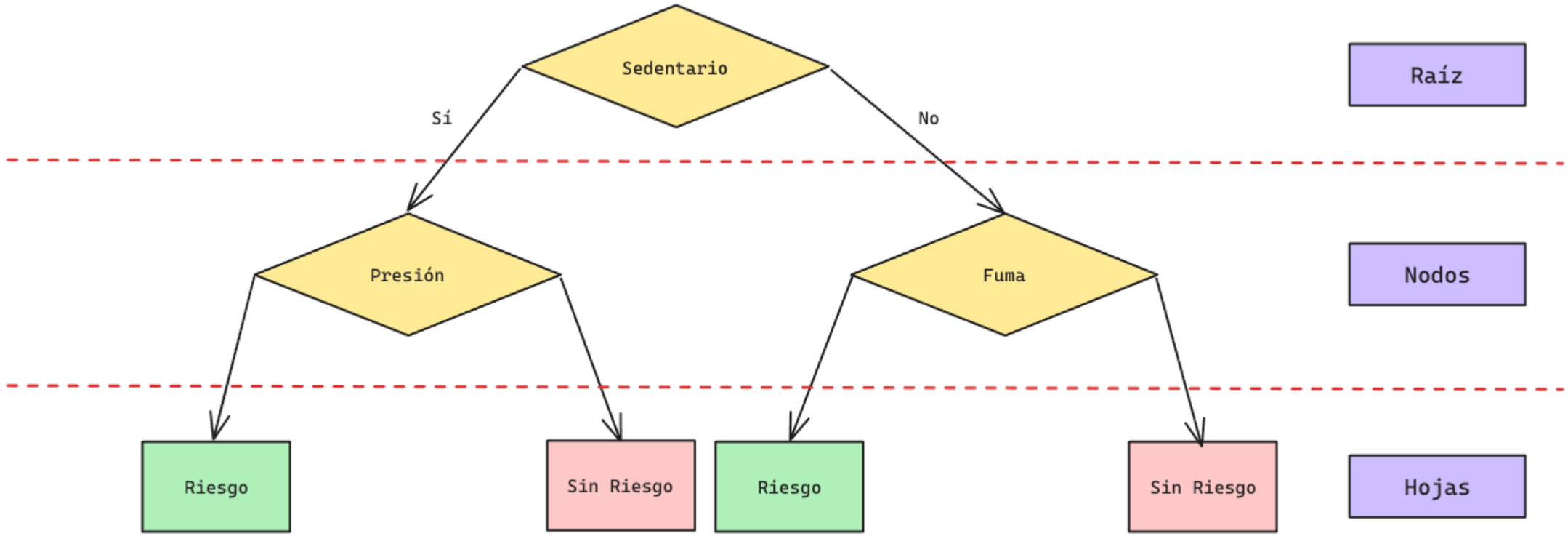
TICS-411 Minería de Datos

Clase 10: Árboles de Decisión

Alfonso Tobar-Arancibia
alfonso.tobar.a@edu.uai.cl

Árboles de Decisión

> Técnica de clasificación supervisada que genera una decisión basada en `árboles de decisión` para clasificar instancias no conocidas.

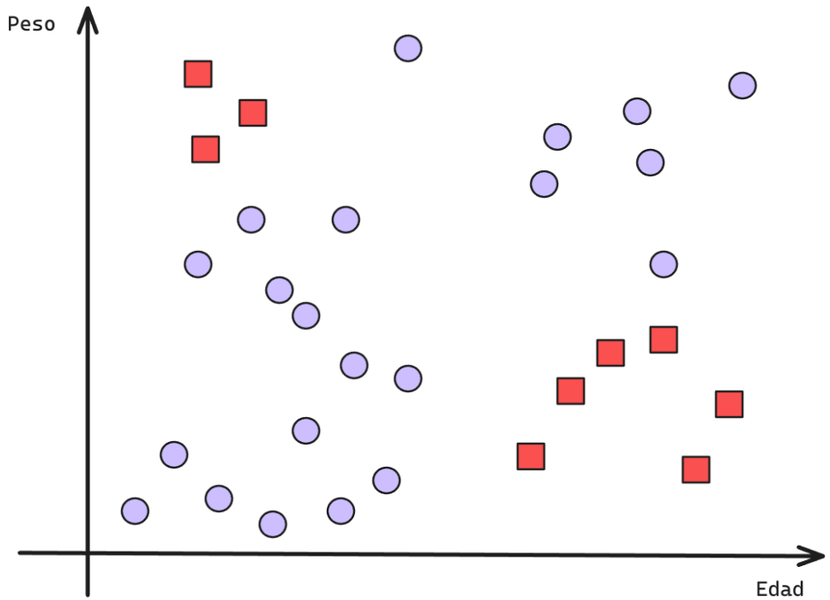


Árboles de Decisión: Ejemplo

> Visualmente, un árbol de decisión segmenta el espacio separando los datos en subgrupos.

Esto permite la generacion de fronteras de decisión sumamente complejas.

Supongamos el siguiente ejemplo:

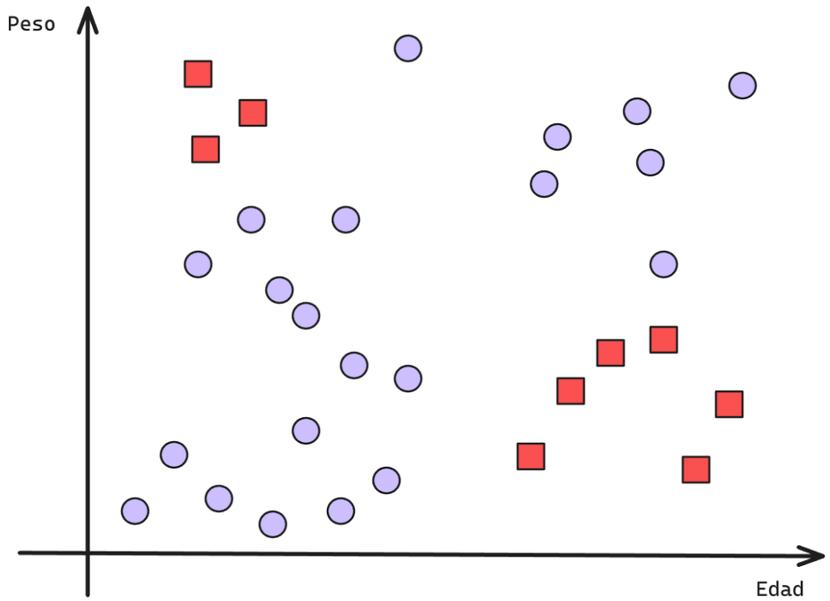


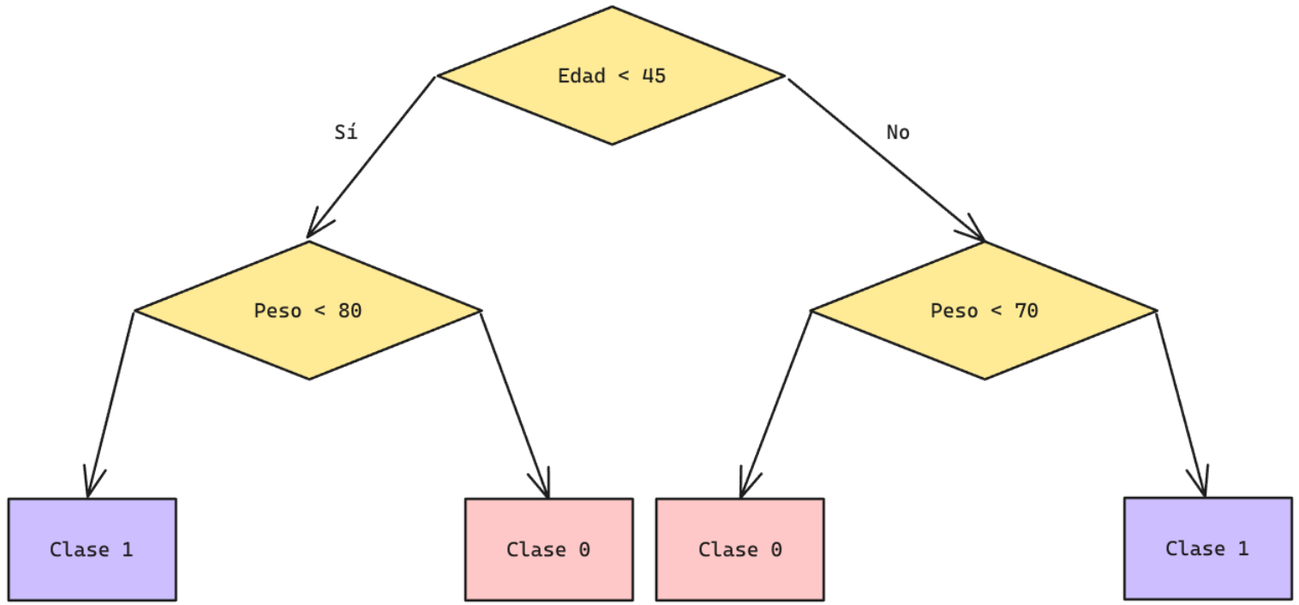
Árboles de Decisión: Ejemplo

> Visualmente, un árbol de decisión segmenta el espacio separando los datos en subgrupos.

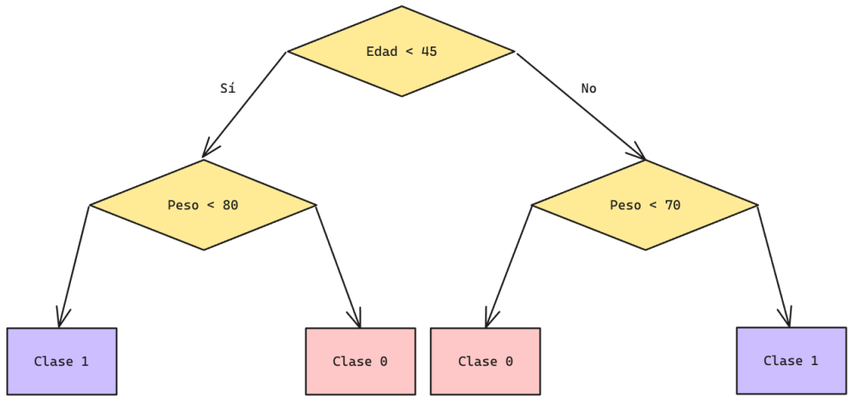
Esto permite la generacion de fronteras de decisión sumamente complejas.

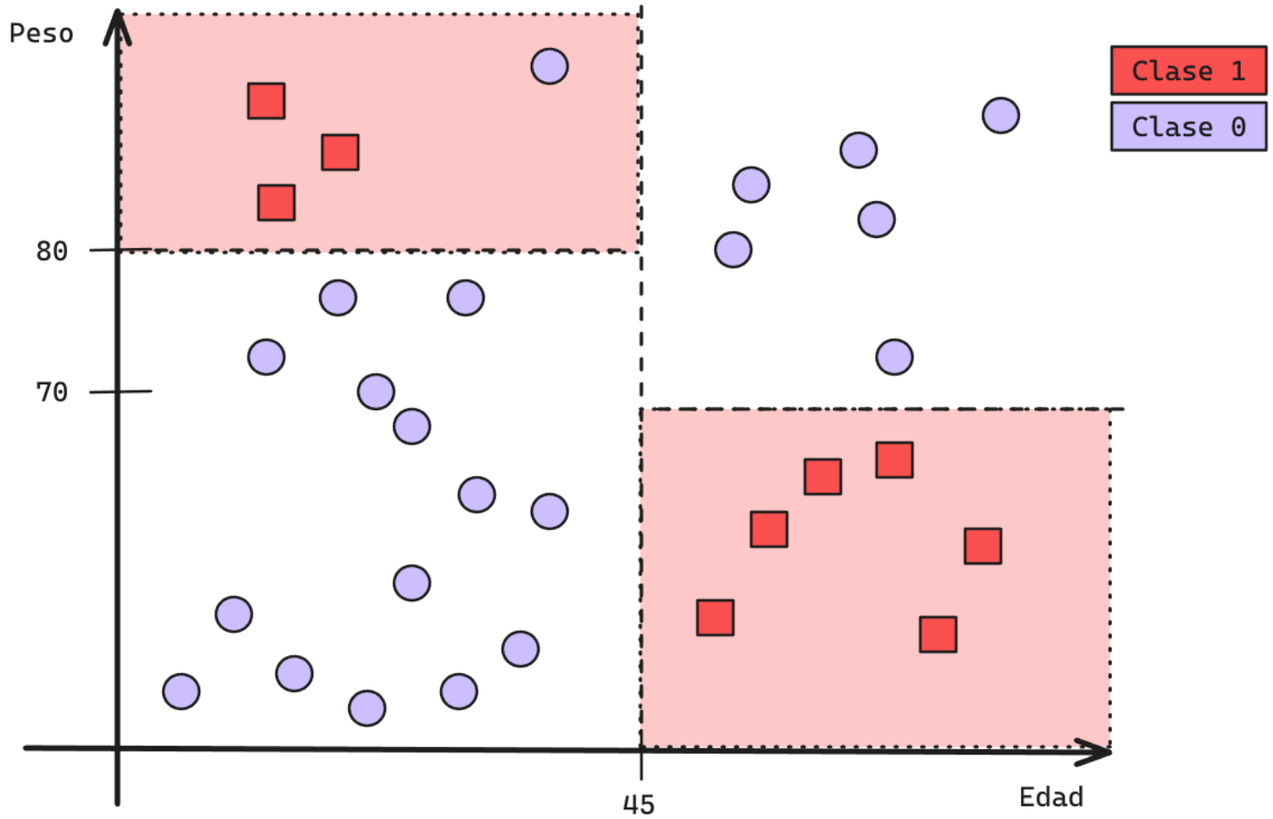
Supongamos el siguiente ejemplo:



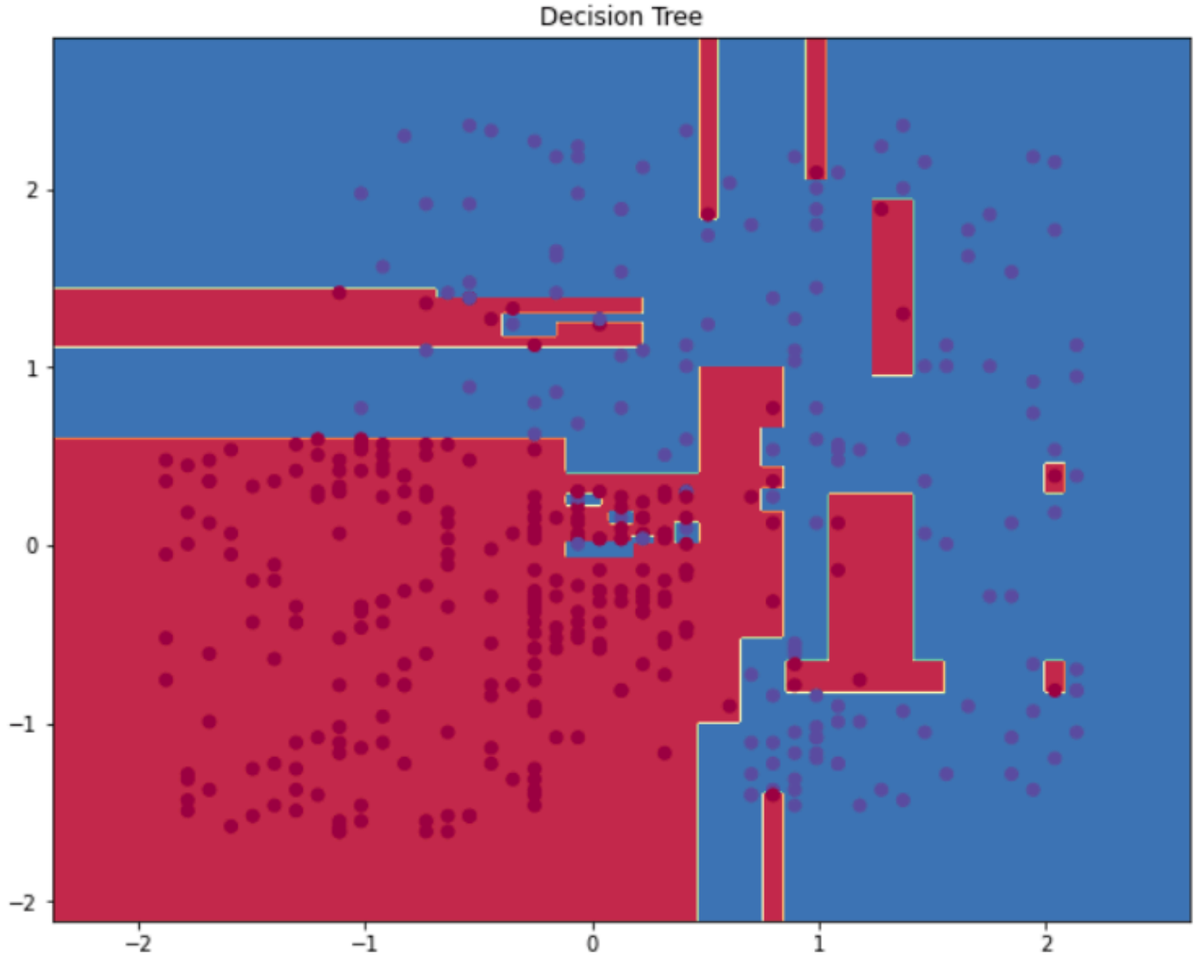


Árboles de Decisión: Frontera de Decisión





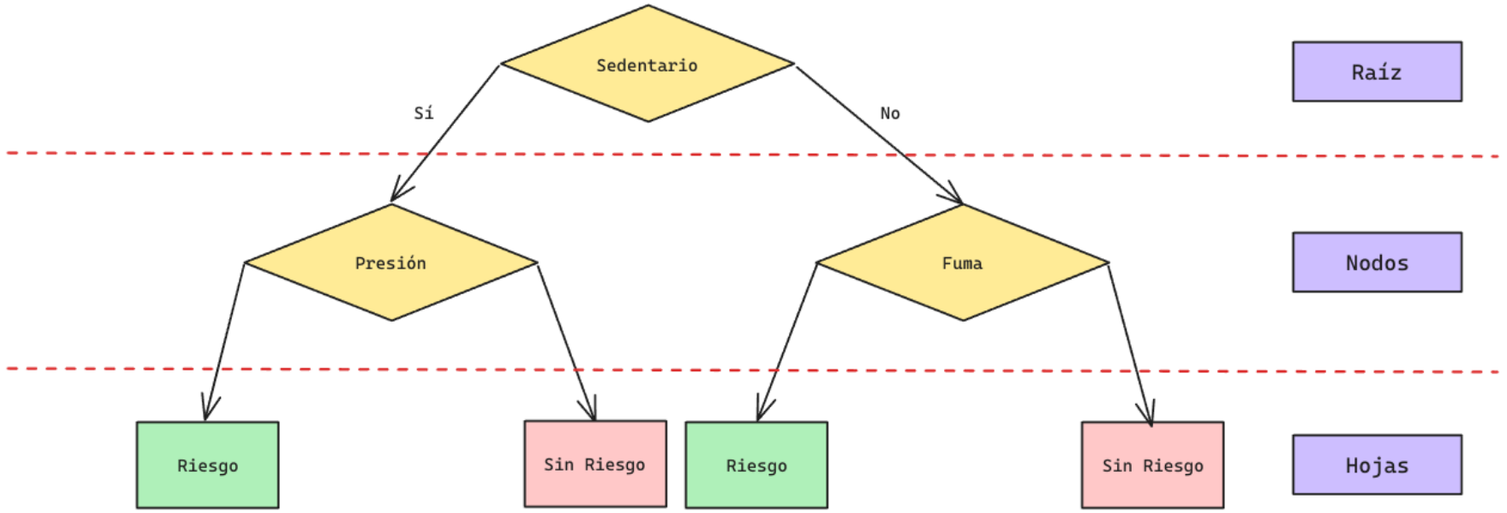
Árboles de Decisión: Frontera de Decisión



¿Cuál sería el ***Nivel de Ajuste*** de un modelo de este tipo?

Árboles de Decisión: Inferencia

Una vez construido el `árbol de decisión` basta con recorrerlo para poder generar la predicción para una instancia dada:



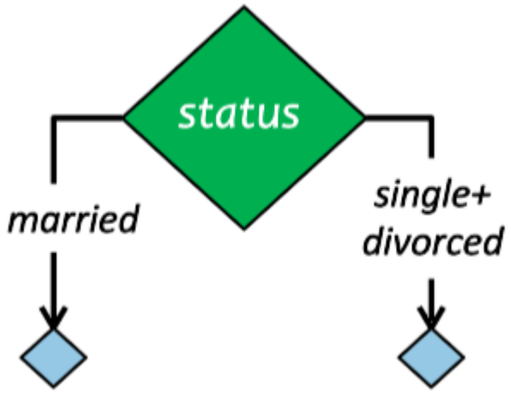
Sedentario	Fumador	Presión	Riesgo
Si	Si	Alta	Sí
Si	Si	Baja	No
Si	No	Baja	No
No	Si	Alta	Sí
No	No	Baja	No
No	No	Alta	?

Características de Árboles

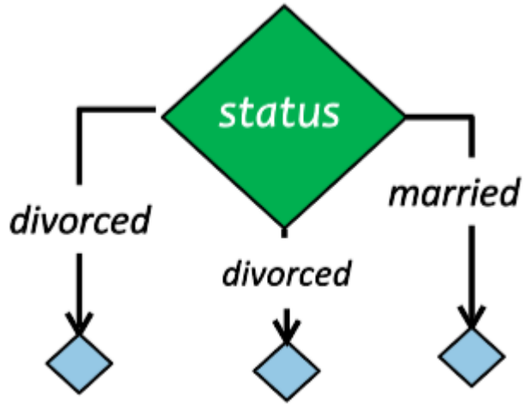
- Pueden trabajar con valores discretos o continuos. Además pueden ser usados como modelos de `Clasificación` o `Regresíon`.
- Una vez seleccionado un atributo no es posible devolverse (backtracking).
- Debido al poder de un árbol de Decisión la mayoría de las veces tienden al Overfitting. Una forma de evitar esto es usar técnicas de `Pruning`.
- Es preferible usar árboles cortos (Principio de Parsimonia o `Occam's Razor`).

> El principio de Parsimonia recomienda encontrar soluciones a problemas utilizando la menor cantidad de elementos/parámetros.

Tipos de Árboles de Decisión



Binary Split



Multi-way Split

- Hunt's Algorithm $\Longrightarrow$ Primer Método.
- ID3 $\Longrightarrow$ Sólo utiliza variables categóricas.
- C4.5 $\Longrightarrow$ incluye variables continuas.
- C5.0 $\Longrightarrow$ Permite separación en Múltiples Splits (No ha sido implementado en Sklearn).
- CART (Classification and Regression Trees) $\Longrightarrow$ Permite que el output sea continuo pero solo utilizando Splits binarios.

Los CARTs son por lejos los árboles más utilizados en las librerías más famosas y potentes: `Scikit-Learn`, `XGboost`, `LightGBM`, `Catboost`.

Creación de un Árbol de Decisión

Pureza

Corresponde a la probabilidad de no sacar dos registros de un Nodo que pertenezcan a la misma clase.

El árbol de Decisión busca crear `Nodos lo más puro posible`. Para ello puede utilizar las siguentes métricas:

Índice Gini

$$Gini(X) = 1 - \sum_{x_i} p(x_i)^2$$

Entropía

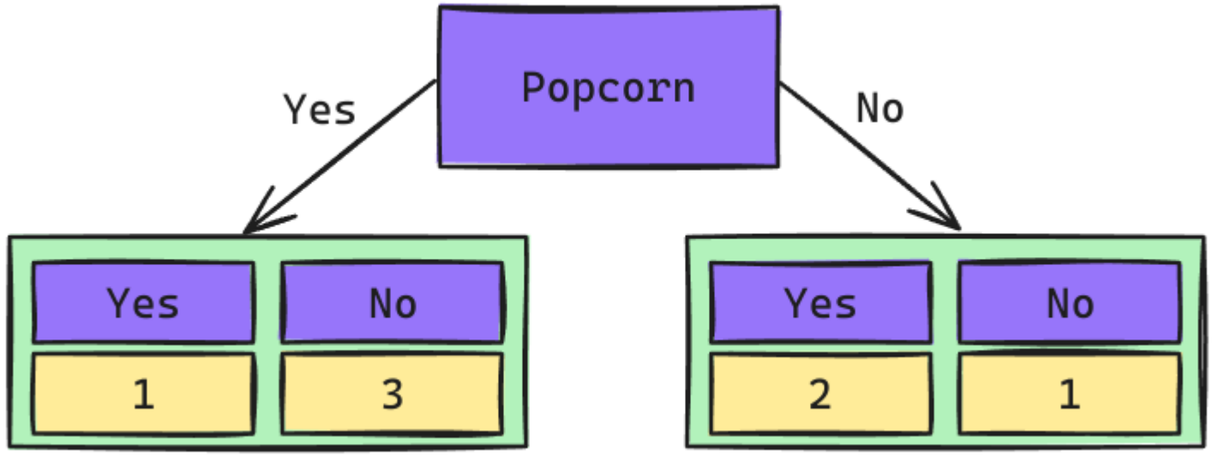
$$H(X) = -\sum_{x_i} p(x_i) log_2 p(x_i)$$

A mayor valor, mayor nivel de impureza. 0 implica Nodo completamente puro.

Árbol de Decisión: Ejemplo

Árbol de Decisión: Raíz Popcorn

Popcorn?	Soda?	Age	Like Avengers Endgame
Yes	Yes	7	No
Yes	No	12	No
No	Yes	18	Yes
No	Yes	35	Yes
Yes	Yes	38	Yes
Yes	No	50	No
No	No	83	No



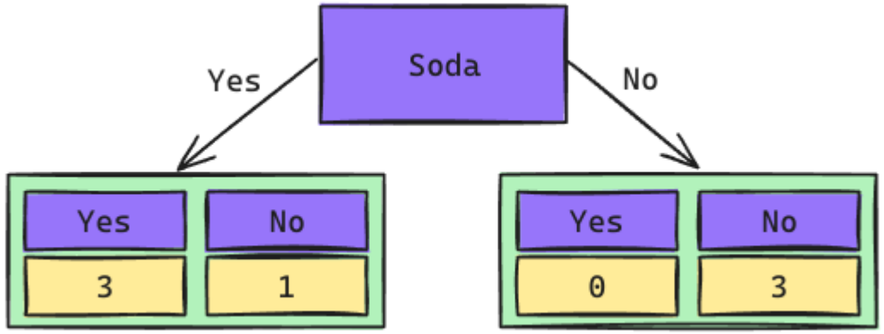
$$Gini_{(yes)} = 1 - \left(\frac{1}{4}\right)^2 - \left(\frac{3}{4}\right)^2 = 0.375$$

$$Gini_{(no)} = 1 - \left(\frac{2}{3}\right)^2 - \left(\frac{1}{3}\right)^2 = 0.444$$

$$Gini_{(split)} = \frac{4}{7} \cdot 0.375 + \frac{3}{7} \cdot 0.444 = 0.405$$

Árbol de Decisión: Raíz Soda

Popcorn?	Soda?	Age	Like Avengers Endgame
Yes	Yes	7	No
Yes	No	12	No
No	Yes	18	Yes
No	Yes	35	Yes
Yes	Yes	38	Yes
Yes	No	50	No
No	No	83	No



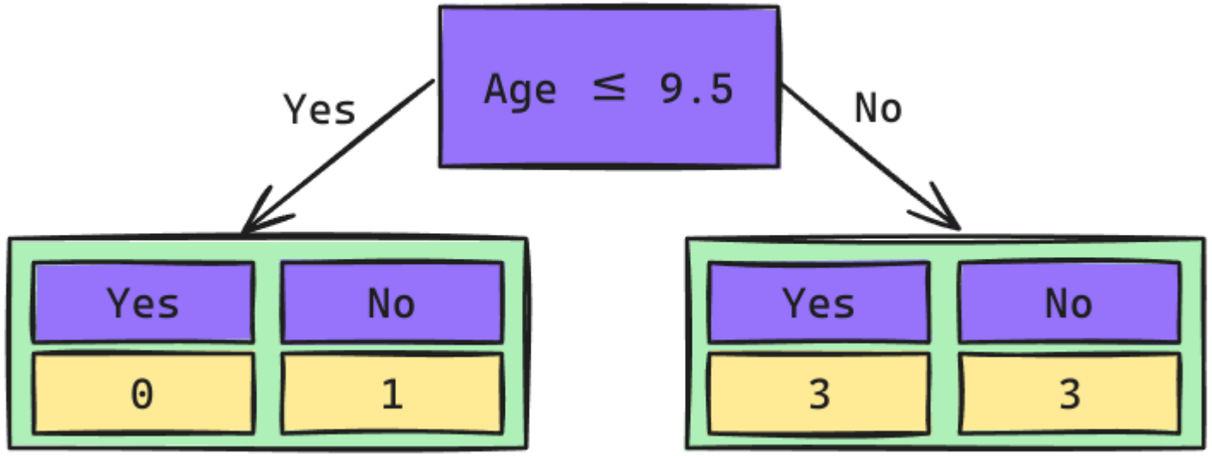
$$Gini_{(yes)} = 1 - \left(\frac{3}{4}\right)^2 - \left(\frac{1}{4}\right)^2 = 0.375$$

$$Gini_{(no)} = 1 - \left(\frac{0}{3}\right)^2 - \left(\frac{3}{3}\right)^2 = 0$$

$$Gini_{(split)} = \frac{4}{7} \cdot 0.375 = 0.214$$

Árbol de Decisión: Raíz Age

Popcorn?	Soda?	Age		Like Avengers Endgame
Yes	Yes	7	9.5	No
Yes	No	12	15	No
No	Yes	18	26.5	Yes
No	Yes	35	36.5	Yes
Yes	Yes	38	44	Yes
Yes	No	50	66.5	No
No	No	83		No



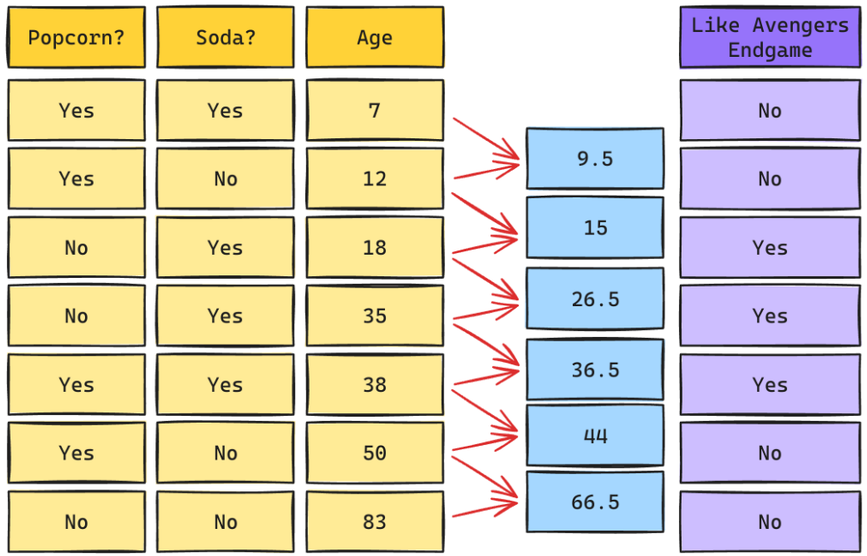
Los cortes de posibles Splits se calculan como el `promedio de los valores adyacentes` una vez que han sidos `ordenados de mayor a menor`.

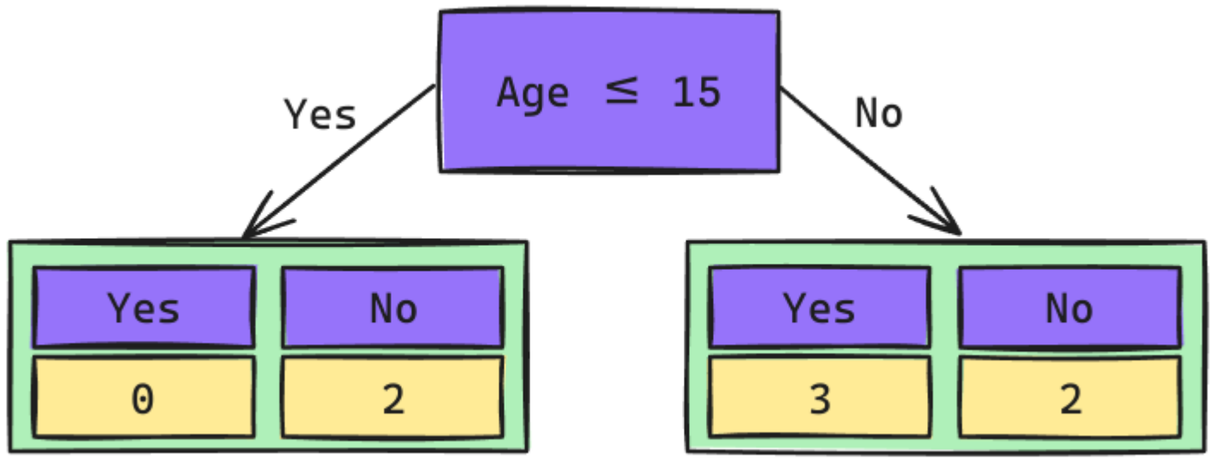
$$Gini_{(yes)} = 1 - \left(\frac{0}{1}\right)^2 - \left(\frac{1}{1}\right)^2 = 0$$

$$Gini_{(no)} = 1 - \left(\frac{3}{6}\right)^2 - \left(\frac{3}{6}\right)^2 = 0.5$$

$$Gini_{(split)} = \frac{6}{7} \cdot 0.5 = 0.429$$

Árbol de Decisión: Raíz Age





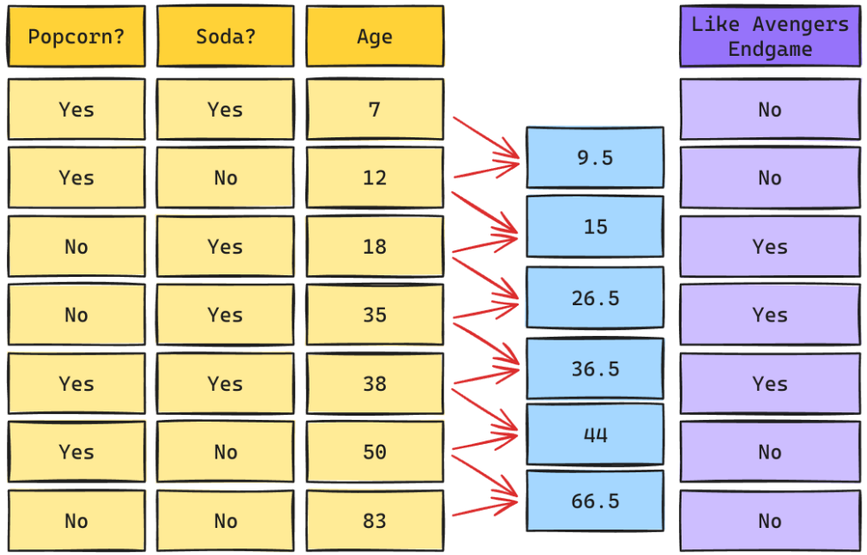
> Los cortes de posibles Splits se calculan como el `promedio de los valores adyacentes` una vez que han sidos `ordenados de mayor a menor`.

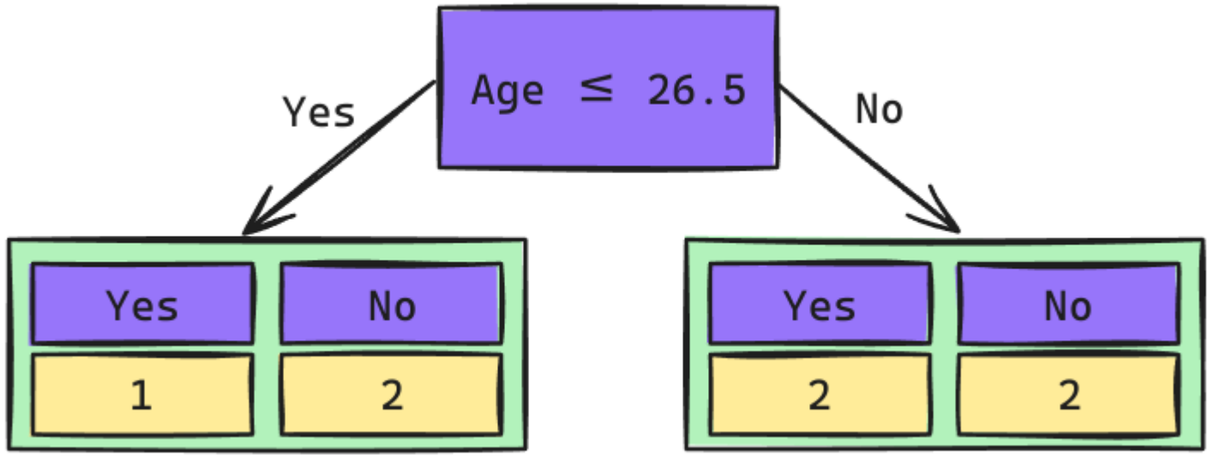
$$Gini_{(yes)} = 1 - \left(\frac{0}{2}\right)^2 - \left(\frac{2}{2}\right)^2 = 0$$

$$Gini_{(no)} = 1 - \left(\frac{3}{5}\right)^2 - \left(\frac{2}{5}\right)^2 = 0.48$$

$$Gini_{(split)} = \frac{5}{7} \cdot 0.48 = 0.343$$

Árbol de Decisión: Raíz Age





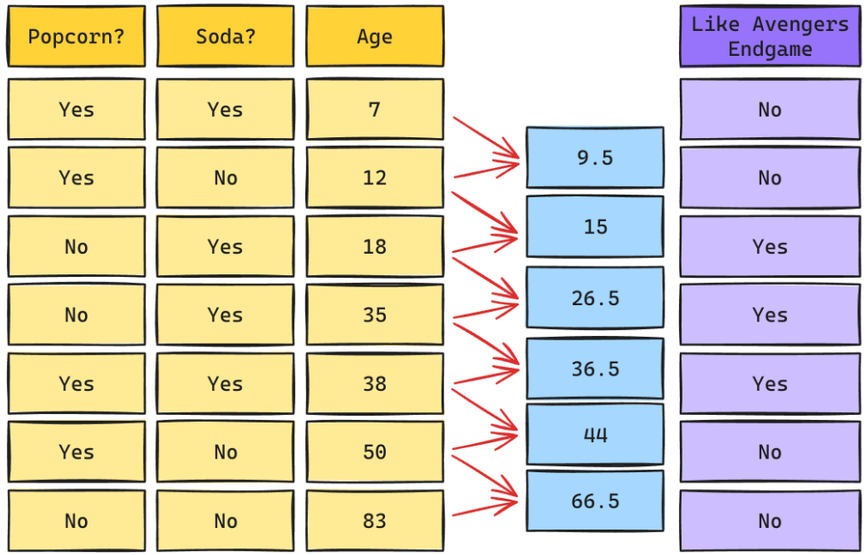
> Los cortes de posibles Splits se calculan como el `promedio de los valores adyacentes` una vez que han sidos `ordenados de mayor a menor`.

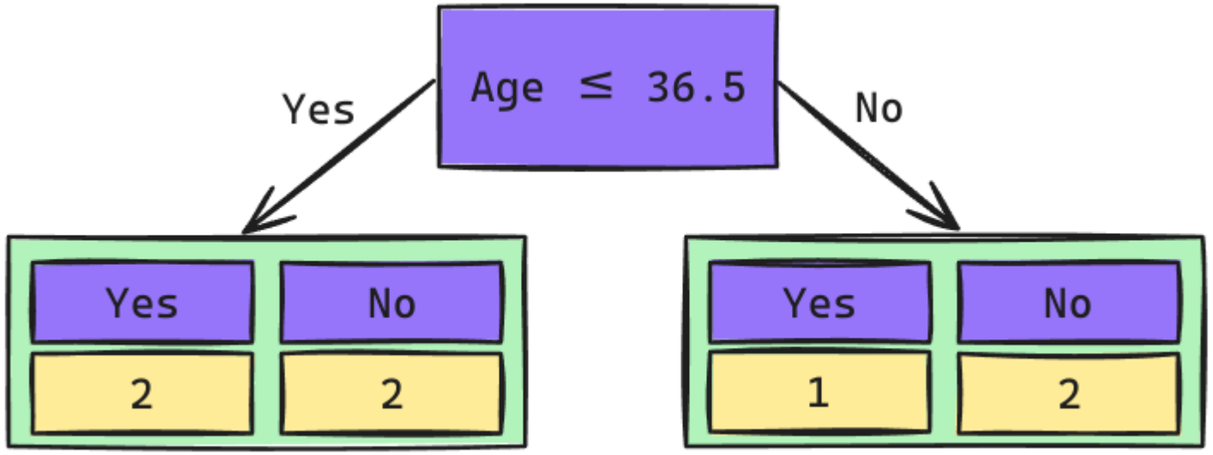
$$Gini_{(yes)} = 1 - \left(\frac{1}{3}\right)^2 - \left(\frac{2}{3}\right)^2 = 0.444$$

$$Gini_{(no)} = 1 - \left(\frac{2}{4}\right)^2 - \left(\frac{2}{4}\right)^2 = 0.5$$

$$Gini_{(split)} = \frac{3}{7} \cdot 0.444 + \frac{4}{7} \cdot 0.5 = 0.476$$

Árbol de Decisión: Raíz Age





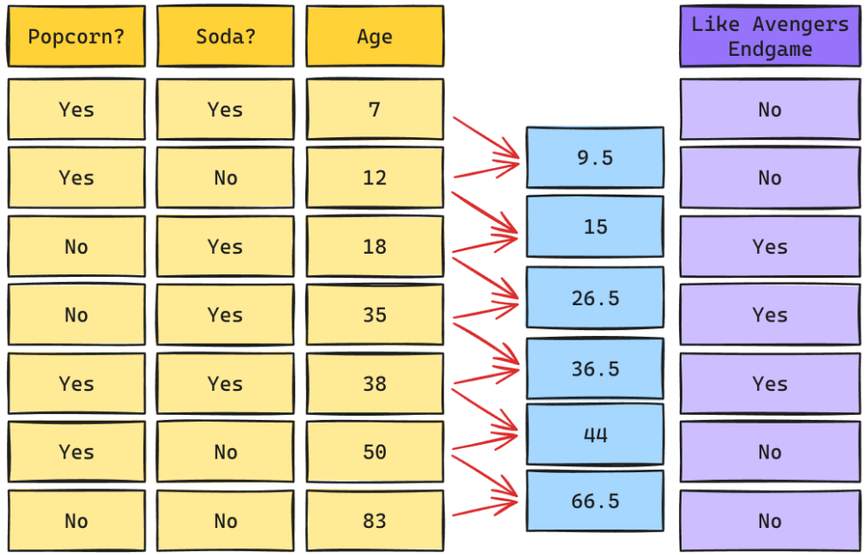
> Los cortes de posibles Splits se calculan como el `promedio de los valores adyacentes` una vez que han sidos `ordenados de mayor a menor`.

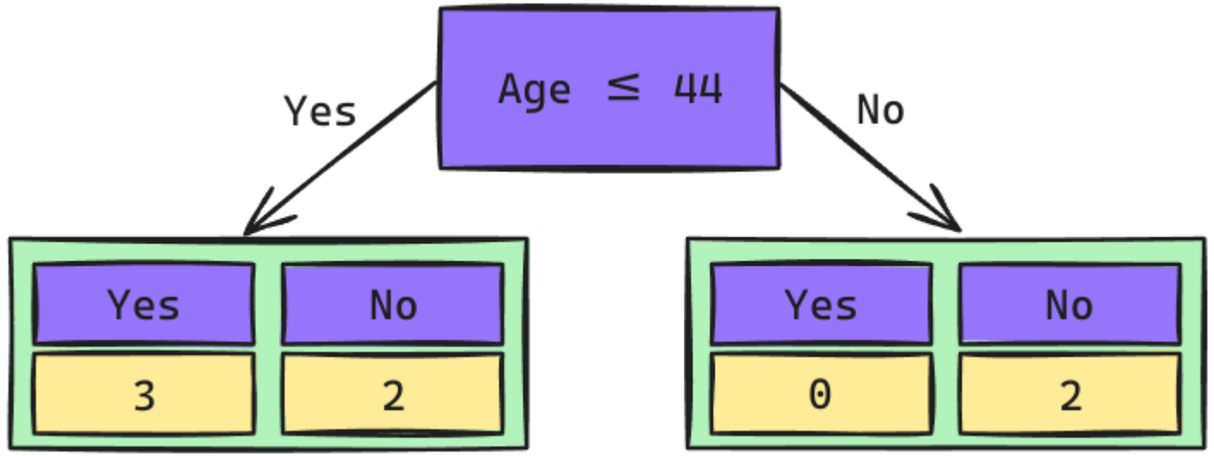
$$Gini_{(yes)} = 1 - \left(\frac{2}{4}\right)^2 - \left(\frac{2}{4}\right)^2 = 0.5$$

$$Gini_{(no)} = 1 - \left(\frac{1}{3}\right)^2 - \left(\frac{2}{3}\right)^2 = 0.444$$

$$Gini_{(split)} = \frac{4}{7} \cdot 0.5 + \frac{3}{7} \cdot 0.444 = 0.476$$

Árbol de Decisión: Raíz Age





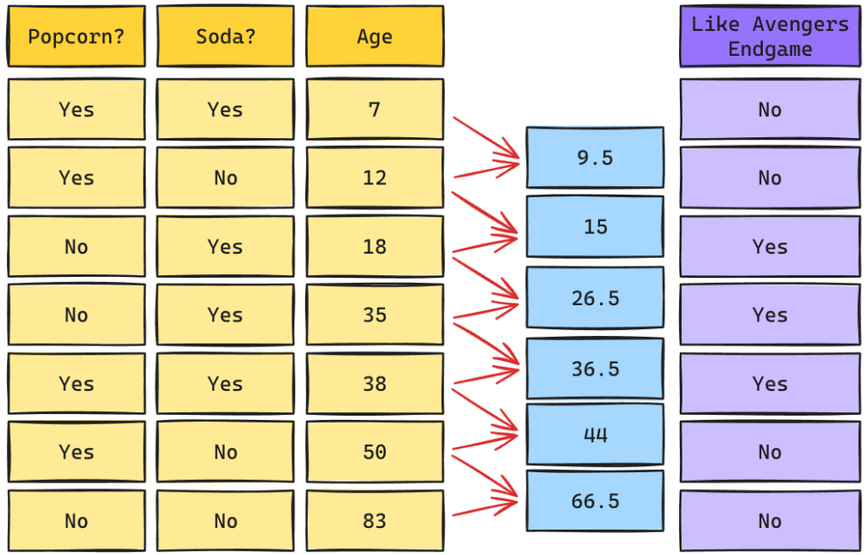
> Los cortes de posibles Splits se calculan como el `promedio de los valores adyacentes` una vez que han sidos `ordenados de mayor a menor`.

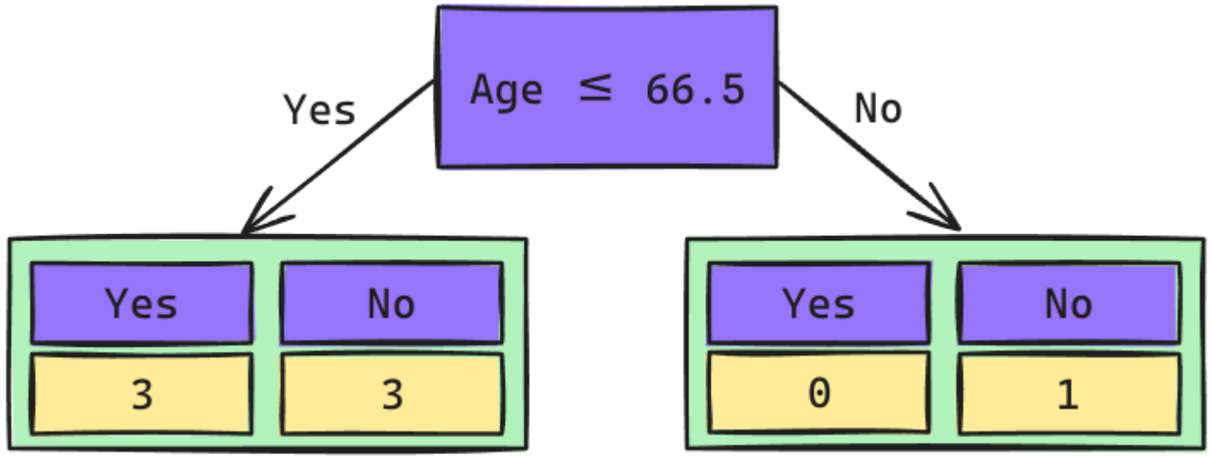
$$Gini_{(yes)} = 1 - \left(\frac{3}{2}\right)^2 - \left(\frac{2}{5}\right)^2 = 0.48$$

$$Gini_{(no)} = 1 - \left(\frac{0}{2}\right)^2 - \left(\frac{2}{2}\right)^2 = 0$$

$$Gini_{(split)} = \frac{5}{7} \cdot 0.48 = 0.343$$

Árbol de Decisión: Raíz Age





Los cortes de posibles Splits se calculan como el `promedio de los valores adyacentes` una vez que han sidos `ordenados de mayor a menor`.

$$Gini_{(yes)} = 1 - \left(\frac{3}{6}\right)^2 - \left(\frac{3}{6}\right)^2 = 0.5$$

$$Gini_{(no)} = 1 - \left(\frac{0}{1}\right)^2 - \left(\frac{1}{1}\right)^2 = 0$$

$$Gini_{(split)} = \frac{6}{7} \cdot 0.5 = 0.429$$

¿Qué Split elegiremos?

Split	Gini
Popcorn	0.405
Soda	0.214
Age ≤ 9.5	0.429
Age ≤ 15	0.343
Age ≤ 26.5	0.476
Age ≤ 36.5	0.476
Age ≤ 44	0.343
Age ≤ 66.5	0.429

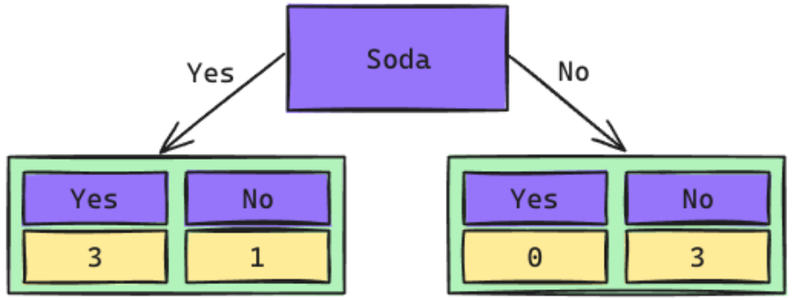
¿Qué Split elegiremos?

Split	Gini
Popcorn	0.405
Soda	0.214
Age ≤ 9.5	0.429
Age ≤ 15	0.343
Age ≤ 26.5	0.476
Age ≤ 36.5	0.476
Age ≤ 44	0.343
Age ≤ 66.5	0.429

Escogeremos el Split más pequeño que representa el que genera más pureza.

¿Qué Split elegiremos?

Split	Gini
Popcorn	0.405
Soda	0.214
Age ≤ 9.5	0.429
Age ≤ 15	0.343
Age ≤ 26.5	0.476
Age ≤ 36.5	0.476
Age ≤ 44	0.343
Age ≤ 66.5	0.429



Popcorn?	Soda?	Age
Yes	Yes	7
No	Yes	18
No	Yes	35
Yes	Yes	38

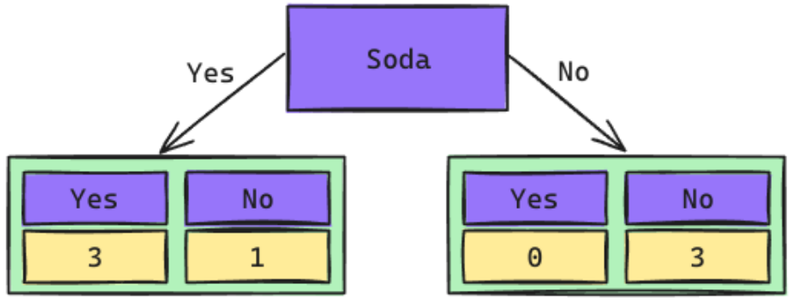
Like Avengers Endgame
No
Yes
Yes
Yes

12.5
26.5
36.5

Escogeremos el Split más pequeño que representa el que genera más pureza.

¿Qué Split elegiremos?

Split	Gini
Popcorn	0.405
Soda	0.214
Age ≤ 9.5	0.429
Age ≤ 15	0.343
Age ≤ 26.5	0.476
Age ≤ 36.5	0.476
Age ≤ 44	0.343
Age ≤ 66.5	0.429



Popcorn?	Soda?	Age
Yes	Yes	7
No	Yes	18
No	Yes	35
Yes	Yes	38

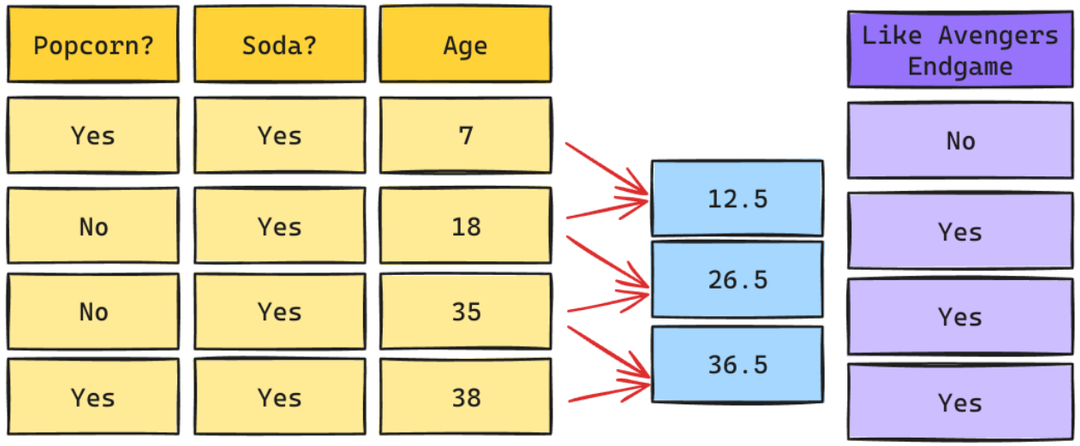
Like Avengers Endgame
No
Yes
Yes
Yes

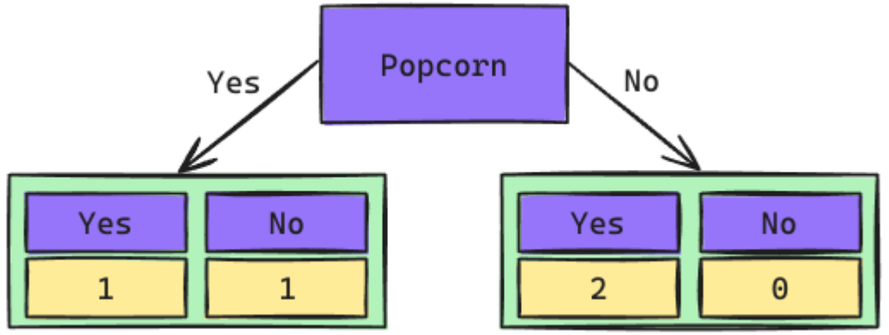
12.5
26.5
36.5

Escogeremos el Split más pequeño que representa el que genera más pureza.

El nodo que no le gusta la Soda quedó completamente puro. Por lo tanto, no puede seguir dividiéndose. Seguiremos trabajando sólo con aquellos `que sí les gusta la Soda`.

Árbol de Decisión: 2do Nivel



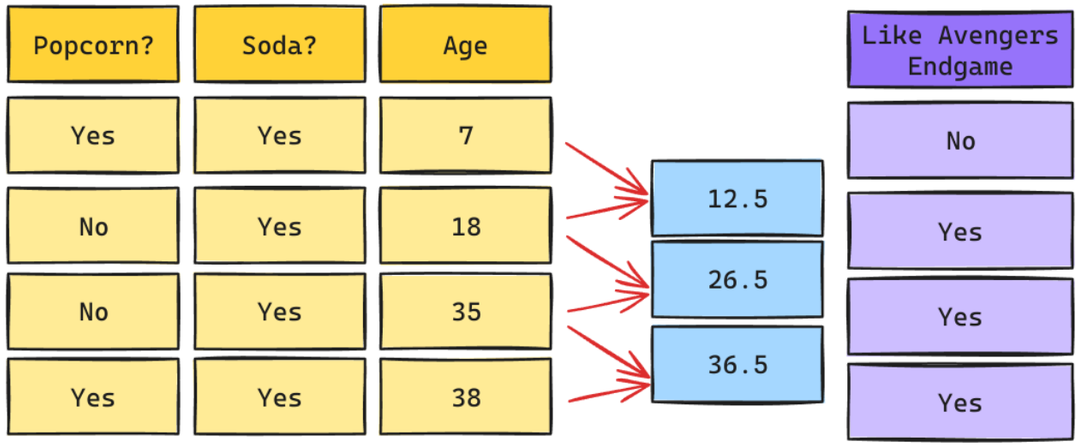


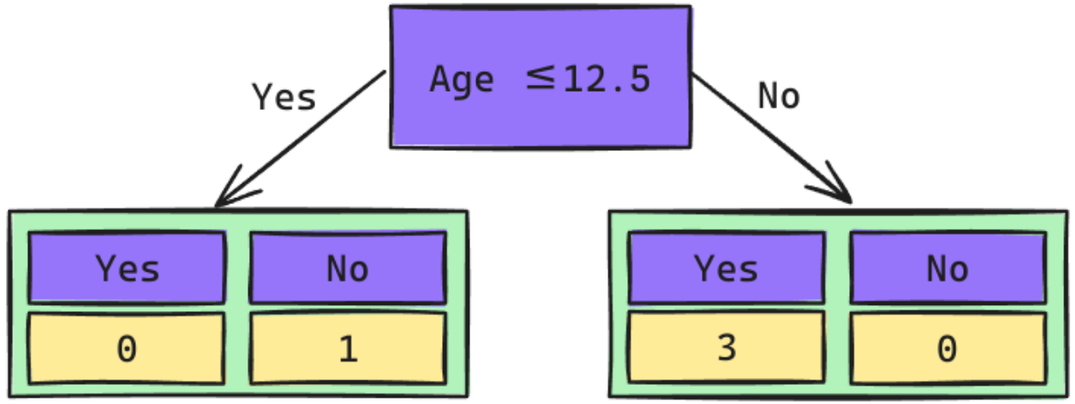
$$Gini_{(yes)} = 1 - \left(\frac{1}{2}\right)^2 - \left(\frac{1}{2}\right)^2 = 0.5$$

$$Gini_{(no)} = 1 - \left(\frac{2}{2}\right)^2 - \left(\frac{0}{2}\right)^2 = 0$$

> $$Gini_{(split)} = \frac{2}{4} \cdot 0.5 = 0.25$$

Árbol de Decisión: 2do Nivel



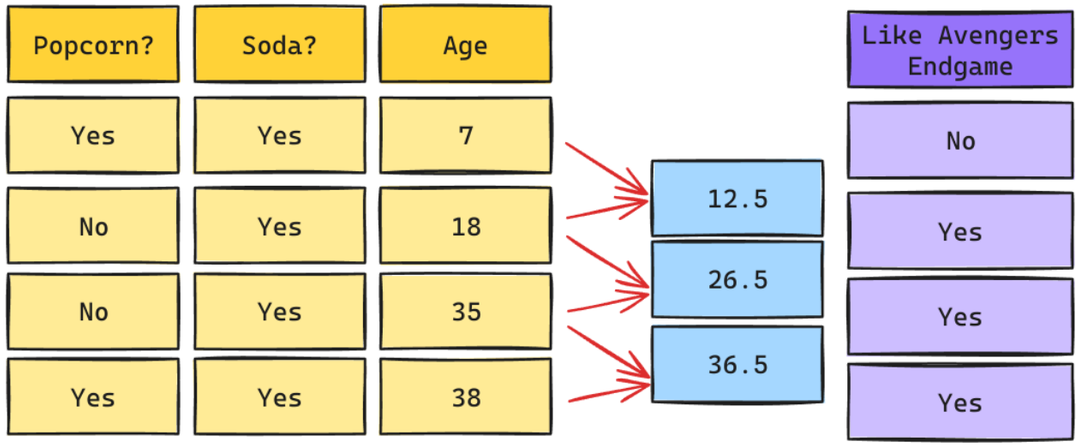


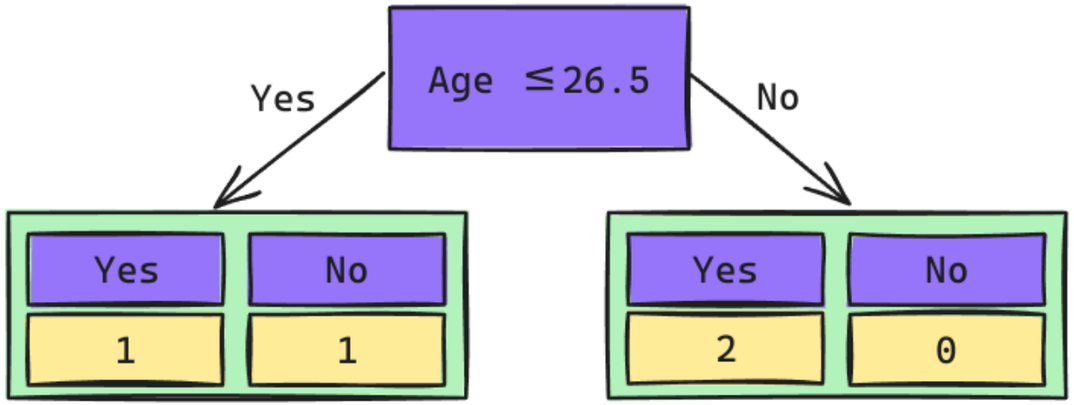
$$Gini_{(yes)} = 1 - \left(\frac{0}{1}\right)^2 - \left(\frac{1}{1}\right)^2 = 0$$

$$Gini_{(no)} = 1 - \left(\frac{3}{3}\right)^2 - \left(\frac{0}{3}\right)^2 = 0$$

> $Gini_{(split)} = 0$

Árbol de Decisión: 2do Nivel



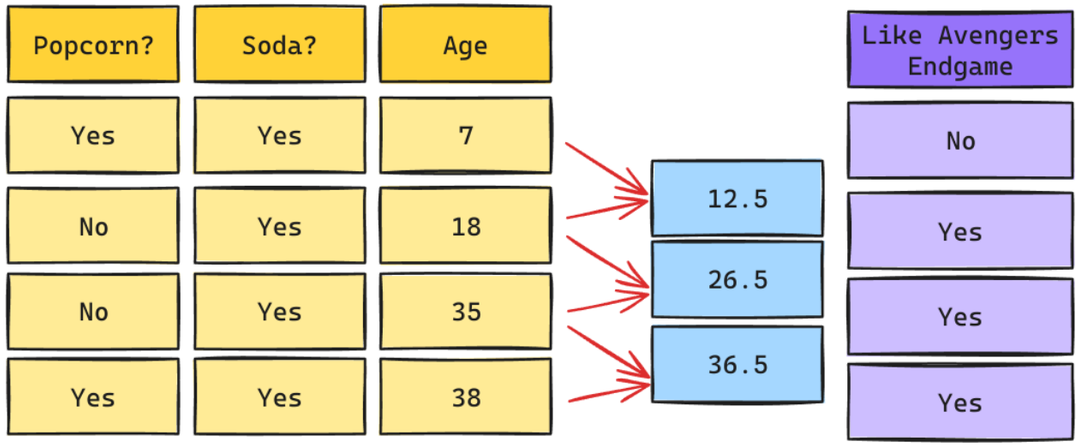


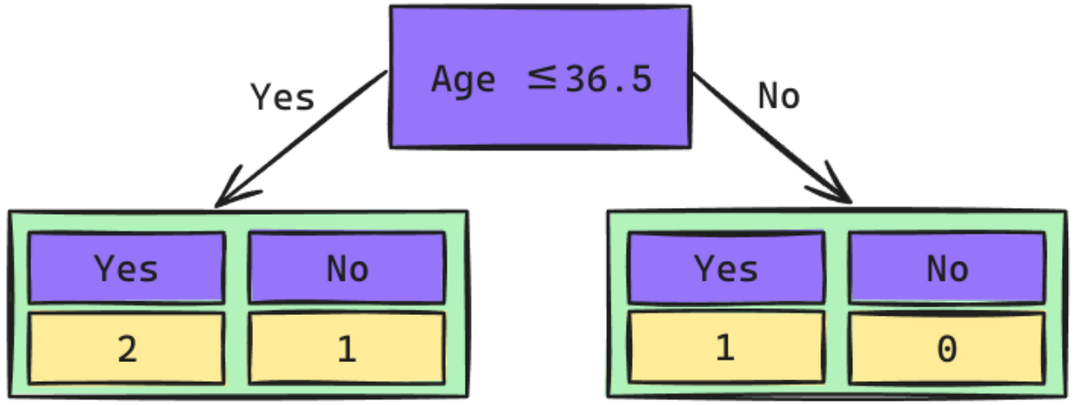
$$Gini_{(yes)} = 1 - \left(\frac{1}{2}\right)^2 - \left(\frac{1}{2}\right)^2 = 0.5$$

$$Gini_{(no)} = 1 - \left(\frac{2}{2}\right)^2 - \left(\frac{0}{2}\right)^2 = 0$$

> $$Gini_{(split)} = \frac{2}{4} \cdot 0.5 = 0.25$$

Árbol de Decisión: 2do Nivel





$$Gini_{(yes)} = 1 - \left(\frac{2}{3}\right)^2 - \left(\frac{1}{3}\right)^2 = 0.444$$

$$Gini_{(no)} = 1 - \left(\frac{1}{1}\right)^2 - \left(\frac{0}{1}\right)^2 = 0$$

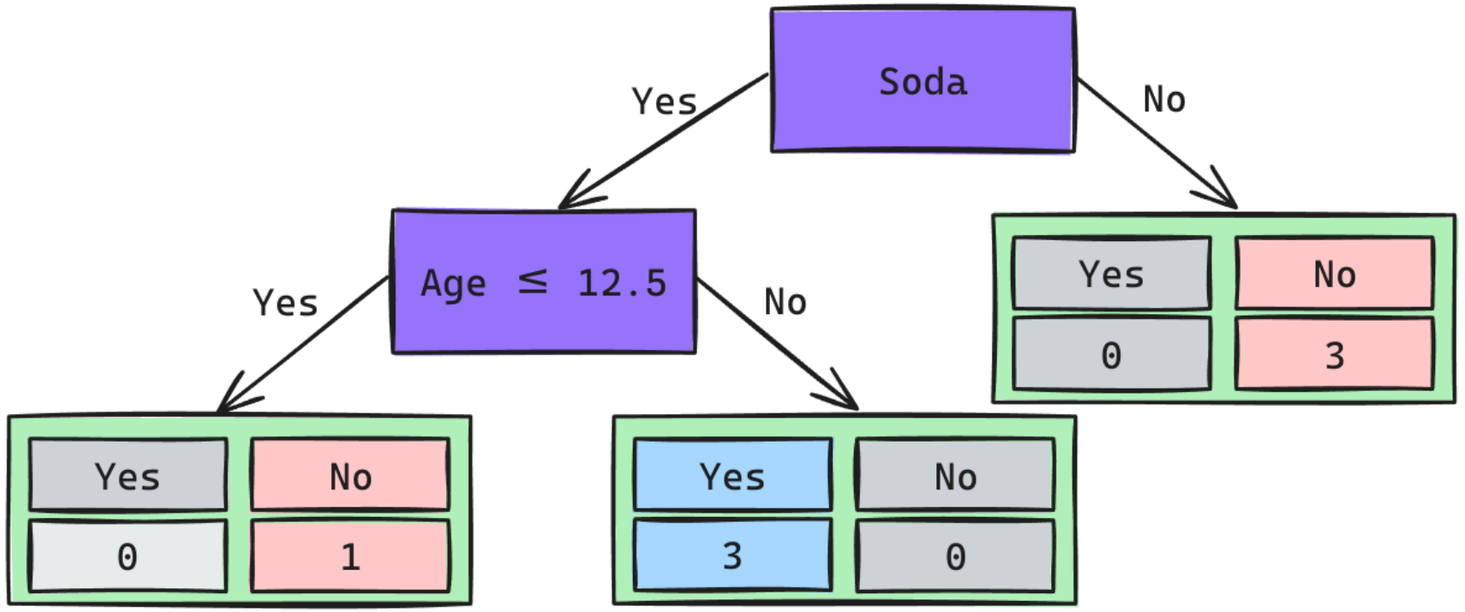
> $$Gini_{(split)} = \frac{3}{4} \cdot 0.444 = 0.333$$

Árbol de Decisión

Split	Gini
Popcorn	0.25
Age ≤ 12.5	0
Age ≤ 26.5	0.25
Age ≤ 36.5	0.333

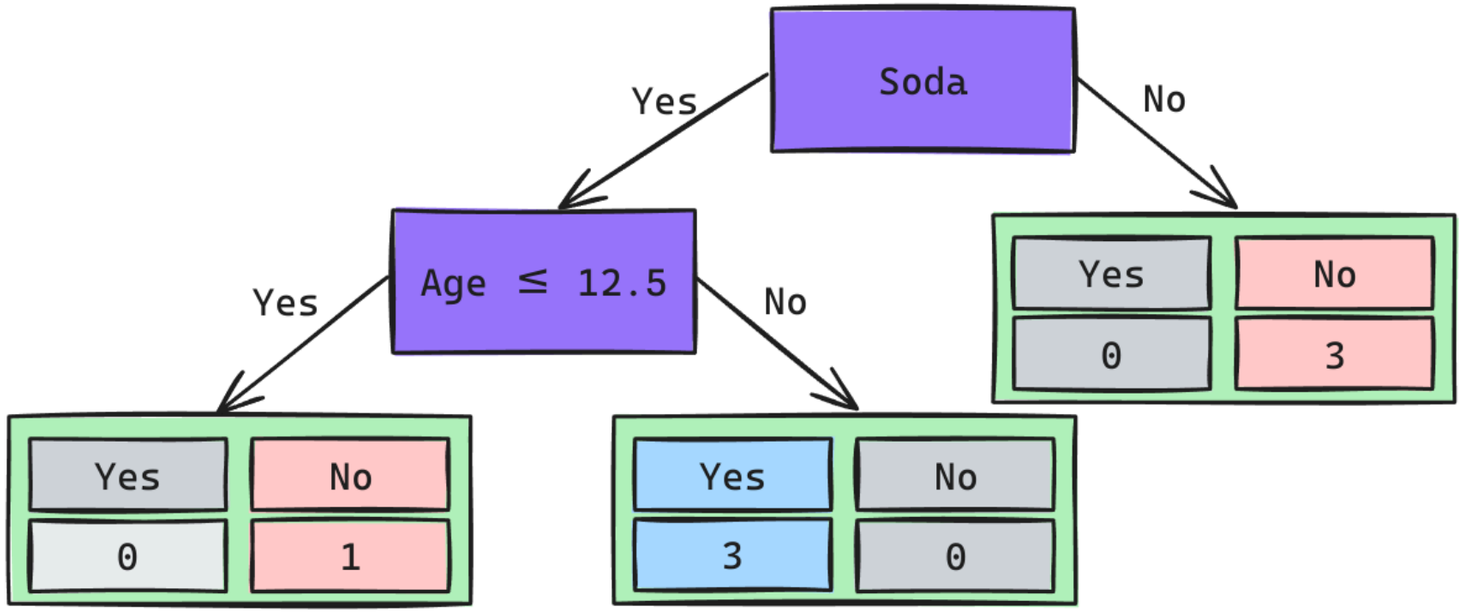
Árbol de Decisión

Split	Gini
Popcorn	0.25
Age ≤ 12.5	0
Age ≤ 26.5	0.25
Age ≤ 36.5	0.333



Árbol de Decisión

Split	Gini
Popcorn	0.25
Age ≤ 12.5	0
Age ≤ 26.5	0.25
Age ≤ 36.5	0.333



¿Cuál sería la predicción?

Popcorn?	Soda?	Age	Like Avengers Endgame
Yes	Yes	37	?

Crecimiento de un Árbol

- Un árbol sólo dejará de crecer si:
  - No hay más puntos a separar.
    - Todas las muestras de un nodo pertenecen a la misma clase.
  - No hay más variables a separar.

- Esto normalmente termina en `Overfitting`.

Para solucionar esto se aplica regularización. En el caso de Árboles esto se denomina `prunning`.

Pruning

Prepruning: Define/Evita que el árbol crezca hasta:

- Un cierto nivel o número de hojas.
- Aplicar un test estadístico (normalmente un proceso muy costoso).
- Usar medidas de complejidad para penalizar árboles de gran tamaño.

Postpruning: Decide eliminar nodos, luego de que el árbol crezca.

- Usar un parámetro de Costo de Impureza.

Hiperparámetros

- **criterion**: Elegir bajo qué criterio se mide la impureza.
- **max_depth**: El nivel es la altura que tendrá el árbol. Niveles más bajos generan árboles más simples.
- **min_samples_split**: Número de instancias necesarias para generar un split. Un mayor número o proporción generará árboles más simples.
- **min_samples_leaf**: Número mínimo de instancias necesarias para que un nodo sea hoja. Un número o proporción más alta generará árboles más simples.
- **ccp_alpha**: Está asociado a la pureza total del árbol. Para más información ver acá.

¿Cómo se ve la complejidad/simplicidad en un árbol de Decisión?

Implementación en Scikit-Learn

```python
from sklearn.tree import DecisionTreeClassifier, plot_tree

dt = DecisionTreeClassifier(criterion="gini", max_depth=None, min_sample_split=2,
                            min_samples_leaf=1,min_impurity_decrease=0,
                            ccp_alpha=0, random_state=42)
dt.fit(X_train, y_train)

y_pred = dt.predict(X_test)
y_proba = dt.predict_proba(X_test)

## Permite Visualizar el Árbol de Decisión
plt_tree(dt, filled = True, feature_names=None, class_names=None)
```

- **criterion**: Puede ser gini o entropía. Por defecto `"gini"`.
- **max_depth**: Número de niveles que se permita que crezca el nivel, por defecto `None`, significa todos los que pueda.
- **min_samples_split**: El número mínimo de elementos dentro de un nodo para permitir el split. Por defecto 2.
- **min_samples_leaf**: El número mínimo de elementos para que un nodo pueda ser considerado hoja. Por defecto 1.
- **min_impurity_decreased**: Decrecimiento mínimo de la impureza. Si no se cumple, no hay Split. Por defecto 0.
- **ccp_alpha**: Parámetro de Post-Pruning. Valores más altos genera la poda de más nodos.

Implementación en Scikit-Learn

```python
from sklearn.tree import DecisionTreeClassifier, plot_tree

dt = DecisionTreeClassifier(criterion="gini", max_depth=None, min_sample_split=2,
                            min_samples_leaf=1,min_impurity_decrease=0,
                            ccp_alpha=0, random_state=42)
dt.fit(X_train, y_train)

y_pred = dt.predict(X_test)
y_proba = dt.predict_proba(X_test)

## Permite Visualizar el Árbol de Decisión
plt_tree(dt, filled = True, feature_names=None, class_names=None)
```

- **criterion**: Puede ser gini o entropía. Por defecto `"gini"`.
- **max_depth**: Número de niveles que se permita que crezca el nivel, por defecto `None`, significa todos los que pueda.
- **min_samples_split**: El número mínimo de elementos dentro de un nodo para permitir el split. Por defecto 2.
- **min_samples_leaf**: El número mínimo de elementos para que un nodo pueda ser considerado hoja. Por defecto 1.
- **min_impurity_decreased**: Decrecimiento mínimo de la impureza. Si no se cumple, no hay Split. Por defecto 0.
- **ccp_alpha**: Parámetro de Post-Pruning. Valores más altos genera la poda de más nodos.

Implementación en Scikit-Learn

```python
from sklearn.tree import DecisionTreeClassifier, plot_tree

dt = DecisionTreeClassifier(criterion="gini", max_depth=None, min_sample_split=2,
                            min_samples_leaf=1,min_impurity_decrease=0,
                            ccp_alpha=0, random_state=42)
dt.fit(X_train, y_train)

y_pred = dt.predict(X_test)
y_proba = dt.predict_proba(X_test)

## Permite Visualizar el Árbol de Decisión
plt_tree(dt, filled = True, feature_names=None, class_names=None)
```

- **criterion**: Puede ser gini o entropía. Por defecto `"gini"`.
- **max_depth**: Número de niveles que se permita que crezca el nivel, por defecto `None`, significa todos los que pueda.
- **min_samples_split**: El número mínimo de elementos dentro de un nodo para permitir el split. Por defecto 2.
- **min_samples_leaf**: El número mínimo de elementos para que un nodo pueda ser considerado hoja. Por defecto 1.
- **min_impurity_decreased**: Decrecimiento mínimo de la impureza. Si no se cumple, no hay Split. Por defecto 0.
- **ccp_alpha**: Parámetro de Post-Pruning. Valores más altos genera la poda de más nodos.

Implementación en Scikit-Learn

```python
from sklearn.tree import DecisionTreeClassifier, plot_tree

dt = DecisionTreeClassifier(criterion="gini", max_depth=None, min_sample_split=2,
                            min_samples_leaf=1,min_impurity_decrease=0,
                            ccp_alpha=0, random_state=42)
dt.fit(X_train, y_train)

y_pred = dt.predict(X_test)
y_proba = dt.predict_proba(X_test)

## Permite Visualizar el Árbol de Decisión
plt_tree(dt, filled = True, feature_names=None, class_names=None)
```

- **criterion**: Puede ser gini o entropía. Por defecto `"gini"`.
- **max_depth**: Número de niveles que se permita que crezca el nivel, por defecto `None`, significa todos los que pueda.
- **min_samples_split**: El número mínimo de elementos dentro de un nodo para permitir el split. Por defecto 2.
- **min_samples_leaf**: El número mínimo de elementos para que un nodo pueda ser considerado hoja. Por defecto 1.
- **min_impurity_decreased**: Decrecimiento mínimo de la impureza. Si no se cumple, no hay Split. Por defecto 0.
- **ccp_alpha**: Parámetro de Post-Pruning. Valores más altos genera la poda de más nodos.

Implementación en Scikit-Learn

```python
from sklearn.tree import DecisionTreeClassifier, plot_tree

dt = DecisionTreeClassifier(criterion="gini", max_depth=None, min_sample_split=2,
                            min_samples_leaf=1,min_impurity_decrease=0,
                            ccp_alpha=0, random_state=42)
dt.fit(X_train, y_train)

y_pred = dt.predict(X_test)
y_proba = dt.predict_proba(X_test)

## Permite Visualizar el Árbol de Decisión
plt_tree(dt, filled = True, feature_names=None, class_names=None)
```

- **criterion**: Puede ser gini o entropía. Por defecto `"gini"`.
- **max_depth**: Número de niveles que se permita que crezca el nivel, por defecto `None`, significa todos los que pueda.
- **min_samples_split**: El número mínimo de elementos dentro de un nodo para permitir el split. Por defecto 2.
- **min_samples_leaf**: El número mínimo de elementos para que un nodo pueda ser considerado hoja. Por defecto 1.
- **min_impurity_decreased**: Decrecimiento mínimo de la impureza. Si no se cumple, no hay Split. Por defecto 0.
- **ccp_alpha**: Parámetro de Post-Pruning. Valores más altos genera la poda de más nodos.

Implementación en Scikit-Learn

```python
from sklearn.tree import DecisionTreeClassifier, plot_tree

dt = DecisionTreeClassifier(criterion="gini", max_depth=None, min_sample_split=2,
                            min_samples_leaf=1,min_impurity_decrease=0,
                            ccp_alpha=0, random_state=42)
dt.fit(X_train, y_train)

y_pred = dt.predict(X_test)
y_proba = dt.predict_proba(X_test)

## Permite Visualizar el Árbol de Decisión
plt_tree(dt, filled = True, feature_names=None, class_names=None)
```

- **criterion**: Puede ser gini o entropía. Por defecto `"gini"`.
- **max_depth**: Número de niveles que se permita que crezca el nivel, por defecto `None`, significa todos los que pueda.
- **min_samples_split**: El número mínimo de elementos dentro de un nodo para permitir el split. Por defecto 2.
- **min_samples_leaf**: El número mínimo de elementos para que un nodo pueda ser considerado hoja. Por defecto 1.
- **min_impurity_decreased**: Decrecimiento mínimo de la impureza. Si no se cumple, no hay Split. Por defecto 0.
- **ccp_alpha**: Parámetro de Post-Pruning. Valores más altos genera la poda de más nodos.

Implementación en Scikit-Learn

```python
from sklearn.tree import DecisionTreeClassifier, plot_tree

dt = DecisionTreeClassifier(criterion="gini", max_depth=None, min_sample_split=2,
                            min_samples_leaf=1,min_impurity_decrease=0,
                            ccp_alpha=0, random_state=42)
dt.fit(X_train, y_train)

y_pred = dt.predict(X_test)
y_proba = dt.predict_proba(X_test)

## Permite Visualizar el Árbol de Decisión
plt_tree(dt, filled = True, feature_names=None, class_names=None)
```

- **criterion**: Puede ser gini o entropía. Por defecto `"gini"`.
- **max_depth**: Número de niveles que se permita que crezca el nivel, por defecto `None`, significa todos los que pueda.
- **min_samples_split**: El número mínimo de elementos dentro de un nodo para permitir el split. Por defecto 2.
- **min_samples_leaf**: El número mínimo de elementos para que un nodo pueda ser considerado hoja. Por defecto 1.
- **min_impurity_decreased**: Decrecimiento mínimo de la impureza. Si no se cumple, no hay Split. Por defecto 0.
- **ccp_alpha**: Parámetro de Post-Pruning. Valores más altos genera la poda de más nodos.

لقد إنتهينا