 (cr...
Tree View
edad
= youth
= middle_aged
= senior
estudiante
yes (4.0)
calificacion_credito
= no
= yes
= fair
= excellent
no (3.0)
yes (2.0)
yes (3.0)
no (2.0)

Árboles CART

- 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**.







…Árboles CART

- 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 nodos 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.
- Estos árboles pueden manipular fácilmente variables numéricas y/o categóricas.
- Son robustos a la presencia de **outliers** y al trabajar solo con particiones binarias son **fáciles de interpretar**.
- Buscan 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** (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 **n distintos** valores discretos, **$\{a_1, a_2, \ldots, a_n\}$.**
  - Para determinar la **mejor partición** binaria sobre **A**, es necesario examinar **todos los posibles subconjuntos** que pueden formarse usando los valores de **A**.
  - Cada subconjunto **S_A**, se considera como una prueba binaria sobre el atributo **A**, tomando la forma **"¿A ∈ S_A?".**
  - Dado que **A** tiene **n distintos valores**, tendríamos entonces **2^n** posibles subconjuntos: 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.
- Cuando se considera una partición binaria, es necesario calcular una **suma ponderada** de la impureza de cada partición resultante.

…CART: GINI Index

- Si una partición binaria sobre **A** divide a **D** en $\mathbf{D_1}$ y $\mathbf{D_2}$, el **GNI Index** de cada partición estará 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: reducción de impureza

- 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	joven	alto	no	favorable	no
2	joven	alto	no	excelente	no
3	adulto	alto	no	favorable	si
4	a_mayor	medio	no	favorable	si
5	a_mayor	bajo	si	favorable	si
6	a_mayor	bajo	si	excelente	no
7	adulto	bajo	si	excelente	si
8	joven	medio	no	favorable	no
9	joven	bajo	si	favorable	si
10	a_mayor	medio	si	favorable	si
11	joven	medio	si	excelente	si
12	adulto	medio	no	excelente	si
13	adulto	alto	si	favorable	si
14	a_mayor	medio	no	excelente	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 **mejor 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:
    - ✓ **{bajo, medio, alto}**
    - ✓ **{bajo, medio}**
    - ✓ **{bajo, alto}**
    - ✓ **{medio, alto}**
    - ✓ **{bajo}**
    - ✓ **{medio}**
    - ✓ **{alto}**
    - ✓ **{ }**

…Ejemplo: GINI Index

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

¿ingreso ∈ {bajo,medio}?

ID	edad	ingreso	estudiante	calificacion_credito	comprar_computadora
1	joven	alto	no	favorable	no
2	joven	alto	no	excelente	no
3	adulto	alto	no	favorable	si
4	a_mayor	medio	no	favorable	si
5	a_mayor	bajo	si	favorable	si
6	a_mayor	bajo	si	excelente	no
7	adulto	bajo	si	excelente	si
8	joven	medio	no	favorable	no
9	joven	bajo	si	favorable	si
10	a_mayor	medio	si	favorable	si
11	joven	medio	si	excelente	si
12	adulto	medio	no	excelente	si
13	adulto	alto	si	favorable	si
14	a_mayor	medio	no	excelente	no

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

CART: GINI Index

- De esta forma:

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

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

ID	edad	ingreso	estudiante	calificacion_credito	comprar_computadora
1	joven	alto	no	favorable	no
2	joven	alto	no	excelente	no
3	adulto	alto	no	favorable	si
4	a_mayor	medio	no	favorable	si
5	a_mayor	bajo	si	favorable	si
6	a_mayor	bajo	si	excelente	no
7	adulto	bajo	si	excelente	si
8	joven	medio	no	favorable	no
9	joven	bajo	si	favorable	si
10	a_mayor	medio	si	favorable	si
11	joven	medio	si	excelente	si
12	adulto	medio	no	excelente	si
13	adulto	alto	si	favorable	si
14	a_mayor	medio	no	excelente	no

CART: GINI Index

- Entonces:

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

$$Gini_{ingreso \in \{bajo,medio\}}(D) = \boxed{0.443}$$

- Por otro lado, es fácil notar que:

$$Gini_{ingreso \in \{bajo,medio\}}(D) = Gini_{ingreso \in \{alto\}}(D)$$

CART: GINI Index

$$Gini_{ingreso \in \{bajo,alto\}}(D) =$$

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

$$Gini_{ingreso \in \{bajo,alto\}}(D) = \boxed{0.458} = Gini_{ingreso \in \{medio\}}(D)$$

$$Gini_{ingreso \in \{medio,alto\}}(D) =$$

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

$$Gini_{ingreso \in \{medio,alto\}}(D) = Gini_{ingreso \in \{bajo\}}(D) = \boxed{0.450}$$

- Por lo tanto, **la mejor partición binaria** para el atributo **ingreso** sería **{bajo,medio} (o {alto})**

CART: GINI Index

Realizando las operaciones para los demás atributos:

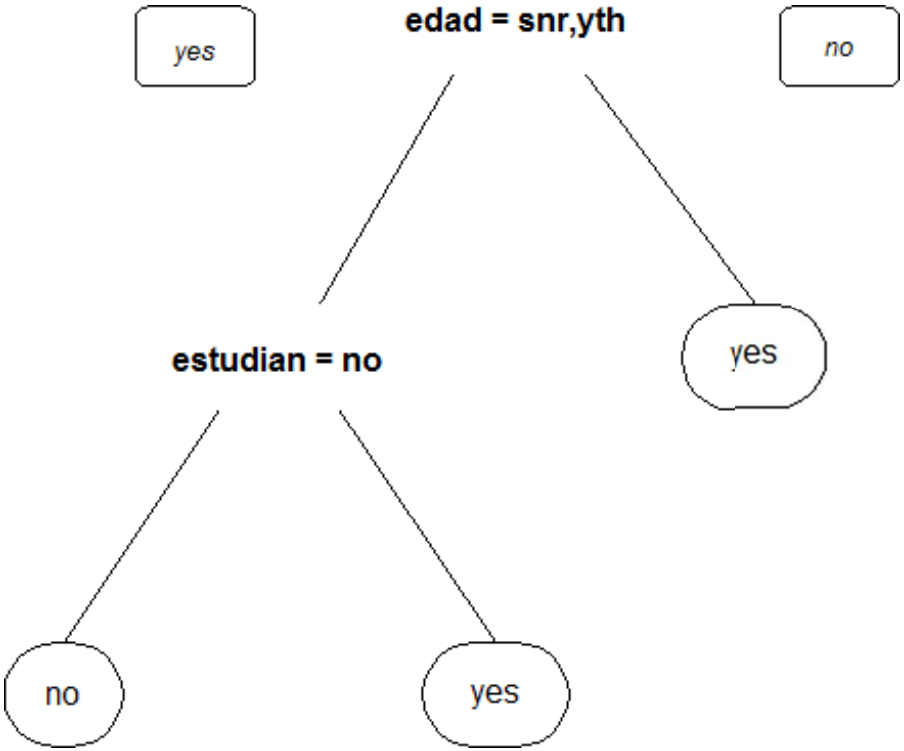
Atributo	Combinación	gini	giniA	delta
Ingreso	{bajo,medio}	0.459	0.443	0.016
	{bajo,alto}	0.459	0.458	0.001
	{medio,alto}	0.459	0.450	0.009
	{bajo}	0.459	0.450	0.009
	{medio}	0.459	0.458	0.001
	{alto}	0.459	0.443	0.016

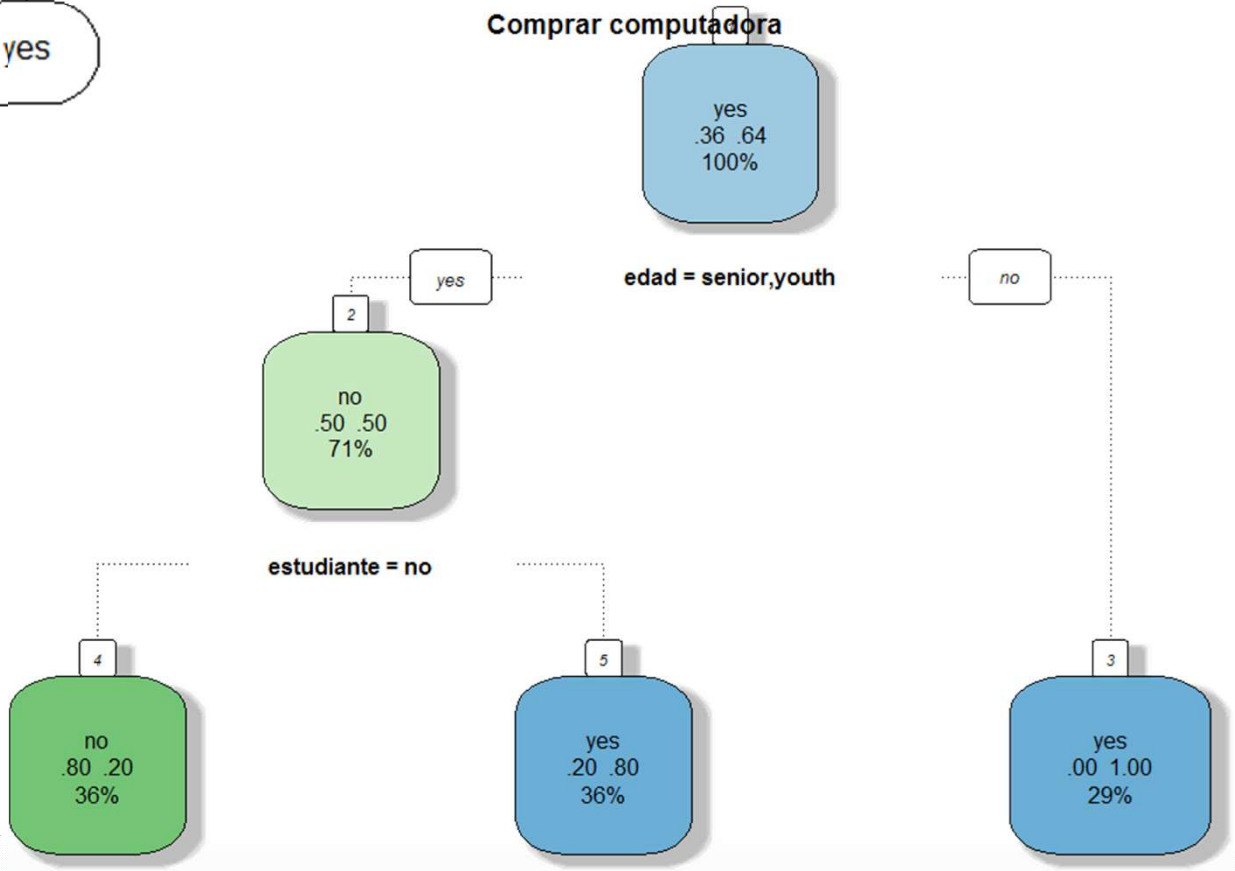
Atributo	Combinación	gini	giniA	delta
Edad	{joven,adulto}	0.459	0.457	0.002
	{joven,a_mayor}	0.459	0.357	0.102
	{adulto,a_mayor}	0.459	0.394	0.066
	{joven}	0.459	0.394	0.066
	{adulto}	0.459	0.357	0.102
	{a_mayor}	0.459	0.457	0.002

Atributo	Combinación	gini	giniA	delta
Califi_cred	{favorable}	0.459	0.429	0.031
	{excelente}	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.*

…Pre-poda

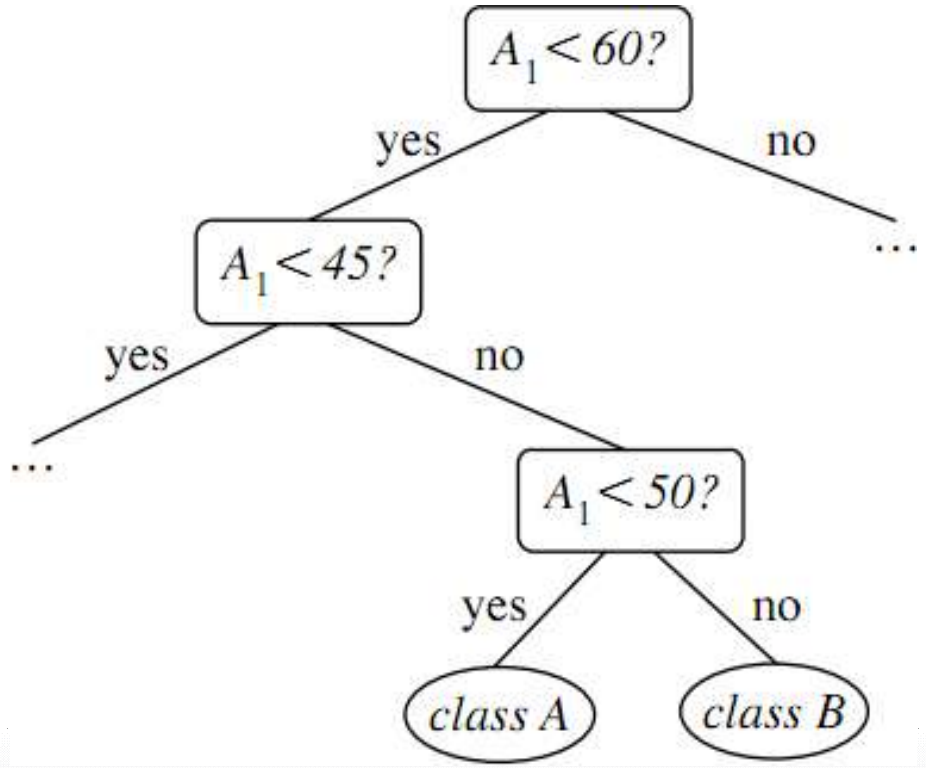
- Medidas como la ***significancia estadística, ganancia de información, GINI index***, se utilizan para asegurar que una partición sea correcta.
- Si una partición de tuplas en un nodo pudiera resultar en una partición que cae por debajo de un umbral especificado previamente, entonces las particiones 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

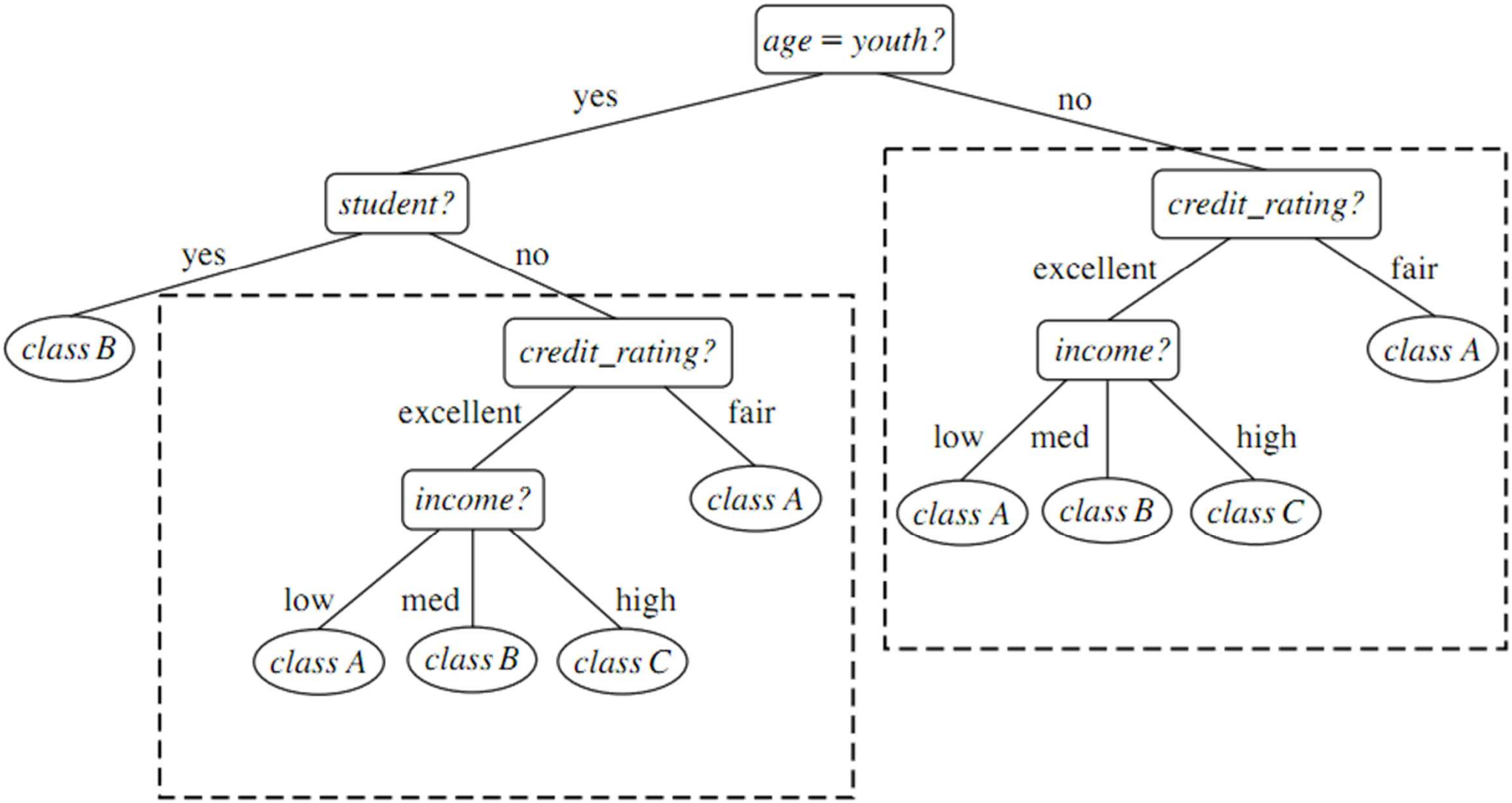
- Es el enfoque que se utiliza con mayor frecuencia.
- Su objetivo es remover sub-árboles de un árbol que ha crecido mucho:
  - Permite que los datos se **sobreajusten** y después se poda reemplazando sub-árboles por una hoja.
  - Para podar un sub-árbol en un nodo dado, se retiran todas sus ramas y 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).

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




¡Gracias!