

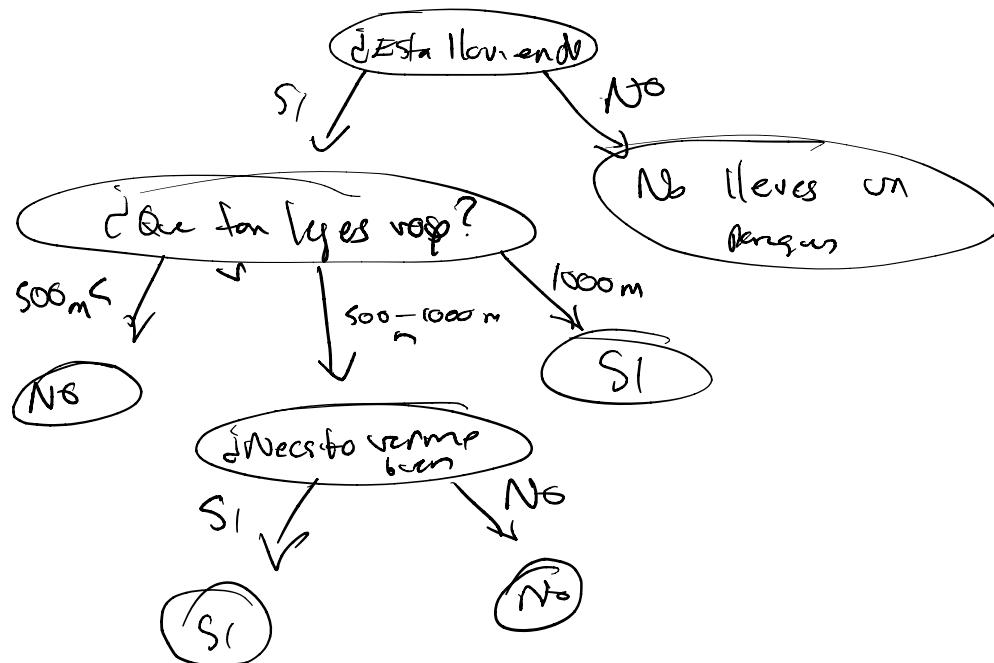
# **"Árboles de Decisión y Random Forest: Fundamentos, Ejemplos y Aplicaciones Reales"**

# ¿Qué es un Árbol de Decisión?

- **Definición básica:** Es una estructura jerárquica de datos no paramétricos que implementa la estrategia de divide y vencerás, utiliza decisiones para clasificar o predecir resultados.
- **Estructura del árbol:** Nodos, hojas y ramas.
- **Ejemplos de uso:** Diagnóstico médico, análisis de riesgo, segmentación de clientes.

necesariamente

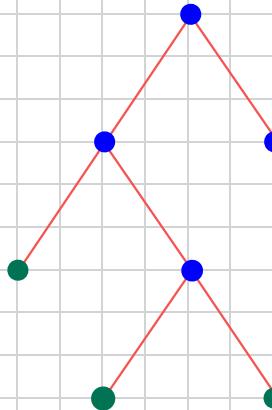
¿ Debo llevar un paraguas?



# Componentes Clave

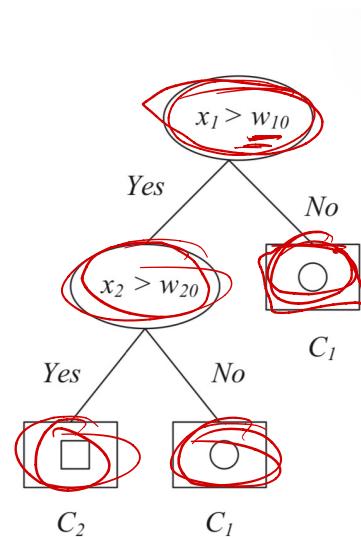
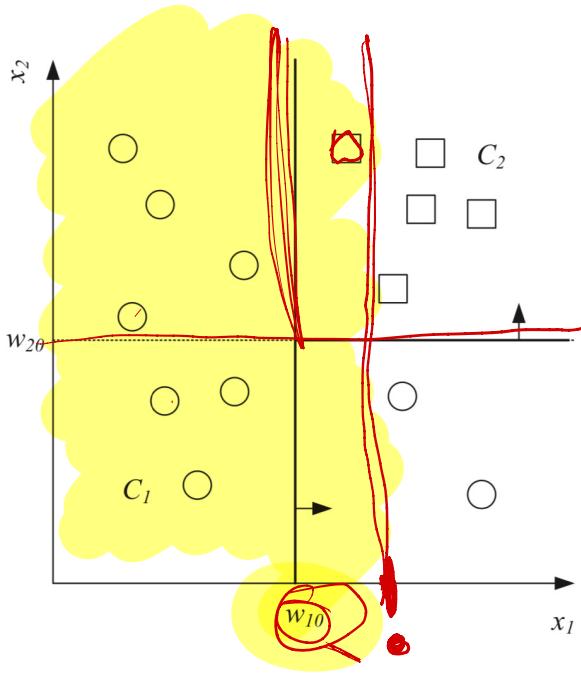
Nodos  
de  
Decisión  
 $f_m(x)$

- **Raíz del Árbol:** El nodo inicial que contiene todo el conjunto de datos.
- **Nodos Internos:** Representan pruebas o decisiones basadas en características de los datos.
- **Hojas (Nodos Terminales):** Los resultados finales (predicciones).
- **Ramas:** Conectan los nodos y muestran el flujo de decisiones.



# Construcción de un Árbol de Decisión

- **Selección de la característica:** ¿Cuál característica dividirá el nodo?
- **Criterio de división:** Medidas comunes como Gini, entropía, y reducción de varianza.
- **División recursiva:** El proceso se repite hasta que se cumple un criterio de detención (profundidad máxima, mínimo de muestras por hoja, etc.).



¿Cuál es la entropía?

$$N = 14$$

$$N = 9 \quad N^2 = 5$$

$$P(C_1 | x_{i,1}) = \frac{9}{14}$$

$$P(C_2 | \cdot) = \frac{5}{14}$$

$$\begin{aligned} I &= - \left( \frac{9}{14} \log_2 \frac{9}{14} + \frac{5}{14} \log_2 \frac{5}{14} \right) \\ &= 1.152 \end{aligned}$$

$$\approx 0.94$$

Ejemplo de un conjunto de datos y su árbol de decisión correspondiente.

# Pros y Contras de los Árboles de Decisión

- **Ventajas:**
  - Fácil de entender e interpretar.
  - Requiere poca preparación de los datos.
- **Desventajas:**
  - Propenso a sobreajuste (overfitting).
  - Puede ser inestable con cambios en los datos.

# Introducción a Random Forest

- **Definición:** Random Forest es un conjunto de árboles de decisión que operan como un modelo de conjunto.
- **Concepto clave:** Bagging (Bootstrap Aggregating).
- Mejora la precisión y reduce el riesgo de sobreajuste.

# Construcción de un Random Forest

- **Muestras aleatorias:** Cada árbol se entrena con una muestra diferente del conjunto de datos.
- **Selección aleatoria de características:** Solo un subconjunto de características se considera para dividir en cada nodo.
- **Agregación:** Se toman las predicciones de todos los árboles y se promedian (regresión) o se vota (clasificación).

# Ventajas de Random Forest y Aplicaciones Comunes

- **Ventajas:**

- Mayor precisión que un solo árbol.
- Menor riesgo de sobreajuste.
- Robusto frente a ruido y datos faltantes.

- **Aplicaciones:**

- Detección de fraudes.
- Reconocimiento de voz.
- Clasificación de imágenes.

# Árboles de Decisión vs Random Forest

- **Simplicidad vs Complejidad:** Un árbol es más simple y fácil de interpretar, mientras que Random Forest es más complejo pero más preciso.
- **Propensión al sobreajuste:** Los árboles de decisión pueden sobreajustar fácilmente, mientras que Random Forest tiende a generalizar mejor.

# Resumen

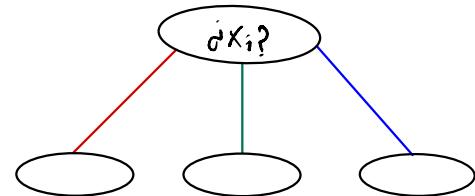
- Los árboles de decisión son una herramienta poderosa pero con limitaciones.
- Random Forest es una mejora significativa que aborda muchas de estas limitaciones.
- Importancia de elegir el modelo adecuado según el problema y los datos disponibles.

## Árboles univariados

En cada nodo interior, la prueba usa solamente una de las dimensiones de entrada (un único atributo).

Si  $x_j$  es la entrada usada y toma uno de  $n$  valores posibles, entonces el nodo de decisión checa el valor de  $x_j$  y toma la rama correspondiente, implementando una división de  $n$  caminos

$$x_j = \text{Color} \in \{\text{Rojo}, \text{Verde}, \text{Azul}\}$$



Si  $x_j$  es un valor numérico la prueba es una comparación

$$f_m(x) : x_j > w_{m0}$$

El nodo de decisión divide en dos el espacio de entrada (división binaria):

$$L_m = \{x | x_j > w_{m0}\}, R_m = \{x | x_j \leq w_{m0}\}$$



$x$  = input

$y$  = output

$r$  = salida respuesta ,  $R$  = número de salidas (clases)

$N$  = número de instancias de entrenamiento

$C_i$  = Clase  $i$

$X$  = muestra de entrenamiento

$\{x^t\}_{t=1}^N$  = conjunto de  $x$  con índice  $t$  de 1 a  $N$

$\{(x^t, r^t)\}_t$  = conjunto de pares ordenados de entradas y salidas deseadas  
con índice  $t$

# Árboles de clasificación

Son árboles de decisión usados para clasificar instancias en diferentes clases.

**Impureza de la División:** Al crear divisiones en el árbol (es decir, al dividir los datos en subconjuntos), es importante evaluar cuán buena es esa división.

La medida de impureza indica cuán "mezcladas" están las clases en un nodo. Si, después de una división, todas las instancias en una rama pertenecen a la misma clase, la división es "pura".

Una mayor pureza implica que la división es mejor, ya que las instancias están claramente separadas por clases.

## Número de Instancias:

- $\underline{N_m}$  es el número total de instancias que llegan a un nodo específico  $m$ .
- $\underline{N_m^i}$  es el número de instancias en el nodo  $m$  que pertenecen a la clase  $C_i$

$$\sum_i^K N_m^i = \underline{N_m}$$

Número de  
instancias de entramado

- En el nodo raíz (que es el nodo inicial del árbol), el número total de instancias es  $\underline{N}$

$$m \bullet \leftarrow A = \underbrace{\{a_1, \dots, a_{N_m}\}}_{a_1 = (a_{1,1}, a_{1,2}, \dots, a_{1,d})}$$

$$N_m^i \text{ pertenecen} \quad N_m = \# A$$

$$a \in C_i \text{ clase} \quad A = C_1 \cup C_2 \cup \dots \cup C_K, \quad C_i \subseteq C_i$$

$$N_m^i = \# C_i$$

**Probabilidad de Clase:** Si una instancia llega a un nodo m, la probabilidad de que pertenezca a la clase  $C_i$  se estima como

$$P(C_i|x, m) \equiv p_m^i = \frac{N_m^i}{N_m}$$

Si,  $p_m^i = 1$   
ent.  $p_m^i = 0$  si

Todos los sustancias  
en el nodo m  
pertenece a la  
clase  $C_1$

Un nodo m es puro si  $p_m^i$  es 0 o 1

**Entropía:** Función para medir la impureza en un nodo m

$$\text{Entropía } I_m = - \sum_{i=1}^K p_m^i \log_2 p_m^i$$

Ya no necesitamos  
seguir dividiendo  
agregamos una  
nueva

donde  $0 \log_2 0 \equiv 0$

$$\frac{1}{2} = \frac{1}{2} (\log_2 \frac{1}{2}) = -1$$

Ejemplo con dos clase  $C_1$  y  $C_2$

$$\begin{aligned} P^1 &= P & P^2 &= (1-P) \\ I &= -P \log_2 P - (1-P) \log_2 (1-P) \\ &= - \end{aligned}$$

# Entropía

La entropía mide cuántos bits son necesarios, como mínimo, para codificar la clase a la que pertenece una instancia en un conjunto de datos.

**Mínimo de bits:** El número mínimo de bits necesarios para codificar una clase está directamente relacionado con la probabilidad de que una instancia pertenezca a esa clase. Si la clase de una instancia es muy probable (predecible), se necesita menos información (menos bits) para codificarla. Si hay más incertidumbre (la clase es menos predecible), se necesitarán más bits.

**Uso en árboles de decisión:** En el contexto de un árbol de decisión, la entropía se usa para medir la impureza o la incertidumbre en un nodo. Una entropía más alta indica que las instancias en ese nodo están distribuidas entre muchas clases de manera poco clara, mientras que una entropía baja significa que las instancias pertenecen principalmente a una sola clase.

Otras funciones de impureza para cuando  $K = 2$

$p^1 \equiv p, \quad p^2 \equiv 1 - p, \quad \phi(p, 1 - p)$  es una función de medida de impureza de una partición si satisface las siguientes propiedades

- $\phi(1/2, 1/2) \geq \phi(p, 1 - p), \forall p \in [0, 1]$
- $\phi(0, 1) = \phi(1, 0) = 0$
- $\phi(p, 1 - p)$  es creciente en  $p \in [0, 1/2]$   
y decreciente en  $p \in [1/2, 1]$

Entropía:  $\phi(p, 1 - p) = -p \log_2 p - (1 - p) \log_2 (1 - p)$

Índice de Gini:  $\phi(p, 1 - p) = 2p(1 - p)$

Misclassification error:  $\phi(p, 1 - p) = 1 - \max(p, 1 - p)$

Error de mala clasificación)

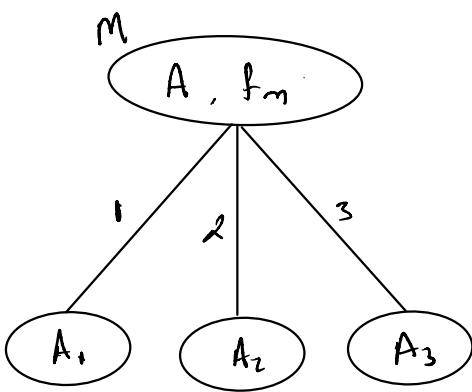
Si  $M$  no es puro, entonces las instancias se deben dividir para disminuir la impureza.  
Hay múltiples atributos sobre los cuales podemos dividir, y para un atributo numérico  
hay varias posiciones de división.

¿Respecto a qué atributo me conviene dividir?

Si este atributo es numérico, ¿Cuál es el mejor umbral?

Respecto al atributo (y umbral) que tras dividir

genere menor impureza.



$$A = \{a_1, \dots, a_{N_m}\}, \quad N_m = |A|$$

$$A_j = \{x \in A : f_m(x^b) = j\}, \quad N_{mj} = |A_j|$$

Obs. Para un atributo discreto con  $n$  valores, hay  $n$  salidas; para uno numérico,  $n=2$ .

$$A_j = C_{1j} \cup C_{2j} \cup \dots \cup C_{kj}$$

$$C_{ij} \subseteq C_{ej}$$

$$\sum_{i=1}^n N_{mj} = N_m$$

$C_{ej}$

$$C_{ij} = \{x \in A_j : x \in C_{ij}\}, \quad N_{mj}^i = |C_{ij}|,$$

$$\sum_{i=1}^k N_{mj}^i = N_{mj}$$

y

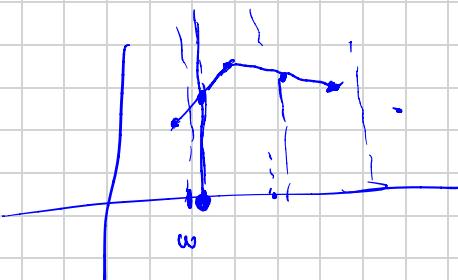
$$\sum_{j=1}^n N_m^i = N_m^i$$

$x$  llega al nodo  $m$ ,  $f_m(x) = j$ , el estimado para la probabilidad de la clase  $C_i$ ,

J toma los valores

que puede tomar  
la variable

$$\hat{P}(C_i|x, m, j) = p_{mj}^i = \frac{N_{mj}^i}{N_{mj}}$$



La impureza total después de una partición está dada por

$$\mathcal{I}'_m = - \sum_{j=1}^n \frac{N_{mj}}{N_m} \sum_{i=1}^K p_{mj}^i \log_2 p_{mj}^i$$

Obs. En el caso de un atributo numérico, es necesario conocer el umbral  $w_m$ , para calcular  $p_{mj}^i$ . Entre  $N_m$  datos hay  $N_m - 1$  posibilidades para  $w_m$ . Hacemos la prueba usando el punto medio de entre cada dato.

Sup.	Edad	close
25		0
30		0
35		1
40		1
45		1

El algoritmo probará entre los puntos medios

25 y 30

30 y 35

35 y 40

40 y 45

y calculará la pureza en cada caso.

(Uso de entropía o Gini). La mejor pureza se obtiene al dividir entre 30 y 35,

esa será la división óptima para el atributo Edad.

Hago a un do m, calculo la impureza.

- Si el nodo es puro, paramos
- S. no, dividimos: Usamos el atributo (y umbral) que minimice la impureza ( $G_m$ , Entropía)

La construcción del árbol continúa recursivamente y en paralelo para todas las ramas que no son puras, hasta que todos los nodos sean puros.

CART (Breiman 1984)

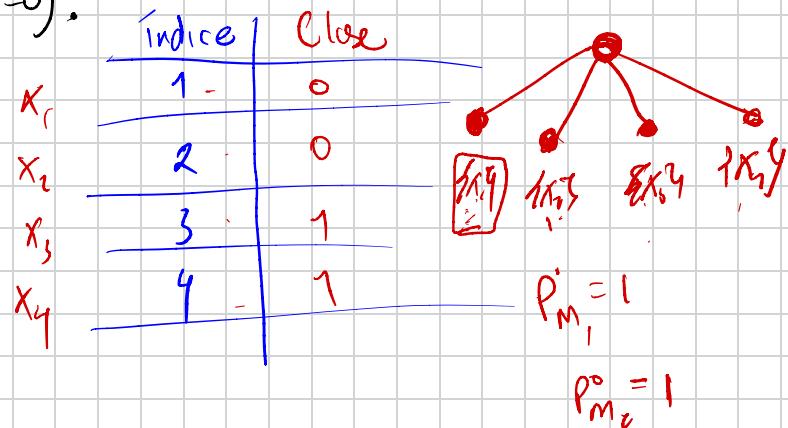
Algoritmos que se basan en costo: ID3 (Quinlan 1986)

Clasificación y regresión

C4.5 (Quinlan 1993)

Problema: Tal partición favorece a atributos con muchos valores: cuando hay muchas variables, hay muchas ramas y la impureza puede ser mucho menor.

Por ejemplo, si turnamos el índice del dato de entrenamiento como un atributo, entonces la medida de impureza elegirá este atributo porque la impureza en cada rama es cero ( $P_{mj}^i = 1$ ,  $\text{Log}_2 P_{mj}^i = 0$ ).



# Árboles de regresión

$$= f(x^t, r^t)^t$$

Para un nodo  $m$ ,  $X_m \subseteq X$  que llegan al nodo  $m$ ; es decir es el conjunto de todos  $x \in X$  que satisfacen todas las condiciones en los nodos de decisión en el camino de la raíz hasta el nodo  $m$ .

Definimos

$$b_m(x) = \begin{cases} 1 & \text{si } x \in X_m : x \text{ llega al nodo } m \\ 0 & \text{en otro caso.} \end{cases}$$

En regresión, el qué tan buena es una partición se mide por el error cuadrático medio de los valores estimados. Supón que  $g_m$  es el valor estimado en un nodo  $m$

$$\sum_m = \frac{1}{N_m} \sum_t (r^t - g_m)^2 b_m(x^t)$$

donde  $N_m = \sum_t b_m(x^t) = |X_m|$ .

$$g_m = \frac{\sum_t b_m(x^t) r^t}{\sum_t b_m(x^t)}$$

O (Si hay mucho ruido)  $g_m$  = Mediana.

Si  $r^t \in \{0, 1\}$   
 $000111$

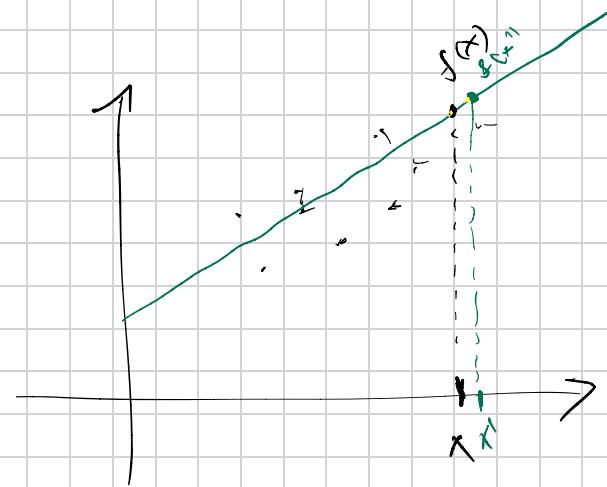
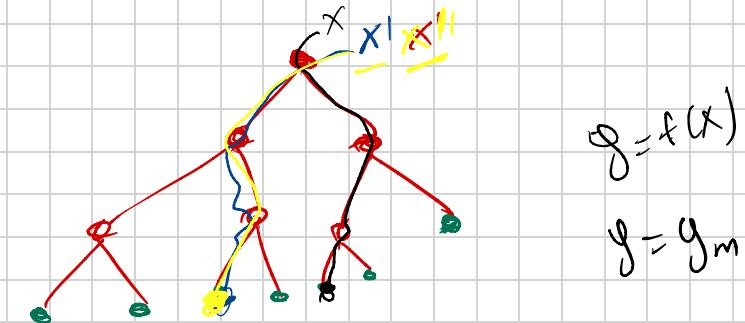
$$\frac{3}{6} = \frac{1}{2}$$

- Si en un nodo el error es aceptable, i.e.,  $E_m < \underline{\theta_r}$ , entonces se crea una hoja con el valor  $g_m$ .

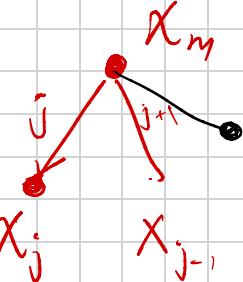
- Si el error no es aceptable, los datos que llegan al nodo m se dividen de tal forma que la suma de los errores en las ramas sea mínima.

- Buscamos el atributo (y umbral para variable numérica) que minimice el error

- Continuamos recursivamente.



- $X_{m_j} \subseteq X_m$  que forman la rama  $j$ :  $\bigcup_{j=1}^n X_{m_j} = X_m$ .



- $b_{mj}(x) = \begin{cases} 1 & \text{si } x \in X_{mj} \\ 0 & \text{de otro modo} \end{cases}$

- $g_{mj}$  es el valor estimado en la rama  $j$  del dato  $m$ :

$$g_{mj} = \frac{\sum_t b_{mj}(x^t) r^t}{\sum_t b_{mj}(x^t)}$$

- Error después de la partición:

$$\hat{E}_m = \frac{1}{N_m} \sum_j \sum_t (r^t - g_{mj})^2 b_{mj}(x^t).$$

Otra función para medir el error:

$$E_m = \max_j \max_t |r^t - g_{mj}| b_{mj}(x^t)$$

Obs. El umbral de error aceptable es el parámetro de complejidad;

Si este es pequeño se generan órboles grandes y se corre el riesgo de sobreajustar; cuando es grande, subajustamos y suavizamos demasiado.

Obs. El modelo se puede mejorar usando Regresión lineal en las hojas:

$$g_m(x) = w_m^T x + w_{mo}$$

# Poda (Pruning)

Técnica utilizada para reducir el tamaño de un árbol que ha crecido demasiado, con el fin de mejorar su capacidad de generalización y evitar el sobreajuste.

## Poda Previa (Pre-pruning):

- Se detiene la construcción del árbol antes de que alcance su máximo crecimiento. Esto se hace estableciendo un criterio de parada, como un umbral de impureza o un número mínimo de instancias por nodo.
- La poda previa previene que el árbol crezca demasiado y se ajuste en exceso a los datos de entrenamiento, lo que reduce el riesgo de sobreajuste.
- Si se aplica demasiado temprano, puede resultar en subajuste.

## Poda Posterior (Post-pruning):

- El árbol se construye completamente hasta su máximo crecimiento, y luego se eliminan retrospectivamente los nodos menos significativos. Esto se hace colapsando ramas o eliminando nodos que no mejoran significativamente el rendimiento del árbol.
- La poda posterior permite crear un árbol más pequeño y generalizable, eliminando nodos que podrían haber sido creados debido a ruido o variaciones irrelevantes en los datos de entrenamiento.
- Un enfoque común es la poda de subárboles completos, donde se reemplaza una rama entera por una hoja si esa simplificación no empeora la precisión del árbol.

## **Ventajas del pruning:**

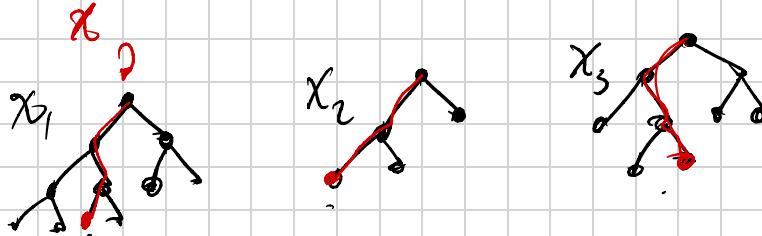
- **Evita el sobreajuste:** Los árboles de decisión sin poda tienden a volverse demasiado específicos para los datos de entrenamiento, lo que les dificulta generalizar con nuevos datos. La poda reduce este riesgo.
- **Mejora la generalización:** Al eliminar nodos innecesarios, el árbol se simplifica, lo que mejora su capacidad para hacer predicciones más precisas en nuevos datos.
- **Reduce la complejidad:** Un árbol podado es más pequeño y menos complejo, lo que facilita su interpretación y reduce los costos computacionales.

## **Criterios comunes para la poda:**

- **Validación cruzada:** Se puede utilizar un conjunto de validación para probar cómo afecta la eliminación de nodos al rendimiento del árbol.
- **Medida de impureza:** Si eliminar una rama aumenta ligeramente la impureza pero no impacta significativamente la precisión del modelo, es una señal de que la poda es apropiada.

# Random Forest.

- $X$  conjunto de datos de prueba
- Tomas subconjuntos de  $X$ , digamos  $\{X_0, \dots, X_N\} \subseteq P(X)$
- creamos un árbol de decisión en base cada  $X_i$
- Genera  $N$  árboles de decisión (de forma aleatoria)



Clasificación: Tomando la media

Regresión: Promedio