Arboles de Desición

Minería de Datos (CC3074) - 2026

Árboles de Decisión

Semestre 01, 2026

El Problema

- Tenemos datos de pacientes: edad, presión arterial, colesterol.
- Queremos predecir si un paciente tiene riesgo cardíaco.
- ¿Cómo toma esa decisión un sistema de forma automática y explicable?

Intuición

- Los humanos tomamos decisiones haciendo preguntas encadenadas.
- "¿Tiene más de 50 años? → SÍ → ¿Presión alta? → SÍ → Alto riesgo."
- Un árbol de decisión formaliza exactamente ese proceso.

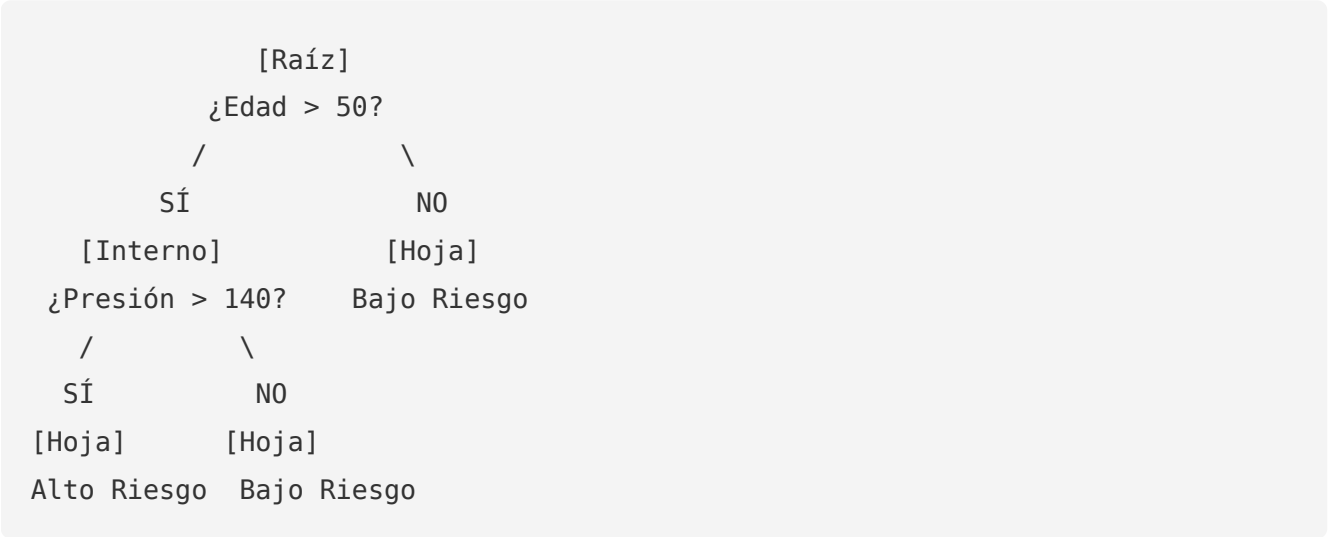
¿Por qué no usar regresión logística?

- La regresión produce una ecuación: difícil de explicar a alguien sin matemáticas.
- El árbol produce un diagrama de preguntas: cualquier persona lo puede leer.
- Interpretabilidad es la ventaja principal de los árboles.

Definición

- Modelo de aprendizaje supervisado para clasificación y regresión.
- Aprende reglas de decisión simples directamente de los datos.
- Resultado: un diagrama de flujo que se puede leer y explicar.

Estructura



- **Nodo Raíz:** Primera pregunta, sobre todos los datos.
- **Nodos Internos:** Preguntas sobre una característica.
- **Ramas:** Resultado de la condición (Sí / No).
- **Nodos Hoja:** Predicción final.

Funcionamiento

1. Se parte del nodo raíz con todos los datos.
2. Se elige la característica que mejor separa los grupos.
3. Se divide recursivamente hacia abajo.
4. Se detiene al alcanzar una condición de parada.
5. Cada hoja entrega una predicción o clase.

División (Splitting)

- Queremos grupos lo más homogéneos posible tras cada corte.
- Un nodo puro contiene solo elementos de una clase.
- Un nodo impuro mezcla clases → predicción poco confiable.

| Situación del nodo | Calidad del corte | | ---------- | ---------- | | Todos de una clase | Excelente (puro) | | Mayoría en una clase | Aceptable | | Mitad y mitad | Inútil |

Criterio: Gini Impurity

- Mide la probabilidad de clasificar mal un elemento elegido al azar.
- Rango: 0 (nodo puro) → 0.5 (máximo desorden).

Fórmula: $Gini = 1 - \Sigma(p_i^2)$

Ejemplo: 4 datos — 3 "Jugar", 1 "No Jugar".

$Gini = 1 - (0.75^2 + 0.25^2) = 1 - (0.5625 + 0.0625) = 0.375$

Criterio: Entropía

- Mide el desorden o incertidumbre de los datos.
- Basada en teoría de información de Shannon.
- Rango: 0 (puro) → 1 (máximo desorden, base 2).

Fórmula: $Entropía = -\Sigma(p_i \cdot \log_2(p_i))$

Ejemplo: mismos 4 datos (3 Jugar, 1 No Jugar).

$Entropía = -(0.75 \cdot \log_2(0.75) + 0.25 \cdot \log_2(0.25)) \approx 0.811$

Criterio: MSE (Regresión)

- Se usa cuando la variable objetivo es numérica, no una clase.
- Mide el error cuadrático medio dentro de cada nodo.
- El corte que minimice el MSE promedio ponderado es el elegido.

Ejemplo: ¿Jugar al Tenis?

| Clima | Humedad | Viento | ¿Jugar? | | ----- | ------- | ------ | ------- | | Soleado | Alta | No | No | | Nublado | Alta | No | Sí | | Lluvia | Normal | No | Sí | | Soleado | Normal | No | Sí |

El algoritmo calcula el Gini para cada atributo y elige el que produce nodos más puros.

Árbol Resultante

```
          ¿Clima?
        /    |    \
  Nublado Soleado  Lluvia
     |        |       |
   [Sí]  ¿Humedad?  [Sí]
         /       \
      Alta     Normal
      [No]      [Sí]
```

- "Clima" fue elegido primero: menor Gini de todos los atributos.
- Solo el subconjunto "Soleado" requería una segunda pregunta.

Overfitting

El Árbol Sin Restricciones

- Sin límites, el árbol crece hasta memorizar cada fila del dataset.
- Crea una regla distinta para cada ejemplo de entrenamiento.
- Error en entrenamiento ≈ 0%, pero falla con datos nuevos.

Comparación

| Característica | Sin Restricciones | Árbol Optimizado | | ---------- | ---------- | ---------- | | Profundidad | Máxima posible | Limitada | | Reglas aprendidas | Una por fila | Patrones generales | | Error en entrenamiento | ~0% | Moderado |

Comparación

| Característica | Sin Restricciones | Árbol Optimizado | | ---------- | ---------- | ---------- | | Error en datos nuevos | Alto | Bajo | | Utilidad real | Pobre | Buena |

Solución

Dos estrategias complementarias:

- **Poda (Pruning):** Eliminar ramas que aportan poco valor.
- **Hiperparámetros:** Imponer límites antes de entrenar.

Poda (Pruning)

- Técnica para reducir el tamaño del árbol sin perder capacidad predictiva.
- Elimina ramas cuya contribución no justifica su complejidad.
- Objetivo: mejorar la generalización a datos nuevos.

Pre-Pruning (Parada Temprana)

- Detiene el crecimiento durante el entrenamiento.

- Condiciones típicas: profundidad máxima, mínimo de datos por nodo.
- Ventaja: Más eficiente computacionalmente.
- Riesgo: Puede detener el árbol antes de aprender patrones útiles.

Post-Pruning

- Deja crecer el árbol completo y luego elimina ramas redundantes.
- Evalúa si quitar una rama mejora el rendimiento en datos de validación.
- Ventaja: Considera el árbol entero antes de podar.
- Riesgo: Más costoso; requiere un conjunto de validación separado.

Hiperparámetros

- Configuraciones que definimos antes de entrenar el modelo.
- No se aprenden de los datos; son decisiones del diseñador.
- Controlan el crecimiento del árbol para evitar sobreajuste.

`max_depth`: Profundidad Máxima

- Límite de niveles de preguntas que puede tener el árbol.
- Es el parámetro con mayor impacto sobre el overfitting.

| `max_depth` | Efecto | | ---------- | ------ | | Sin límite | Memoriza los datos | | Muy bajo (1–2) | Underfitting, modelo muy simple | | Moderado (3–8) | Balance habitual |

`min_samples_leaf`

- Número mínimo de registros que debe contener un nodo hoja.
- Evita que el modelo cree reglas basadas en 1 o 2 ejemplos aislados.

Ejemplo: `min_samples_leaf = 10` → ninguna hoja puede tener menos de 10 datos.

`min_samples_split`

- Cantidad mínima de datos para que un nodo pueda dividirse.
- Si un nodo tiene menos datos que este valor → se convierte en hoja automáticamente.
- Actúa como filtro de relevancia antes de cada división.

Optimización

Grid Search (Búsqueda en Rejilla)

- Definimos una lista de valores candidatos por hiperparámetro.
- El sistema entrena y evalúa todas las combinaciones posibles.
- Se elige la combinación con mejor rendimiento promedio.

| `max_depth` | `min_samples_leaf` | `min_samples_split` | | ---------- | ---------- | ---------- | | 3 | 10 | 20 | | 5 | 20 | 50 | | 8 | 50 | 100 |

Validación Cruzada (K-Fold)

- Divide los datos en K partes (folds) iguales.
- Entrena con $K-1$ partes y evalúa con la restante.
- Repite K veces, rotando el fold de evaluación.
- El resultado final es el promedio de los K errores.

¿Por qué combinar ambos?

- Grid Search sin validación cruzada puede elegir parámetros que funcionan "por suerte".
- K-Fold garantiza que el resultado sea estable y confiable.
- Juntos: búsqueda exhaustiva + evaluación robusta.

Ventajas y Limitaciones

Ventajas

- **Interpretable:** Cualquier persona puede leer el árbol y entender la decisión.
- **Versátil:** Funciona para clasificación y regresión.
- **Sin preprocesamiento complejo:** Tolera valores faltantes y variables mixtas.
- **Rápido:** Entrenamiento e inferencia eficientes.

Limitaciones

- **Inestable:** Pequeños cambios en los datos pueden producir árboles muy distintos.
- **Tendencia al overfitting:** Crece demasiado sin restricciones.
- **Sesgado:** Favorece atributos con muchos valores posibles.
- **Modelo base débil:** Solo; suele superarse con métodos ensemble (Random Forest).