



UTEC Posgrado



MAESTRÍA

Árboles de decisión



Antecedentes



Métodos Basados en Árboles

La idea fundamental es **particionar el espacio de características** (las variables de entrada X) en un conjunto de rectángulos disjuntos.

Una vez dividido el espacio, ajustamos un modelo muy simple dentro de cada región: generalmente una **constante** (promedio).

Son conceptualmente simples pero poderosos para capturar estructuras no lineales.

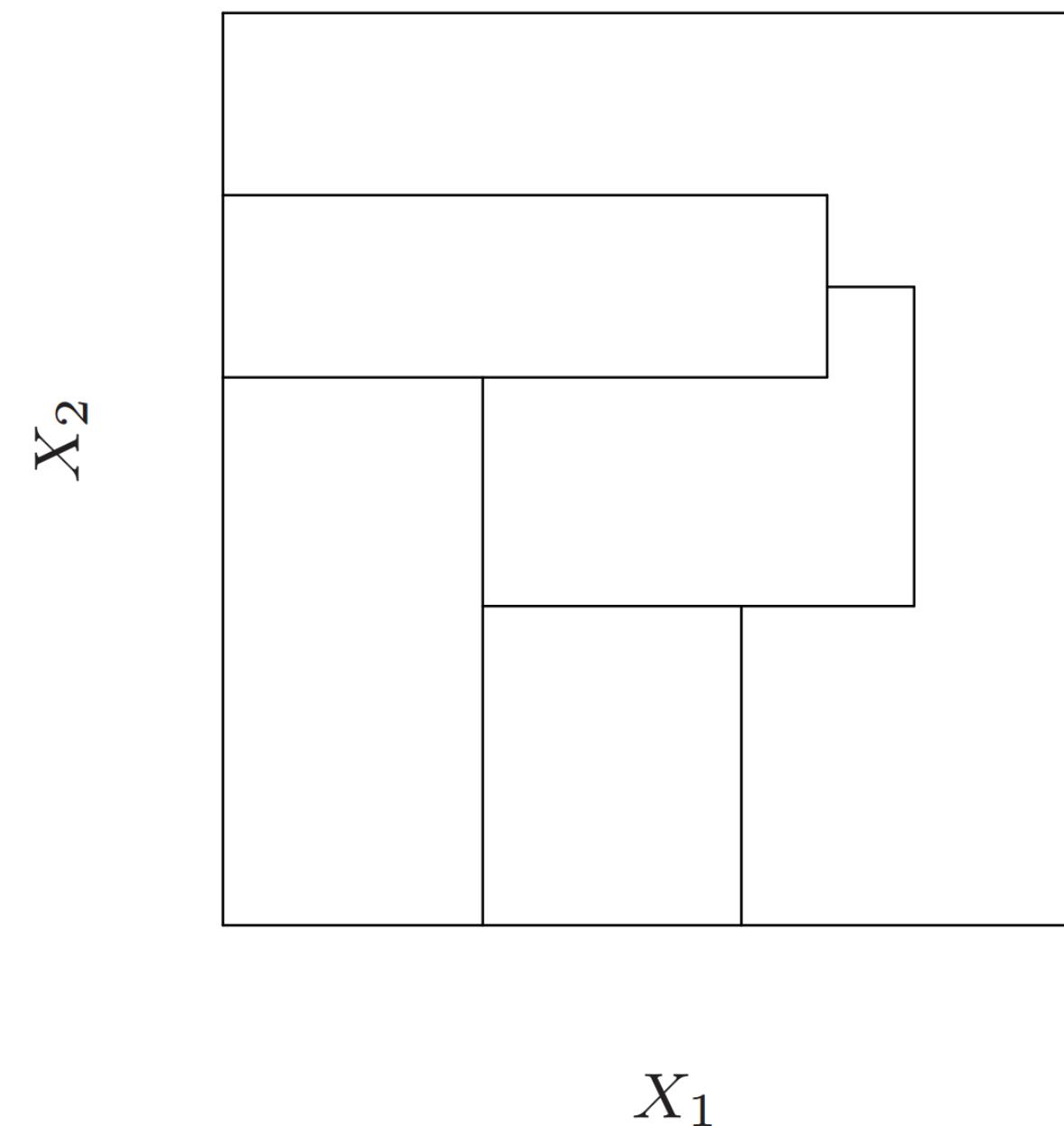


El Problema de la Partición General

Muestra una división del espacio mediante líneas paralelas a los ejes ($X_1 = c$), creando bloques rectangulares

La estructura es **arbitraria**; los cortes no necesariamente atraviesan todo el espacio o una región previa completa.

Aunque las líneas son simples, describir matemáticamente las regiones resultantes es **complicado** (no sigue reglas anidadas simples).



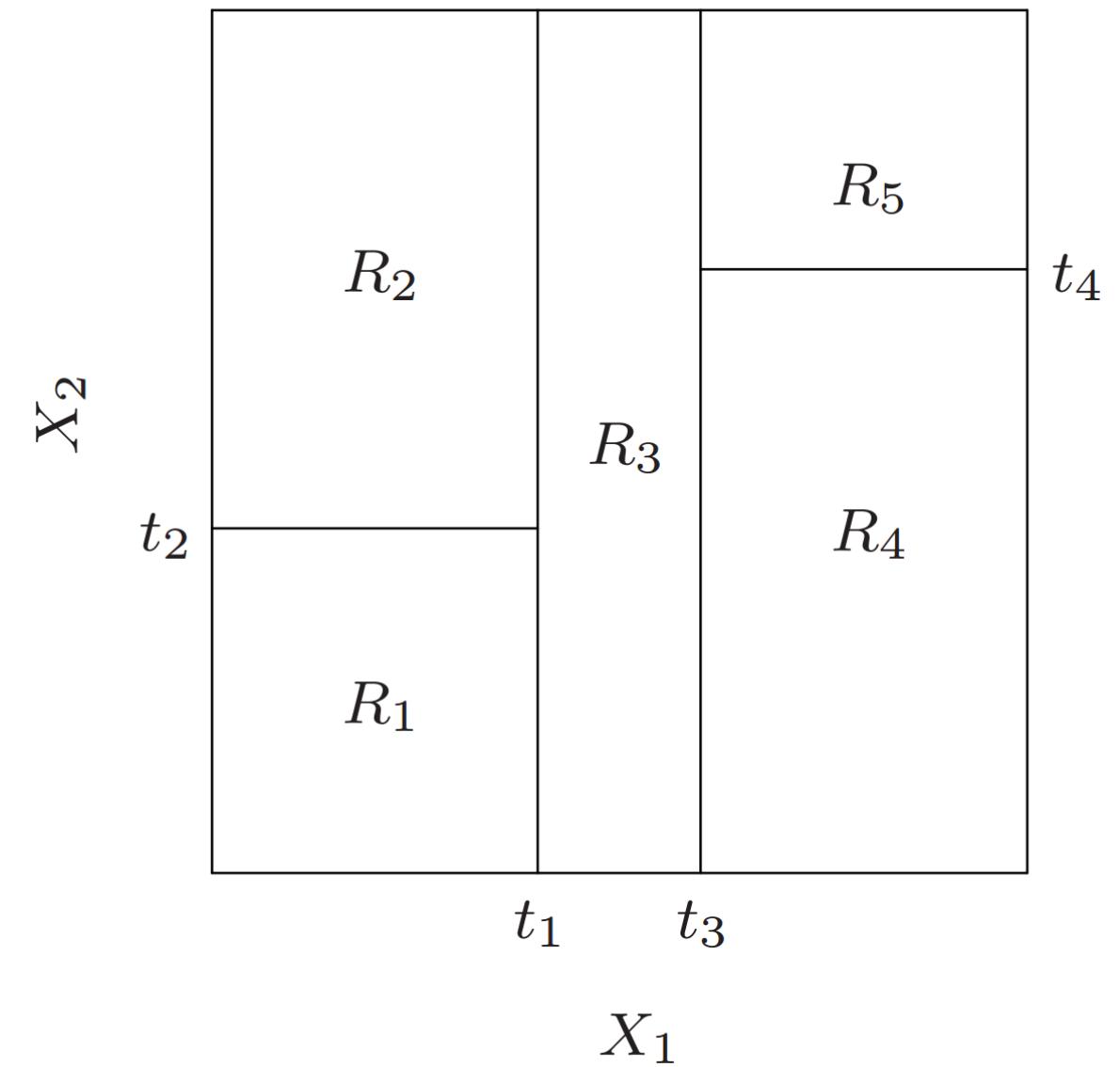
La Solución CART (Partición Binaria Recursiva)

Restringimos el método a divisiones **secuenciales** y **jerárquicas**

Secuencia del Ejemplo:

1. Primer corte en $X_1 = t_1$ (divide todo el espacio)
2. La región izquierda se divide en $X_2 = t_2$ (genera R_1, R_2).
3. La región derecha se divide en $X_1 = t_3$, y luego una parte de esta en $X_2 = t_4$

Resultado: 5 regiones finales (R_1, \dots, R_5).

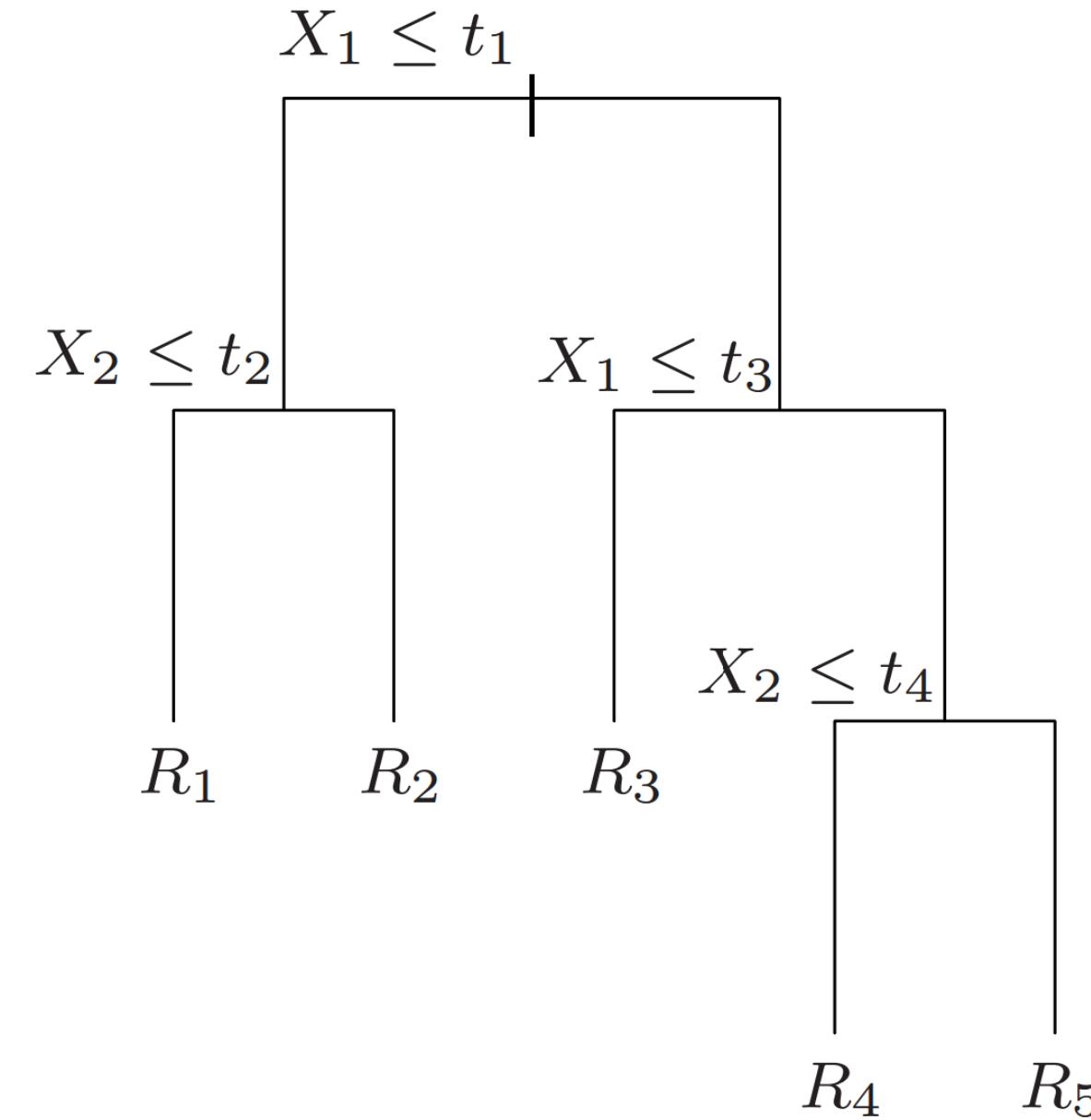


La Solución CART (Partición Binaria Recursiva)

Representa el mismo modelo como un diagrama de flujo.

- **Raíz:** Contiene todos los datos.
- **Nodos Intermedios:** Reglas de decisión ($X_j \leq t_k$).
- **Hojas (Nodos Terminales):** Corresponden a las regiones finales R_m .

Estrategia: Utilizamos un algoritmo "**Greedy**" (**Voraz**), eligiendo la mejor variable y punto de corte (j, s) en cada paso para minimizar el error localmente



Formulación Matemática del Árbol de Regresión

$$\hat{f}(X) = \sum_{m=1}^5 c_m I\{(X_1, X_2) \in R_m\}.$$

Función de **predicción** estimada.

La constante de predicción para la región m

(Función Indicadora):

- Vale **1** si el input (X_1, X_2) está dentro de la región R_m .
- Vale **0** en caso contrario.

La m-ésima región rectangular disjunta resultante de la partición recursiva



Definición Formal y algorítmico



Árboles de Regresión

Para construir un árbol de regresión, partimos de un conjunto de datos con:

- **N observaciones:** (x_i, y_i) para $i = 1, 2, \dots, N$.
- **p variables de entrada:** $(x_i = x_{i1}, x_{i2}, \dots, x_{ip})$
- **Una variable de respuesta:** y_i .

El algoritmo debe **decidir** automáticamente las **variables de división** y los **puntos de división**, y también qué topología (forma) debe tener el árbol.



Definición formal del Modelo

Si dividimos el espacio de las variables en **M regiones** ($R_1, R_2 \dots, R_m$), modelamos la respuesta como una **constante c_m** en cada región:

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

Nota: $I(x \in R_m)$ es una función indicadora que vale 1 si x pertenece a la región R_m y 0 en caso contrario.

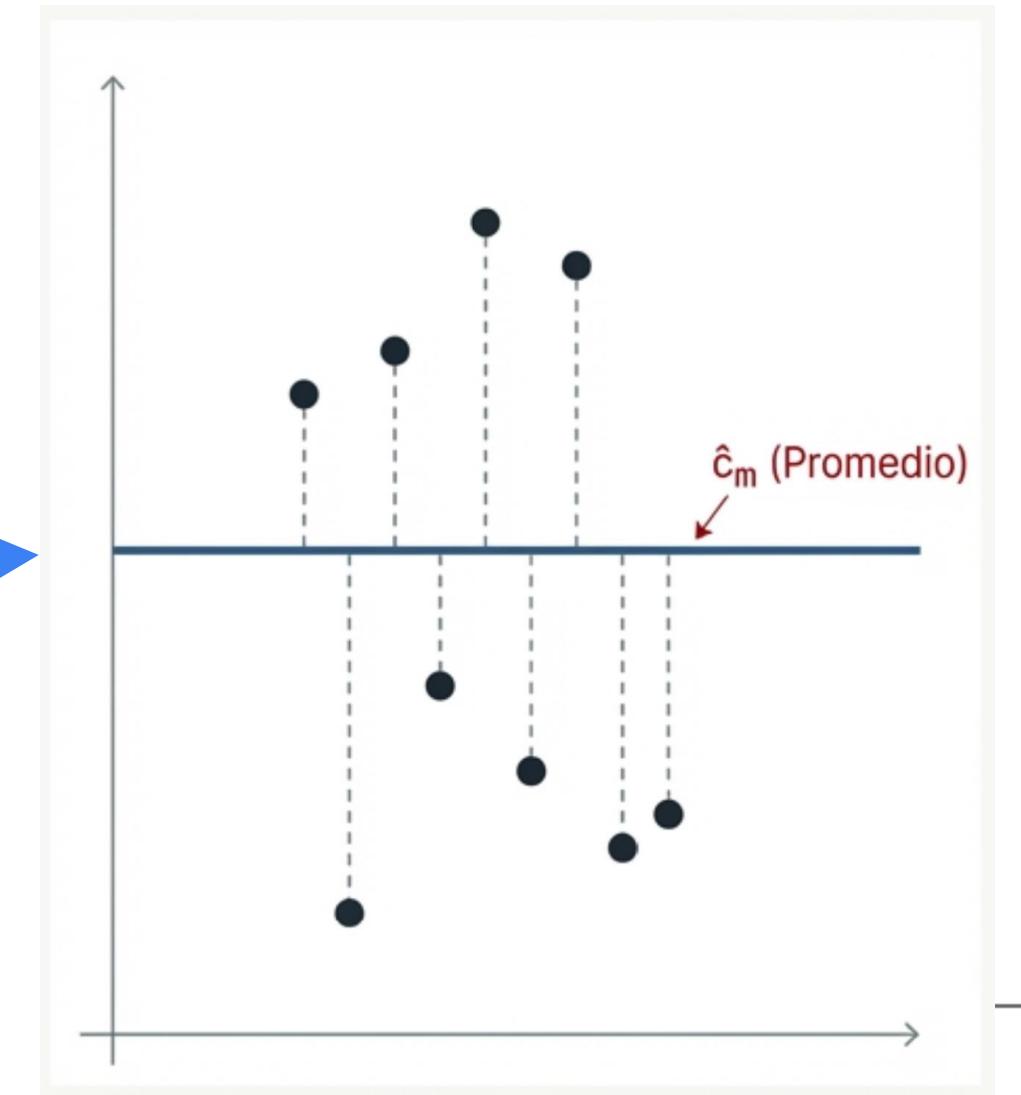


El Criterio de Minimización

Para minimizar la suma de los errores al cuadrado en una región dada, la estadística nos dice que la mejor predicción es simplemente el **promedio** de los datos observados en esa región:

$$\hat{c}_m = \text{ave}(y_i | x_i \in R_m).$$

Bajo este criterio, el mejor valor para la constante \hat{c}_m es simplemente el promedio de los y_i en esa región.



Algoritmo de Partición Binaria Recursiva

El Desafío Computacional

Encontrar la partición óptima global (la mejor división posible de todo el conjunto) es computacionalmente inviable.

Por ello, se utiliza un **algoritmo "greedy"** (codicioso): toma la mejor decisión posible en el paso actual sin considerar las implicaciones futuras.



Algoritmo de Partición Binaria Recursiva

Definición de la División (Semiplanos)

Para cada variable de división j y punto de corte s , definimos un par de semiplanos:

Región 1: $R_1(j, s) = \{X | X_j \leq s\}$ (valores menores o iguales al corte).

Región 2: $R_2(j, s) = \{X | X_j > s\}$ (valores mayores al corte).



Algoritmo de Partición Binaria Recursiva

Luego, **se optimiza**, buscando la variable j y el punto de corte s que resuelve el siguiente problema de minimización:

$$\min_{j, s} \left[\min_{c_1} \sum_{\substack{x_i \in R_1(j, s)}} (y_i - c_1)^2 + \min_{c_2} \sum_{\substack{x_i \in R_2(j, s)}} (y_i - c_2)^2 \right].$$



Error Región 1. **Error Región 2.**

Para cualquier elección de j y s , la minimización interna se resuelve mediante los promedios:

$$\hat{c}_1 = ave(y_i | x_i \in R_1) \quad \text{y} \quad \hat{c}_2 = ave(y_i | x_i \in R_2)$$



Algoritmo de Partición Binaria Recursiva

Finalmente,

Tras encontrar la **mejor división**, dividimos los datos en las dos regiones resultantes y repetimos el proceso de división en cada una de ellas. A continuación, **este proceso se repite** en todas las **regiones** resultantes.



Criterio de Costo complejidad



Criterio de Costo complejidad

Se busca una **estrategia** adaptativa para equilibrar el tamaño del árbol y la bondad de ajuste.

El tamaño del árbol es un **parámetro de ajuste** que gobierna la complejidad.

La estrategia preferida es hacer crecer un árbol tan grande, deteniendo el proceso solo cuando se alcanza un tamaño mínimo de nodo. (Ej. 5 observaciones).

Luego, este árbol masivo se poda utilizando poda de **Costo-Complejidad**.



Criterio de Costo complejidad

Definiciones y notación

Observaciones en la región

Conteo de puntos de datos que caen en el nodo terminal m



$$N_m = \#\{x_i \in R_m\},$$

Predicción media

El valor promedio de la variable respuesta y_i para las observaciones de esta región.



$$\hat{c}_m = \frac{1}{N_m} \sum_{x_i \in R_m} y_i,$$

Error cuadrático medio (MSE).

Varianza de las observaciones dentro del nodo m respecto a la media predicha.



$$Q_m(T) = \frac{1}{N_m} \sum_{x_i \in R_m} (y_i - \hat{c}_m)^2$$



Criterio de Costo complejidad

El objetivo es encontrar un subárbol que minimice este valor.

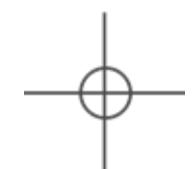
$$C_\alpha(T) = \sum_{m=1}^{|T|} N_m Q_m(T) + \alpha |T|$$

Penalización por complejidad

Costo lineal añadido por cada nodo terminal extra

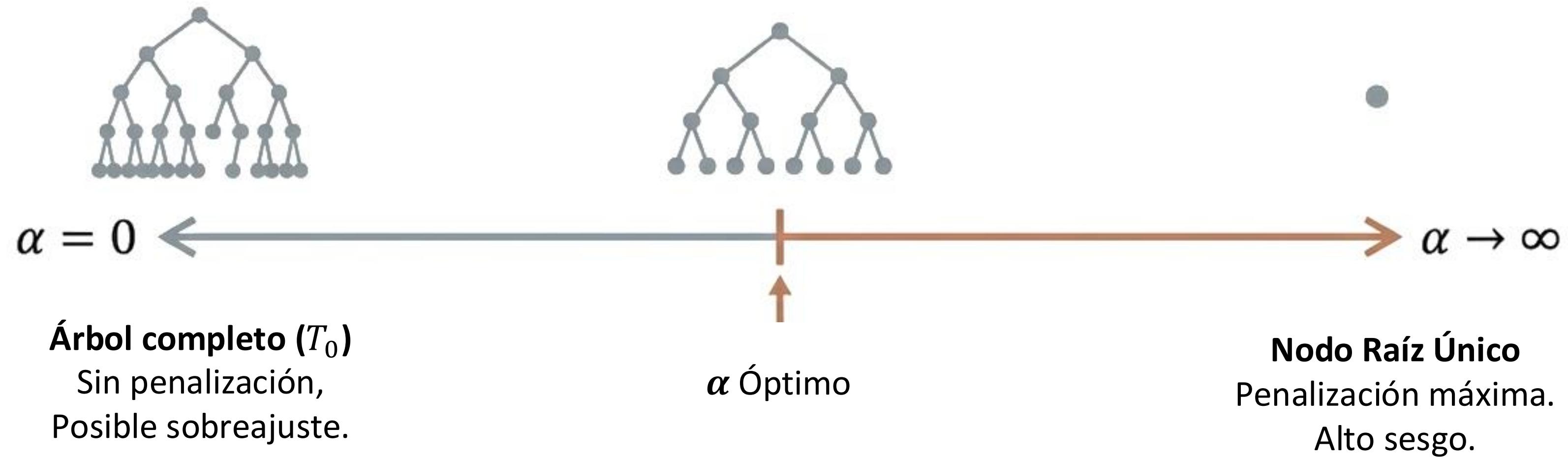
Error de Ajuste

Mide que tan bien se adapta el árbol a los datos de entrenamiento.



Criterio de Costo complejidad

La función del parámetro de ajuste (α).



Valores grande de α resultan en árboles más pequeños. El valor óptimo debe elegirse adaptativamente.



MAESTRÍA

Random Forests

Theory and practice.



IN
GE
NIE
CI
RA
BLE



Introducción

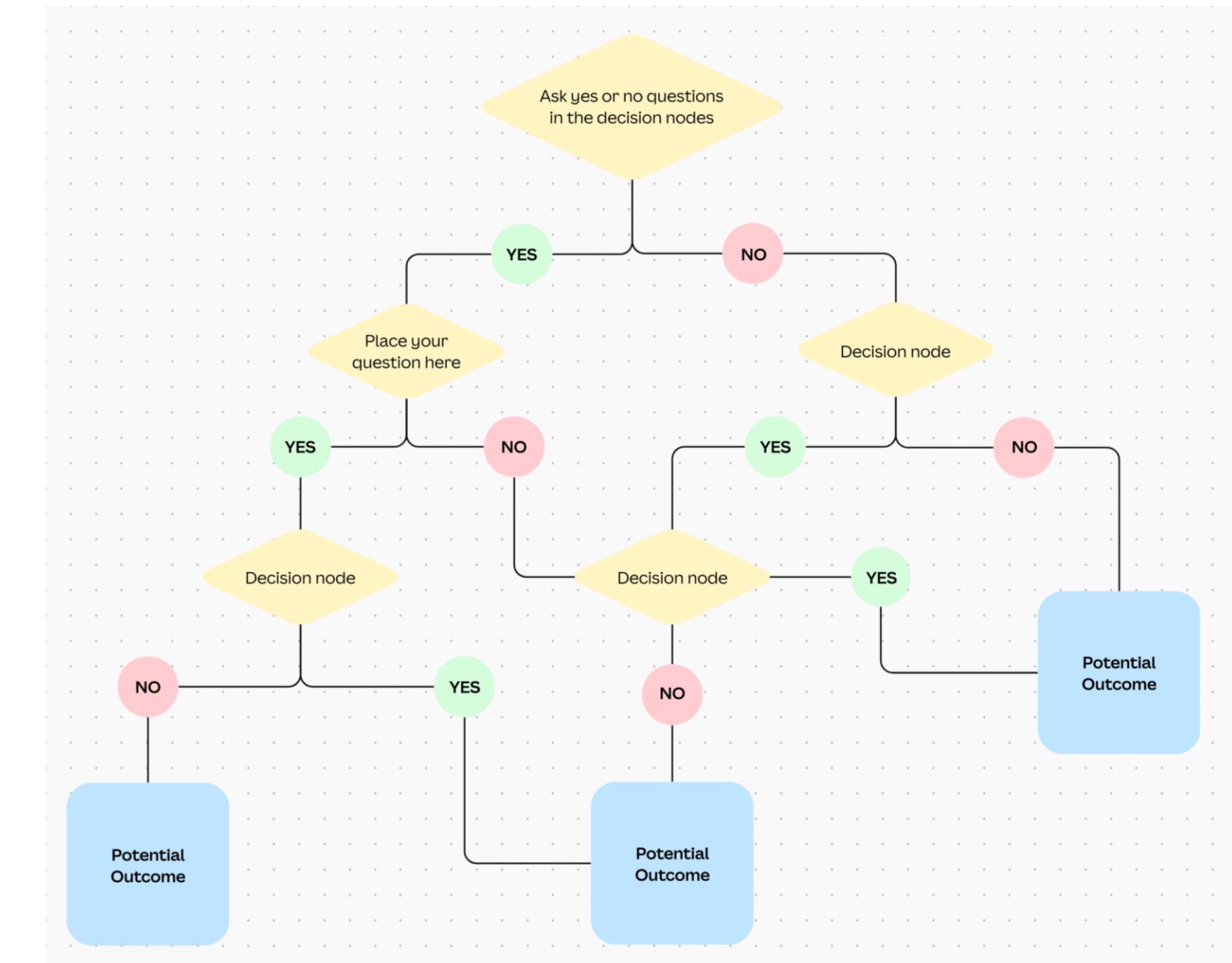
Bagging y Random Forests



El problema: árboles inestables

Los árboles de decisión son muy **sensibles** a los **datos**.

Si **cambiamos** un poco el conjunto de entrenamiento, el árbol puede cambiar **mucho**, y por tanto la **predicción** final también.



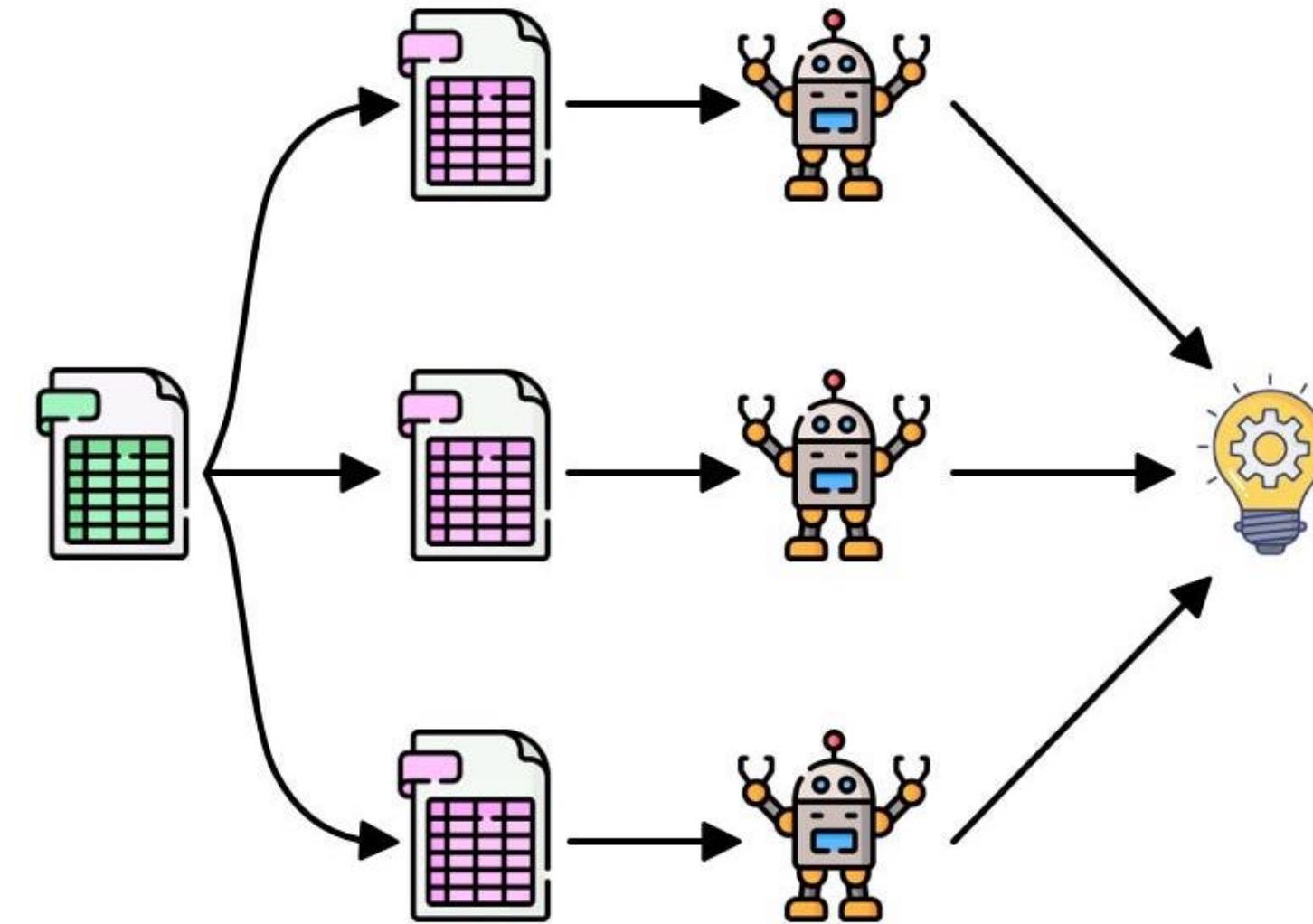
La idea de Bagging

Bagging

En lugar de confiar en un solo árbol, **usamos muchos.**

- Creamos muchos conjuntos de datos ligeramente distintos a partir del original.
- Entrenamos un árbol en cada uno.
- Cada árbol da su propia predicción.
- Combinamos todas las predicciones.

Promediar muchas predicciones inestables produce una predicción más estable.



¿Por qué funciona Bagging?

Los errores aleatorios se cancelan al promediar.

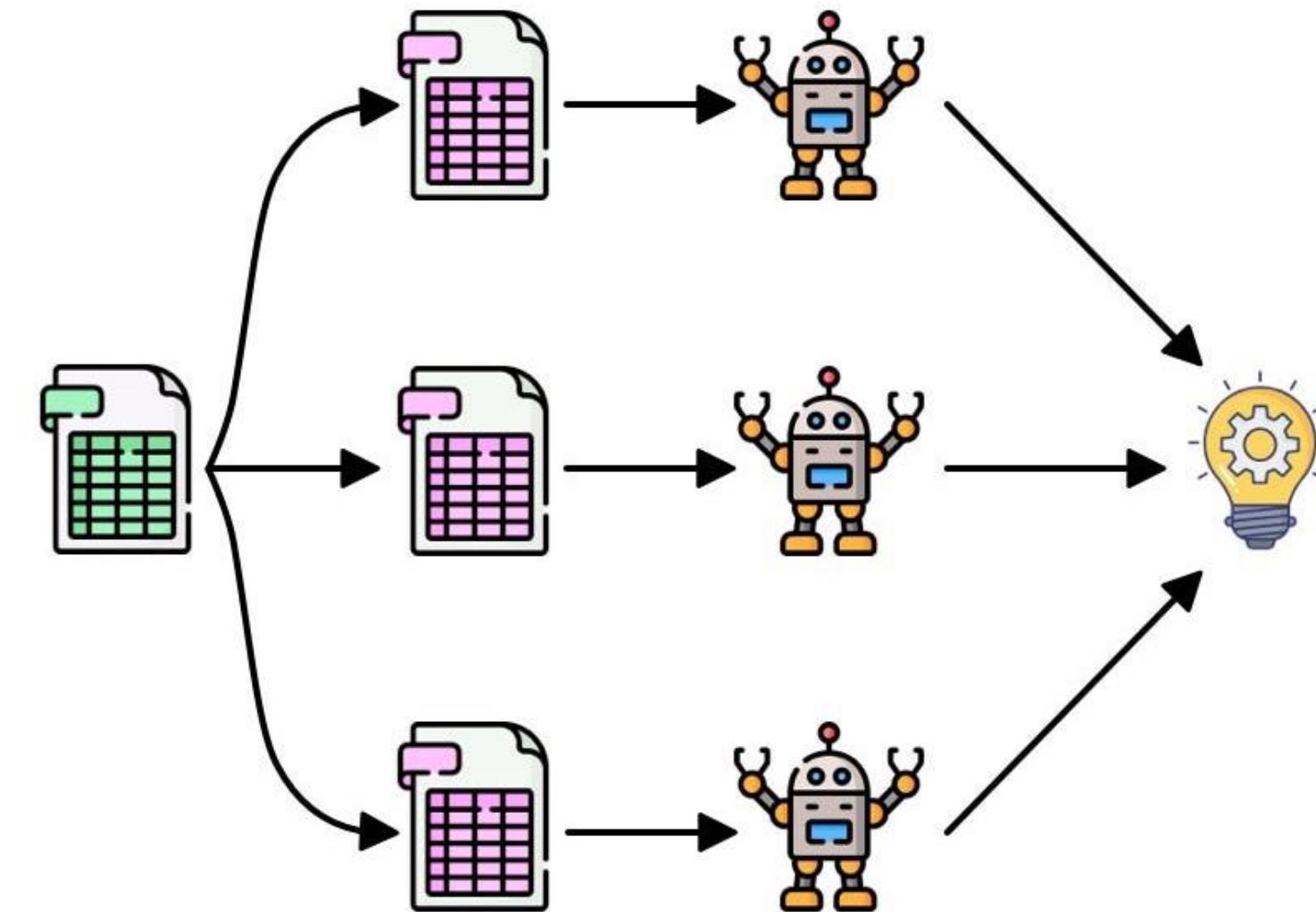
Cada árbol comete errores diferentes.

Al promediar:

- ✓ Los errores extremos se suavizan.
- ✓ La predicción final es más estable.

Bagging no hace los árboles más inteligentes, los hace menos variables.

Bagging

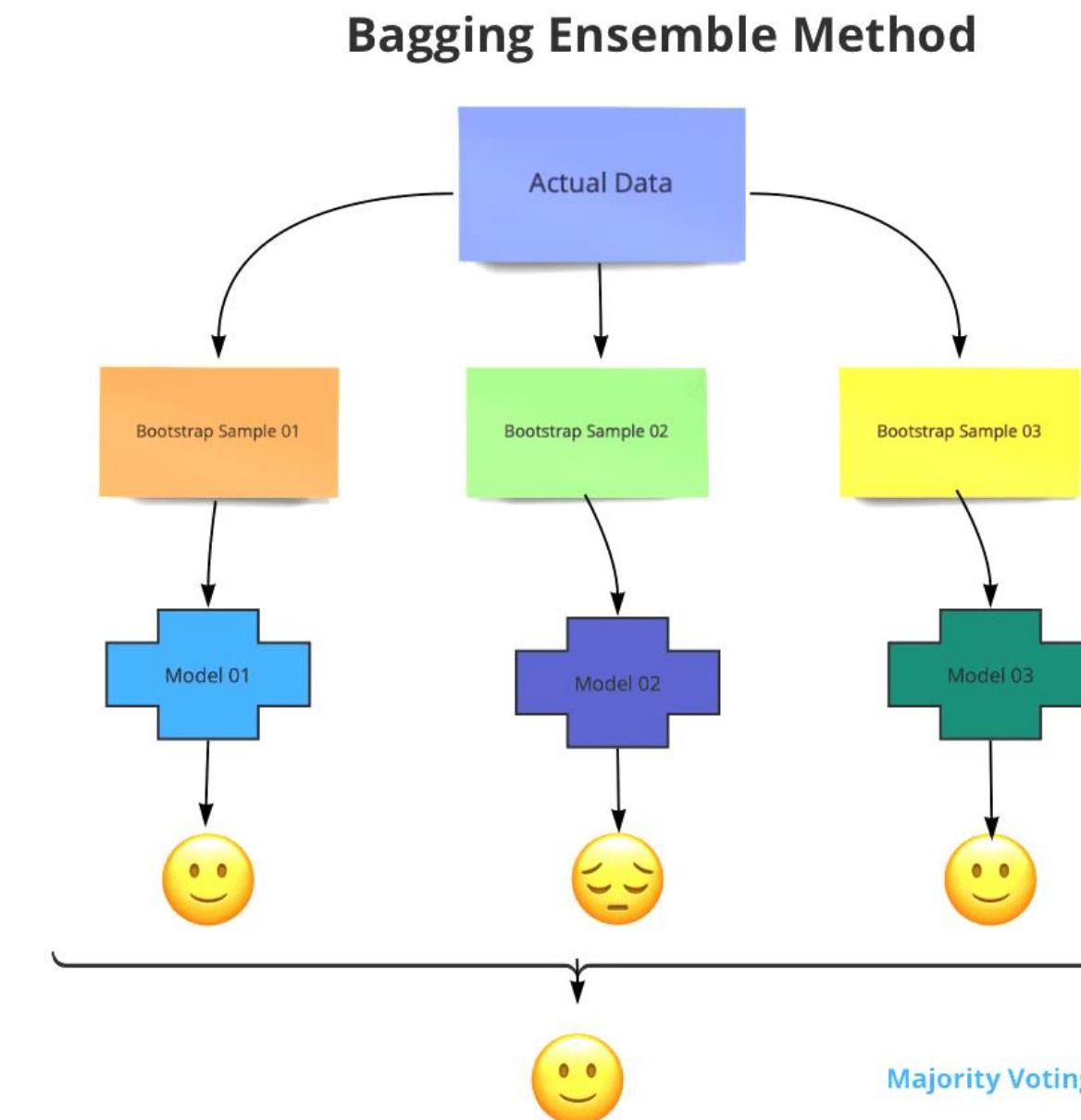


Problemas del Bagging

Bagging mejora los árboles, pero muchos árboles siguen siendo parecidos.

En Bagging, cada árbol ve datos ligeramente distintos. Aun así, si hay variables muy **dominantes**, muchos árboles toman **decisiones similares**.

Si los árboles son muy parecidos, el promedio no reduce tanto el error.



Límite matemático de Bagging

$$\text{Varianza} = \rho\sigma^2 + \frac{1-\rho}{B}\sigma^2$$

Límite inferior causado por la correlación
 ↓
 ↑ Varianza de cualquier árbol individual
 ↑ Número de árboles
 ↗ Correlación positiva por pares
 ↑ Desaparece cuando $B \rightarrow \infty$

A medida que aumentas el número de árboles (B), el segundo término desaparece, pero el primero ($\rho\sigma^2$) permanece. Esto significa que **la correlación entre los árboles limita los beneficios del promedio**. Si los árboles son muy **parecidos entre sí**, el *Bagging* deja de mejorar el modelo.



Modelado

Random Forests



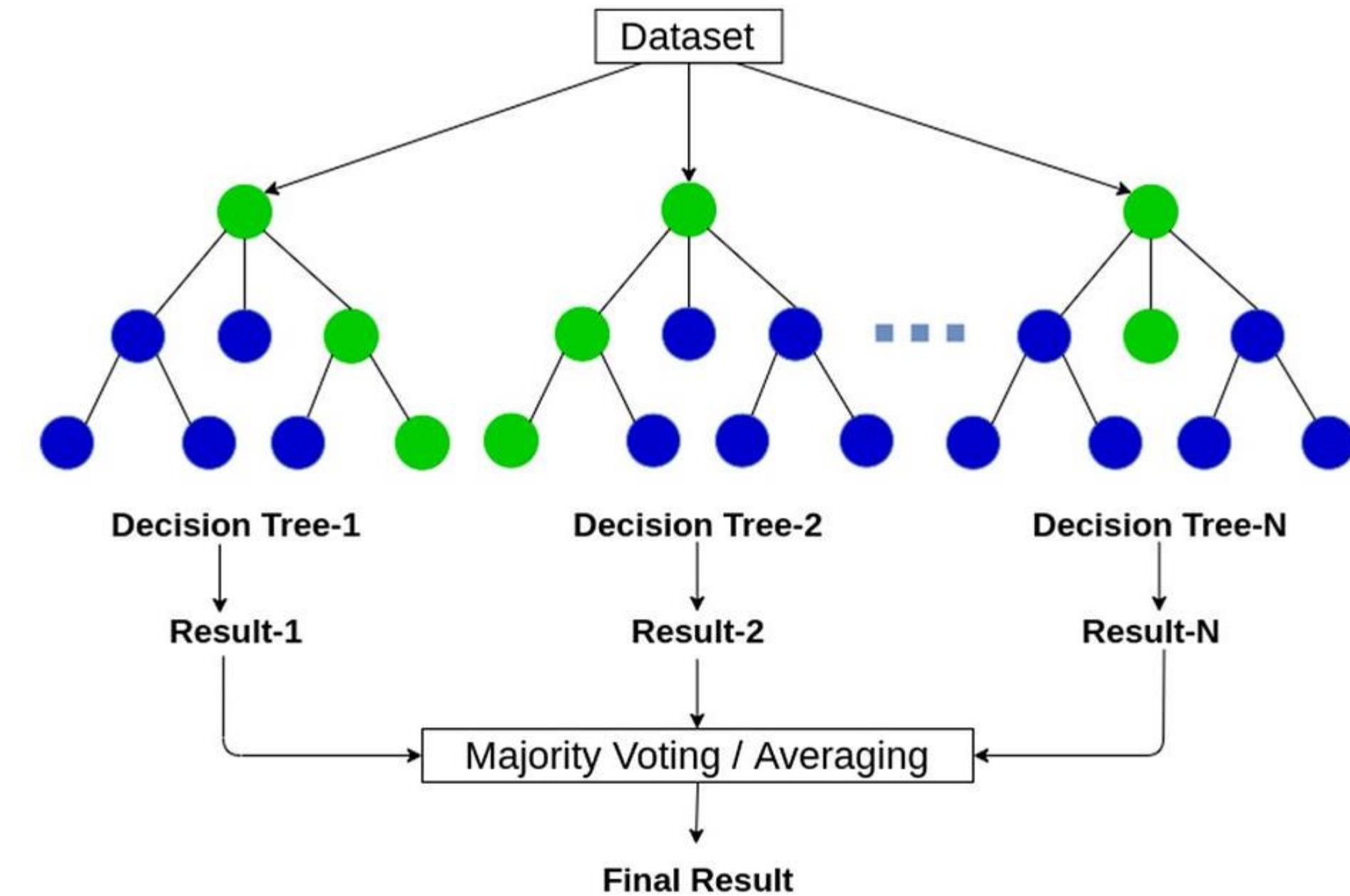
Random Forests

Son una **modificación** sustancial del **bagging** que construye una gran colección de árboles **no** correlacionados y luego los promedia.

En otras palabras, Random Forest **fuerza a que los árboles sean diferentes entre sí**.

Random Forest:

- ✓ Sigue usando muchos árboles (como bagging).
- ✓ Pero introduce más aleatoriedad al construir cada árbol.
- ✓ Cada árbol ve el problema desde un ángulo distinto.



Random Forests

Predicción final del **Random Forest** para una nueva entrada x .

indica que está basado en un conjunto de B árboles.

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

Indica "Random Forest"

! individual (el árbol número b) para la entrada x .

Importante!

Entonces, la **predicción de un Random Forest en regresión es el promedio de las predicciones de todos sus árboles.**



Algoritmo

Random Forests



Random Forests - Algorítmoo

1. Para $b = 1$ hasta B :
 - (a) Extraer una muestra bootstrap \mathbf{Z}^* de tamaño N a partir de los datos de entrenamiento.
 - (b) Hacer crecer un árbol T_b del bosque aleatorio con los datos bootstrappeados, repitiendo recursivamente los siguientes pasos para cada nodo terminal del árbol, hasta que el tamaño mínimo de nodo n_{min} sea alcanzado.
 - i. Seleccionar m variables al azar de entre las p variables.
 - ii. Escoger la mejor variable/punto de división de entre los m .
 - iii. Dividir el nodo en dos nodos hijos.
2. Dar como salida el ensamble de árboles $\{T_b\}_1^B$.

Para hacer una predicción en un nuevo punto x :

$$\text{Regresión: } \hat{f}_{\text{rf}}^B(x) = \frac{1}{B} \sum_{b=1}^B T_b(x).$$

Clasificación: Sea $\hat{C}_b(x)$ la predicción de clase en el b ésimo árbol aleatorio. Entonces

$$\hat{C}_{\text{rf}}^B(x) = \text{mayoria } \{\hat{C}_b(x)\}_1^B.$$

1. *Entrenamiento: El bucle de construcción del ensamble (Pasos 1 y 2).*
2. *Inferencia: El mecanismo de predicción para nuevos datos x .*



Random Forests - Algorítmo

DEFINICIÓN DE VARIABLES E INPUTS

N

Tamaño de Muestra
El número total de observaciones en la data de entrenamiento original.

p

Predictors
El número total de variables o features disponibles en el dataset.

B

Bootstrap Trees
La cantidad de árboles que cultivaremos en el bosque (el límite del bucle).

Z*

Bootstrap Sample
La muestra aleatoria generada para entrenar un árbol específico.



Random Forests - Algorítmoo

1. Para $b = 1$ hasta B :

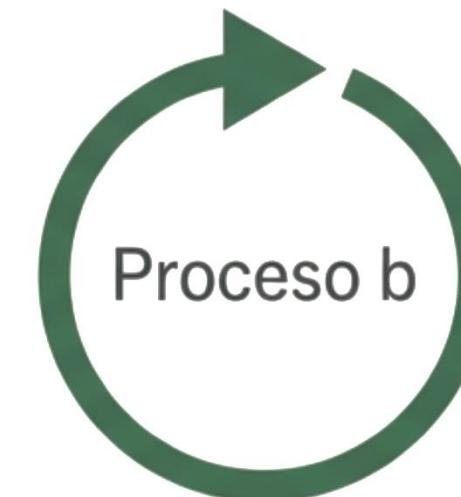
- (a) Extraer una muestra bootstrap \mathbf{Z}^* de tamaño N a partir de los datos de entrenamiento.
- (b) Hacer crecer un árbol T_b del bosque aleatorio con los datos bootstrappeados, repitiendo recursivamente los siguientes pasos para cada nodo terminal del árbol, hasta que el tamaño mínimo de nodo n_{min} sea alcanzado.
 - i. Seleccionar m variables al azar de entre las p variables.
 - ii. Escoger la mejor variable/punto de división de entre los m .
 - iii. Dividir el nodo en dos nodos hijos.

2. Dar como salida el ensamble de árboles $\{T_b\}_1^B$.

Para hacer una predicción en un nuevo punto x :

$$\text{Regresión: } \hat{f}_{\text{rf}}^B(x) = \frac{1}{B} \sum_{b=1}^B T_b(x).$$

Clasificación: Sea $\hat{C}_b(x)$ la predicción de clase en el b ésimo árbol aleatorio. Entonces
 $\hat{C}_{\text{rf}}^B(x) = \text{mayoria } \{\hat{C}_b(x)\}_1^B$.



*El algoritmo opera mediante la repetición.
No se busca un “super árbol”, sino una colección de B estimadores independientes.*

Cada iteración b es un proceso autónomo que resulta en un árbol $T_{(b)}$.

B es un hiperparámetro. Un número mayor de árboles estabiliza el error, pero aumenta el costo computacional.



Random Forests - Algorítmoo

1. Para $b = 1$ hasta B :

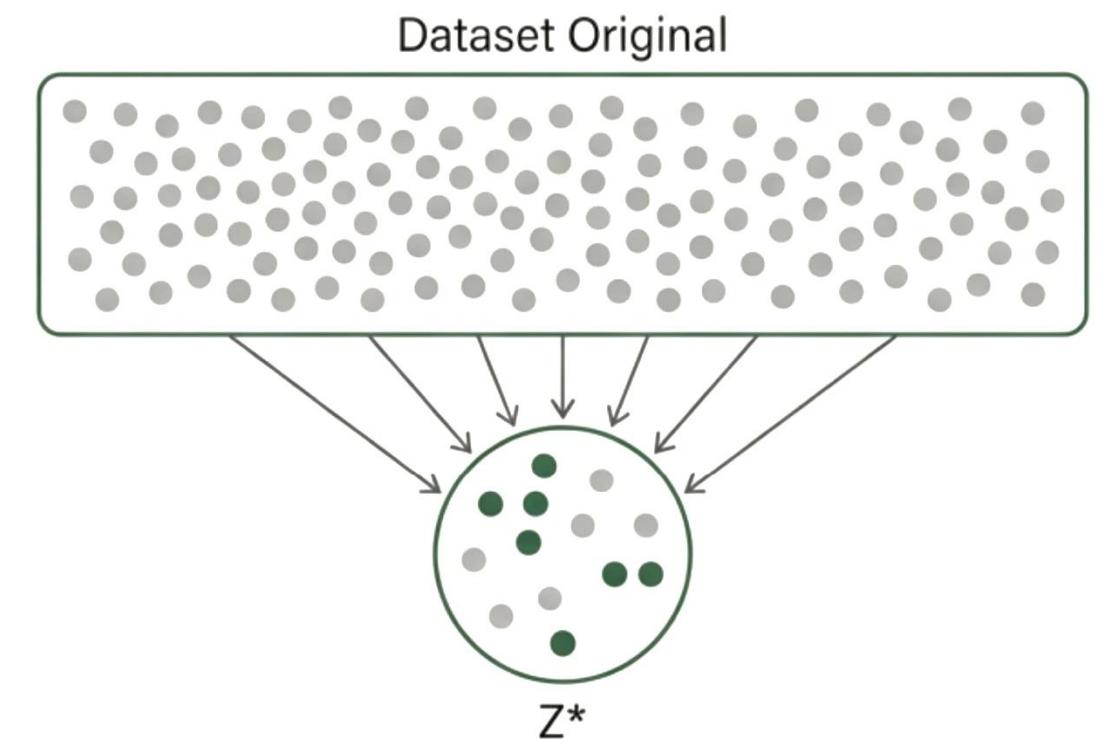
- (a) Extraer una muestra bootstrap Z^* de tamaño N a partir de los datos de entrenamiento.
- (b) Hacer crecer un árbol T_b del bosque aleatorio con los datos bootstrappeados, repitiendo recursivamente los siguientes pasos para cada nodo terminal del árbol, hasta que el tamaño mínimo de nodo n_{min} sea alcanzado.
 - i. Seleccionar m variables al azar de entre las p variables.
 - ii. Escoger la mejor variable/punto de división de entre los m .
 - iii. Dividir el nodo en dos nodos hijos.

2. Dar como salida el ensamble de árboles $\{T_b\}_1^B$.

Para hacer una predicción en un nuevo punto x :

$$\text{Regresión: } \hat{f}_{\text{rf}}^B(x) = \frac{1}{B} \sum_{b=1}^B T_b(x).$$

Clasificación: Sea $\hat{C}_b(x)$ la predicción de clase en el b ésimo árbol aleatorio. Entonces $\hat{C}_{\text{rf}}^B(x) = \text{mayoria } \{\hat{C}_b(x)\}_1^B$.



La Mecánica: Para el árbol b , generamos un nuevo dataset Z^* seleccionando N observaciones aleatorias de la data original.

Con Reemplazo: Una misma observación puede aparecer múltiples veces en Z^* , mientras que otras pueden quedar fuera (Out-of-Bag).

Propósito: Introducir diversidad en los datos de entrenamiento



Random Forests - Algorítmoo

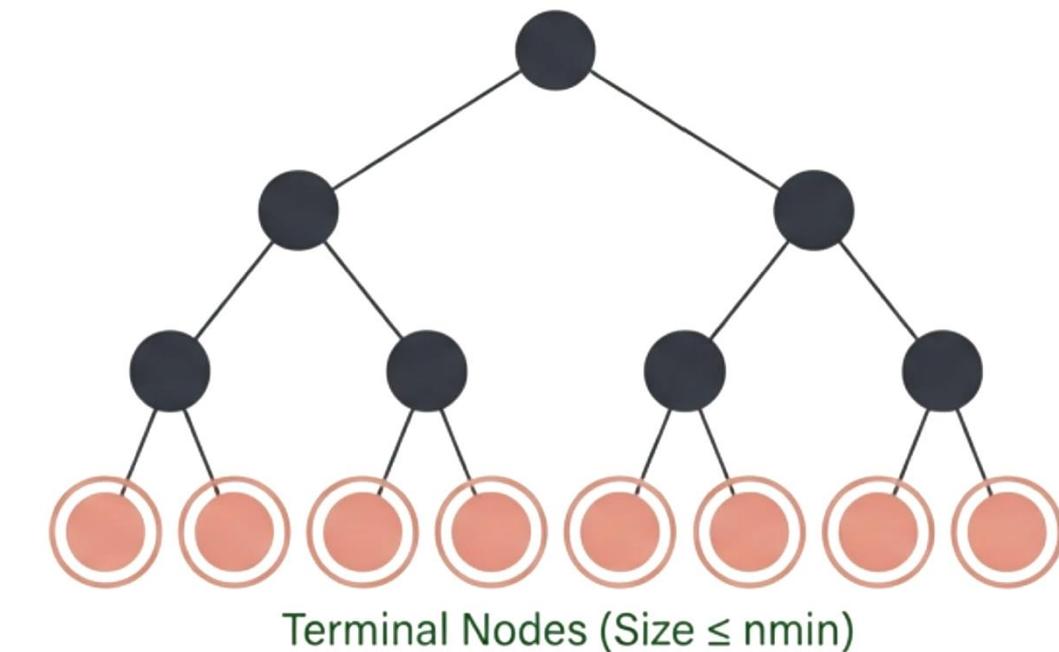
1. Para $b = 1$ hasta B :
 - (a) Extraer una muestra bootstrap \mathbf{Z}^* de tamaño N a partir de los datos de entrenamiento.
 - (b) Hacer crecer un árbol T_b del bosque aleatorio con los datos bootstrappeados, repitiendo recursivamente los siguientes pasos para cada nodo terminal del árbol, hasta que el tamaño mínimo de nodo n_{min} sea alcanzado.
 - i. Seleccionar m variables al azar de entre las p variables.
 - ii. Escoger la mejor variable/punto de división de entre los m .
 - iii. Dividir el nodo en dos nodos hijos.
2. Dar como salida el ensamble de árboles $\{T_b\}_1^B$.

Para hacer una predicción en un nuevo punto x :

$$\text{Regresión: } \hat{f}_{\text{rf}}^B(x) = \frac{1}{B} \sum_{b=1}^B T_b(x).$$

Clasificación: Sea $\hat{C}_b(x)$ la predicción de clase en el b ésimo árbol aleatorio. Entonces $\hat{C}_{\text{rf}}^B(x) = \text{mayoria } \{\hat{C}_b(x)\}_1^B$.

El Concepto



El árbol crece en los datos Z^ .*

Recursividad: *El proceso de división se aplica repetidamente a cada nodo terminal resultante.*

Criterio de Parada: *El crecimiento continúa hasta que el tamaño del nodo es menor o igual a n_{min} . Esto suele resultar en árboles profundos con bajo sesgo pero alta varianza.*



Random Forests - Algorítmoo

1. Para $b = 1$ hasta B :

- (a) Extraer una muestra bootstrap \mathbf{Z}^* de tamaño N a partir de los datos de entrenamiento.
- (b) Hacer crecer un árbol T_b del bosque aleatorio con los datos bootstrappeados, repitiendo recursivamente los siguientes pasos para cada nodo terminal del árbol, hasta que el tamaño mínimo de nodo n_{min} sea alcanzado.
 - i. Seleccionar m variables al azar de entre las p variables.
 - ii. Escoger la mejor variable/punto de división de entre los m .
 - iii. Dividir el nodo en dos nodos hijos.

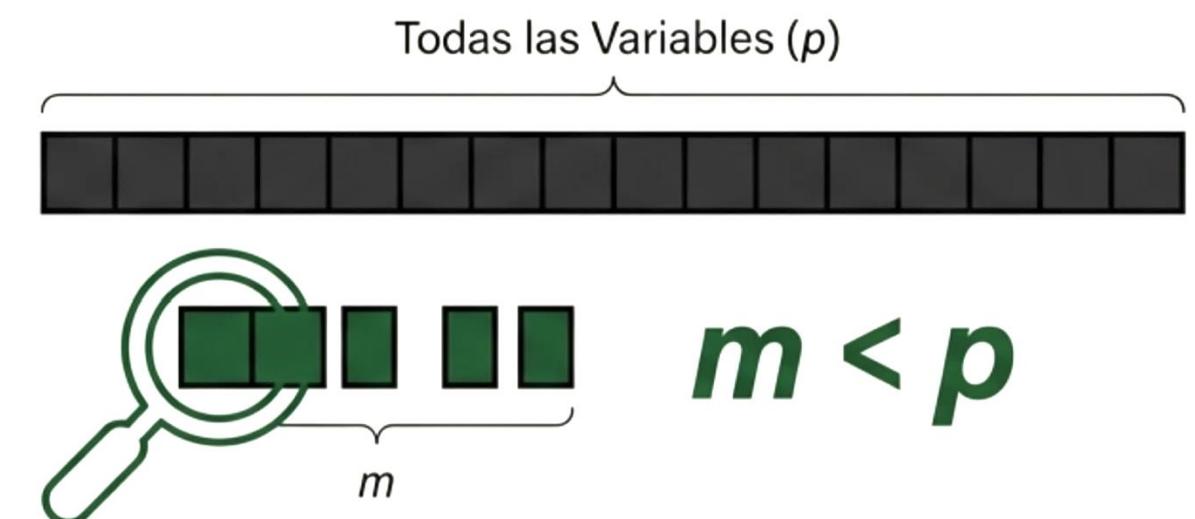
2. Dar como salida el ensamble de árboles $\{T_b\}_1^B$.

Para hacer una predicción en un nuevo punto x :

$$\text{Regresión: } \hat{f}_{\text{rf}}^B(x) = \frac{1}{B} \sum_{b=1}^B T_b(x).$$

Clasificación: Sea $\hat{C}_b(x)$ la predicción de clase en el b ésimo árbol aleatorio. Entonces $\hat{C}_{\text{rf}}^B(x) = \text{mayoria } \{\hat{C}_b(x)\}_1^B$.

El Concepto



En cada nodo, el algoritmo no evalúa todas las variables posibles (p). Se restringe aleatoriamente a un subconjunto de tamaño m .

Al forzar al árbol a considerar divisiones subóptimas, se "descorrelacionan" los árboles del bosque, haciendo el ensamble final más robusto.



Random Forests - Algorítmoo

1. Para $b = 1$ hasta B :
 - (a) Extraer una muestra bootstrap \mathbf{Z}^* de tamaño N a partir de los datos de entrenamiento.
 - (b) Hacer crecer un árbol T_b del bosque aleatorio con los datos bootstrappeados, repitiendo recursivamente los siguientes pasos para cada nodo terminal del árbol, hasta que el tamaño mínimo de nodo n_{min} sea alcanzado.
 - i. Seleccionar m variables al azar de entre las p variables.
 - ii. Escoger la mejor variable/punto de división de entre los m .
 - iii. Dividir el nodo en dos nodos hijos.
2. Dar como salida el ensamble de árboles $\{T_b\}_1^B$.

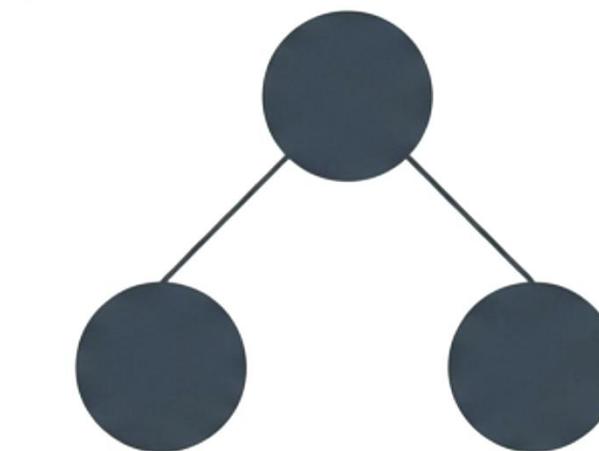
Para hacer una predicción en un nuevo punto x :

$$\text{Regresión: } \hat{f}_{\text{rf}}^B(x) = \frac{1}{B} \sum_{b=1}^B T_b(x).$$

Clasificación: Sea $\hat{C}_b(x)$ la predicción de clase en el b ésimo árbol aleatorio. Entonces $\hat{C}_{\text{rf}}^B(x) = \text{mayoria } \{\hat{C}_b(x)\}_1^B$.

El Concepto

Optimización sobre m variables



Nodos Hijos (Daughter Nodes)

Optimización: Dentro del subconjunto m , se busca la variable y el punto de corte que maximicen la pureza (Clasificación) o minimicen la varianza (Regresión).

Bifurcación: El nodo padre se divide en dos. El ciclo vuelve al paso 1(b) para cada hijo hasta alcanzar n_{min} .



Random Forests - Algorítmoo

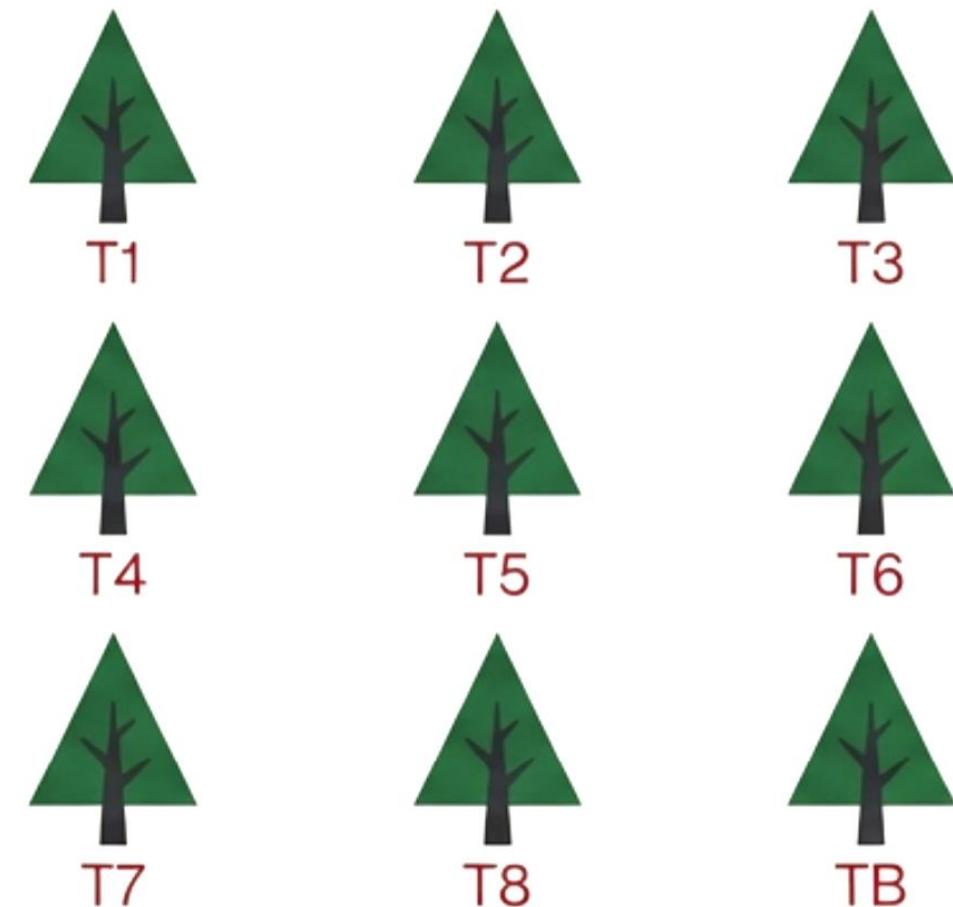
1. Para $b = 1$ hasta B :
 - (a) Extraer una muestra bootstrap \mathbf{Z}^* de tamaño N a partir de los datos de entrenamiento.
 - (b) Hacer crecer un árbol T_b del bosque aleatorio con los datos bootstrappeados, repitiendo recursivamente los siguientes pasos para cada nodo terminal del árbol, hasta que el tamaño mínimo de nodo n_{min} sea alcanzado.
 - i. Seleccionar m variables al azar de entre las p variables.
 - ii. Escoger la mejor variable/punto de división de entre los m .
 - iii. Dividir el nodo en dos nodos hijos.
2. Dar como salida el ensamble de árboles $\{T_b\}_1^B$.

Para hacer una predicción en un nuevo punto x :

$$\text{Regresión: } \hat{f}_{\text{rf}}^B(x) = \frac{1}{B} \sum_{b=1}^B T_b(x).$$

Clasificación: Sea $\hat{C}_b(x)$ la predicción de clase en el b ésimo árbol aleatorio. Entonces $\hat{C}_{\text{rf}}^B(x) = \text{mayoria } \{\hat{C}_b(x)\}_1^B$.

El Concepto



El resultado del entrenamiento es la colección completa de árboles $\{T_b\}$. El modelo ahora está "congelado" y listo para recibir nuevos datos (x) para realizar predicciones.



Random Forests - Algorítmoo

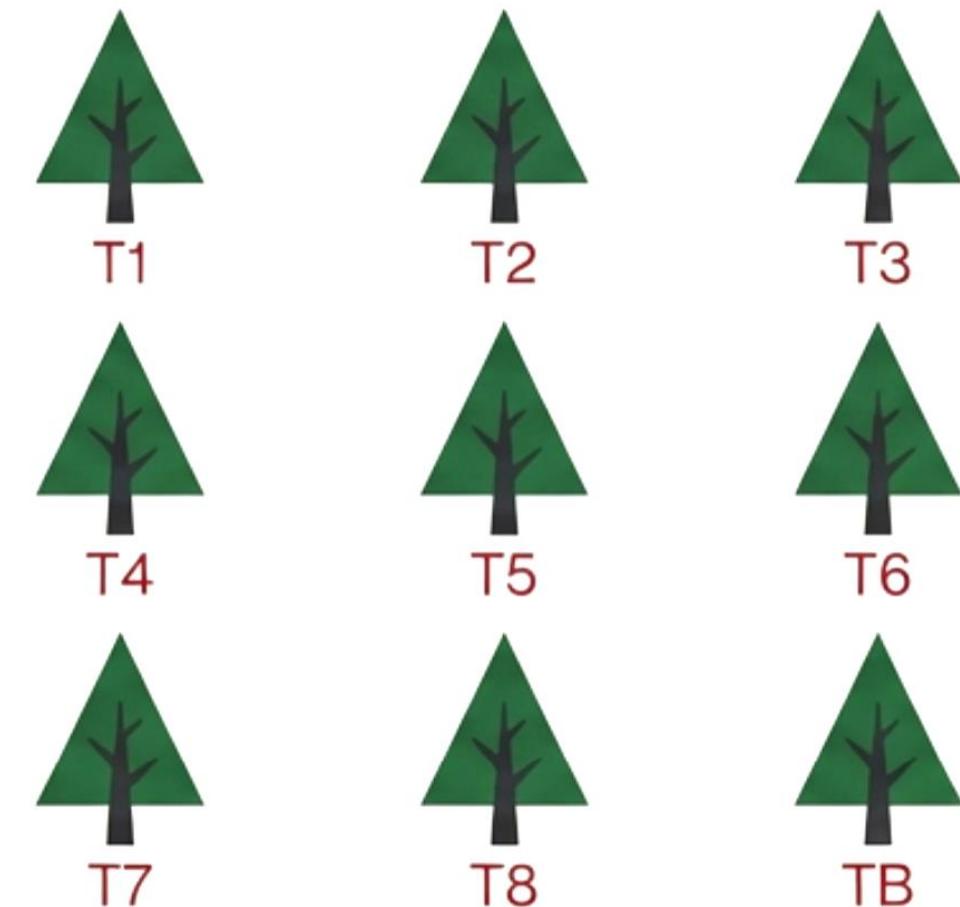
1. Para $b = 1$ hasta B :
 - (a) Extraer una muestra bootstrap \mathbf{Z}^* de tamaño N a partir de los datos de entrenamiento.
 - (b) Hacer crecer un árbol T_b del bosque aleatorio con los datos bootstrappeados, repitiendo recursivamente los siguientes pasos para cada nodo terminal del árbol, hasta que el tamaño mínimo de nodo n_{min} sea alcanzado.
 - i. Seleccionar m variables al azar de entre las p variables.
 - ii. Escoger la mejor variable/punto de división de entre los m .
 - iii. Dividir el nodo en dos nodos hijos.
2. Dar como salida el ensamble de árboles $\{T_b\}_1^B$.

Para hacer una predicción en un nuevo punto x :

$$\text{Regresión: } \hat{f}_{\text{rf}}^B(x) = \frac{1}{B} \sum_{b=1}^B T_b(x).$$

Clasificación: Sea $\hat{C}_b(x)$ la predicción de clase en el b ésimo árbol aleatorio. Entonces $\hat{C}_{\text{rf}}^B(x) = \text{mayoria } \{\hat{C}_b(x)\}_1^B$.

El Concepto



El resultado del entrenamiento es la colección completa de árboles $\{T_b\}$. El modelo ahora está "congelado" y listo para recibir nuevos datos (x) para realizar predicciones.



Random Forests - Algorítmoo

1. Para $b = 1$ hasta B :
 - (a) Extraer una muestra bootstrap \mathbf{Z}^* de tamaño N a partir de los datos de entrenamiento.
 - (b) Hacer crecer un árbol T_b del bosque aleatorio con los datos bootstrappeados, repitiendo recursivamente los siguientes pasos para cada nodo terminal del árbol, hasta que el tamaño mínimo de nodo n_{min} sea alcanzado.
 - i. Seleccionar m variables al azar de entre las p variables.
 - ii. Escoger la mejor variable/punto de división de entre los m .
 - iii. Dividir el nodo en dos nodos hijos.
2. Dar como salida el ensamble de árboles $\{T_b\}_1^B$.

Para hacer una predicción en un nuevo punto x :

$$\text{Regresión: } \hat{f}_{\text{rf}}^B(x) = \frac{1}{B} \sum_{b=1}^B T_b(x).$$

Clasificación: Sea $\hat{C}_b(x)$ la predicción de clase en el b ésimo árbol aleatorio. Entonces
 $\hat{C}_{\text{rf}}^B(x) = \text{mayoria } \{\hat{C}_b(x)\}_1^B$.

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

Predicción final del Random Forest para una nueva entrada x .
 indica que está basado en un conjunto de B árboles.
 todas las variables aleatorias y parámetros que definen al árbol b
 individual (el árbol número b) para la entrada x .

Indica "Random Forest"



Random Forests - Algorítmoo

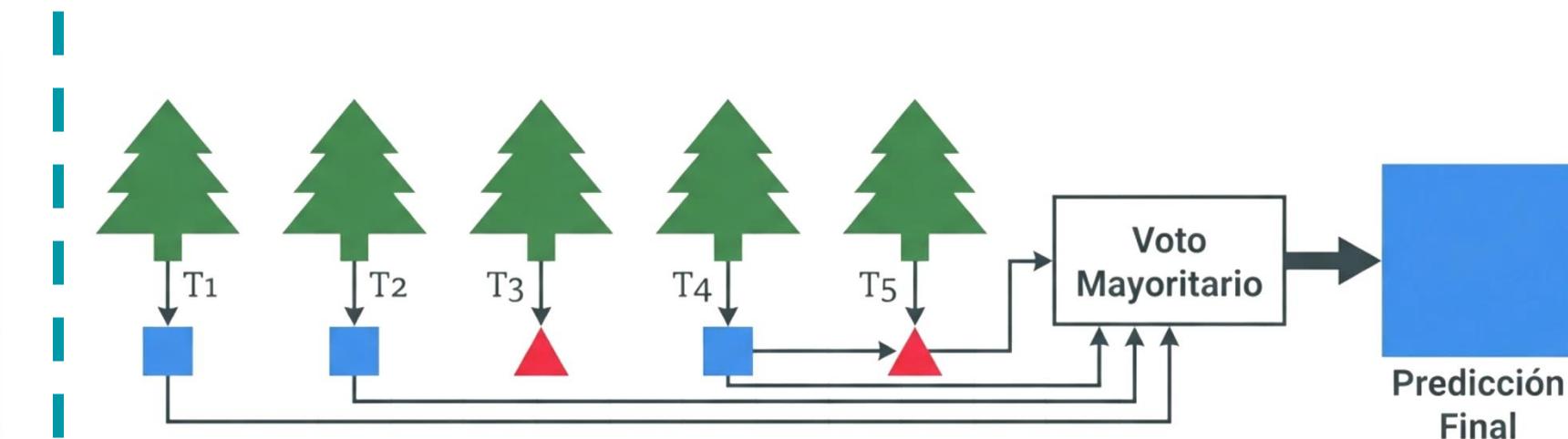
1. Para $b = 1$ hasta B :
 - (a) Extraer una muestra bootstrap \mathbf{Z}^* de tamaño N a partir de los datos de entrenamiento.
 - (b) Hacer crecer un árbol T_b del bosque aleatorio con los datos bootstrappeados, repitiendo recursivamente los siguientes pasos para cada nodo terminal del árbol, hasta que el tamaño mínimo de nodo n_{min} sea alcanzado.
 - i. Seleccionar m variables al azar de entre las p variables.
 - ii. Escoger la mejor variable/punto de división de entre los m .
 - iii. Dividir el nodo en dos nodos hijos.
2. Dar como salida el ensamble de árboles $\{T_b\}_1^B$.

Para hacer una predicción en un nuevo punto x :

$$\text{Regresión: } \hat{f}_{\text{rf}}^B(x) = \frac{1}{B} \sum_{b=1}^B T_b(x).$$

Clasificación: Sea $\hat{C}_b(x)$ la predicción de clase en el b ésimo árbol aleatorio. Entonces

$$\hat{C}_{\text{rf}}^B(x) = \text{mayoria } \{\hat{C}_b(x)\}_1^B.$$



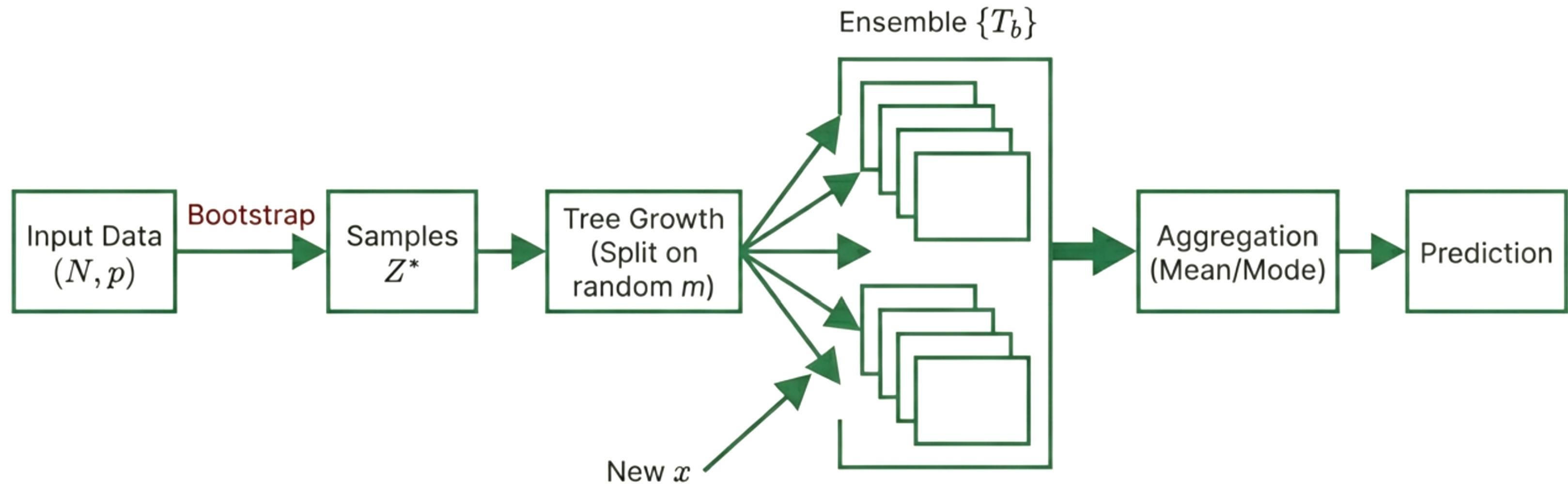
$\hat{c}_b(x)$: La clase asignada por un solo árbol (ej. "Fraude" vs "No Fraude").

mayoria: La clase final es la moda estadística del conjunto de predicciones.

La "**sabiduría de la multitud**" asegura que el consenso es más preciso que cualquier árbol individual.



Random Forests - Arquitectura





UTEC Posgrado

