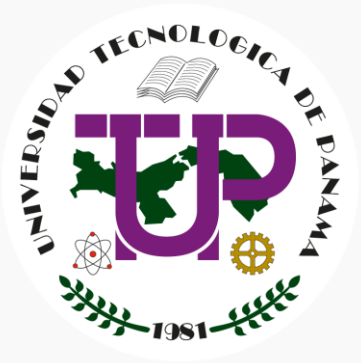


Universidad Tecnológica de Panamá
Facultad de Ingeniería Eléctrica
Maestría en Ingeniería Eléctrica

Semana 2

Introducción a Machine Learning

Árboles de decisión

Árboles de decisión

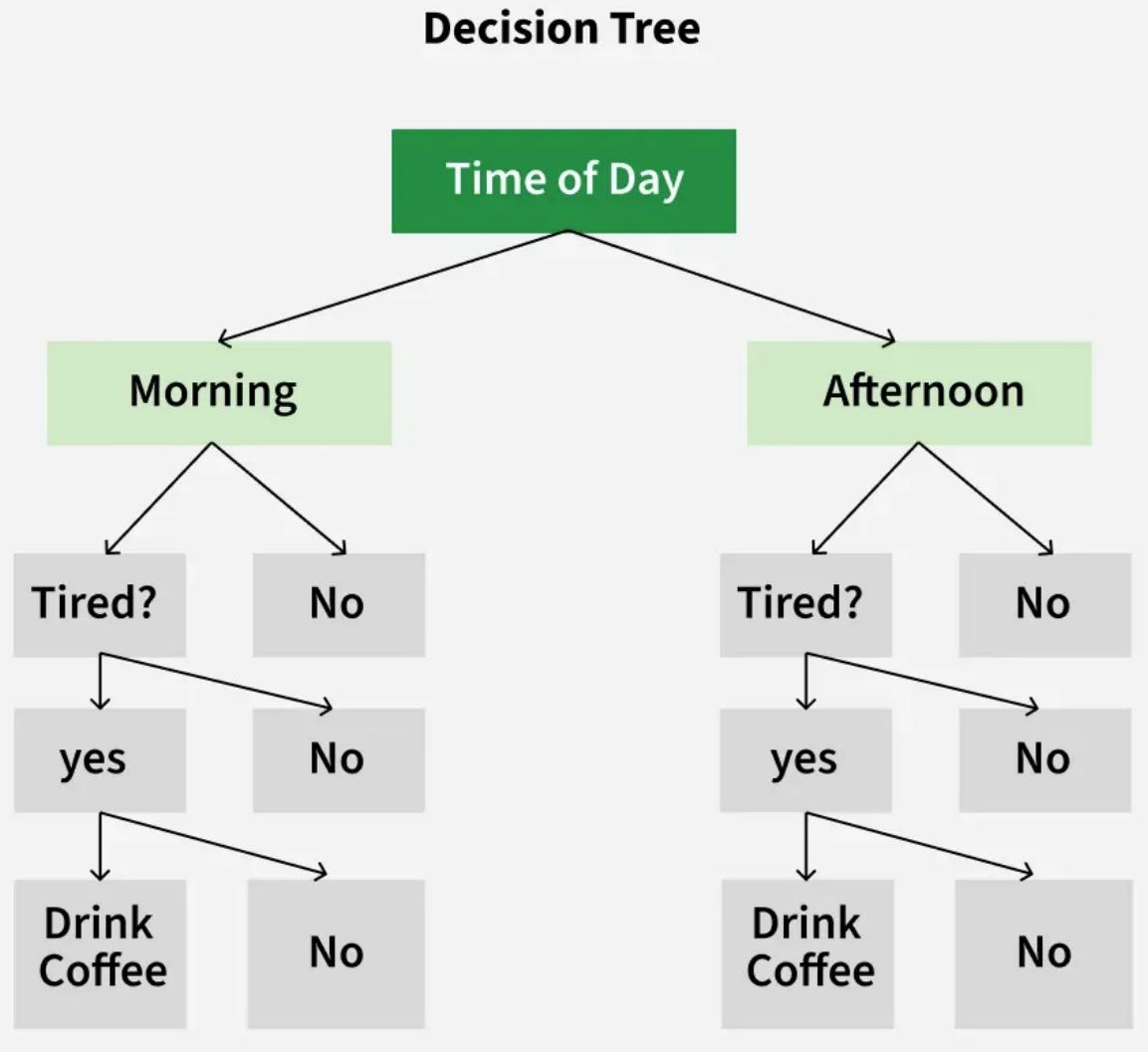
Definición básica

Es una estructura en forma de árbol en la que:

- Cada **nodo interno** representa una prueba o condición sobre un atributo (por ejemplo, "¿edad > 50?").
- Cada **rama** representa el resultado de esa prueba (por ejemplo, "sí" o "no").
- Cada **hoja** representa una **predicción de clase** (para clasificación) o un valor (para regresión).

El modelo divide recursivamente el espacio de los datos en regiones más homogéneas respecto a la variable de salida.

Árboles de decisión



Árboles de decisión

Ventajas

- Muy **interpretables** (explicables).
- Soportan **datos categóricos y continuos**.
- No requieren escalado de las variables.

Desventajas

- Sensibles a los cambios en los datos (**overfitting** si no se poda el árbol).
- No generalizan tan bien como algunos modelos más complejos.

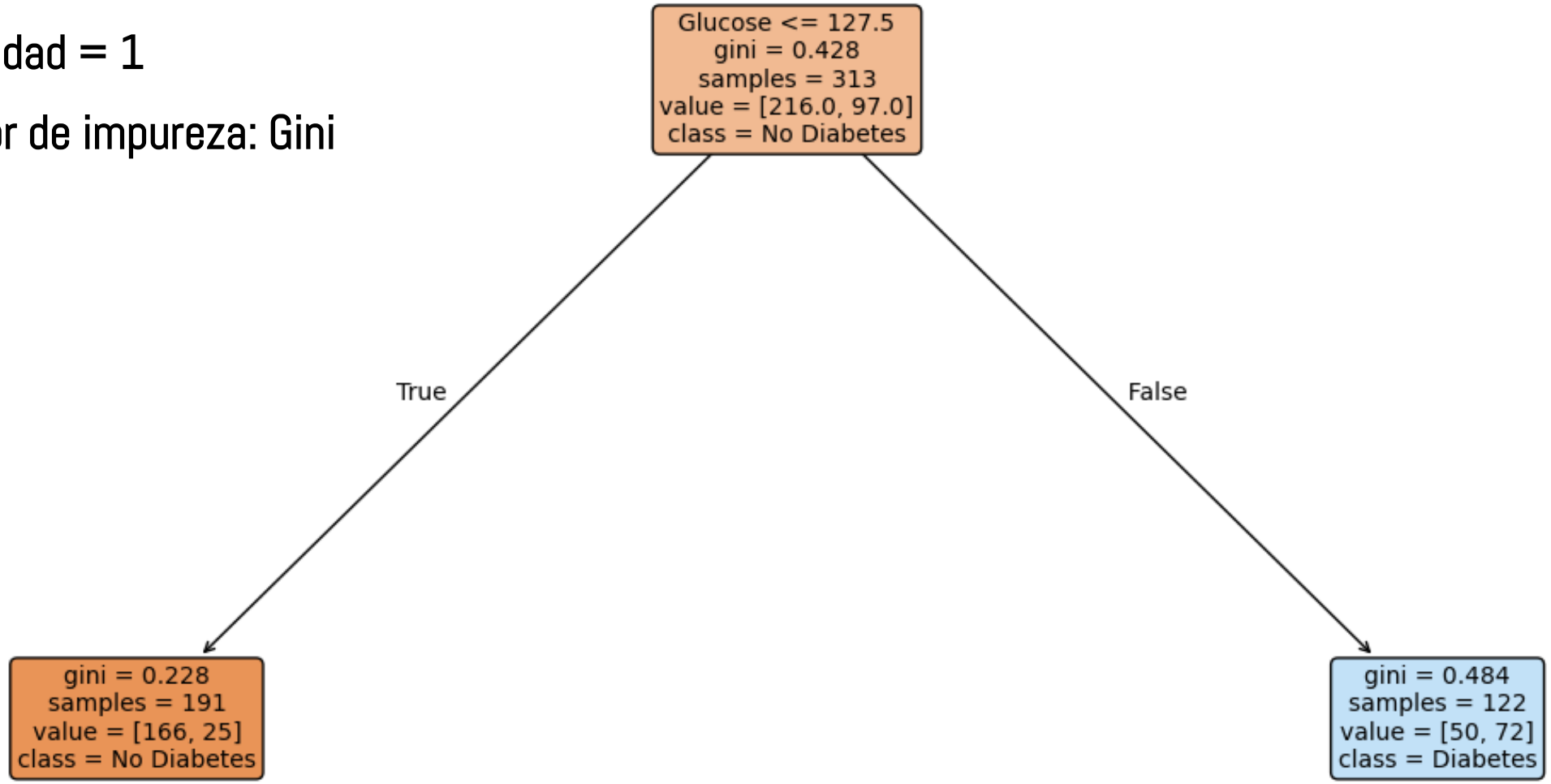
Árboles de decisión

Cuándo se usan

Uso típico	Justificación
Cuando se necesita **explicabilidad**	Fácil de visualizar y entender.
Datos con mezclas de variables categóricas y continuas	No requieren preprocesamiento complejo.
Prototipos rápidos	Entrenamiento y predicción rápidos.

Ejemplo: Pima Indian Diabetes Dataset

- Profundidad = 1
- Indicador de impureza: Gini



Indicador de impureza

- El **indicador de impureza** es una **medida matemática** que evalúa **qué tan mezcladas están las clases** dentro de un nodo en un árbol de decisión.
- Su propósito es ayudar al algoritmo a decidir **dónde dividir** los datos para obtener subgrupos lo más "puros" posible, es decir, que contengan principalmente ejemplos de una sola clase.

¿Qué significa "impureza"?

- En clasificación binaria (por ejemplo, diabetes vs. no diabetes):
- Un nodo es “**puro**” si **todas las observaciones pertenecen a la misma clase**.
- Un nodo es “**impuro**” si las clases están mezcladas.

Tipos comunes de indicadores de impureza

- **Índice de Gini:** Mide qué tan mezcladas están las clases dentro de un grupo. Cuanto más bajo es el valor, más puro es el nodo. Es el más usado por defecto en árboles de decisión.
- **Entropía:** Evalúa el nivel de desorden o incertidumbre en un grupo de datos. Si hay muchas clases mezcladas, la entropía es alta. Se basa en el concepto de información de la teoría de Shannon.
- **Error de clasificación:** Indica qué proporción de ejemplos en un nodo no pertenecen a la clase mayoritaria. Es una medida más simple, pero menos sensible para hacer divisiones.

Índice de Gini

El **índice de Gini** es una medida de impureza utilizada en árboles de decisión para evaluar qué tan homogéneo es un conjunto de datos en un nodo. Su fórmula es:

$$\text{Gini} = 1 - \sum_{i=1}^{C} p_i^2$$

donde:

- C es el número total de clases (por ejemplo, 2 si es binaria),
- p_i es la **proporción** de elementos de la clase i en ese nodo.

Índice de Gini

- **Gini = 0**: el nodo es puro (todos los ejemplos son de la misma clase).
- **Gini cercano a 0.5**: las clases están mezcladas de forma equilibrada (en clasificación binaria).
- El árbol busca divisiones que **minimicen** el valor del índice de Gini.

Ejemplo

Si en un nodo hay 70% de clase 0 y 30% de clase 1:

$$\text{Gini} = 1 - (0.7^2 + 0.3^2) = 1 - (0.49 + 0.09) = 0.42$$

Este valor indica que el nodo no es completamente puro, pero tiene una ligera preferencia por la clase 0.

Entropía

Definición general

La **entropía** mide el **grado de incertidumbre** o **impureza** en un conjunto de ejemplos.

En términos simples:

- Si todos los ejemplos pertenecen a la misma clase → **entropía baja (cero)** → no hay incertidumbre.
- Si las clases están equilibradas (por ejemplo, 50% de cada clase) → **entropía alta** → alta incertidumbre.

Modelo de la entropía

Dado un conjunto S con C clases posibles, la entropía se modela como:

$$\text{Entropía}(S) = -\sum_{i=1}^{C} p_i \log_2(p_i)$$

donde:

- p_i es la proporción de ejemplos de la clase i en el conjunto S.
- Por convención, si $p_i = 0$, se define $0 \log 0 = 0$.

Ejemplo práctico (binario)

Supongamos un conjunto S de ejemplos con dos clases:

- 8 ejemplos de clase 1
- 2 ejemplos de clase 0

Entonces:

$$p_1 = \frac{8}{10} = 0.8, \quad p_0 = \frac{2}{10} = 0.2$$

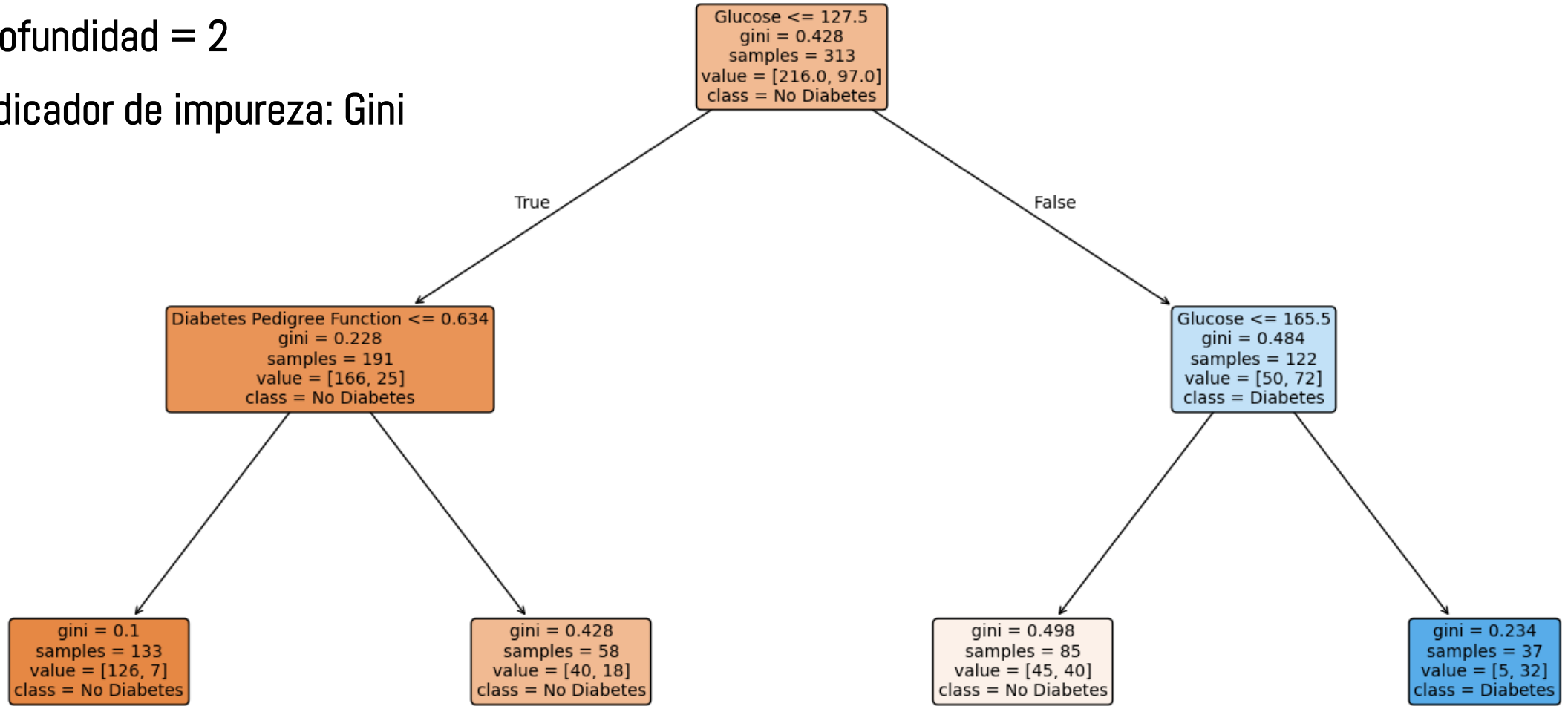
El modelo de entropía sería:

$$\text{Entropía}(S) = -\left(0.8 \log_2(0.8) + 0.2 \log_2(0.2)\right)$$

$$\text{Entropía}(S) \approx 0.7219$$

Ejemplo: Pima Indian Diabetes Dataset

- Profundidad = 2
- Indicador de impureza: Gini



¿Qué es la **profundidad** de un árbol de decisión?

La **profundidad** (o *depth* en inglés) de un árbol de decisión es la **longitud del camino más largo desde la raíz hasta una hoja**.

Estructura básica

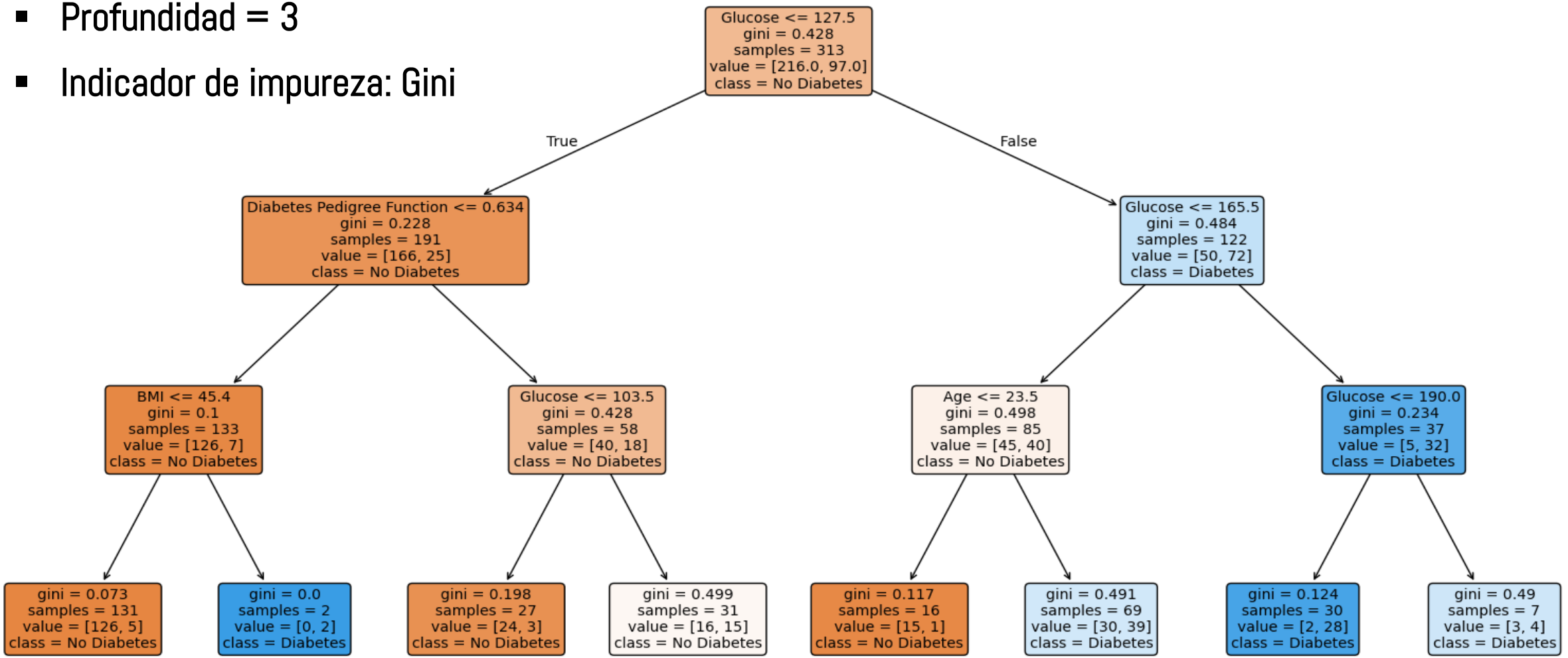
- El **nodo raíz** (donde comienza el árbol) está en la **profundidad 0**.
- Cada vez que se hace una división, se **incrementa la profundidad en 1**.
- Un nodo **hoja** (donde ya no se divide más) está al final de ese camino.

¿Por qué es importante?

- Una **mayor profundidad** permite al árbol **aprender patrones más complejos**, pero también puede llevar a **sobreajuste** (memoriza los datos).
- Una **menor profundidad** genera un árbol más simple, fácil de interpretar, pero puede **perder precisión**.

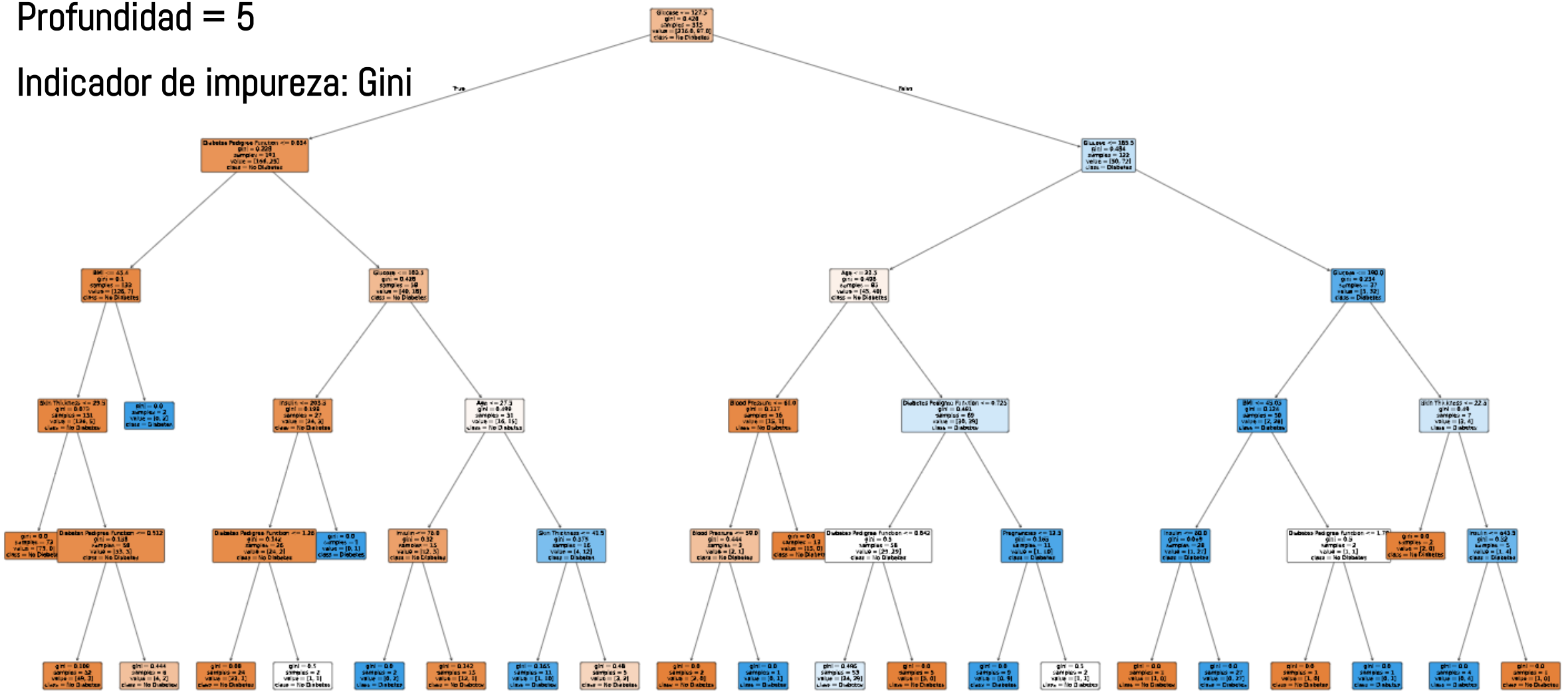
Ejemplo: Pima Indian Diabetes Dataset

- Profundidad = 3
- Indicador de impureza: Gini

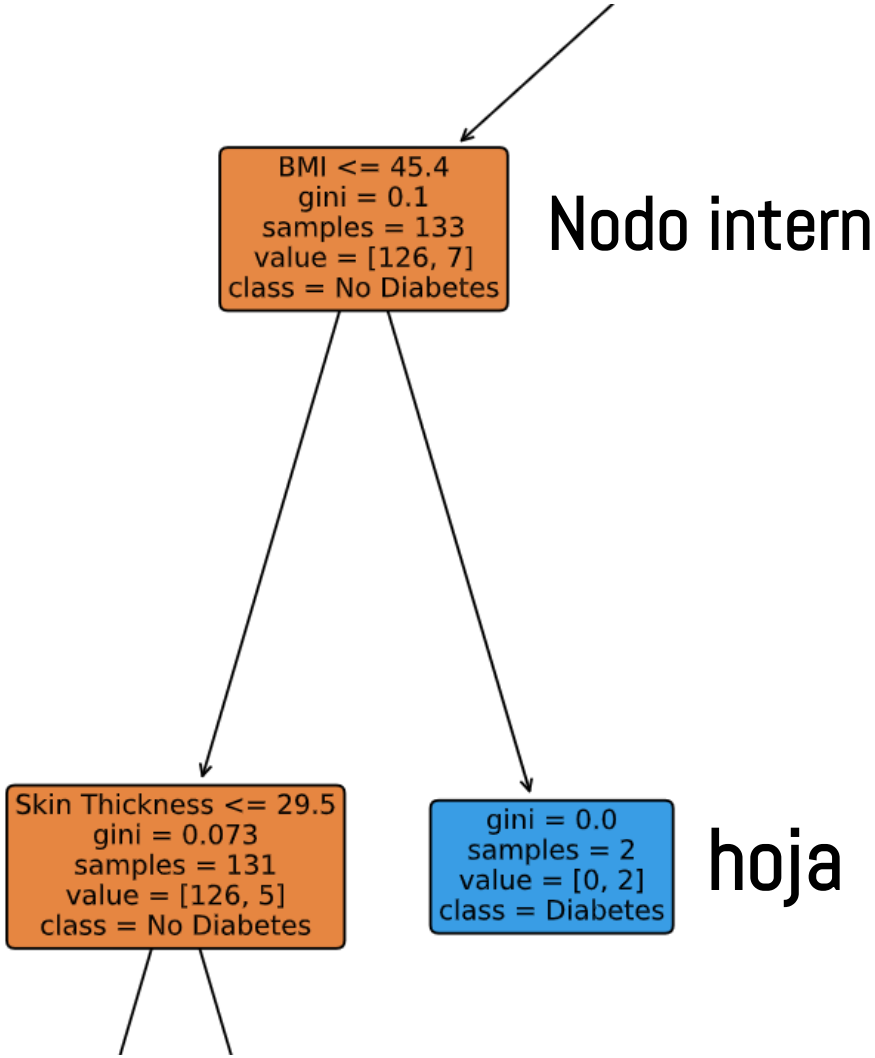


Ejemplo: Pima Indian Diabetes Dataset

- Profundidad = 5
- Indicador de impureza: Gini



Nodos internos y hojas



Nodo interno

hoja

Datos continuos y datos categóricos

¿Qué son los datos categóricos?

Son variables que toman un número **limitado de categorías o etiquetas**.

No representan cantidades, sino **clases** o **grupos**.

Ejemplos:

- Sexo: `masculino`, `femenino`
- Tipo de pago: `efectivo`, `tarjeta`, `cheque`
- Diagnóstico: `positivo`, `negativo`

Tratamiento en árboles:

- Se generan ramas para cada categoría (por ejemplo, una rama para "masculino", otra para "femenino").
- Si hay muchas categorías, a veces se agrupan para evitar árboles muy grandes o poco interpretables.

¿Qué es el overfitting?

El **overfitting** ocurre cuando un modelo **aprende demasiado bien los datos de entrenamiento**, incluyendo **ruido, excepciones o patrones irrelevantes**, y por eso **falla al generalizar** a nuevos datos (por ejemplo, el conjunto de prueba o datos reales).

Ejemplo en árboles de decisión

Un árbol de decisión que tiene **mucha profundidad** (por ejemplo, max_depth=10 o más) puede llegar a:

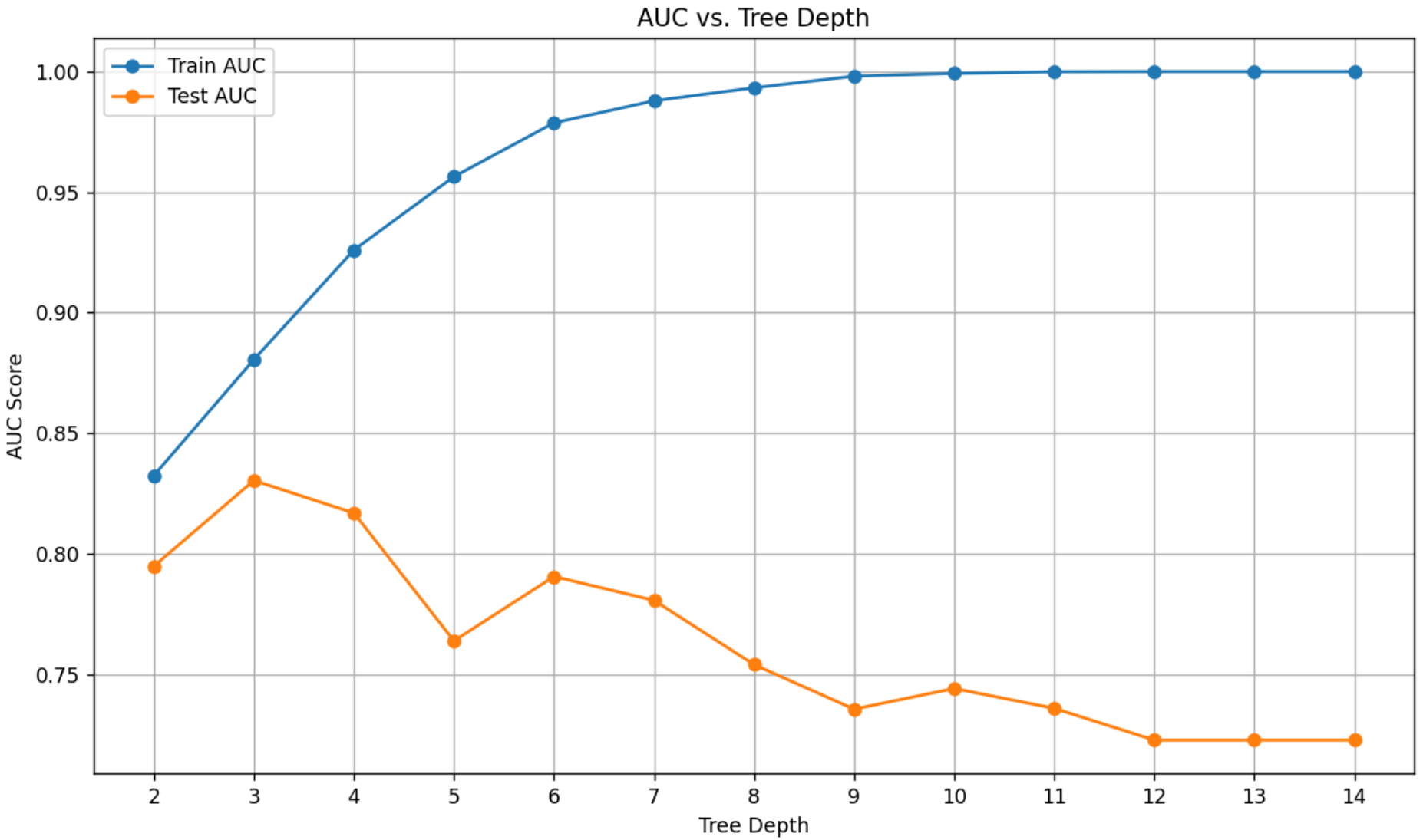
- Dividir los datos tantas veces que termina creando reglas **muy específicas** para unos pocos casos,
- Tener nodos hoja con **1 o 2 muestras**,
- Lograr **exactitud perfecta** en el entrenamiento, pero **mal desempeño** fuera de él.

Indicadores de overfitting

Un indicador de overtiffing es cuando un modelo funciona mucho mejor en el dataset de entrenamiento en comparación al dataset de pruebas.

Depth	Train AUC	Test AUC	Train Accuracy	Test Accuracy	Train Precision	Test Precision	Train Recall	Test Recall	Train F1	Test F1
2	0.8324	0.7948	0.7764	0.6962	0.8649	1.0000	0.3299	0.2727	0.4776	0.4286
3	0.8805	0.8304	0.8115	0.8228	0.6759	0.8276	0.7526	0.7273	0.7122	0.7742
4	0.9260	0.8169	0.8466	0.7468	0.8889	0.8095	0.5773	0.5152	0.7000	0.6296
5	0.9564	0.7638	0.8850	0.7595	0.7748	0.7333	0.8866	0.6667	0.8269	0.6984
6	0.9787	0.7905	0.9201	0.7722	0.8158	0.7419	0.9588	0.6970	0.8815	0.7188
7	0.9879	0.7806	0.9361	0.7722	0.9529	0.8000	0.8351	0.6061	0.8901	0.6897
8	0.9933	0.7540	0.9553	0.7722	0.9029	0.7586	0.9588	0.6667	0.9300	0.7097
9	0.9981	0.7355	0.9808	0.7468	0.9691	0.7407	0.9691	0.6061	0.9691	0.6667
10	0.9993	0.7441	0.9840	0.7468	1.0000	0.7600	0.9485	0.5758	0.9735	0.6552
11	0.9999	0.7358	0.9968	0.7595	0.9898	0.7692	1.0000	0.6061	0.9949	0.6780
12	1.0000	0.7227	1.0000	0.7468	1.0000	0.7600	1.0000	0.5758	1.0000	0.6552
13	1.0000	0.7227	1.0000	0.7468	1.0000	0.7600	1.0000	0.5758	1.0000	0.6552
14	1.0000	0.7227	1.0000	0.7468	1.0000	0.7600	1.0000	0.5758	1.0000	0.6552

Indicadores de overfitting



Feature importance

La **importancia de características (feature importance)** en los árboles de decisión mide cuánto contribuye cada variable a reducir la **impureza** (como el índice Gini o la entropía) a lo largo del árbol.

¿Cómo se calcula?

- **Cada vez que se divide un nodo**, el árbol elige una característica y un umbral que mejor reduce la impureza (Gini, entropía).
- La **reducción de impureza** lograda por esa división se **asigna como ganancia** a esa característica.
- Se **suma** esta ganancia para cada vez que la característica se usa en el árbol.
- Finalmente, las ganancias se **normalizan** (dividen por la suma total) para que todas las importancias sumen 1.

Pima Indian Diabetes Dataset

Depth = 3

Feature	Importance
Glucose	0.742541
Age	0.108523
Diabetes Pedigree Function	0.088684
BMI	0.060252
Pregnancies	0.000000
Blood Pressure	0.000000
Skin Thickness	0.000000
Insulin	0.000000

Depth = 5

Feature	Importance
Glucose	0.502935
Diabetes Pedigree Function	0.154384
Age	0.125948
Insulin	0.090955
BMI	0.049820
Skin Thickness	0.045803
Blood Pressure	0.020994
Pregnancies	0.009161

Cálculo de feature importance

La **reducción de impureza** en el nodo t es:

$$\Delta i(t) = w(t) \cdot i(t) - w_L(t) \cdot i_L(t) - w_R(t) \cdot i_R(t)$$

La **importancia de una característica** x_j es la suma de las reducciones de impureza en todos los nodos donde se usó:

$$\mathrm{FI}(x_j) = \sum_{t \in T : f_t = x_j} \Delta i(t)$$

Y la **importancia normalizada** (para que todas sumen 1) es:

$$\text{Normalized FI}(x_j) = \frac{\mathrm{FI}(x_j)}{\sum_k \mathrm{FI}(x_k)}$$

Sea T el conjunto de nodos internos del árbol. Para cada nodo $t \in T$, f_t es la característica usada para dividir, $i(t)$ su impureza antes de dividir, $i_L(t)$ e $i_R(t)$ las impurezas de los nodos hijo, y $w(t), w_L(t), w_R(t)$ las cantidades de muestras en el nodo y sus hijos.

Pruning

- **Pruning** (poda) en árboles de decisión es una técnica fundamental utilizada para reducir el tamaño del árbol y controlar su complejidad.
- Su objetivo principal es evitar el **sobreajuste** (*overfitting*), que ocurre cuando el modelo se ajusta demasiado a los datos de entrenamiento, capturando ruido o patrones irrelevantes, lo cual disminuye su capacidad de generalizar bien a datos nuevos.

Pruning

- Durante el entrenamiento, un árbol sin restricciones puede crecer hasta clasificar perfectamente los datos de entrenamiento, generando una estructura muy profunda y ramificada.
- Aunque este árbol puede mostrar alta precisión en entrenamiento, su rendimiento en datos no vistos suele deteriorarse.
- La poda actúa como una forma de regularización, eliminando ramas del árbol que aportan poca o ninguna mejora al rendimiento del modelo.
- Al simplificar la estructura, se obtiene un modelo más robusto, interpretable y con mejor capacidad de generalización.

Pruning

Existen dos enfoques principales para aplicar poda:

- la **poda anticipada** (*pre-pruning*), que detiene el crecimiento del árbol de forma temprana según ciertos criterios (como la profundidad máxima o el número mínimo de muestras por nodo)
- **La poda posterior** (*post-pruning*), que elimina ramas innecesarias después de construir el árbol completo, evaluando su impacto en el rendimiento mediante un conjunto de validación o una función de costo.

Ambos métodos buscan el equilibrio entre exactitud y simplicidad.

Extrategias de pruning

Pre-poda (pre-pruning)

Estas técnicas **detienen el crecimiento del árbol antes de completarlo**, según ciertas condiciones:

Parámetro	Descripción
max_depth	Profundidad máxima del árbol.
min_samples_split	Mínimo de muestras requerido para dividir un nodo.
min_samples_leaf	Mínimo de muestras requerido en cada hoja.
max_leaf_nodes	Número máximo de hojas permitidas.
max_features	Número máximo de características consideradas en cada división.

Extrategias de pruning

Post-poda (post-pruning)

Estas técnicas recortan un árbol ya construido:

Técnica	Descripción
ccp_alpha (cost-complexity)	Elimina ramas que no justifican su complejidad con ganancia en rendimiento.
Poda basada en validación	Se entrena el árbol completo y se podan ramas usando un conjunto de validación como referencia para conservar solo las divisiones útiles.

Cost Complexity Pruning alpha

- ccp_alpha (del inglés Cost Complexity Pruning alpha) es un parámetro de regularización en los árboles de decisión que se utiliza para controlar el proceso de poda (pruning), es decir, la reducción del tamaño del árbol después de su construcción inicial.
- Su propósito principal es evitar el overfitting, lo que ocurre cuando el árbol se adapta demasiado a los datos de entrenamiento, capturando ruido y patrones específicos que no generalizan bien a datos nuevos.

Cost Complexity Pruning alpha

Este parámetro se basa en el principio de equilibrio entre **precisión y simplicidad del modelo**. La idea es penalizar árboles más complejos que no aportan una mejora significativa en el rendimiento. Formalmente, se minimiza una función de costo que incorpora tanto el error del árbol como su tamaño:

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

donde:

- $R(T)$ es el error del árbol (por ejemplo, basado en impureza),
- $|T|$ es el número de hojas del árbol,
- α es el valor de `ccp_alpha`.

Extrategias de pruning

Otras técnicas indirectas de regularización

Método	Descripción
Random Forests	Usa múltiples árboles pequeños con diferentes subconjuntos de datos y características; esto reduce el overfitting.
Gradient Boosting	Combina árboles pequeños (generalmente con max_depth bajo) entrenados secuencialmente para mejorar el rendimiento.
Bagging	Promedia múltiples árboles sin poda, pero cada uno entrenado sobre distintos subconjuntos de datos.