Introducción a la Ciencia de Datos

Maestría en Probabilidad y Estadística

Dr. Marco Antonio Aquino López

Centro de Investigación en Matemáticas

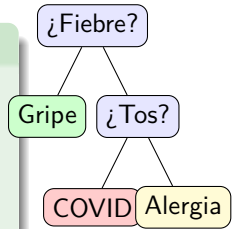
Agosto-Diciembre 2025

¿Qué es un árbol de decisión?

Pensemos en un juego de 20 preguntas

Ejemplo: Diagnosticar una enfermedad

- **Pregunta 1:** ¿Tiene fiebre?
  - **Sí:** ¿Tiene tos?
  - **No:** ¿Tiene dolor de cabeza?
- **Pregunta 2:** ¿La fiebre es mayor a 38°C?
- **Pregunta 3:** ¿Los síntomas empezaron hace más de 3 días?



Idea central

Un árbol de decisión es un **flujograma** que hace preguntas sucesivas para clasificar o predecir.

Partes de un árbol de decisión: Componentes

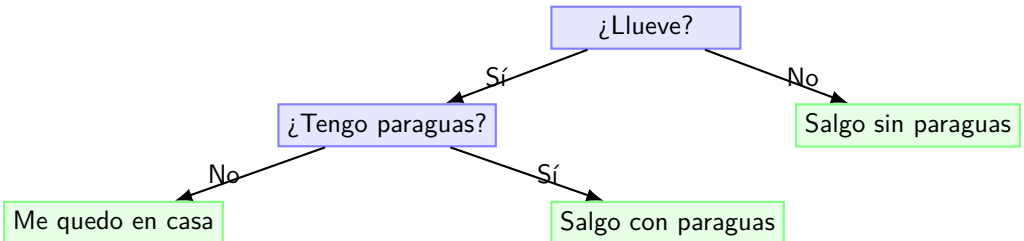
Componentes básicos

- **Nodo raíz:** Primera pregunta del árbol.
- **Nodos internos:** Preguntas intermedias que dividen el conjunto de datos.
- **Ramas:** Respuestas posibles (Sí/No, Mayor/Menor, etc.).
- **Hojas:** Decisiones finales o predicciones del modelo.

Ejemplo simple

- **Problema:** Decidir si salir a caminar.
- **Variables:** Lluvia, paraguas, temperatura.
- **Decisión:** Salir o quedarse en casa.

Partes de un árbol de decisión: Ejemplo visual



Este diagrama muestra la estructura típica de un árbol de decisión: preguntas en los nodos, ramas con respuestas y hojas con decisiones finales.

Árboles de decisión y teoría de decisiones

Perspectiva teórica

Un árbol de decisión representa un proceso secuencial de toma de decisiones bajo incertidumbre. Desde la teoría de decisiones, cada nodo implica:

- **Una decisión** (*acción*) elegida por el agente.
- **Un evento aleatorio** con distintas probabilidades.
- **Un resultado** asociado a una **utilidad esperada**.

Relación con la teoría clásica

- Cada camino desde la raíz hasta una hoja representa una **política de decisión**.
- El objetivo del agente racional es elegir el camino que **maximiza la utilidad esperada** o **minimiza la pérdida esperada**.
- Esta estructura es equivalente al principio de *decisión bayesiana* si se asignan probabilidades y pérdidas.

Componentes formales en la teoría de decisiones

Estructura general

Un árbol de decisión puede describirse mediante:

- Un conjunto de **estados del mundo** $\Theta = \{\theta_1, \theta_2, \ldots\}$.
- Un conjunto de **acciones posibles** $A = \{a_1, a_2, \ldots\}$.
- Una **función de pérdida** $L(a, \theta)$ o de utilidad $U(a, \theta)$.
- Una **distribución de probabilidad** sobre los estados: $p(\theta)$.

Criterio de decisión óptima

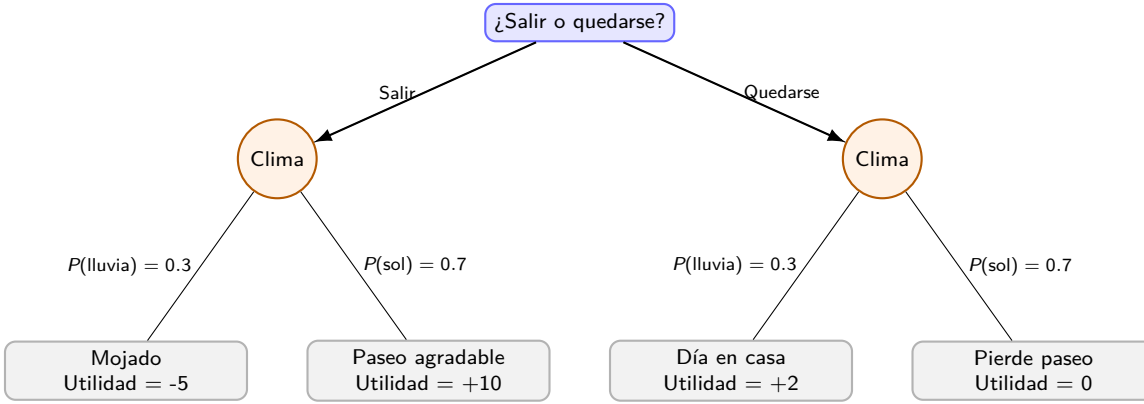
- La acción óptima a^* se elige para minimizar la pérdida esperada:

$$a^* = \arg\min_a \mathbb{E}_\theta[L(a, \theta)]$$

  o equivalentemente maximizar $\mathbb{E}_\theta[U(a, \theta)]$.

- En un árbol de decisión, esta evaluación se realiza **de abajo hacia arriba**, propagando las

Ejemplo: Árbol de decisión bajo incertidumbre



$\mathbb{E}[U_{\text{salir}}] = 0.3(-5) + 0.7(10) = +5.5$

$\mathbb{E}[U_{\text{quedarse}}] = 0.3(2) + 0.7(0) = +0.6$

Decisión óptima: Salir maximiza la utilidad esperada ($5.5 > 0.6$), por lo tanto, un agente racional elige **salir**.

De la teoría de decisiones al aprendizaje automático

Conexión conceptual

Los árboles de decisión en **aprendizaje automático** heredan la lógica de la teoría de decisiones:

- Cada nodo representa una **decisión óptima local** basada en una variable predictora.
- Cada rama describe una **condición observada** en los datos.
- Cada hoja corresponde a una **estimación esperada** — ya sea:
  - una **probabilidad** (clasificación), o
  - un **valor promedio** (regresión).

Del marco teórico al empírico

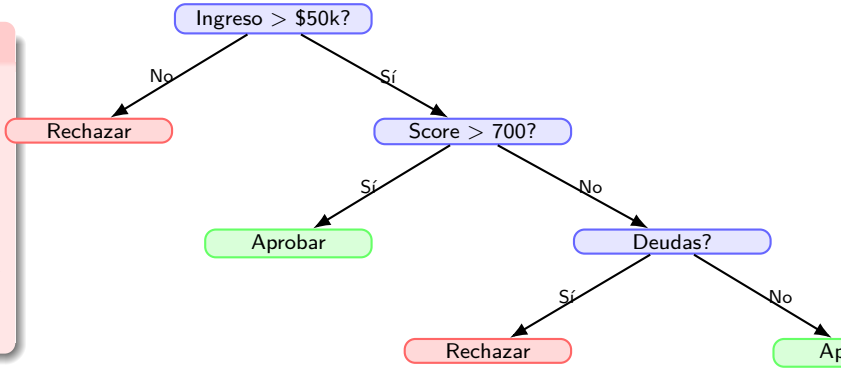
- En teoría de decisiones, se busca **maximizar la utilidad esperada**.
- En aprendizaje automático, se busca **minimizar una función de pérdida** en los datos:
- En ambos casos, el árbol aprende a **tomar decisiones que reducen la incertidumbre.**

Así, un árbol de decisión moderno es la implementación estadística del razonamiento secuencial de la teoría

¿Para qué sirven los árboles de decisión?

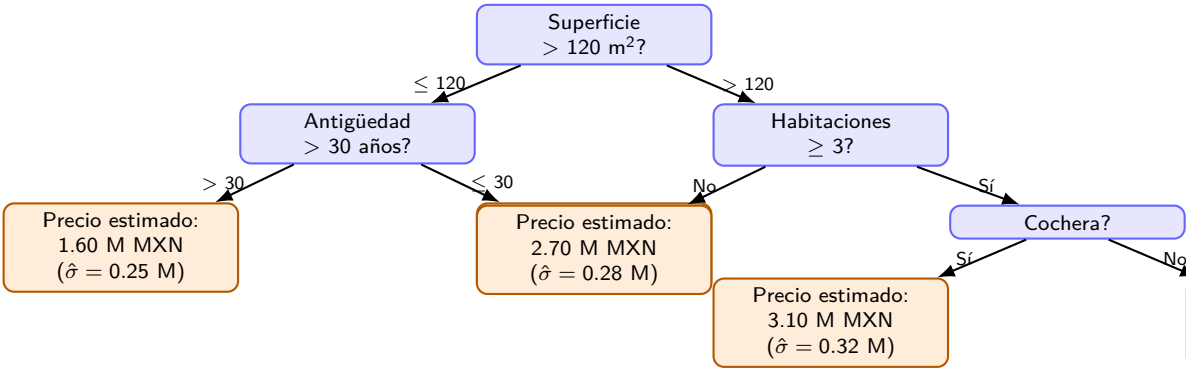
Ventajas principales

- **Fáciles de interpretar.**
- **Manejan variables numéricas y categóricas.**
- **No requieren supuestos distribucionales complejos.**



Ejemplo de árbol de decisión usado para clasificación (aprobación de crédito).

Árbol de decisión para regresión: precio de vivienda



¿Por qué árboles de decisión en estadística?

Del modelo paramétrico al no paramétrico

- **Enfoque clásico:** Modelos lineales, GLMs, supuestos distribucionales

$$Y = \beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p + \varepsilon$$

- **Limitaciones:** Interacciones complejas, no linealidades, heterocedasticidad
- **Árboles:** Aproximación **no paramétrica** por funciones constantes a trozos

$$\hat{f}(x) = \sum_{m=1}^{M} c_m \cdot \mathbb{I}\{x \in R_m\}$$

Ventajas estadísticas

- Captura automáticamente interacciones entre variables
- Robustez a outliers y valores faltantes
- Interpretabilidad: reglas *si-entonces* naturales
- Base para métodos ensemble (Random Forests, Gradient Boosting)

Motivación

- Buscamos un estimador $\hat{f}$ que prediga Y a partir de $X \in \mathcal{X}$.
- Los árboles de decisión inducen una **partición recursiva** de $\mathcal{X}$:

$$\mathcal{X} = \bigcup_{m=1}^{M} R_m, \quad R_m \cap R_{m'} = \emptyset.$$

- En cada región R_m, el modelo es *constante*:

$$\hat{f}(x) = c_m \quad \text{si } x \in R_m.$$

- En regresión: $c_m = \frac{1}{|R_m|} \sum_{i: x_i \in R_m} y_i$.
- En clasificación: $c_m = \arg\max_k p_{mk}$ donde p_{mk} es la proporción de clase k en R_m.

Motivación: el problema de clasificación

Objetivo general

Queremos construir una regla $\hat{f}: \mathcal{X} \to \{1, \ldots, K\}$ que minimice el riesgo esperado:

$$R(f) = \mathbb{E}[\mathbb{I}\{f(X) \neq Y\}].$$

- El clasificador de Bayes $f^*(x) = \arg\max_k P(Y = k \mid X = x)$ es óptimo.
- En la práctica, $P(Y \mid X)$ es desconocida.
- Los árboles buscan una aproximación empírica a f^* mediante particiones sucesivas del espacio $\mathcal{X}$.
- Idea: aproximar regiones donde una clase domina de forma clara.

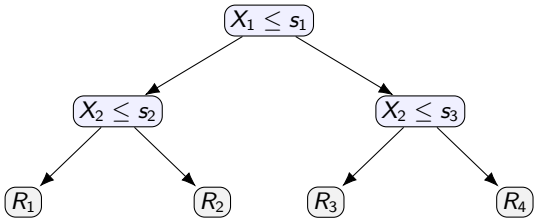
Motivación geométrica

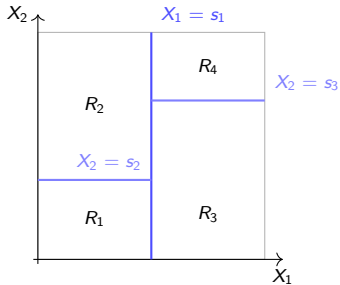
- La frontera de decisión $f^*(x)$ separa regiones de distinta clase.
- Un árbol genera una **aproximación poligonal a trozos** de esa frontera:

$$\mathcal{X} = \bigcup_{m=1}^{M} R_m, \quad \hat{f}(x) = c_m \text{ si } x \in R_m.$$

- Cada R_m es un **rectángulo axis–aligned** (hiperrectángulo).
- Al aumentar el número de divisiones, la frontera se ajusta mejor.

Árbol binario y partición inducida en $\mathbb{R}^2$

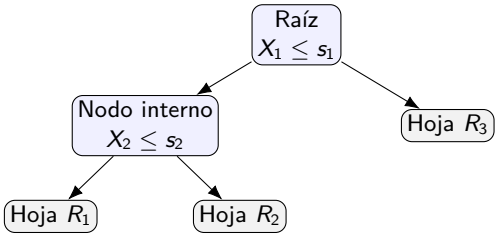




- Cada hoja R_m define una región del espacio.
- La etiqueta de la hoja: clase mayoritaria en R_m.

- El árbol induce hiperrectángulos *axis-aligned*.
- La frontera de decisión se aproxima a trozos.

Estructura de un árbol de decisión



Ejemplo: Un árbol con una raíz, un nodo interno y tres hojas.

Componentes principales

- **Nodo raíz:** contiene todos los datos; representa el espacio completo $\mathcal{X}$.
- **Nodo interno:** define una condición de partición (p. ej. $X_j \leq s$).
- **Hoja (o nodo terminal):** conjunto de observaciones que no se subdividen más. Cada hoja corresponde a una región $R_m \subset \mathcal{X}$.

Regla inducida

$$\mathcal{X} = \bigcup_{m=1}^{M} R_m, \quad R_m \cap R_{m'} = \emptyset.$$

Cómo se construye un árbol de decisión

Idea central

Un árbol se construye **recursivamente**: cada nodo divide el espacio de entrada en dos subregiones más homogéneas en términos de la variable respuesta.

1. Comenzar con todos los datos en el **nodo raíz**.
2. Evaluar todas las posibles divisiones (X_j, s):

$$R_L(j, s) = \{x : x_j \leq s\}, \quad R_R(j, s) = \{x : x_j > s\}.$$

3. Escoger la división que maximiza la **disminución de impureza**.
4. Repetir el proceso en los nodos hijos hasta cumplir un criterio de parada:
   - tamaño mínimo de hoja,
   - profundidad máxima, o
   - impureza mínima.
5. Etiquetar cada hoja R_m con la clase o valor promedio correspondiente.

Concepto de impureza

En un nodo t, definimos las proporciones de clases

$$p_{k|t} = P(Y = k \mid X \in t), \quad k = 1, \ldots, K.$$

Una medida de impureza $I(t)$ debe:

- Ser mínima cuando todas las observaciones pertenecen a una sola clase ($p_{k|t} \in \{0, 1\}$).
- Ser máxima cuando las clases están equiprobables ($p_{k|t} = 1/K$).
- Ser simétrica respecto a las clases.

Ejemplo (dos clases):

$$p_{1|t} = p, \quad p_{2|t} = 1 - p.$$

Queremos una función $I(p)$ que mida la "mezcla" entre ambas.

Ejemplos de funciones de impureza

Criterio	Expresión	Nombre común
$1 - \max_k p_{k\|t}$	Error de clasificación	"Misclassification"
$-\sum_k p_{k\|t} \log p_{k\|t}$	Entropía	"Information gain"
$\sum_k p_{k\|t}(1 - p_{k\|t})$	Índice de Gini	"Gini impurity"

Ejemplo binario:

$$I_{\text{Gini}}(p) = 2p(1-p), \quad I_{\text{Entropía}}(p) = -p\log p - (1-p)\log(1-p).$$

Ambas son máximas en $p = 0.5$ y nulas en $p = 0, 1$.

Criterio de división óptima

En cada nodo t, probamos posibles divisiones (X_j, s) que separan los datos en:

$$t_L = \{x : x_j \leq s\}, \quad t_R = \{x : x_j > s\}.$$

La **ganancia de impureza** se define como:

$$\Delta I(s, j) = I(t) - p_L I(t_L) - p_R I(t_R),$$

donde $p_L = N_L/N_t$ y $p_R = N_R/N_t$.

Elegimos la división que **maximiza** $\Delta I(s, j)$.

$$(X_j^*, s^*) = \arg \max_{(j,s)} \Delta I(s, j).$$

Ejemplo numérico: criterio de Gini

Supongamos un nodo con 10 observaciones de dos clases:

Clase	1	1	1	2	2	1	2	2	1	2
X_1	2.0	3.1	3.5	4.0	4.3	5.2	5.4	6.1	6.8	7.0

Para cada posible punto de corte s, calculamos:

$$I_{\text{Gini}}(t) = 2p(1-p), \quad \Delta I(s) = I(t) - p_L I(t_L) - p_R I(t_R).$$

El mejor corte es el que maximiza $\Delta I(s)$ (mayor reducción de impureza).

Ejemplo numérico (resuelto): preparación

Datos en el nodo (n=10): cinco de clase 1 y cinco de clase 2.

$$I_{\text{Gini}}(t) = 2p(1-p), \qquad p = \frac{\#\{Y=1\}}{n}.$$

Como $p = 5/10 = 0.5$, entonces $I(t) = 2(0.5)(0.5) = 0.5$.

Cortes candidatos (puntos medios entre X_1 ordenados):

$$\mathcal{S} = \{2.55,\ 3.30,\ 3.75,\ 4.15,\ 4.75,\ 5.30,\ 5.75,\ 6.45,\ 6.90\}.$$

Para cada $s \in \mathcal{S}$:

$$\Delta I(s) \ = \ I(t) \ - \ p_L\, I(t_L) \ - \ p_R\, I(t_R), \quad p_L = \frac{N_L}{n},\ p_R = \frac{N_R}{n}.$$

Ejemplo numérico (resuelto): resultados por punto de corte (1/2)

s	N_L	(n_{1L}, n_{2L})	$I(t_L)$	N_R	(n_{1R}, n_{2R})	$I(t_R)$	p_L	$\Delta I(s)$
2.55	1	(1,0)	0.000	9	(4,5)	0.494	0.10	0.0556
3.30	2	(2,0)	0.000	8	(3,5)	0.469	0.20	0.1250
3.75	3	(3,0)	0.000	7	(2,5)	0.408	0.30	**0.2143**
4.15	4	(3,1)	0.375	6	(2,4)	0.444	0.40	0.0833
4.75	5	(3,2)	0.480	5	(2,3)	0.480	0.50	0.0200

Cálculo de Gini binario: $I = 2p(1-p)$ con $p = \frac{n_1}{n_1+n_2}$. Por ejemplo, para $s = 3.75$:

$$t_L:\ (n_{1L}, n_{2L}) = (3,0) \Rightarrow I(t_L) = 0, \qquad t_R:\ (n_{1R}, n_{2R}) = (2,5) \Rightarrow p = \tfrac{2}{7},\ I(t_R) = 2 \cdot \tfrac{2}{7} \cdot \tfrac{5}{7} = \tfrac{20}{49} \approx 0.408.$$

Entonces $p_L = 0.3$, $p_R = 0.7$ y $\Delta I(3.75) = 0.5 - 0.3(0) - 0.7(0.408) = 0.2143$.

Ejemplo numérico (resuelto): resultados por punto de corte (2/2)

s	N_L	(n_{1L}, n_{2L})	$I(t_L)$	N_R	(n_{1R}, n_{2R})	$I(t_R)$	p_L	$\Delta I(s)$
5.30	6	(4,2)	0.444	4	(1,3)	0.375	0.60	0.0833
5.75	7	(4,3)	0.490	3	(1,2)	0.444	0.70	0.0238
6.45	8	(4,4)	0.500	2	(1,1)	0.500	0.80	0.0000
6.90	9	(5,4)	0.494	1	(0,1)	0.000	0.90	0.0556

Conclusión numérica: el máximo de $\Delta I(s)$ se obtiene en $s^\star = 3.75$ con $\Delta I(s^\star) = 0.2143$.

Regla inducida por el mejor corte y etiqueta de hojas

Con $s^\star = 3.75$:

$$\begin{cases} X_1 \leq 3.75 \Rightarrow t_L \text{ con } (n_{1L}, n_{2L}) = (3, 0) \Rightarrow \text{clase mayoritaria } = 1, \\ X_1 > 3.75 \Rightarrow t_R \text{ con } (n_{1R}, n_{2R}) = (2, 5) \Rightarrow \text{clase mayoritaria } = 2. \end{cases}$$

Regla de decisión (clasificador por mayoría):

$$\hat{f}(x) = \begin{cases} 1, & \text{si } x_1 \leq 3.75, \\ 2, & \text{si } x_1 > 3.75. \end{cases}$$

Comentario teórico: equivalencia de fórmulas de Gini

Para dos clases, las formas usuales del índice de Gini son equivalentes:

$$I_{\text{Gini}} \;=\; 2p(1-p) \;=\; 1 - \sum_{k=1}^{2} p_k^2, \qquad p_1 = p,\; p_2 = 1-p.$$

En multiclase ($K \geq 2$) se usa

$$I_{\text{Gini}} \;=\; 1 - \sum_{k=1}^{K} p_k^2,$$

y la fórmula de ganancia de impureza permanece:

$$\Delta I(s) \;=\; I(t) \;-\; p_L\, I(t_L) \;-\; p_R\, I(t_R).$$

Esto garantiza que una división nunca *aumenta* la impureza ($\Delta I \geq 0$).

Propiedades matemáticas

- Las funciones de impureza son **cóncavas** en p.
- Implica que $\Delta I \geq 0$: una división nunca incrementa impureza.
- Para dos clases, cualquier $I(p)$ cóncava y simétrica genera un árbol consistente.
- En clasificación multiclase, Gini y entropía generalizan naturalmente:

$$I_{\text{Gini}} = 1 - \sum_k p_k^2, \quad I_{\text{Entropía}} = -\sum_k p_k \log p_k.$$

ver código

Sobreajuste (Overfitting) en árboles de decisión

Problema: Los árboles de decisión tienden a crecer hasta capturar todas las particularidades de los datos de entrenamiento.

Causa: El algoritmo de construcción busca siempre la división que maximiza la reducción de impureza, incluso cuando ésta es espuria.

Consecuencia:

- Altas varianzas: pequeñas perturbaciones en los datos generan árboles distintos.
- Pérdida de capacidad predictiva en datos nuevos (error de generalización alto).

Formalmente, si $R(f)$ es el riesgo esperado y $\widehat{R}(f)$ el empírico:

$$R(f) = E_{(X,Y)}[l(Y, f(X))], \quad \widehat{R}(f) = \frac{1}{n}\sum_{i=1}^{n} l(y_i, f(x_i)).$$

El sobreajuste ocurre cuando $\widehat{R}(f)$ es pequeño pero $R(f)$ es grande.

Criterios de paro (pre–pruning): motivación y definición

Motivación: evitar sobreajuste controlando la complejidad del árbol *durante* el crecimiento.

Idea general: detener una división si la ganancia esperada es "pequeña" o si el nodo es "pequeño".

Criterios típicos (clasificación/regresión):

- **Profundidad máxima** $d_{\text{máx}}$: detener si $\text{depth}(t) \geq d_{\text{máx}}$.
- **Tamaño mínimo para dividir** $n_{\text{mín}}^{\text{split}}$: detener si $N_t < n_{\text{mín}}^{\text{split}}$.
- **Tamaño mínimo de hoja** $n_{\text{mín}}^{\text{leaf}}$: proponer división sólo si $N_{t_L}, N_{t_R} \geq n_{\text{mín}}^{\text{leaf}}$.
- **Ganancia mínima de impureza** ε: detener si

$$\Delta I(s, j) \;=\; I(t) - p_L I(t_L) - p_R I(t_R) \;<\; \varepsilon$$

  (o, en regresión, ganancia mínima de SSE/MSE).
- **Número máximo de hojas** $L_{\text{máx}}$: detener cuando $|T| \geq L_{\text{máx}}$.

Efecto estadístico: mayor sesgo, menor varianza $\Rightarrow$ mejor generalización si se eligen bien los umbrales.

Poda (Pruning): Controlar la complejidad del árbol

Idea: Simplificar el árbol para reducir varianza y mejorar generalización.

Cost Complexity Pruning (CART):

$$R_\alpha(T) = R(T) + \alpha|T|,$$

donde $R(T)$ es el error de validación y $|T|$ es el número de hojas.

- $\alpha = 0$: árbol completo (máximo ajuste).
- $\alpha \to \infty$: árbol con una sola hoja (máxima simplicidad).

Algoritmo:

1. Generar una secuencia de árboles anidados $T_0 \supset T_1 \supset \cdots \supset T_K$.
2. Evaluar $R_\alpha(T_k)$ en validación cruzada.
3. Elegir el árbol T_α que minimiza $R_\alpha(T)$.

Algoritmos de construcción de árboles

Existen varios algoritmos con diferencias en los criterios de división y manejo de datos:

- **CART (Breiman et al., 1984):** Usa el índice de Gini o el error cuadrático. Soporta clasificación y regresión.
- **ID3 (Quinlan, 1986):** Utiliza la *ganancia de información* basada en entropía. Sólo para clasificación.
- **C4.5 (Quinlan, 1993):** Extiende ID3 con manejo de valores continuos, missing values y poda.

Comparación:

Algoritmo	Criterio	Tipo de salida
CART	Gini / Varianza	Clasificación / Regresión
ID3	Ganancia de información	Clasificación
C4.5	Ganancia de información normalizada	Clasificación

Ejemplo paso a paso: criterio de Gini

Supongamos un conjunto con dos variables X_1, X_2 y una respuesta binaria Y.

Para cada posible punto de corte s en X_j:

$$\Delta I(s, j) = I(t) - p_L I(t_L) - p_R I(t_R),$$

donde $I(t) = 2p(1-p)$.

Ejemplo:

- Nodo raíz: $p = 0.5$, $I(t) = 0.5$.
- División $X_1 < 4.5$: $I(t_L) = 0.2$, $I(t_R) = 0.4$, con $p_L = 0.6$, $p_R = 0.4$.

$$\Delta I = 0.5 - (0.6)(0.2) - (0.4)(0.4) = 0.22.$$

La mejor división es aquella con mayor ΔI.

Manejo de variables continuas

Para variables continuas X_j, se consideran cortes posibles s entre valores ordenados:

$$s \in \left\{ \frac{x_{(i)} + x_{(i+1)}}{2} : i = 1, \ldots, n-1 \right\}.$$

Cada punto de corte define una partición:

$$R_L = \{x : x_j \leq s\}, \quad R_R = \{x : x_j > s\}.$$

El algoritmo selecciona (X_j, s) que maximiza $\Delta I(s, j)$.

Ventaja: Se exploran todos los posibles umbrales, permitiendo detectar no linealidades.

Manejo de variables categóricas

Para variables categóricas con K niveles $\{c_1, \ldots, c_K\}$:

- El espacio de posibles divisiones crece exponencialmente ($2^{K-1} - 1$ particiones posibles).
- CART busca la mejor partición binaria $A \subset \{c_1, \ldots, c_K\}$ tal que:

$$\Delta I(A) = I(t) - p_L I(t_L) - p_R I(t_R)$$

sea máxima.

Problema: Sesgo hacia variables con muchos niveles (más oportunidades de reducción de impureza).
Solución: Penalización por complejidad o usar árboles honestos (basados en muestras independientes).

Ventajas y limitaciones de los árboles

Ventajas:

- Interpretables: las reglas son fáciles de comunicar.
- No paramétricos: no requieren suposiciones sobre la distribución de los datos.
- Robustez a valores atípicos y missing values.

Limitaciones:

- Alta varianza: inestabilidad ante pequeños cambios.
- Sesgo hacia variables con muchos niveles.
- Fronteras de decisión ortogonales (rectangulares).
- Baja precisión comparada con métodos ensemble.

De árboles individuales a métodos *ensemble*

Motivación: Reducir la varianza de los árboles individuales combinando múltiples modelos.

Bagging (Bootstrap Aggregating):

- Se generan B muestras bootstrap del conjunto de entrenamiento.
- Se ajusta un árbol T_b en cada muestra.
- Predicción final:

$$\hat{f}_{\text{bag}}(x) = \frac{1}{B} \sum_{b=1}^{B} \hat{f}_b(x).$$

Random Forest: Extiende el bagging introduciendo aleatoriedad adicional en la selección de variables en cada división.

Boosting: Combina árboles secuencialmente, ponderando más los errores previos (e.g., Gradient Boosting Machines).

Referencias I


Breiman, L., Friedman, J. H., Olshen, R. A., & Stone, C. J. (1984). *Classification and Regression Trees*. Wadsworth International Group.


Quinlan, J. R. (1986).
Induction of Decision Trees.
Machine Learning, **1**(1), 81–106.


Quinlan, J. R. (1993). *C4.5: Programs for Machine Learning*. Morgan Kaufmann.


Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. 2nd ed. Springer.


James, G., Witten, D., Hastie, T., & Tibshirani, R. (2021). *An Introduction to Statistical Learning with Applications in R*. 2nd ed. Springer.


Loh, W.-Y. (2014). Fifty Years of Classification and Regression Trees. *International Statistical Review*, **82**(3), 329–348.


Breiman, L. (1996). Bagging Predictors. *Machine Learning*, **24**(2), 123–140.

Referencias II


Friedman, J. H. (2001). Greedy Function Approximation: A Gradient Boosting Machine. *Annals of Statistics*, **29**(5), 1189–1232.

Motivación: la inestabilidad de los árboles

- Los árboles de decisión son **modelos de alta varianza**: pequeños cambios en los datos pueden generar estructuras de árbol muy diferentes.
- Este fenómeno se conoce como **inestabilidad** o **sensibilidad al muestreo**.
- Consecuencia: bajo rendimiento en datos nuevos (alto error de generalización).

Ejemplo conceptual

Entrenamos dos árboles sobre muestras distintas del mismo conjunto:

$$T_1(x),\ T_2(x) \Rightarrow f_1(x) \neq f_2(x)$$

Idea clave: Si combinamos sus predicciones, el promedio puede ser más estable.

De la varianza al promedio: idea general

- Si cada árbol tiene varianza σ^2 y correlación ρ entre sí, el promedio de B árboles tiene varianza aproximada:

$$\mathrm{Var}\left[\bar{T}(x)\right] = \rho\sigma^2 + \frac{1-\rho}{B}\sigma^2$$

- Cuando ρ es pequeña y B grande, la varianza se reduce significativamente.

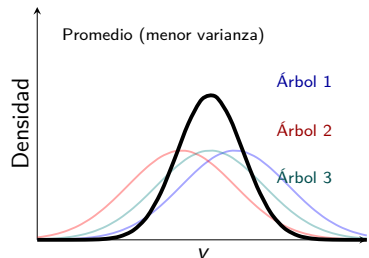


Figura: Promediar modelos reduce varianza.

Bagging: Bootstrap Aggregating

- **Idea:** generar múltiples subconjuntos de entrenamiento mediante remuestreo bootstrap.
- Entrenar un árbol en cada muestra bootstrap:

$$D_b = \{(x_i, y_i)\}_{i=1}^{n_b}, \quad T_b(x) = \text{árbol ajustado en } D_b.$$

- Predicción final (regresión):

$$\hat{f}_{\text{bag}}(x) = \frac{1}{B} \sum_{b=1}^{B} T_b(x)$$

- En clasificación: voto mayoritario.

Bagging: Bootstrap Aggregating

- **Idea:** generar múltiples subconjuntos de entrenamiento mediante remuestreo bootstrap.
- Entrenar un árbol en cada muestra bootstrap:

$$D_b = \{(x_i, y_i)\}_{i=1}^{n_b}, \quad T_b(x) = \text{árbol ajustado en } D_b.$$

- Predicción final (regresión):

$$\hat{f}_{\text{bag}}(x) = \frac{1}{B} \sum_{b=1}^{B} T_b(x)$$

- En clasificación: voto mayoritario.

Intuición

Cada árbol ve una versión ligeramente distinta del mundo → la combinación promedia sus errores.

Bagging: Fundamentos Teóricos

Suposiciones para un x fijo:

- $\mathbb{E}[T_b(x)] = \mu(x)$ (mismo sesgo)
- $\mathrm{Var}[T_b(x)] = \sigma^2(x)$ (misma varianza)
- $\mathrm{Cov}[T_b(x), T_{b'}(x)] = \rho(x)\sigma^2(x)$ (correlación entre modelos)

Bagging en regresión:

$$\hat{f}_{\mathrm{bag}}(x) = \frac{1}{B}\sum_{b=1}^{B} T_b(x)$$

Análisis del sesgo:

$$\mathbb{E}\left[\hat{f}_{\mathrm{bag}}(x)\right] = \frac{1}{B}\sum_{b=1}^{B} \mathbb{E}[T_b(x)] = \mu(x)$$

Conclusión importante

Bagging **no afecta el sesgo** - si cada T_b tiene sesgo $b(x)$, el promedio mantiene el mismo sesgo.

Bagging: Reducción de Varianza y MSE

Varianza del bagging:

$$\begin{aligned}\mathrm{Var}\left[\hat{f}_{\mathrm{bag}}(x)\right] &= \frac{1}{B^2}\left(\sum_{b=1}^{B}\mathrm{Var}[T_b(x)] + \sum_{b\neq b'}\mathrm{Cov}[T_b(x),\, T_{b'}(x)]\right) \\ &= \frac{B\sigma^2 + B(B-1)\rho\sigma^2}{B^2}\end{aligned}$$

Resultado clave:

$$\boxed{\mathrm{Var}\left[\hat{f}_{\mathrm{bag}}(x)\right] = \rho\sigma^2 + \frac{1-\rho}{B}\sigma^2}$$

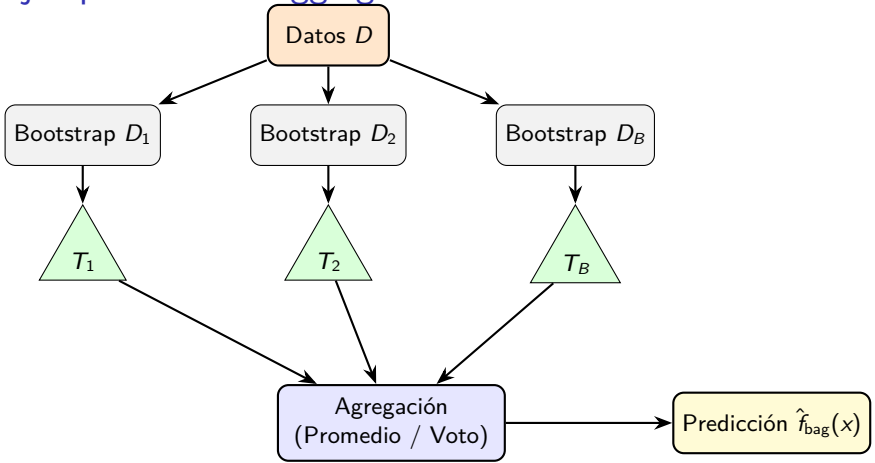
Error Cuadrático Medio (MSE):

$$\mathrm{MSE}\big(\hat{f}_{\mathrm{bag}}(x)\big) = \underbrace{b(x)^2}_{\text{sesgo}} + \underbrace{\rho\sigma^2}_{\text{límite por correlación}} + \underbrace{\frac{1-\rho}{B}\sigma^2}_{\text{decae con } B}$$

Implicaciones prácticas

- Si $\rho < 1$, bagging **reduce varianza**
- Cuando $B \to \infty$: $\mathrm{Var} \to \rho\sigma^2$ (**límite por correlación**)
- **Clave:** Reducir ρ (motivación para Random Forest)

Ejemplo visual: Bagging de árboles



Regresión: $\frac{1}{B}\sum_{b=1}^{B} T_b(x)$

Clasificación: voto mayoritario

Paso 1: Generar B muestras bootstrap.
Paso 2: Ajustar un árbol T_b en cada muestra.
Paso 3: Promediar (regresión) o votar (clasificación).

Ver animaciones

Motivación: Evolución de Bagging a Random Forest

- **Problema con árboles individuales**: Tienen alta varianza - pequeños cambios en los datos producen árboles muy diferentes.
- **Solución Bagging**: Promediar múltiples árboles entrenados en muestras bootstrap:

$$\widehat{f}_{\text{bag}}(x) = \frac{1}{B}\sum_{b=1}^{B} T_b(x)$$

- **Límite del Bagging**: Los árboles siguen correlacionados porque usan las mismas variables importantes.
- **Análisis de varianza** para ensembles:

$$\text{Var}\left[\widehat{f}_{\text{bag}}(x)\right] = \rho\sigma^2 + \frac{1-\rho}{B}\sigma^2$$

  - ρ: correlación entre árboles
  - σ^2: varianza de un árbol individual

- **Insight clave**: ¡Reducir ρ es más efectivo que reducir σ^2!

El problema: Correlación en Bagging

Ejemplo práctico

En datos médicos, si “edad” es muy predictiva:

- Bagging: Todos los árboles split en “edad” primero
- Resultado: Árboles muy similares entre sí
- El ensemble actúa como un solo árbol “amplificado”

Solución: Random Forest - Aleatorización inteligente

Innovación de Breiman (2001)

Dos fuentes de aleatoriedad en lugar de una:

1. **Bootstrap de observaciones** (como en bagging)
2. **Submuestreo de variables** en cada split

Solución: Random Forest - Aleatorización inteligente

Innovación de Breiman (2001)

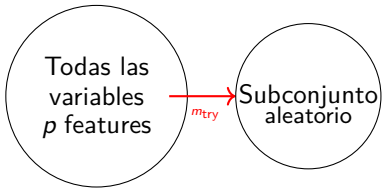
Dos fuentes de aleatoriedad en lugar de una:

1. **Bootstrap de observaciones** (como en bagging)
2. **Submuestreo de variables** en cada split

Algoritmo por nodo

Para cada nodo en cada árbol:

1. Seleccionar m_{try} variables al azar
2. Encontrar mejor split solo entre esas
3. Splitear el nodo
4. Repetir recursivamente



Solución: Random Forest - Aleatorización inteligente

Innovación de Breiman (2001)

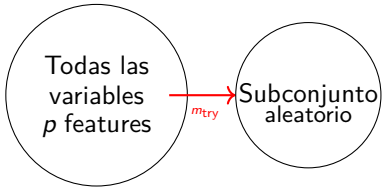
Dos fuentes de aleatoriedad en lugar de una:

1. **Bootstrap de observaciones** (como en bagging)
2. **Submuestreo de variables** en cada split

Algoritmo por nodo

Para cada nodo en cada árbol:

1. Seleccionar m_{try} variables al azar
2. Encontrar mejor split solo entre esas
3. Splitear el nodo
4. Repetir recursivamente



Resultado

Árboles diversos que capturan diferentes patrones → Ensemble más robusto

Definición formal de Random Forest (Parte 1)

Modelo matemático

Para B árboles y parámetros aleatorios Θ_b:

$$\widehat{f}_{\text{RF}}(x) = \frac{1}{B} \sum_{b=1}^{B} T(x; \Theta_b)$$

donde Θ_b incluye: Muestra bootstrap $\mathcal{D}_b$ y selección aleatoria de variables en cada nodo

Definición formal de Random Forest (Parte 1)

Modelo matemático

Para B árboles y parámetros aleatorios Θ_b:

$$\widehat{f}_{\mathrm{RF}}(x) = \frac{1}{B}\sum_{b=1}^{B} T(x;\Theta_b)$$

donde Θ_b incluye: Muestra bootstrap $\mathcal{D}_b$ y selección aleatoria de variables en cada nodo

Interpretación estadística

- Cada $\Theta_b \sim P_\Theta$
- $\mathbb{E}_\Theta[T(x;\Theta)]$: predicción poblacional
- $\widehat{f}_{\mathrm{RF}}(x)$: aproximación Monte Carlo

Definición formal de Random Forest (Parte 2)

Comportamiento con diferentes tamaños de ensemble

B	Comportamiento del Error (EJEMPLO)
1	Alto (árbol individual)
10	Medio (mejora significativa)
100	Bajo (estabilidad buena)
1000	Estable (convergencia)

Definición formal de Random Forest (Parte 2)

Comportamiento con diferentes tamaños de ensemble

B	Comportamiento del Error (EJEMPLO)
1	Alto (árbol individual)
10	Medio (mejora significativa)
100	Bajo (estabilidad buena)
1000	Estable (convergencia)

Convergencia garantizada

Cuando $B \to \infty$:

$$\widehat{f}_{\mathrm{RF}}(x) \xrightarrow{a.s.} \mathbb{E}_{\Theta}[T(x;\Theta)]$$

- El ensemble converge a la expectativa poblacional

Comparación: Evolución de los métodos

De árbol individual a Random Forest

Método	Mecanismo	Resultado
Árbol individual	Entrenamiento determinístico	Alta varianza, sobreajuste fácil
Bagging	Bootstrap de observaciones	Reduce varianza, pero árboles correlacionados
Random Forest	Bootstrap + submuestreo de variables	**Máxima reducción de varianza**

Analogía intuitiva

- **Bagging**: Mismos expertos, distintos casos $\rightarrow$ Opiniones similares
- **Random Forest**: Expertos especializados, distintas perspectivas $\rightarrow$ Análisis más completo

Resumen: Ventajas clave de Random Forest

Ventajas teóricas

- Menor varianza que bagging
- Árboles más diversos y decorrelacionados
- Límite de varianza: $\rho\sigma^2$ con ρ más pequeño
- Convergencia garantizada

Ventajas prácticas

- Funciona "out-of-the-box"
- Poca necesidad de tuning
- Robustez a overfitting
- Maneja bien missing values
- Proporciona importancia de variables

Random Forest = Bagging + **Diversidad forzada**

Teorema de Convergencia de Random Forest

Teorema (Breiman, 2001)

Sea $\{T(x;\Theta_b)\}_{b=1}^{B}$ una secuencia de árboles de decisión construidos según el algoritmo de Random Forest, donde los Θ_b son i.i.d. con distribución P_Θ. Entonces, cuando $B \to \infty$:

$$\widehat{f}_{\mathrm{RF}}(x) = \frac{1}{B}\sum_{b=1}^{B} T(x;\Theta_b) \xrightarrow{a.s.} \mathbb{E}_\Theta[T(x;\Theta)]$$

Teorema de Convergencia de Random Forest

Teorema (Breiman, 2001)

Sea $\{T(x;\Theta_b)\}_{b=1}^{B}$ una secuencia de árboles de decisión construidos según el algoritmo de Random Forest, donde los Θ_b son i.i.d. con distribución P_Θ. Entonces, cuando $B \to \infty$:

$$\widehat{f}_{\text{RF}}(x) = \frac{1}{B}\sum_{b=1}^{B} T(x;\Theta_b) \xrightarrow{a.s.} \mathbb{E}_\Theta[T(x;\Theta)]$$

Hipótesis clave

- Los parámetros Θ_b son **independientes e idénticamente distribuidos**
- Cada árbol $T(x;\Theta_b)$ tiene varianza finita: $\mathbb{E}[T(x;\Theta)^2] < \infty$
- La estructura de los árboles permite suficiente aleatorización

Demostración del Teorema de Convergencia

Esquema de la demostración.

1. **Ley Fuerte de los Grandes Números (LFGN)**: Para variables i.i.d. con media finita:

$$\frac{1}{B}\sum_{b=1}^{B} X_b \xrightarrow{a.s.} \mathbb{E}[X_1]$$

2. **Aplicación a RF**: Definimos $X_b = T(x;\Theta_b)$, entonces:
   - Θ_b i.i.d. $\Rightarrow$ $T(x;\Theta_b)$ i.i.d.
   - Varianza finita garantizada por construcción
3. **Convergencia**: Por LFGN:

$$\widehat{f}_{\mathrm{RF}}(x) = \frac{1}{B}\sum_{b=1}^{B} T(x;\Theta_b) \xrightarrow{a.s.} \mathbb{E}_{\Theta}[T(x;\Theta)]$$

Estructura general del algoritmo

Objetivo

Construir un conjunto $\{T(x;\Theta_b)\}_{b=1}^{B}$ de árboles independientes, cada uno entrenado con una muestra bootstrap y subconjunto aleatorio de predictores, para luego promediar sus salidas:

$$\widehat{f}_{\mathrm{RF}}(x) = \frac{1}{B}\sum_{b=1}^{B} T(x;\Theta_b)$$

Estructura general del algoritmo

Objetivo

Construir un conjunto $\{T(x;\Theta_b)\}_{b=1}^{B}$ de árboles independientes, cada uno entrenado con una muestra bootstrap y subconjunto aleatorio de predictores, para luego promediar sus salidas:

$$\widehat{f}_{\mathrm{RF}}(x) = \frac{1}{B}\sum_{b=1}^{B} T(x;\Theta_b)$$

Esquema general (clasificación o regresión)

1. Para $b = 1, \ldots, B$:
    1. Tomar una muestra bootstrap D_b del conjunto de entrenamiento D.
    2. Construir un árbol de decisión T_b sin poda.
    3. En cada nodo, seleccionar aleatoriamente m_{try} predictores de los p disponibles.
    4. Elegir la mejor división (X_j, s) sólo entre esos m_{try} predictores.
2. Combinar las predicciones:

Pseudocódigo resumido

Algorithm 1: Random Forest (Breiman, 2001)

Input: Datos de entrenamiento $D = \{(x_i, y_i)\}_{i=1}^{n}$, número de árboles B, número de predictores por nodo m_{try}

for $b = 1$ **to** B **do**
- Generar muestra bootstrap D_b de tamaño n.
- Construir árbol T_b con D_b:
- **for** *cada nodo t del árbol* **do**
  - Seleccionar aleatoriamente $M_t \subset \{1, \ldots, p\}$, con $|M_t| = m_{\text{try}}$.
  - Encontrar el predictor $j^* \in M_t$ y punto de corte s^* que maximicen $\Delta I(s, j)$.
  - Dividir el nodo t en t_L y t_R según $(X_{j^*} \leq s^*)$.

Predicción final:

$$\widehat{f}_{\text{RF}}(x) = \begin{cases} \frac{1}{B} \sum_{b=1}^{B} T_b(x), & \text{regresión} \\ \text{moda}\{T_b(x)\}, & \text{clasificación.} \end{cases}$$

Parámetros principales del Bosque Aleatorio

Parámetro	**Símbolo**	**Descripción**
Número de árboles	B	Controla la estabilidad del ensemble.
Predictores por nodo	m_{try}	Controla la correlación entre árboles.
Profundidad máxima	$d_{\text{máx}}$	Controla el sesgo (menor $d_{\text{máx}} \Rightarrow$ mayor sesgo).
Mínimo tamaño de hoja	$n_{\text{mín}}^{(\text{leaf})}$	Evita nodos con muy pocos datos.

Parámetros principales del Bosque Aleatorio

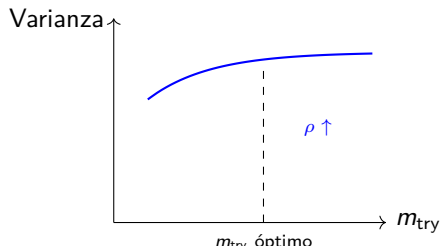
Parámetro	**Símbolo**	**Descripción**
Número de árboles	B	Controla la estabilidad del ensemble.
Predictores por nodo	m_{try}	Controla la correlación entre árboles.
Profundidad máxima	$d_{\text{máx}}$	Controla el sesgo (menor $d_{\text{máx}} \Rightarrow$ mayor sesgo).
Mínimo tamaño de hoja	$n_{\text{mín}}^{(\text{leaf})}$	Evita nodos con muy pocos datos.

Valores típicos:

- Clasificación: $m_{\text{try}} = \sqrt{p}$
- Regresión: $m_{\text{try}} = p/3$
- B grande (ej. 500–1000) asegura estabilidad asintótica.

Efecto de los parámetros

- $B \uparrow$: disminuye la varianza hasta estabilizarse (ley de los grandes números).
- $m_{\text{try}} \downarrow$: reduce la correlación ρ entre árboles $\Rightarrow$ menor varianza total.
- Árboles sin poda $\Rightarrow$ bajo sesgo, varianza controlada por el ensemble.



Ajustar m_{try} es clave: controla el balance entre independencia (varianza) y fuerza individual (sesgo).

Resumen de la sección

- El Bosque Aleatorio combina bagging con selección aleatoria de predictores.
- Cada árbol se construye sobre una muestra bootstrap y con restricciones locales de variables.
- El parámetro m_{try} controla la correlación ρ entre árboles:

$$\text{Var}[\bar{T}] = \rho\sigma^2 + \frac{1-\rho}{B}\sigma^2$$

- Árboles sin poda $\Rightarrow$ bajo sesgo, la agregación controla la varianza.