Instituto Tecnológico y de Estudios Superiores de Monterrey.
Campus Estado de México

Inteligencia artificial avanzada para la ciencia de datos I
TC3006C | (Gpo 501)

Uso de framework o biblioteca de aprendizaje máquina para la implementación de una solución

---

Portafolio Implementación - Módulo 2

Profesor
Jorge Adolfo Ramírez Uresti

Estudiante

Ulises Jaramillo Portilla……. | A01798380

Fecha de entrega: 11 de septiembre del 2025

1. Introducción

En este proyecto se implementó un clasificador de árbol de decisión utilizando la librería scikit-learn con el objetivo fue resolver tareas de clasificación de manera eficiente, reproducible e interpretable, aprovechando los hiperparámetros y utilidades que este framework ofrece. Este modelo se configuró para soportar tanto atributos categóricos como numéricos, permitiendo elegir el criterio de partición, así como definir parámetros de control como profundidad máxima, tamaño mínimo de división o impureza mínima para dividir un nodo. Además, se incluyeron procesos de limpieza de datos, codificación de variables categóricas y división de conjuntos de entrenamiento y prueba con semilla fija para asegurar reproducibilidad.

Por otra parte, se desarrolló un flujo en Python que integra pandas y scikit-learn para la carga, preprocesamiento y evaluación de datasets, complementado con un módulo de métricas para calcular precisión, recall, F1 y matriz de confusión. Para evaluar el algoritmo se utilizaron varios conjuntos de datos con características distintas, abarcando escenarios categóricos, numéricos y mixtos.

1.1. Objetivo

Utilizar frameworks de machine learning, en particular scikit-learn, para implementar un clasificador de árbol de decisión configurable, capaz de trabajar con datos mixtos e implementar la generación y visualización de dichas predicciones mediante una interfaz gráfica, apoyándome en lo visto en otros módulos; y basarme en lo tratado en este módulo para la creación del algoritmo y el entendimiento/interpretación de los resultados.

2. Problema y conjunto de datos

2.1. Tipo de problema

El proyecto aborda un problema de clasificación supervisada, donde el objetivo es predecir una categoría o clase a partir de un conjunto de atributos. Dado que se trabaja con diferentes datasets (mushrooms, heart, heart_D, tennis y stress level), el tipo de problema se mantiene en el ámbito de la clasificación, aunque con variaciones en número de clases, balance de datos y naturaleza de las variables, tanto categóricas como numéricas.

2.2. Dataset

Para validar el clasificador se usaron distintos conjuntos de datos:

- **Mushrooms.csv:** Instancias con atributos categóricos que describen características de hongos (ej. color de sombrero, olor, textura) con el objetivo de predecir si son comestibles o venenosos.

- **Heart.csv / Heart_D.csv:** Instancias con atributos tanto numéricos (edad, presión arterial, colesterol) como categóricos (sexo, tipo de dolor de pecho), buscando predecir la presencia o ausencia de enfermedad cardíaca.
- **Tennis.csv:** Dataset pequeño (≈15 ejemplos) con atributos categóricos como clima, humedad y viento, para predecir si se puede jugar tenis o no.
- **Academic Stress Level – Maintenance 1.csv:** Datos que describen factores académicos y de comportamiento, usados para clasificar el nivel de estrés académico de los estudiantes.

Cada dataset fue dividido en entrenamiento y prueba (80/20, seed=42) para una mejor estimación de resultados, asi como una prueba en donde se utilizó todo el dataset como entrenamiento y testing. Se aplicaron procesos básicos de limpieza de valores faltantes y estandarización de tokens para asegurar consistencia en las predicciones.

3. Algoritmo

En este proyecto se empleó el DecisionTreeClassifier de scikit-learn, que implementa de manera optimizada los árboles de decisión basados en criterios como entropy o gini. El modelo se entrenó tras un proceso de limpieza y codificación de datos, ajustando hiperparámetros como la profundidad máxima y el número mínimo de muestras por nodo para controlar el sobreajuste. Con fit() se entrenó el clasificador, con predict() se generaron predicciones y su desempeño se evaluó con métricas estándar como accuracy, precisión, recall y F1-score. Además, se utilizó la visualización del árbol para interpretar fácilmente las reglas de decisión aprendidas.

3.1. Descripción general

3.1.1. ¿Qué hace?

El modelo construye un árbol de decisión que divide el espacio de atributos en función de los valores que mejor separan las clases, utilizando criterios como ganancia de información entropy. En cada nodo, el algoritmo selecciona automáticamente el umbral o división más informativa y genera ramas que llevan a predicciones finales. Una vez entrenado, el árbol puede recorrer nuevas instancias hasta llegar a una clase predicha, permitiendo interpretar fácilmente las reglas de decisión aprendidas.

3.1.2. ¿Cuándo se usa?

Este modelo resulta especialmente útil en problemas de clasificación supervisada donde se valora la interpretabilidad del proceso de decisión. Es ideal cuando se requiere una solución ligera, implementada sin necesidad de frameworks externos, pero que aún así pueda trabajar con datos mixtos que combinan atributos numéricos y categóricos. Además, su capacidad de

manejar valores incompletos lo convierte en una opción práctica para conjuntos de datos reales, donde la información suele ser imperfecta o estar incompleta.

4. Resultados de experimentación

Para evaluar el desempeño del clasificador implementado con frameworks, se diseñaron dos etapas de pruebas. En la primera, se entrenó y evaluó el modelo utilizando el dataset completo, con el fin de observar su comportamiento en un escenario sin separación de datos. Posteriormente, se aplicó una partición 80/20 (train/test, con semilla fija de 42), lo que permitió medir de forma más realista la capacidad de generalización sobre datos no vistos. Además, según las necesidades del usuario, se incorporó la posibilidad de usar GridSearchCV para realizar una búsqueda sistemática de hiperparámetros (como profundidad máxima, criterio de división o número mínimo de muestras por nodo), lo que facilita encontrar la configuración más adecuada del modelo. En ambos casos se registraron métricas y resultados gráficos para comparar desempeño y robustez.

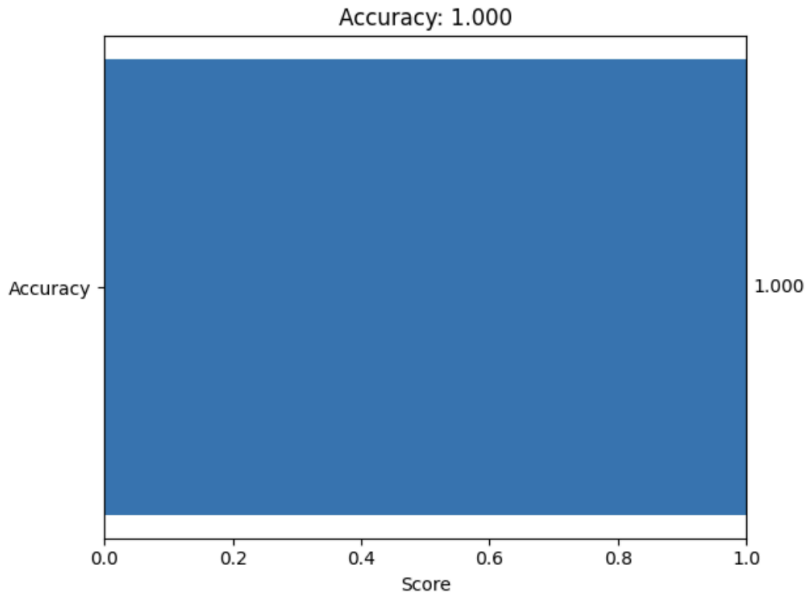
4.1. (Dataset: Tennis)

```
[run] Dataset: data/tennis.csv
[run] Target : Play
[run] GridSearchCV: Desactivado

=== SHOWCASE ===
** Training and testing on the entire dataset **

Training accuracy = 1.0000 (results saved in results/showcase)


=== VALIDATION ===
** Training/testing split: 80/20, seed=42 **

Test accuracy = 1.0000 (results saved in results/validation)
```

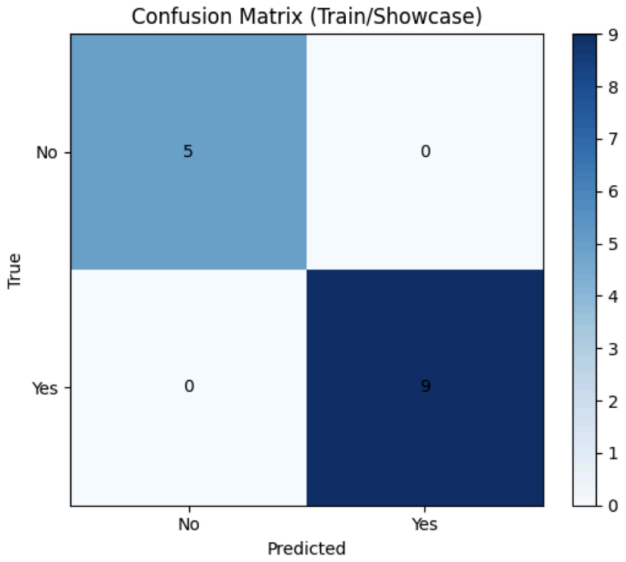
```
[run] Dataset: data/tennis.csv
[run] Target : Play
[run] GridSearchCV: Activado

=== SHOWCASE ===
** Training and testing on the entire dataset **

Training accuracy = 0.8571 (results saved in results/showcase)

/Library/Frameworks/Python.framework/Versions/3.12/lib/python3.12/site-packages/sklearn/model_
selection/_split.py:811: UserWarning: The least populated class in y has only 4 members, which
 is less than n_splits=5.
  warnings.warn(

=== VALIDATION ===
** Training/testing split: 80/20, seed=42 **

Test accuracy = 0.6667 (results saved in results/validation)
```

4.1.1. Showcase (entrenamiento sobre todo el dataset)

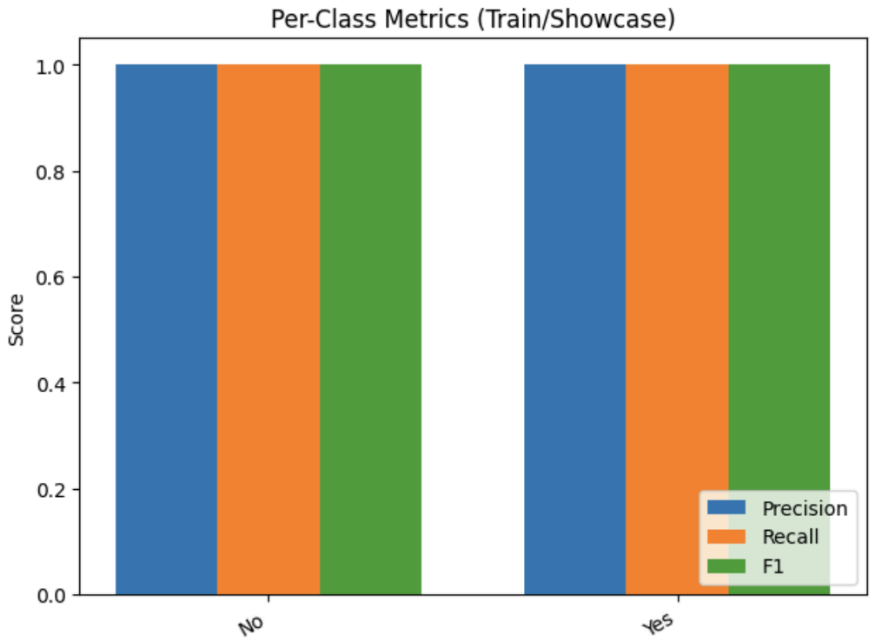
- **Accuracy entrenamiento:** 1.000



- **Matriz de confusión:** 5 aciertos en clase "No" y 9 en clase "Yes". Sin errores.



- **Métricas por clase:** Precisión, Recall y F1 = 1.0 para ambas clases.



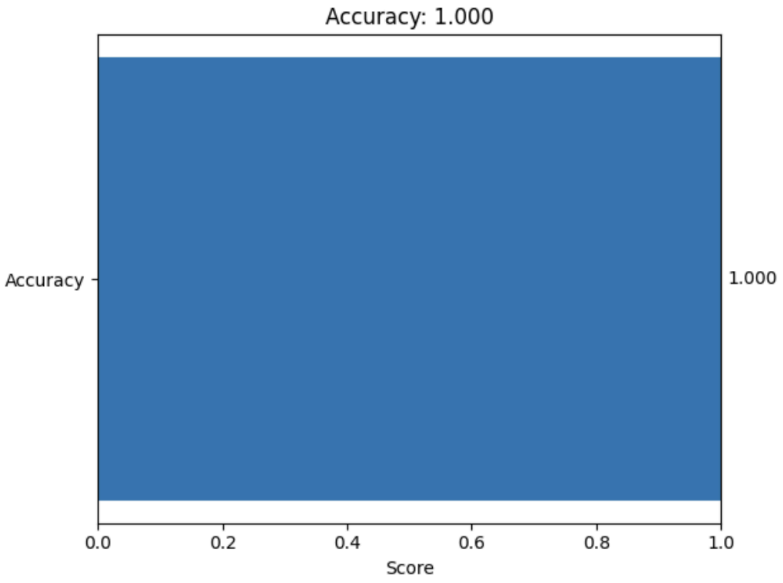
- **Árbol aprendido:** Raíz en Outlook, divisiones sucesivas en Windy y Humidity, perfectamente alineado con las reglas conocidas del dataset.

```
|--- Outlook_Overcast <= 0.50
|   |--- Humidity_Normal <= 0.50
|   |   |--- Outlook_Rain <= 0.50
|   |   |   |--- class: No
|   |   |--- Outlook_Rain >  0.50
|   |   |   |--- Windy_Strong <= 0.50
|   |   |   |   |--- class: Yes
|   |   |   |--- Windy_Strong >  0.50
|   |   |   |   |--- class: No
|   |--- Humidity_Normal >  0.50
|   |   |--- Windy_Strong <= 0.50
|   |   |   |--- class: Yes
|   |   |--- Windy_Strong >  0.50
|   |   |   |--- Outlook_Sunny <= 0.50
|   |   |   |   |--- class: No
|   |   |   |--- Outlook_Sunny >  0.50
|   |   |   |   |--- class: Yes
|--- Outlook_Overcast >  0.50
|   |--- class: Yes
```

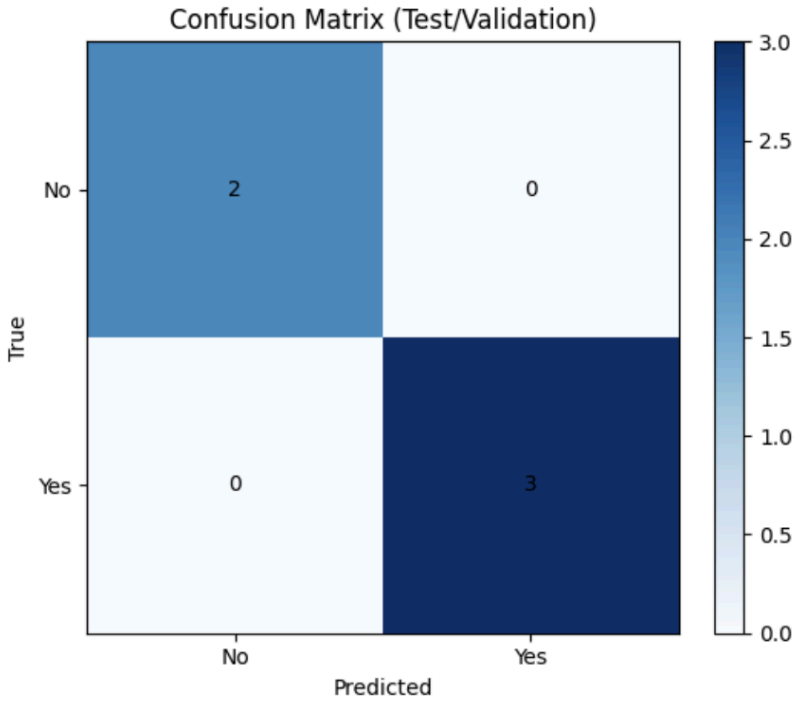
Interpretación: El dataset es pequeño y limpio, por lo que el algoritmo logra memorizar sin problema las reglas exactas, alcanzando precisión perfecta.

4.1.2 Validación (split 80/20, seed=42)

- **Accuracy test:** 1.000



- **Matriz de confusión:** 2 aciertos en clase "No" y 3 en clase "Yes". Sin errores.



- **Métricas por clase:** Precisión, Recall y F1 = 1.0 en ambas clases.

- **Árbol aprendido:** Raíz en Outlook, divisiones sucesivas en Windy y Humidity, perfectamente alineado con las reglas conocidas del dataset.

```
|--- Outlook_Overcast <= 0.50
|   |--- Humidity_Normal <= 0.50
|   |   |--- Outlook_Rain <= 0.50
|   |   |   |--- class: No
|   |   |--- Outlook_Rain >  0.50
|   |   |   |--- Windy_Strong <= 0.50
|   |   |   |   |--- class: Yes
|   |   |   |--- Windy_Strong >  0.50
|   |   |   |   |--- class: No
|   |--- Humidity_Normal >  0.50
|   |   |--- Windy_Strong <= 0.50
|   |   |   |--- class: Yes
|   |   |--- Windy_Strong >  0.50
|   |   |   |--- Outlook_Sunny <= 0.50
|   |   |   |   |--- class: No
|   |   |   |--- Outlook_Sunny >  0.50
|   |   |   |   |--- class: Yes
|--- Outlook_Overcast >  0.50
|   |--- class: Yes
```

Interpretación: El árbol generaliza bien en este caso, pues incluso con división aleatoria mantiene 100% de exactitud. Dado el tamaño reducido (12 instancias en train, 3 en test), no sorprende que pueda separar perfectamente.

4.1.3. Interpretación final

- **Fortaleza:** El algoritmo implementado se ajusta perfecto al dataset Tennis, confirmando que la implementación funciona correctamente en escenarios sencillos.

- **Limitación:** Al ser un dataset muy pequeño y lineal, los resultados no se pueden extrapolar como evidencia de desempeño en conjuntos más grandes. Además, cuando se activa GridSearchCV, los resultados muestran una caída en la exactitud (0.8571 en entrenamiento y 0.6667 en validación), reflejando que el ajuste de hiperparámetros en datasets reducidos puede generar particiones poco representativas y pérdida de generalización. Esto evidencia que, en problemas con pocos datos, la búsqueda sistemática puede no ser adecuada, y confirma la necesidad de usar GridSearchCV solo en datasets con mayor tamaño o balance.

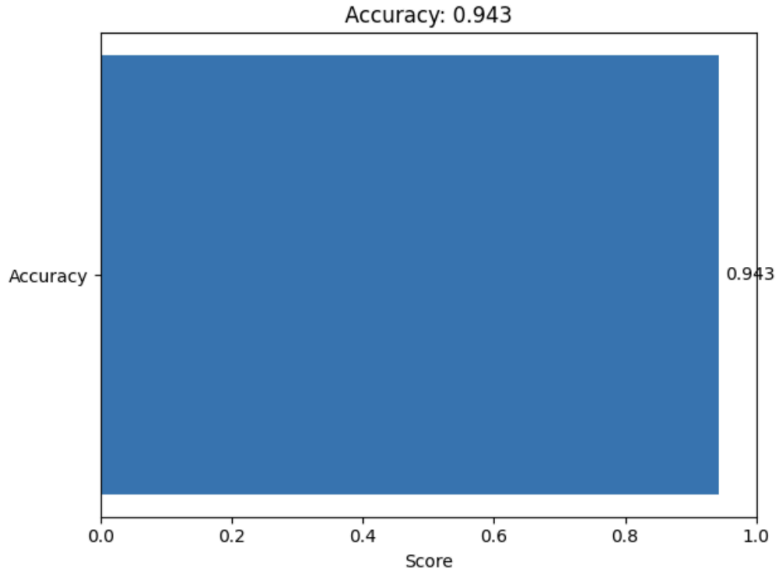
4.2. (Dataset: academic Stress level - maintainance 1)

```
[run] Dataset: data/academic Stress level - maintainance 1.csv
[run] Target : Your Academic Stage
[run] GridSearchCV: Desactivado

=== SHOWCASE ===
** Training and testing on the entire dataset **

Training accuracy = 1.0000 (results saved in results/showcase)


=== VALIDATION ===
** Training/testing split: 80/20, seed=42 **

Test accuracy = 0.7143 (results saved in results/validation)
```

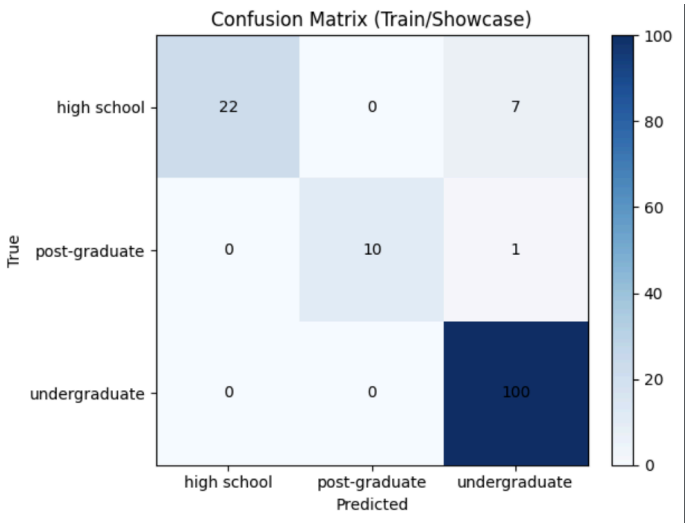
```
[run] Dataset: data/academic Stress level - maintainance 1.csv
[run] Target : Your Academic Stage
[run] GridSearchCV: Activado

=== SHOWCASE ===
** Training and testing on the entire dataset **

Training accuracy = 0.7429 (results saved in results/showcase)


=== VALIDATION ===
** Training/testing split: 80/20, seed=42 **

Test accuracy = 0.7143 (results saved in results/validation)
```

4.2.1. Showcase (entrenamiento sobre todo el dataset)

- **Accuracy entrenamiento:** 0.943

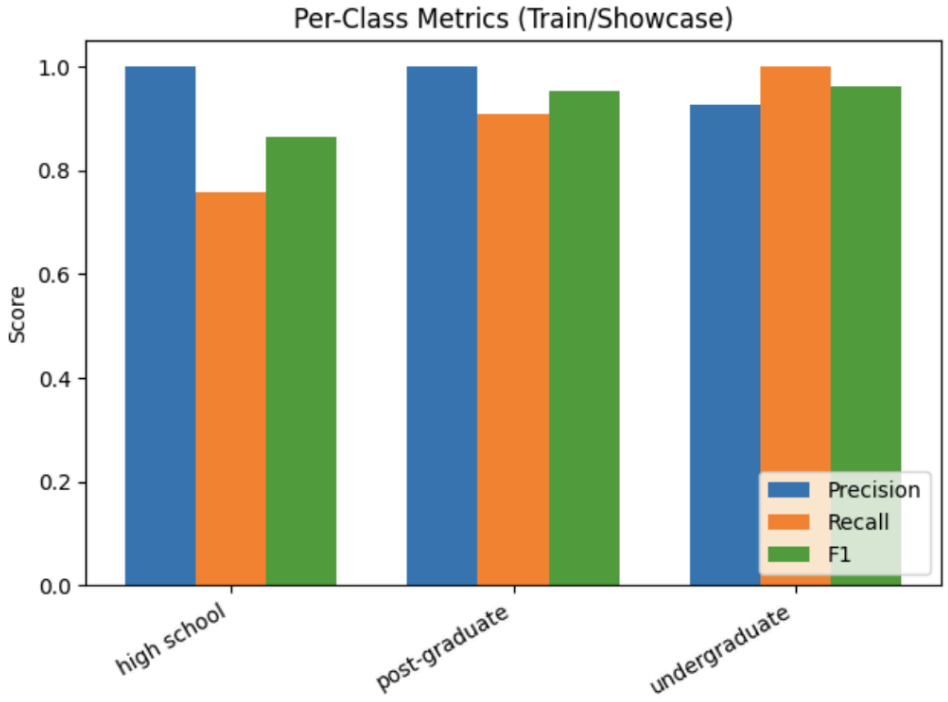


- **Matriz de confusión:**
  - High school: 22 correctos, 7 mal clasificados como undergraduate.
  - Post-graduate: 10 correctos, 1 error hacia undergraduate.
  - Undergraduate: 100 correctos sin errores.



- **Métricas por clase:**
  - High school: Precisión = 1.0, Recall = 0.76, F1 ≈ 0.87

- Post-graduate: Precisión = 1.0, Recall = 0.91, F1 ≈ 0.95
- Undergraduate: Precisión = 0.93, Recall = 1.0, F1 ≈ 0.96



- **Árbol aprendido:** La raíz se forma en la variable Academic pressure from your home, seguida de divisiones por competition y otros atributos, con hojas puras en varias ramas.

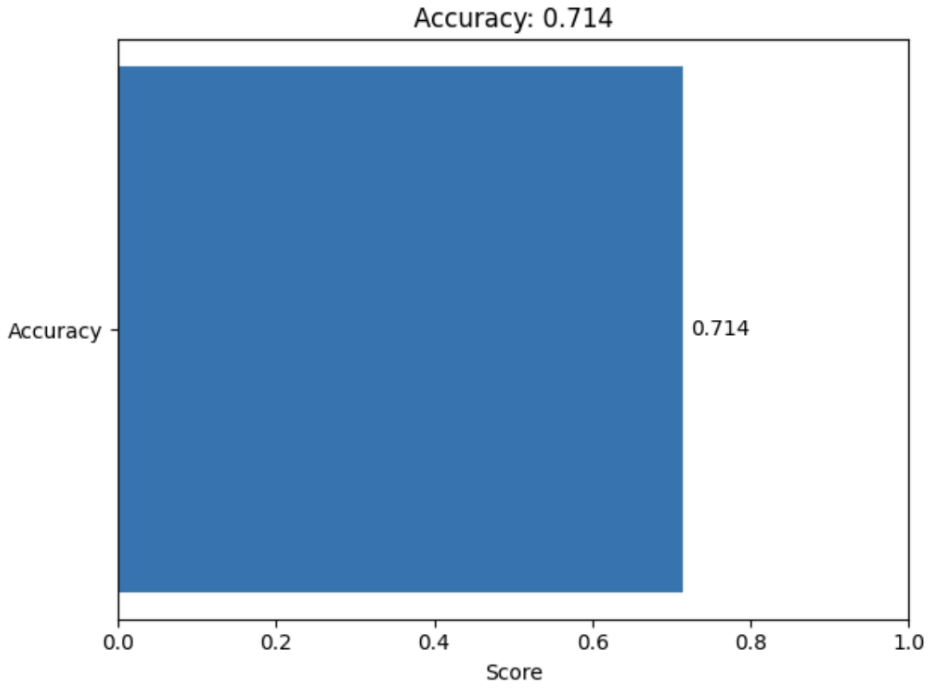
```
└── Academic pressure from your home <= 5 ?
    ├── LE
    │   ├── What would you rate the academic  competition in your student life <= 5 ?
    │   │   ├── LE
    │   │   │   ├── Timestamp?
    │   │   │   │   ├── 11/08/2025 12:15:00 → post-graduate [counts={'post-graduate': 1}]
    │   │   │   │   ├── 11/08/2025 18:07:26 → undergraduate [counts={'undergraduate': 1}]
    │   │   │   │   ├── 11/08/2025 19:29:07 → post-graduate [counts={'post-graduate': 1}]
    │   │   │   │   ├── 12/08/2025 02:28:42 → post-graduate [counts={'post-graduate': 1}]
    │   │   │   │   ├── 12/08/2025 08:11:48 → post-graduate [counts={'post-graduate': 1}]
    │   │   │   │   ├── 12/08/2025 10:03:58 → undergraduate [counts={'undergraduate': 1}]
    │   │   │   │   ├── 12/08/2025 13:16:50 → undergraduate [counts={'undergraduate': 1}]
    │   │   │   │   ├── 12/08/2025 15:01:20 → high school [counts={'high school': 1}]
    │   │   │   │   ├── 13/08/2025 21:45:58 → undergraduate [counts={'undergraduate': 1}]
    │   │   │   │   ├── 14/08/2025 06:10:01 → post-graduate [counts={'post-graduate': 1}]
    │   │   │   │   ├── 14/08/2025 21:06:15 → undergraduate [counts={'undergraduate': 1}]
    │   │   │   │   ├── 17/08/2025 13:02:04 → undergraduate [counts={'undergraduate': 1}]
    │   │   │   │   ├── 18/08/2025 14:36:00 → undergraduate [counts={'undergraduate': 1}]
    │   │   │   │   ├── 18/08/2025 17:13:52 → undergraduate [counts={'undergraduate': 1}]
    │   │   │   │   ├── 18/08/2025 22:40:13 → undergraduate [counts={'undergraduate': 1}]
    │   │   │   │   ├── 24/07/2025 22:05:39 → undergraduate [counts={'undergraduate': 1}]
    │   │   │   │   ├── 24/07/2025 22:05:52 → undergraduate [counts={'undergraduate': 1}]
    │   │   │   │   ├── 24/07/2025 22:06:39 → undergraduate [counts={'undergraduate': 1}]
    │   │   │   │   ├── 24/07/2025 22:06:45 → undergraduate [counts={'undergraduate': 1}]
    │   │   │   │   ├── 24/07/2025 22:08:06 → undergraduate [counts={'undergraduate': 1}]
    │   │   │   │   ├── 24/07/2025 22:08:13 → undergraduate [counts={'undergraduate': 1}]
    │   │   │   │   ├── 24/07/2025 22:09:21 → undergraduate [counts={'undergraduate': 1}]
    │   │   │   │   ├── 24/07/2025 22:10:06 → undergraduate [counts={'undergraduate': 1}]
    │   │   │   │   ├── 24/07/2025 22:11:01 → undergraduate [counts={'undergraduate': 1}]
    │   │   │   │   ├── 24/07/2025 22:11:19 → undergraduate [counts={'undergraduate': 1}]
    │   │   │   │   ├── 24/07/2025 22:12:12 → undergraduate [counts={'undergraduate': 1}]
    │   │   │   │   ├── 24/07/2025 22:12:24 → undergraduate [counts={'undergraduate': 1}]
    │   │   │   │   ├── 24/07/2025 22:12:27 → undergraduate [counts={'undergraduate': 1}]
    │   │   │   │   ├── 24/07/2025 22:12:48 → undergraduate [counts={'undergraduate': 1}]
    │   │   │   │   ├── 24/07/2025 22:14:16 → undergraduate [counts={'undergraduate': 1}]
    │   │   │   │   ├── 24/07/2025 22:15:06 → undergraduate [counts={'undergraduate': 1}]
    │   │   │   │   ├── 24/07/2025 22:15:39 → undergraduate [counts={'undergraduate': 1}]
    │   │   │   │   ├── 24/07/2025 22:16:10 → undergraduate [counts={'undergraduate': 1}]
    │   │   │   │   ├── 24/07/2025 22:16:53 → undergraduate [counts={'undergraduate': 1}]
    │   │   │   │   ├── 24/07/2025 22:17:46 → undergraduate [counts={'undergraduate': 1}]
    │   │   │   │   ├── 24/07/2025 22:18:02 → undergraduate [counts={'undergraduate': 1}]
    │   │   │   │   ├── 24/07/2025 22:18:44 → undergraduate [counts={'undergraduate': 1}]
    │   │   │   │   ├── 24/07/2025 22:18:55 → undergraduate [counts={'undergraduate': 1}]
    │   │   │   │   ├── 24/07/2025 22:18:58 → undergraduate [counts={'undergraduate': 1}]
    │   │   │   │   ├── 24/07/2025 22:19:06 → undergraduate [counts={'undergraduate': 1}]
    │   │   │   │   ├── 24/07/2025 22:19:22 → undergraduate [counts={'undergraduate': 1}]
    │   │   │   │   ├── 24/07/2025 22:19:25 → undergraduate [counts={'undergraduate': 1}]
```

```
|   |   |   |   ├── 24/07/2025 22:19:33 → high school [counts={'high school': 1}]
|   |   |   |   ├── 24/07/2025 22:19:51 → undergraduate [counts={'undergraduate': 1}]
|   |   |   |   ├── 24/07/2025 22:20:28 → undergraduate [counts={'undergraduate': 1}]
|   |   |   |   ├── 24/07/2025 22:21:04 → undergraduate [counts={'undergraduate': 1}]
|   |   |   |   ├── 24/07/2025 22:23:15 → undergraduate [counts={'undergraduate': 1}]
|   |   |   |   ├── 24/07/2025 22:24:13 → undergraduate [counts={'undergraduate': 1}]
|   |   |   |   ├── 24/07/2025 22:24:53 → undergraduate [counts={'undergraduate': 1}]
|   |   |   |   ├── 24/07/2025 22:25:17 → undergraduate [counts={'undergraduate': 1}]
|   |   |   |   ├── 24/07/2025 22:25:27 → undergraduate [counts={'undergraduate': 1}]
|   |   |   |   ├── 24/07/2025 22:25:41 → undergraduate [counts={'undergraduate': 1}]
|   |   |   |   ├── 24/07/2025 22:28:58 → undergraduate [counts={'undergraduate': 1}]
|   |   |   |   ├── 24/07/2025 22:29:26 → undergraduate [counts={'undergraduate': 1}]
|   |   |   |   ├── 24/07/2025 22:30:29 → undergraduate [counts={'undergraduate': 1}]
|   |   |   |   ├── 24/07/2025 22:32:22 → undergraduate [counts={'undergraduate': 1}]
|   |   |   |   ├── 24/07/2025 22:32:36 → post-graduate [counts={'post-graduate': 1}]
|   |   |   |   ├── 24/07/2025 22:32:37 → undergraduate [counts={'undergraduate': 1}]
|   |   |   |   ├── 24/07/2025 22:33:48 → undergraduate [counts={'undergraduate': 1}]
|   |   |   |   ├── 24/07/2025 22:34:34 → undergraduate [counts={'undergraduate': 1}]
|   |   |   |   ├── 24/07/2025 22:34:41 → undergraduate [counts={'undergraduate': 1}]
|   |   |   |   ├── 24/07/2025 22:34:53 → undergraduate [counts={'undergraduate': 1}]
|   |   |   |   ├── 24/07/2025 22:38:47 → undergraduate [counts={'undergraduate': 1}]
|   |   |   |   ├── 24/07/2025 22:39:36 → undergraduate [counts={'undergraduate': 1}]
|   |   |   |   ├── 24/07/2025 22:41:41 → undergraduate [counts={'undergraduate': 1}]
|   |   |   |   ├── 24/07/2025 22:42:11 → undergraduate [counts={'undergraduate': 1}]
|   |   |   |   ├── 24/07/2025 22:42:25 → undergraduate [counts={'undergraduate': 1}]
|   |   |   |   ├── 24/07/2025 22:42:48 → undergraduate [counts={'undergraduate': 1}]
|   |   |   |   ├── 24/07/2025 22:44:07 → undergraduate [counts={'undergraduate': 1}]
|   |   |   |   ├── 24/07/2025 22:44:50 → undergraduate [counts={'undergraduate': 1}]
|   |   |   |   ├── 24/07/2025 23:01:02 → undergraduate [counts={'undergraduate': 1}]
|   |   |   |   ├── 24/07/2025 23:02:49 → post-graduate [counts={'post-graduate': 1}]
|   |   |   |   ├── 24/07/2025 23:03:10 → undergraduate [counts={'undergraduate': 1}]
|   |   |   |   ├── 24/07/2025 23:04:01 → undergraduate [counts={'undergraduate': 1}]
|   |   |   |   ├── 24/07/2025 23:15:48 → undergraduate [counts={'undergraduate': 1}]
|   |   |   |   ├── 24/07/2025 23:22:17 → undergraduate [counts={'undergraduate': 1}]
|   |   |   |   ├── 24/07/2025 23:31:01 → undergraduate [counts={'undergraduate': 1}]
|   |   |   |   ├── 24/07/2025 23:32:26 → undergraduate [counts={'undergraduate': 1}]
|   |   |   |   ├── 24/07/2025 23:36:22 → undergraduate [counts={'undergraduate': 1}]
|   |   |   |   ├── 24/07/2025 23:44:35 → undergraduate [counts={'undergraduate': 1}]
|   |   |   |   ├── 25/07/2025 00:10:10 → undergraduate [counts={'undergraduate': 1}]
|   |   |   |   ├── 25/07/2025 00:13:46 → undergraduate [counts={'undergraduate': 1}]
|   |   |   |   ├── 25/07/2025 00:21:30 → undergraduate [counts={'undergraduate': 1}]
|   |   |   |   ├── 25/07/2025 00:21:38 → undergraduate [counts={'undergraduate': 1}]
|   |   |   |   ├── 25/07/2025 08:13:01 → undergraduate [counts={'undergraduate': 1}]
|   |   |   |   ├── 25/07/2025 08:42:22 → undergraduate [counts={'undergraduate': 1}]
|   |   |   |   ├── 25/07/2025 10:30:12 → undergraduate [counts={'undergraduate': 1}]
|   |   |   |   ├── 25/07/2025 10:37:39 → undergraduate [counts={'undergraduate': 1}]
|   |   |   |   ├── 25/07/2025 10:43:24 → undergraduate [counts={'undergraduate': 1}]
|   |   |   |   ├── 25/07/2025 11:00:10 → undergraduate [counts={'undergraduate': 1}]
|   |   |   |   ├── 25/07/2025 11:06:32 → undergraduate [counts={'undergraduate': 1}]
|   |   |   |   ├── 25/07/2025 11:06:40 → undergraduate [counts={'undergraduate': 1}]
|   |   |   |   ├── 25/07/2025 11:47:24 → undergraduate [counts={'undergraduate': 1}]
|   |   |   |   ├── 25/07/2025 13:17:44 → post-graduate [counts={'post-graduate': 1}]
|   |   |   |   ├── 25/07/2025 13:18:47 → undergraduate [counts={'undergraduate': 1}]
|   |   |   |   ├── 25/07/2025 13:38:10 → undergraduate [counts={'undergraduate': 1}]
|   |   |   |   ├── 25/07/2025 16:00:32 → undergraduate [counts={'undergraduate': 1}]
|   |   |   |   ├── 25/07/2025 16:04:38 → undergraduate [counts={'undergraduate': 1}]
|   |   |   |   ├── 25/07/2025 17:08:00 → undergraduate [counts={'undergraduate': 1}]
|   |   |   |   ├── 25/07/2025 17:51:17 → post-graduate [counts={'post-graduate': 1}]
|   |   |   |   ├── 25/07/2025 18:18:38 → undergraduate [counts={'undergraduate': 1}]
|   |   |   |   ├── 25/07/2025 18:27:24 → undergraduate [counts={'undergraduate': 1}]
|   |   |   |   ├── 25/07/2025 19:34:59 → undergraduate [counts={'undergraduate': 1}]
|   |   |   |   ├── 25/07/2025 20:37:52 → post-graduate [counts={'post-graduate': 1}]
|   |   |   |   ├── 25/07/2025 23:34:18 → undergraduate [counts={'undergraduate': 1}]
|   |   |   |   ├── 25/07/2025 23:39:15 → undergraduate [counts={'undergraduate': 1}]
|   |   |   |   ├── 26/07/2025 07:33:35 → high school [counts={'high school': 1}]
|   |   |   |   ├── 26/07/2025 07:50:38 → high school [counts={'high school': 1}]
|   |   |   |   ├── 26/07/2025 08:13:18 → high school [counts={'high school': 1}]
|   |   |   |   ├── 26/07/2025 08:25:17 → high school [counts={'high school': 1}]
|   |   |   |   ├── 26/07/2025 08:25:18 → high school [counts={'high school': 1}]
|   |   |   |   ├── 26/07/2025 08:27:10 → high school [counts={'high school': 1}]
|   |   |   |   ├── 26/07/2025 09:36:09 → high school [counts={'high school': 1}]
|   |   |   |   ├── 26/07/2025 09:47:04 → high school [counts={'high school': 1}]
|   |   |   |   ├── 26/07/2025 09:56:32 → high school [counts={'high school': 1}]
|   |   |   |   ├── 26/07/2025 10:01:33 → high school [counts={'high school': 1}]
|   |   |   |   ├── 26/07/2025 10:04:32 → high school [counts={'high school': 1}]
|   |   |   |   ├── 26/07/2025 10:38:24 → high school [counts={'high school': 1}]
|   |   |   |   ├── 26/07/2025 10:39:37 → high school [counts={'high school': 1}]
|   |   |   |   ├── 26/07/2025 11:36:30 → high school [counts={'high school': 1}]
|   |   |   |   ├── 26/07/2025 13:39:21 → undergraduate [counts={'undergraduate': 1}]
|   |   |   |   ├── 26/07/2025 14:21:53 → high school [counts={'high school': 1}]
|   |   |   |   ├── 26/07/2025 14:23:45 → high school [counts={'high school': 1}]
|   |   |   |   ├── 26/07/2025 18:45:13 → high school [counts={'high school': 1}]
|   |   |   |   ├── 26/07/2025 21:37:46 → high school [counts={'high school': 1}]
|   |   |   |   ├── 26/07/2025 22:11:02 → high school [counts={'high school': 1}]
|   |   |   |   ├── 27/07/2025 13:45:00 → undergraduate [counts={'undergraduate': 1}]
|   |   |   |   ├── 27/07/2025 22:56:54 → high school [counts={'high school': 1}]
|   |   |   |   └── 30/07/2025 16:02:16 → undergraduate [counts={'undergraduate': 1}]
|   |   |   └── GT → post-graduate [counts={'post-graduate': 1}]
|   |   └── GT
|   |       └── Academic pressure from your home <= 5 ?
|   |           ├── LE → high school [counts={'high school': 7}]
|   |           └── GT → undergraduate [counts={'undergraduate': 3}]
```

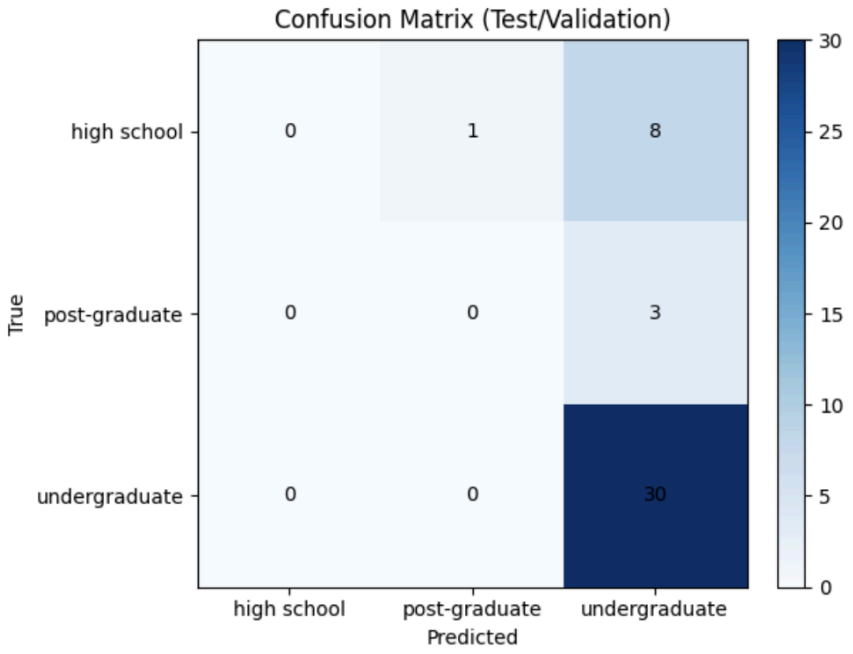
Interpretación: El modelo logra un buen ajuste (94.3%), aunque se observa que la clase High school es la más difícil de separar, por menor cantidad de datos o mayor solapamiento con Undergraduate.

4.2.2 Validación (split 80/20, seed=42)

- **Accuracy test:** 0.714

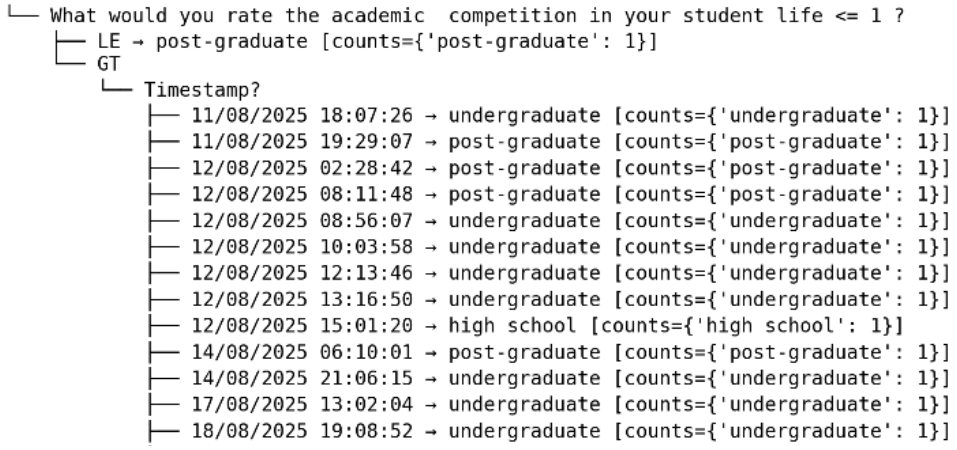


- **Matriz de confusión:**
    - High school: 0 correctos, 9 mal clasificados (8 como undergrad, 1 como post-grad).
    - Post-graduate: 0 correctos, 3 mal clasificados como undergrad.
    - Undergraduate: 30 correctos.



- **Métricas por clase:**
  - High school: Precisión = 0.0, Recall = 0.0, F1 = 0.0
  - Post-graduate: Precisión = 0.0, Recall = 0.0, F1 = 0.0
  - Undergraduate: Precisión ≈ 0.71, Recall = 1.0, F1 ≈ 0.83

- **Árbol aprendido:** La raíz se forma en la variable competition, con ramas que colapsan hacia Undergraduate. Esto muestra que el árbol se sesgó fuertemente a la clase mayoritaria.



```
├── 18/08/2025 19:08:52 → undergraduate [counts={'undergraduate': 1}]
├── 18/08/2025 22:40:13 → undergraduate [counts={'undergraduate': 1}]
├── 24/07/2025 22:05:39 → undergraduate [counts={'undergraduate': 1}]
├── 24/07/2025 22:05:52 → undergraduate [counts={'undergraduate': 1}]
├── 24/07/2025 22:06:45 → undergraduate [counts={'undergraduate': 1}]
├── 24/07/2025 22:08:06 → undergraduate [counts={'undergraduate': 1}]
├── 24/07/2025 22:08:13 → undergraduate [counts={'undergraduate': 1}]
├── 24/07/2025 22:09:21 → undergraduate [counts={'undergraduate': 1}]
├── 24/07/2025 22:10:06 → undergraduate [counts={'undergraduate': 1}]
├── 24/07/2025 22:11:01 → undergraduate [counts={'undergraduate': 1}]
├── 24/07/2025 22:12:12 → undergraduate [counts={'undergraduate': 1}]
├── 24/07/2025 22:12:48 → undergraduate [counts={'undergraduate': 1}]
├── 24/07/2025 22:14:16 → undergraduate [counts={'undergraduate': 1}]
├── 24/07/2025 22:15:06 → undergraduate [counts={'undergraduate': 1}]
├── 24/07/2025 22:15:39 → undergraduate [counts={'undergraduate': 1}]
├── 24/07/2025 22:16:10 → undergraduate [counts={'undergraduate': 1}]
├── 24/07/2025 22:18:02 → undergraduate [counts={'undergraduate': 1}]
├── 24/07/2025 22:18:55 → undergraduate [counts={'undergraduate': 1}]
├── 24/07/2025 22:18:58 → undergraduate [counts={'undergraduate': 1}]
├── 24/07/2025 22:19:06 → undergraduate [counts={'undergraduate': 1}]
├── 24/07/2025 22:19:22 → undergraduate [counts={'undergraduate': 1}]
├── 24/07/2025 22:19:25 → undergraduate [counts={'undergraduate': 1}]
├── 24/07/2025 22:20:28 → undergraduate [counts={'undergraduate': 1}]
├── 24/07/2025 22:24:53 → undergraduate [counts={'undergraduate': 1}]
├── 24/07/2025 22:25:17 → undergraduate [counts={'undergraduate': 1}]
├── 24/07/2025 22:25:27 → undergraduate [counts={'undergraduate': 1}]
├── 24/07/2025 22:25:41 → undergraduate [counts={'undergraduate': 1}]
├── 24/07/2025 22:29:26 → undergraduate [counts={'undergraduate': 1}]
├── 24/07/2025 22:30:29 → undergraduate [counts={'undergraduate': 1}]
├── 24/07/2025 22:32:22 → undergraduate [counts={'undergraduate': 1}]
├── 24/07/2025 22:32:36 → post-graduate [counts={'post-graduate': 1}]
├── 24/07/2025 22:32:37 → undergraduate [counts={'undergraduate': 1}]
├── 24/07/2025 22:34:41 → undergraduate [counts={'undergraduate': 1}]
├── 24/07/2025 22:34:53 → undergraduate [counts={'undergraduate': 1}]
├── 24/07/2025 22:39:36 → undergraduate [counts={'undergraduate': 1}]
├── 24/07/2025 22:41:41 → undergraduate [counts={'undergraduate': 1}]
├── 24/07/2025 22:42:11 → undergraduate [counts={'undergraduate': 1}]
├── 24/07/2025 22:42:25 → undergraduate [counts={'undergraduate': 1}]
├── 24/07/2025 22:44:07 → undergraduate [counts={'undergraduate': 1}]
├── 24/07/2025 22:44:50 → undergraduate [counts={'undergraduate': 1}]
├── 24/07/2025 23:01:02 → undergraduate [counts={'undergraduate': 1}]
├── 24/07/2025 23:03:10 → undergraduate [counts={'undergraduate': 1}]
├── 24/07/2025 23:15:48 → undergraduate [counts={'undergraduate': 1}]
├── 24/07/2025 23:22:17 → undergraduate [counts={'undergraduate': 1}]
├── 24/07/2025 23:32:26 → undergraduate [counts={'undergraduate': 1}]
├── 24/07/2025 23:44:35 → undergraduate [counts={'undergraduate': 1}]
├── 25/07/2025 00:10:10 → undergraduate [counts={'undergraduate': 1}]
├── 25/07/2025 00:13:46 → undergraduate [counts={'undergraduate': 1}]
├── 25/07/2025 00:21:38 → undergraduate [counts={'undergraduate': 1}]
├── 25/07/2025 10:30:12 → undergraduate [counts={'undergraduate': 1}]
├── 25/07/2025 10:37:39 → undergraduate [counts={'undergraduate': 1}]
├── 25/07/2025 10:43:24 → undergraduate [counts={'undergraduate': 1}]
├── 25/07/2025 11:06:32 → undergraduate [counts={'undergraduate': 1}]
├── 25/07/2025 11:06:40 → undergraduate [counts={'undergraduate': 1}]
├── 25/07/2025 13:17:44 → post-graduate [counts={'post-graduate': 1}]
├── 25/07/2025 13:38:10 → undergraduate [counts={'undergraduate': 1}]
├── 25/07/2025 16:00:32 → undergraduate [counts={'undergraduate': 1}]
├── 25/07/2025 16:31:45 → post-graduate [counts={'post-graduate': 1}]
├── 25/07/2025 17:08:00 → undergraduate [counts={'undergraduate': 1}]
├── 25/07/2025 18:27:24 → undergraduate [counts={'undergraduate': 1}]
├── 25/07/2025 19:34:59 → undergraduate [counts={'undergraduate': 1}]
├── 25/07/2025 23:34:18 → undergraduate [counts={'undergraduate': 1}]
├── 25/07/2025 23:39:15 → undergraduate [counts={'undergraduate': 1}]
├── 26/07/2025 07:25:43 → high school [counts={'high school': 1}]
├── 26/07/2025 07:33:35 → high school [counts={'high school': 1}]
├── 26/07/2025 07:34:28 → high school [counts={'high school': 1}]
├── 26/07/2025 07:50:38 → high school [counts={'high school': 1}]
├── 26/07/2025 08:13:18 → high school [counts={'high school': 1}]
├── 26/07/2025 08:25:17 → high school [counts={'high school': 1}]
├── 26/07/2025 08:25:18 → high school [counts={'high school': 1}]
├── 26/07/2025 08:57:40 → high school [counts={'high school': 1}]
├── 26/07/2025 09:28:08 → high school [counts={'high school': 1}]
├── 26/07/2025 09:47:04 → high school [counts={'high school': 1}]
├── 26/07/2025 09:56:32 → high school [counts={'high school': 1}]
├── 26/07/2025 10:01:33 → high school [counts={'high school': 1}]
├── 26/07/2025 10:38:24 → high school [counts={'high school': 1}]
├── 26/07/2025 10:39:37 → high school [counts={'high school': 1}]
├── 26/07/2025 11:36:30 → high school [counts={'high school': 1}]
├── 26/07/2025 13:39:21 → undergraduate [counts={'undergraduate': 1}]
├── 26/07/2025 14:21:53 → high school [counts={'high school': 1}]
├── 26/07/2025 21:37:46 → high school [counts={'high school': 1}]
├── 26/07/2025 22:11:02 → high school [counts={'high school': 1}]
├── 27/07/2025 13:45:00 → undergraduate [counts={'undergraduate': 1}]
├── 27/07/2025 22:56:54 → high school [counts={'high school': 1}]
└── 30/07/2025 16:02:16 → undergraduate [counts={'undergraduate': 1}]
```

Interpretación: El rendimiento baja a 71.4%, evidenciando sobreajuste en entrenamiento y dificultad del modelo para reconocer las clases minoritarias (High school y Post-graduate).

4.2.3. Interpretación final

- **Fortaleza:**El modelo detecta muy bien la clase mayoritaria (Undergraduate), confirmando que la implementación funciona en datasets reales y más ruidosos.

- **Limitación:** El algoritmo presenta dificultades para reconocer las clases minoritarias (High school y Post-graduate), lo que evidencia su sensibilidad a desbalances de clase. Al activar GridSearchCV, el entrenamiento bajó de 100% a 74.3% de exactitud, mientras que la validación se mantuvo en 71.4%. Esto sugiere que la búsqueda sistemática de hiperparámetros puede evitar el sobreajuste extremo, pero no resuelve el problema de desbalance ni mejora el desempeño en test. En consecuencia, se requieren técnicas adicionales para mejorar la capacidad predictiva en clases minoritarias.

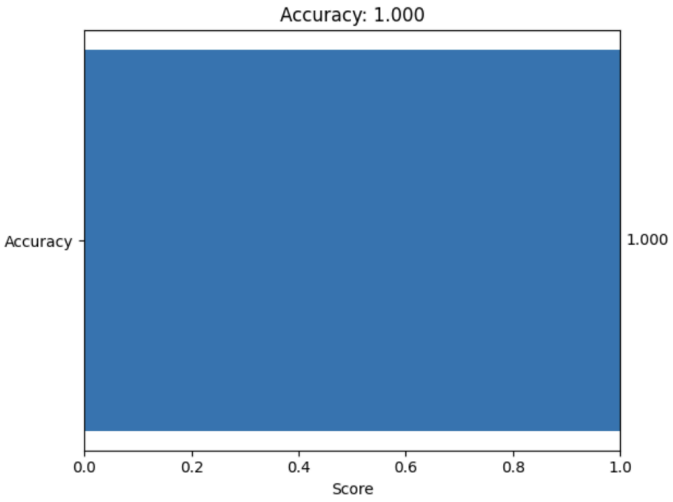
4.3. (Dataset: mushrooms)

```
[run] Dataset: data/mushrooms.csv
[run] Target : class
[run] GridSearchCV: Desactivado

=== SHOWCASE ===
** Training and testing on the entire dataset **

Training accuracy = 1.0000 (results saved in results/showcase)


=== VALIDATION ===
** Training/testing split: 80/20, seed=42 **

Test accuracy = 1.0000 (results saved in results/validation)
```

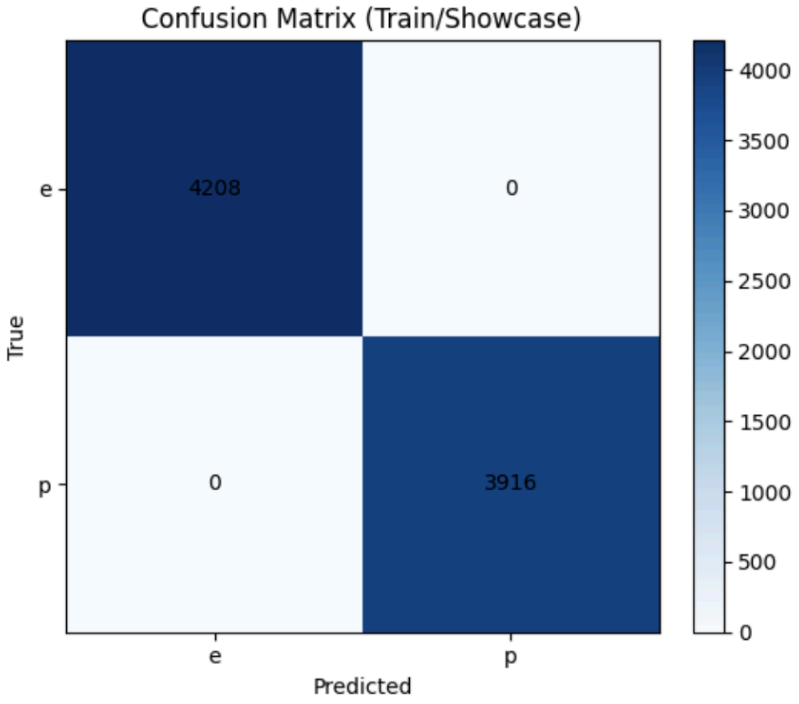
```
[run] Dataset: data/mushrooms.csv
[run] Target : class
[run] GridSearchCV: Activado

=== SHOWCASE ===
** Training and testing on the entire dataset **

Training accuracy = 1.0000 (results saved in results/showcase)


=== VALIDATION ===
** Training/testing split: 80/20, seed=42 **

Test accuracy = 1.0000 (results saved in results/validation)
```

4.3.1. Showcase (entrenamiento sobre todo el dataset)

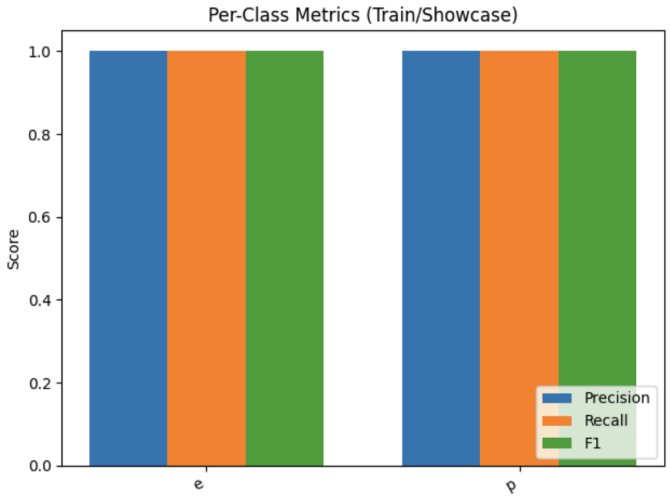
- **Accuracy entrenamiento:** 1.000



- **Matriz de confusión:** Todos los ejemplos cayeron en la diagonal correctamente clasificados.

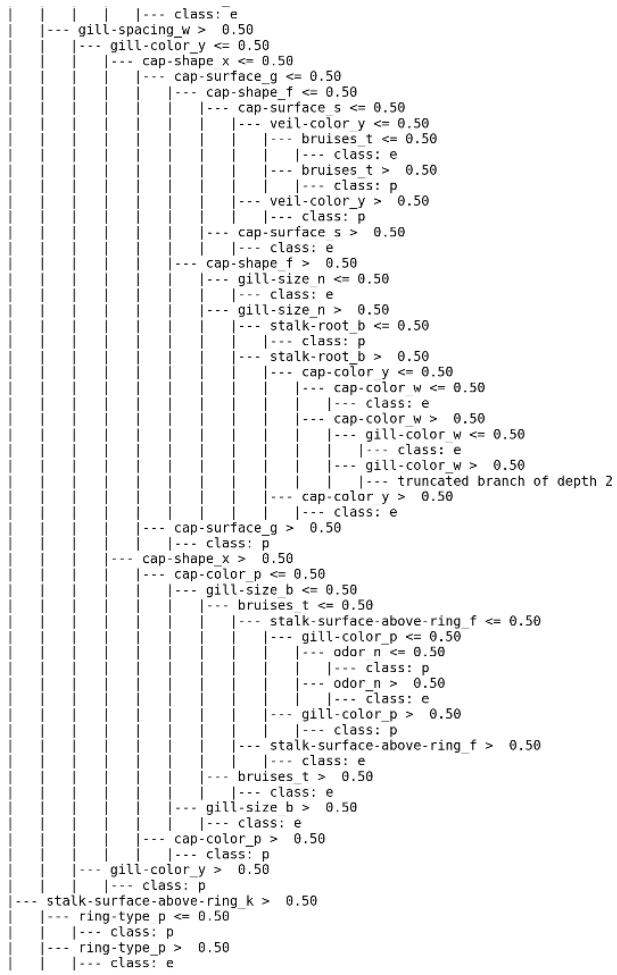


- **Métricas por clase:** Para ambas clases (e y p), las métricas alcanzaron el valor máximo (1.0). Esto confirma que el árbol logra una separación perfecta.



- **Árbol aprendido:** La raíz comienza en la característica odor, lo cual es lógico ya que el olor es un predictor muy fuerte en este dataset. Las ramas dividen de inmediato entre clases e/p de manera clara.

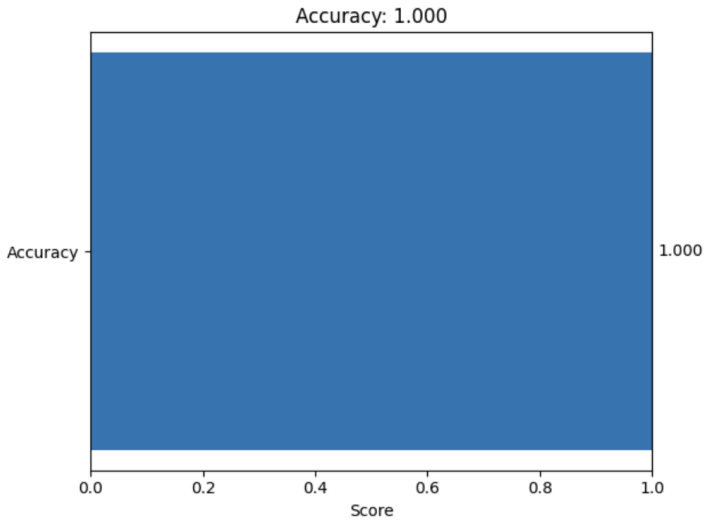
```
|--- stalk-surface-above-ring_k <= 0.50
|   |--- gill-spacing_w <= 0.50
|   |   |--- bruises_t <= 0.50
|   |   |   |--- stalk-shape_t <= 0.50
|   |   |   |   |--- stalk-color-above-ring_w <= 0.50
|   |   |   |   |   |--- class: e
|   |   |   |   |--- stalk-color-above-ring_w >  0.50
|   |   |   |   |   |--- cap-color_r <= 0.50
|   |   |   |   |   |   |--- stalk-root_b <= 0.50
|   |   |   |   |   |   |   |--- class: e
|   |   |   |   |   |   |--- stalk-root_b >  0.50
|   |   |   |   |   |   |   |--- class: p
|   |   |   |   |   |--- cap-color_r >  0.50
|   |   |   |   |   |   |--- class: e
|   |   |   |--- stalk-shape_t >  0.50
|   |   |   |   |--- class: p
|   |   |--- bruises_t >  0.50
|   |   |   |--- population_n <= 0.50
|   |   |   |   |--- gill-size_n <= 0.50
|   |   |   |   |   |--- cap-shape_f <= 0.50
|   |   |   |   |   |   |--- habitat_u <= 0.50
|   |   |   |   |   |   |   |--- spore-print-color_h <= 0.50
|   |   |   |   |   |   |   |   |--- stalk-shape_e <= 0.50
|   |   |   |   |   |   |   |   |   |--- class: e
|   |   |   |   |   |   |   |   |--- stalk-shape_e >  0.50
|   |   |   |   |   |   |   |   |   |--- gill-color_n <= 0.50
|   |   |   |   |   |   |   |   |   |   |--- gill-color_r <= 0.50
|   |   |   |   |   |   |   |   |   |   |   |--- truncated branch of depth 5
|   |   |   |   |   |   |   |   |   |   |--- gill-color_r >  0.50
|   |   |   |   |   |   |   |   |   |   |   |--- class: p
|   |   |   |   |   |   |   |   |   |--- gill-color_n >  0.50
|   |   |   |   |   |   |   |   |   |   |--- class: e
|   |   |   |   |   |   |   |--- spore-print-color_h >  0.50
|   |   |   |   |   |   |   |   |--- class: p
|   |   |   |   |   |   |--- habitat_u >  0.50
|   |   |   |   |   |   |   |--- class: p
|   |   |   |   |   |--- cap-shape_f >  0.50
|   |   |   |   |   |   |--- spore-print-color_n <= 0.50
|   |   |   |   |   |   |   |--- gill-color_r <= 0.50
|   |   |   |   |   |   |   |   |--- habitat_g <= 0.50
|   |   |   |   |   |   |   |   |   |--- cap-color_b <= 0.50
|   |   |   |   |   |   |   |   |   |   |--- stalk-color-above-ring_g <= 0.50
|   |   |   |   |   |   |   |   |   |   |   |--- truncated branch of depth 9
|   |   |   |   |   |   |   |   |   |   |--- stalk-color-above-ring_g >  0.50
|   |   |   |   |   |   |   |   |   |   |   |--- class: e
|   |   |   |   |   |   |   |   |   |--- cap-color_b >  0.50
|   |   |   |   |   |   |   |   |   |   |--- ring-type_e <= 0.50
|   |   |   |   |   |   |   |   |   |   |   |--- class: p
|   |   |   |   |   |   |   |   |   |   |--- ring-type_e >  0.50
|   |   |   |   |   |   |   |   |   |   |   |--- class: e
|   |   |   |   |   |   |   |   |--- habitat_g >  0.50
|   |   |   |   |   |   |   |   |   |--- stalk-root_r <= 0.50
|   |   |   |   |   |   |   |   |   |   |--- class: p
|   |   |   |   |   |   |   |   |   |--- stalk-root_r >  0.50
|   |   |   |   |   |   |   |   |   |   |--- class: e
|   |   |   |   |   |   |   |--- gill-color_r >  0.50
|   |   |   |   |   |   |   |   |--- class: p
|   |   |   |   |   |   |   |--- spore-print-color_n >  0.50
|   |   |   |   |   |   |   |   |--- class: e
|   |   |   |   |   |--- gill-size_n >  0.50
|   |   |   |   |   |   |--- class: p
|   |   |   |--- population_n >  0.50
|   |   |   |   |--- class: e
```



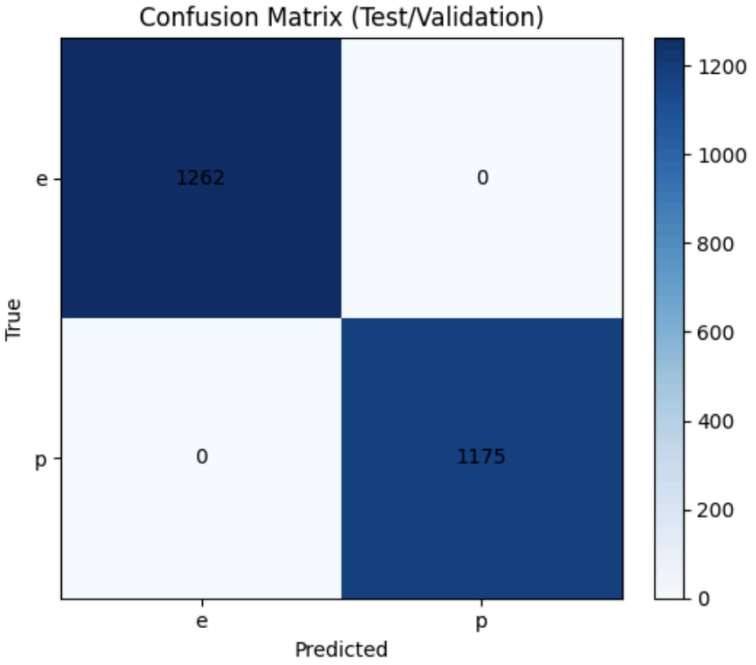
Interpretación: la importancia del olor confirma lo esperado: es un atributo determinante para saber si una seta es venenosa o comestible.

4.3.2 Validación (split 80/20, seed=42)

- **Accuracy test:** 1.000



- **Matriz de confusión:** Todos los ejemplos de test se encuentran en la diagonal, el árbol no pierde generalización, incluso al evaluar con datos no vistos.



- **Métricas por clase:** Precision, Recall y F1 = 1.0 en ambas clases, confirmando que no solo memorizó, sino que generalizó perfectamente en este dataset.

- **Árbol aprendido:** Nuevamente comienza en odor, seguido por atributos secundarios como gill-size, spore-print-color y stalk-surface-above-ring.

```
|--- odor_n <= 0.50
|   |--- bruises_f <= 0.50
|   |   |--- stalk-root_c <= 0.50
|   |   |   |--- stalk-root_r <= 0.50
|   |   |   |   |--- habitat_d <= 0.50
|   |   |   |   |   |--- class: p
|   |   |   |   |--- habitat_d >  0.50
|   |   |   |   |   |--- class: e
|   |   |   |--- stalk-root_r >  0.50
|   |   |   |   |--- class: e
|   |   |--- stalk-root_c >  0.50
|   |   |   |--- class: e
|   |--- bruises_f >  0.50
|   |   |--- class: p
|--- odor_n >  0.50
|   |--- spore-print-color_r <= 0.50
|   |   |--- stalk-surface-below-ring_y <= 0.50
|   |   |   |--- cap-surface_g <= 0.50
|   |   |   |   |--- gill-size_b <= 0.50
|   |   |   |   |   |--- bruises_f <= 0.50
|   |   |   |   |   |   |--- class: p
|   |   |   |   |   |--- bruises_f >  0.50
|   |   |   |   |   |   |--- class: e
|   |   |   |   |--- gill-size_b >  0.50
|   |   |   |   |   |--- class: e
|   |   |   |--- cap-surface_g >  0.50
|   |   |   |   |--- class: p
|   |   |--- stalk-surface-below-ring_y >  0.50
|   |   |   |--- population_y <= 0.50
|   |   |   |   |--- class: p
|   |   |   |--- population_y >  0.50
|   |   |   |   |--- class: e
|   |--- spore-print-color_r >  0.50
|   |   |--- class: p
```

Interpretación: Esto refleja que pocas características bastan para lograr separación perfecta.

4.3.3. Interpretación final

- **Fortaleza:** El algoritmo implementado demostró ser totalmente capaz de capturar las reglas de decisión en un dataset real y de gran tamaño (8124 instancias), manteniendo exactitud perfecta tanto en entrenamiento como en validación. Esto confirma que la integración con scikit-learn soporta datasets categóricos complejos y de gran escala.

- **Limitación:** La perfección en las métricas también refleja que este dataset es relativamente "fácil" para un árbol de decisión, ya que atributos como odor son casi deterministas. Además, al activar GridSearchCV, aunque los resultados fueron igualmente perfectos, el tiempo de cómputo aumentó considerablemente debido al tamaño del dataset y la búsqueda sistemática de combinaciones de hiperparámetros. Esto muestra que, en escenarios donde el dataset ya separa de forma casi lineal las clases, el uso de GridSearchCV puede ser redundante y poco eficiente.

5. Conclusiones

En este proyecto se implementó un clasificador de árbol de decisión utilizando scikit-learn, aprovechando su capacidad para manejar atributos numéricos y categóricos, ajustar hiperparámetros y visualizar reglas de decisión. Al evaluarlo en distintos datasets, se confirmó tanto la utilidad práctica del enfoque como sus limitaciones.

- **Tennis:** El modelo alcanzó 100% de exactitud en entrenamiento y validación cuando GridSearchCV estuvo desactivado, lo que muestra que la configuración básica es suficiente en escenarios sencillos. Sin embargo, al activar la búsqueda de hiperparámetros, la exactitud cayó (85.7% en entrenamiento y 66.7% en validación), evidenciando que en datasets pequeños el uso de GridSearchCV puede introducir particiones poco representativas y pérdida de generalización.
- **Academic Stress:** Los resultados fueron mixtos: sin GridSearchCV, el entrenamiento alcanzó 100% pero la validación bajó a 71.4%, mostrando fuerte sesgo hacia la clase mayoritaria. Con GridSearchCV, el entrenamiento se redujo a 74.3% y la validación se mantuvo en 71.4%, lo que indica que la búsqueda ayudó a mitigar el sobreajuste pero no resolvió el problema del desbalance. Esto confirma que en datasets ruidosos y desbalanceados se requieren técnicas adicionales como re-muestreo, ponderación de clases o ensambles.
- **Mushrooms:** El modelo clasificó perfectamente (100% de exactitud en entrenamiento y validación), tanto con como sin GridSearchCV. Esto valida que el algoritmo funciona en datasets grandes y categóricos, aunque también revela que este dataset es muy "fácil" para un árbol de decisión, ya que un solo atributo (odor) es casi determinista. Cabe destacar que con GridSearchCV el tiempo de cómputo fue considerablemente mayor, sin mejorar resultados.

5.1. Reflexión final

Trabajar con scikit-learn me permitió comprender cómo un framework robusto automatiza gran parte de la lógica de los árboles de decisión y facilita la experimentación con hiperparámetros. Confirmé que los árboles son modelos potentes, interpretables y fáciles de entrenar, pero también sensibles al sobreajuste, al desbalance de clases y a la complejidad de los datos.

El uso de GridSearchCV demostró ser una herramienta valiosa para optimizar hiperparámetros, aunque su beneficio depende del contexto: en datasets pequeños puede perjudicar el rendimiento, mientras que en datasets grandes y fáciles puede ser redundante y costoso en tiempo. A futuro, integraría estrategias como Random Forest, Bagging o Boosting, junto con manejo de pesos de clase y técnicas de balanceo, para mejorar la robustez y la generalización del modelo.