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

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

Documentación de Arquitectura de Software y Reflexión

Portafolio Implementación - Módulo 2

Profesor
Jorge Adolfo Ramírez Uresti

Estudiante

Ulises Jaramillo Portilla……. | A01798380

Fecha de entrega: 5 de septiembre del 2025

1. Introducción

En este proyecto se implementó desde cero un clasificador de árbol de decisión basado en el aprendido en clase ID3, al que se le asignó el nombre ***DecisionTreeID3Plus***, con el fin de resolver tareas de clasificación de forma interpretable y reproducible. Sin embargo, a diferencia de un ID3 básico, el modelo que se planteó en esta entrega, soporta variables categóricas y numéricas, permitiendo elegir el criterio de partición, es decir, la ganancia de información. Asimismo, expone hiperparámetros de control como profundidad máxima, tamaño mínimo de división y ganancia mínima, manejando valores faltantes o no vistos mediante retroceso a la clase mayoritaria. Finalmente, y apoyándome con herramientas de IA para poder entender el proceso esperado, calculé probabilidades con suavizado de Laplace y se aplicó poda por error reducido usando un conjunto de validación determinado

Por otra parte, se desarrolló una CLI en Python que limpia valores, detectando automáticamente la columna objetivo y fija una semilla para asegurar reproducibilidad. Para evaluar el algoritmo utilicé varios datasets de clasificación con características distintas, con el fin de evaluar completamente el comportamiento y eficiencia de mi algoritmo, con lo que se cubrieron escenarios categóricos y mixtos.

1.1. Objetivo

Implementar desde cero un clasificador tipo ID3, sin frameworks de aprendizaje automático, con soporte para atributos numéricos y categóricos; además, probar el algoritmo con un conjunto de datos y realizar algunas predicciones. 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 (70/30, seed=42), 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

Implementación propia de un árbol de decisión ID3, la cual no solo conserva la esencia del algoritmo original, sino que lo extiende para hacerlo más eficiente y aplicable a un mayor número de casos. Entre las mejoras principales está la capacidad de trabajar tanto con atributos numéricos como con atributos categóricos, lo cual amplía el alcance del modelo y permite aplicarlo a conjuntos de datos más variados y complejos. Además, la estructura de la implementación fue diseñada pensando en la flexibilidad, de modo que pueda adaptarse fácilmente a diferentes escenarios de clasificación y análisis de datos.

3.1. Descripción general

3.1.1. ¿Qué hace?

El modelo aprende un árbol de decisión capaz de particionar el espacio de atributos para predecir la clase de un ejemplo. En cada nodo, el algoritmo selecciona la división más informativa, utilizando como criterio la ganancia de información o la razón de ganancia, según se requiera. Para atributos numéricos, calcula un umbral óptimo que separa los datos; mientras que para atributos categóricos, genera ramas específicas para cada valor posible, incluyendo una rama dedicada a los valores faltantes. Durante la fase de inferencia, el modelo recorre el árbol hasta llegar a una predicción. Si en el camino encuentra valores faltantes o no vistos previamente, aplica una estrategia de retroceso, asignando la clase mayoritaria del nodo para mantener la robustez del resultado.

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

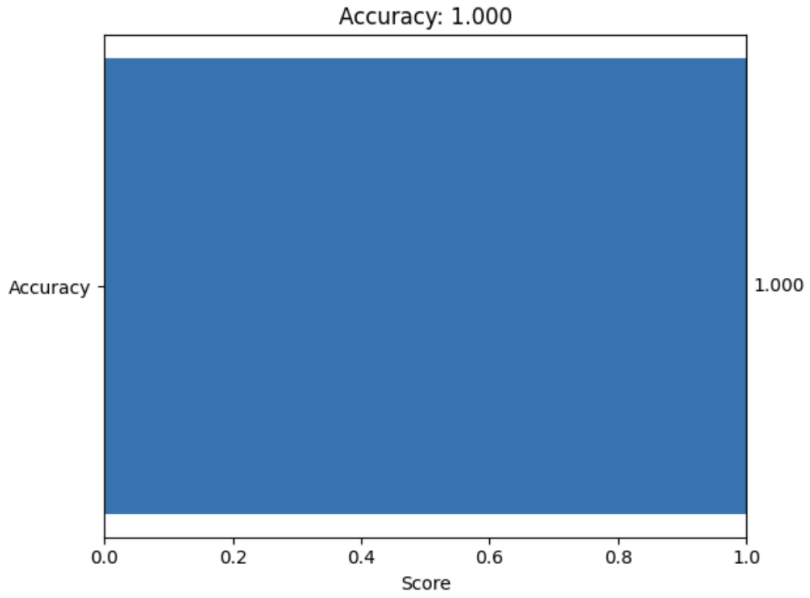
Para evaluar el desempeño del clasificador ID3 implementado a mano, se diseñaron dos etapas de pruebas. En la primera, se entrenó y evaluó el modelo utilizando el dataset completo, con el fin de observar el comportamiento en un escenario sin separación de datos. Posteriormente, se aplicó una partición en proporción 70/30 (train/test, con semilla fija de 42), lo que generó dos conjuntos en archivos separados (train.csv y test.csv). Esta división permitió probar de manera más realista la capacidad de generalización del algoritmo sobre datos no vistos, registrando métricas y resultados gráficos en ambas configuraciones.

4.1. (Dataset: Tennis)

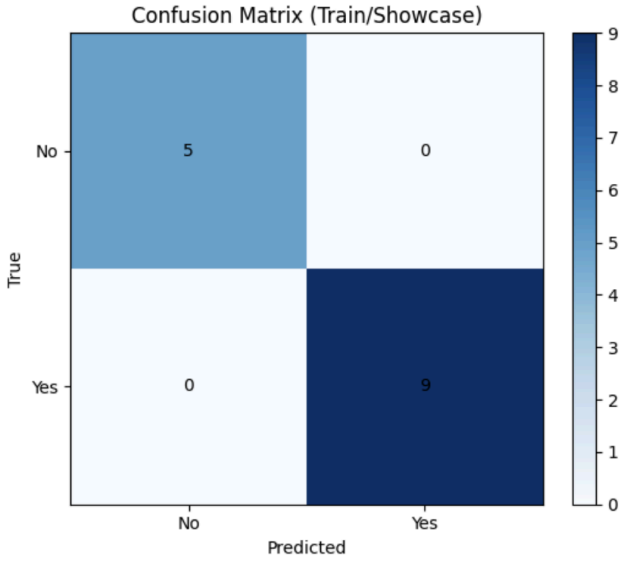
```
[run] Dataset: data/tennis.csv
[run] Target : Play

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

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


=== VALIDATION ===
** Training/testing split: 70/30, seed=42 **

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

4.1.1. Showcase (entrenamiento sobre todo el dataset)

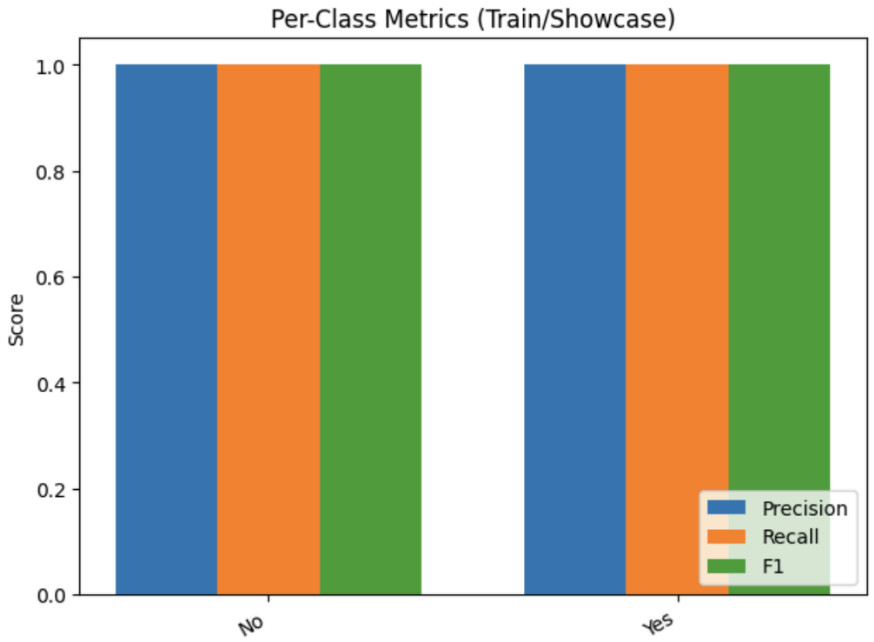
- **Accuracy entrenamiento:** 1.000



- **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.



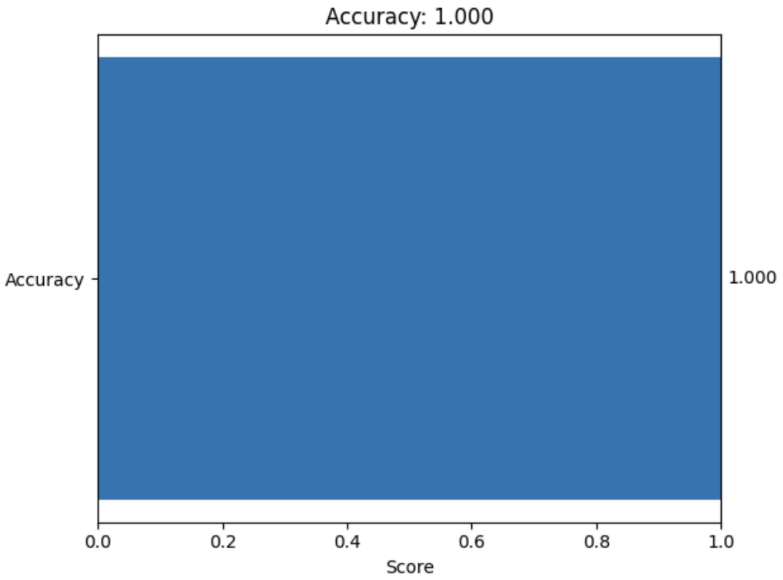
- **Árbol aprendido:** Raíz en Outlook, divisiones sucesivas en Windy y Humidity, perfectamente alineado con las reglas conocidas del dataset.

```
└── Outlook?
    ├── Overcast → Yes [counts={'Yes': 4}]
    ├── Rain
    │   ├── Windy?
    │   │   ├── Strong → No [counts={'No': 2}]
    │   │   └── Weak → Yes [counts={'Yes': 3}]
    └── Sunny
        └── Humidity?
            ├── High → No [counts={'No': 3}]
            └── Normal → Yes [counts={'Yes': 2}]
```

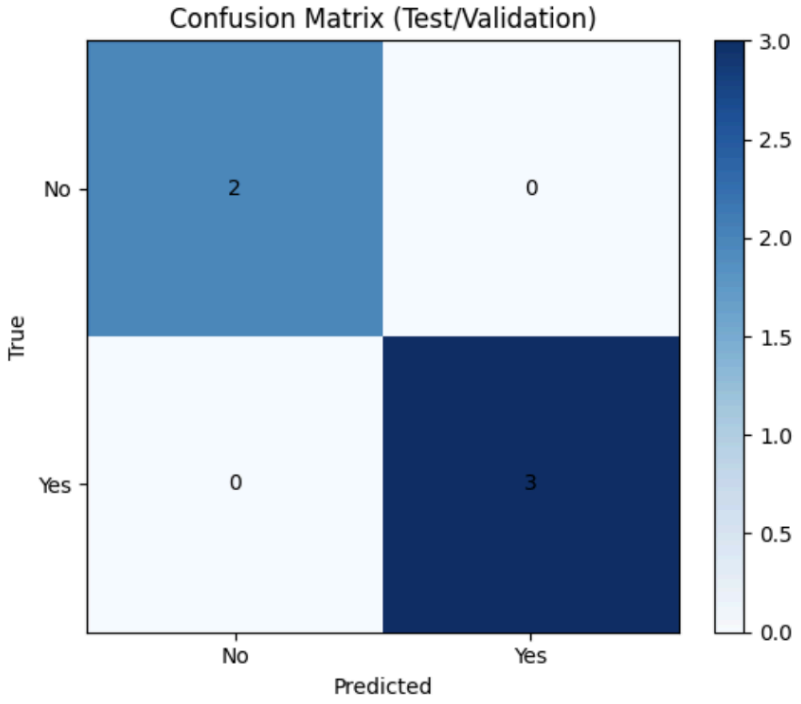
Interpretación: El dataset es pequeño y limpio, por lo que el algoritmo logra memorizar sin problema las reglas exactas, alcanzando precisión perfecta.

4.1.2 Validación (split 70/30, seed=42)

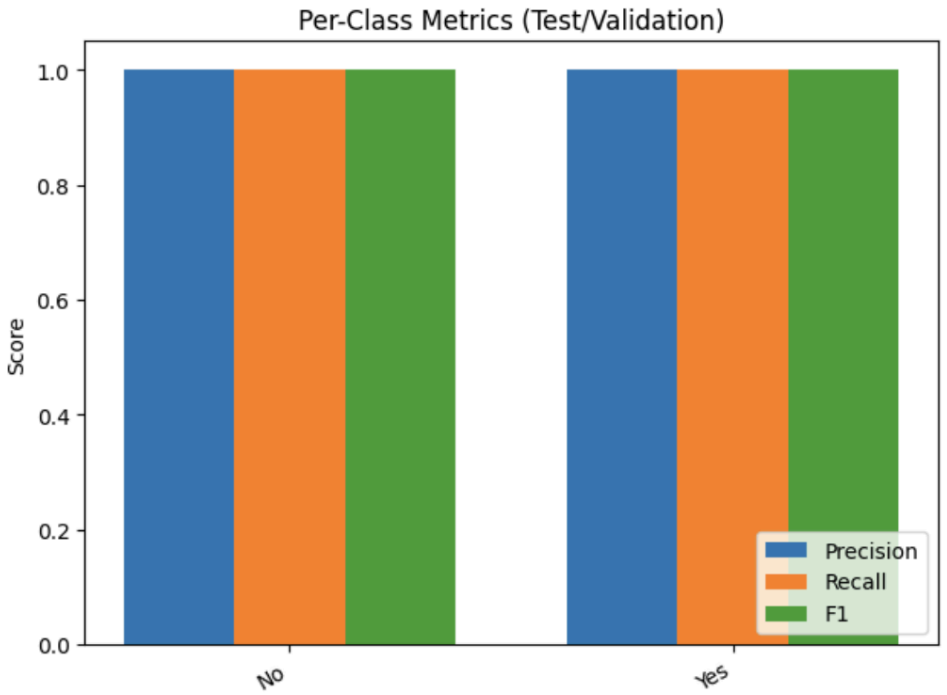
- **Accuracy test:** 1.000



- **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 → Yes [counts={'Yes': 2}]
    ├── Rain
    │   ├── Windy?
    │   │   ├── Strong → No [counts={'No': 1}]
    │   │   └── Weak → Yes [counts={'Yes': 2}]
    └── Sunny
        └── Humidity?
            ├── High → No [counts={'No': 2}]
            └── Normal → Yes [counts={'Yes': 2}]
```

Interpretación: El árbol generaliza bien en este caso, pues incluso con división aleatoria mantiene 100% de exactitud. Dado el tamaño reducido (10 instancias en train, 5 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.

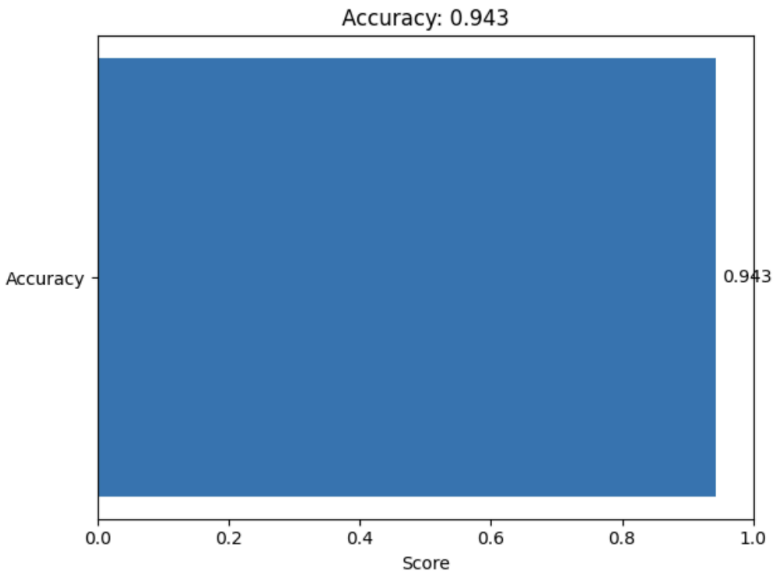
- **Limitación:** Al ser un dataset muy pequeño y con reglas lineales claras, no es una prueba dura de sobreajuste. Es decir, los resultados no se pueden extrapolar como evidencia de desempeño en datasets más grandes.

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

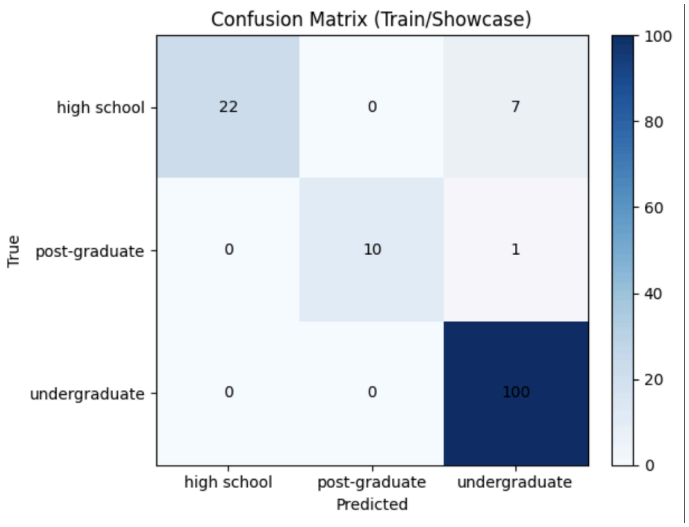
```
[run] Dataset: data/academic Stress level - maintainance 1.csv
[run] Target : Your Academic Stage

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

Training accuracy = 0.9429 (results saved in results/showcase)


=== VALIDATION ===
** Training/testing split: 70/30, seed=42 **

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

4.2.1. Showcase (entrenamiento sobre todo el dataset)

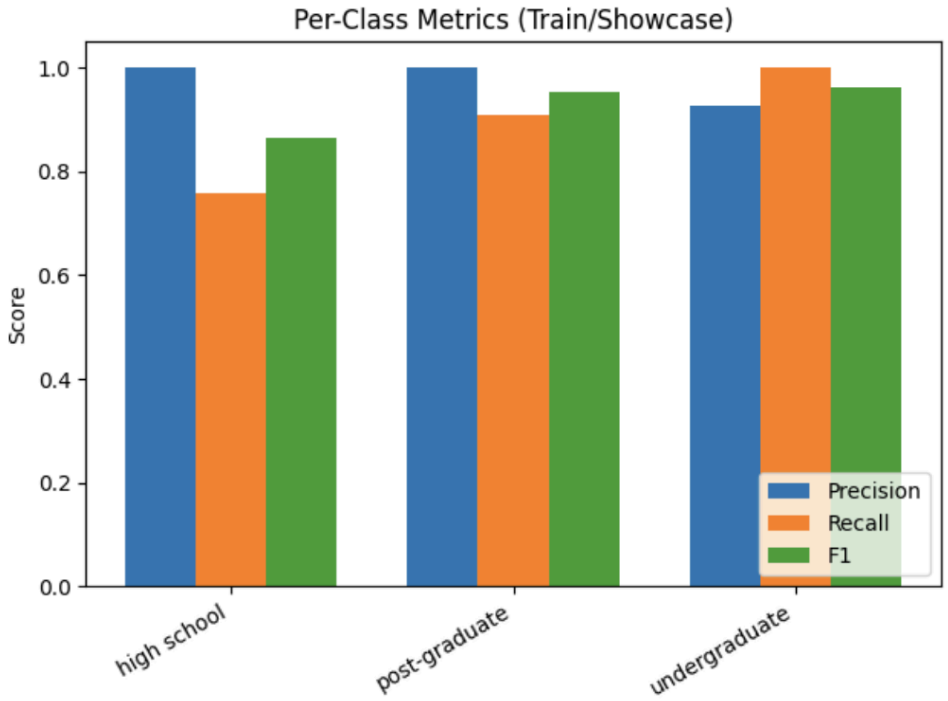
- **Accuracy entrenamiento:** 0.943



- **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.

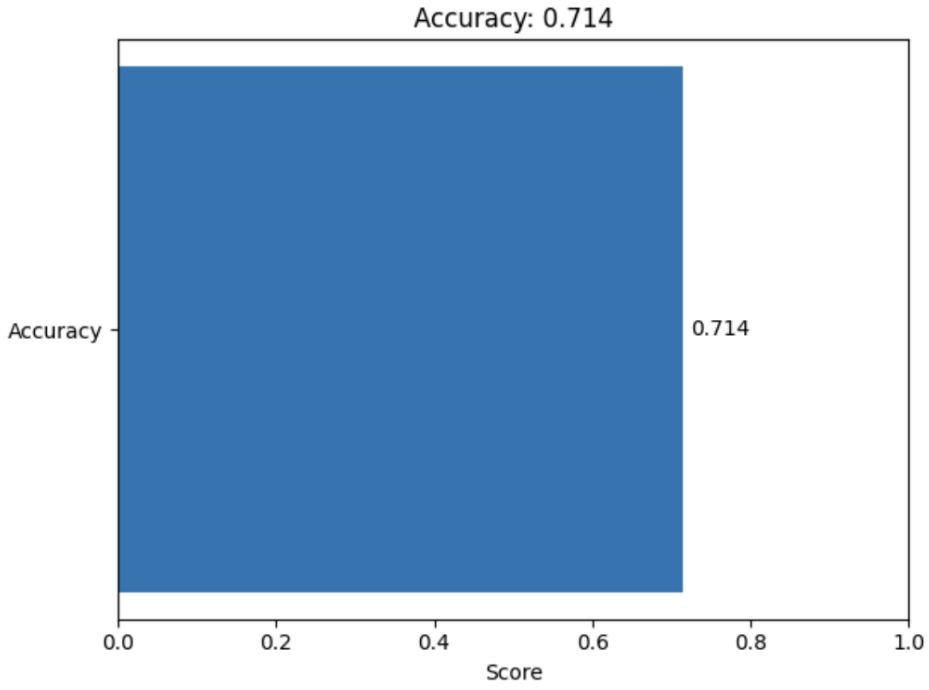
```
└── Academic pressure from your home <= 5 ?
    ├── LE
    │   ├── What would you rate the academic  competition in your student life <= 5 ?
    │   │   ├── LE
    │   │   │   ├── Timestamp?
    │   │   │   │   ├── 11/08/2025 12:15:00 → post-graduate [counts={'post-graduate': 1}]
    │   │   │   │   ├── 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}]
```

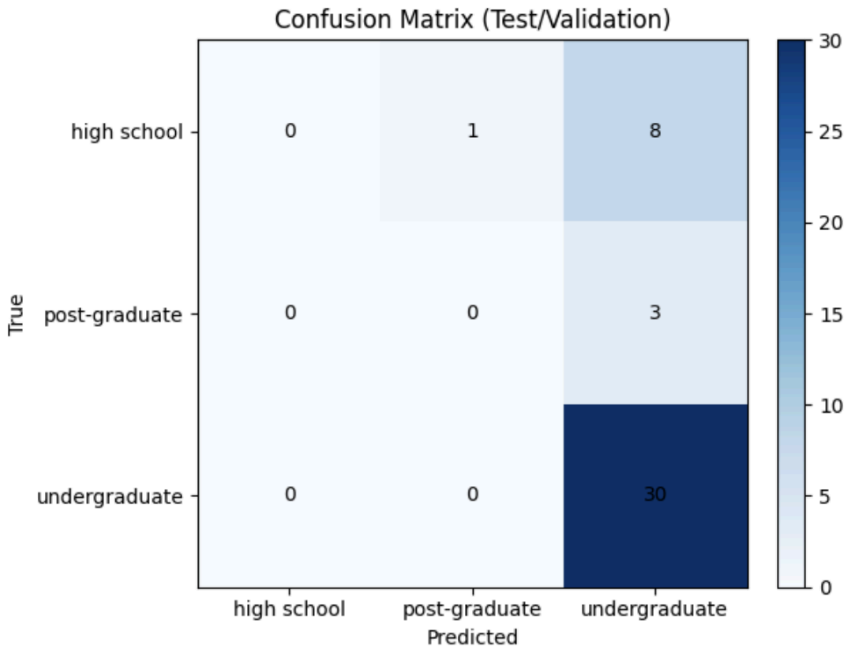
Interpretación: El modelo logra un buen ajuste (94.3%), aunque se observa que la clase High school es la más difícil de separar, por menor cantidad de datos o mayor solapamiento con Undergraduate.

4.2.2 Validación (split 70/30, seed=42)

- **Accuracy test:** 0.714

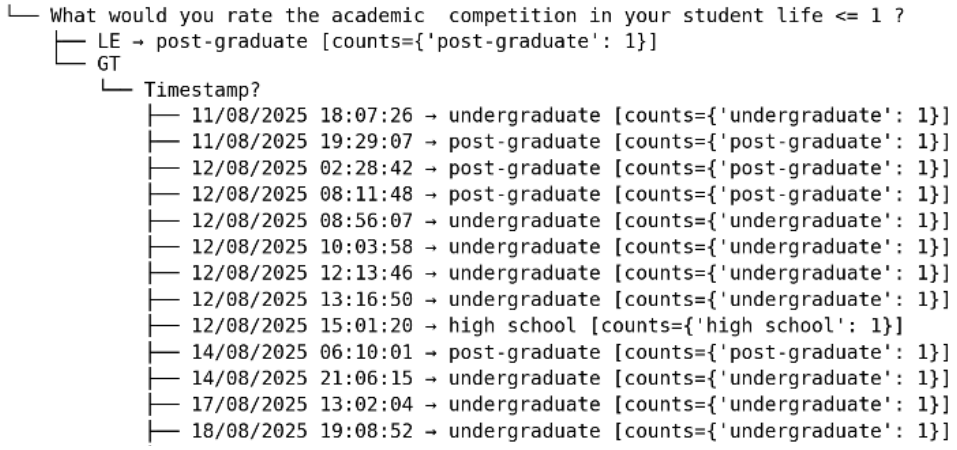


- **Matriz de confusión:**
    - High school: 0 correctos, 9 mal clasificados (8 como undergrad, 1 como post-grad).
    - Post-graduate: 0 correctos, 3 mal clasificados como undergrad.
    - Undergraduate: 30 correctos.



- **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.

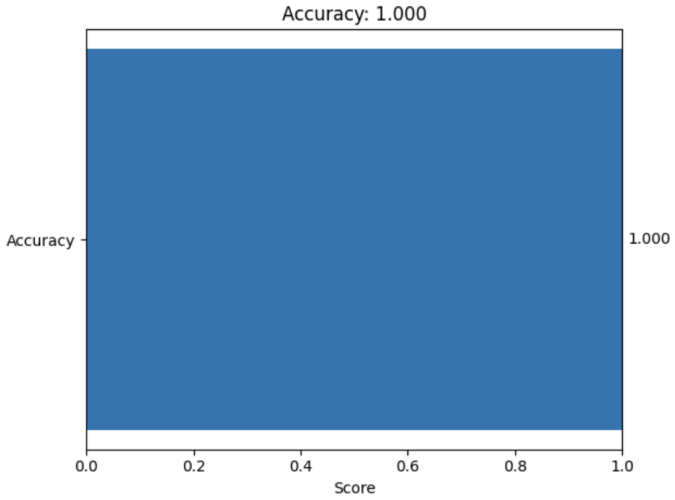
- **Limitación:** El algoritmo es sensible a desbalances de clase; el bajo desempeño en High school y Post-graduate refleja que se requiere poda más agresiva o técnicas de balanceo específicas.

4.3. (Dataset: mushrooms)

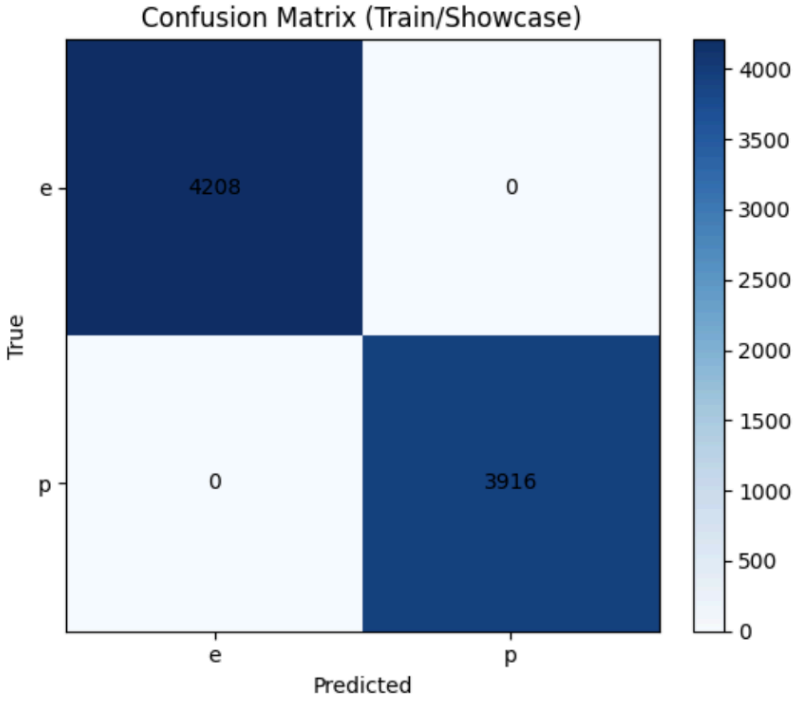
```
[run] Dataset: data/mushrooms.csv
[run] Target : class

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

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


=== VALIDATION ===
** Training/testing split: 70/30, seed=42 **

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

4.3.1. Showcase (entrenamiento sobre todo el dataset)

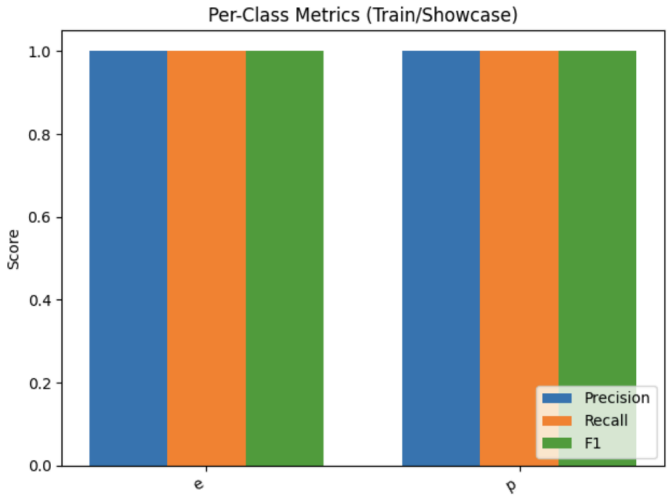
- **Accuracy entrenamiento:** 1.000



- **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.



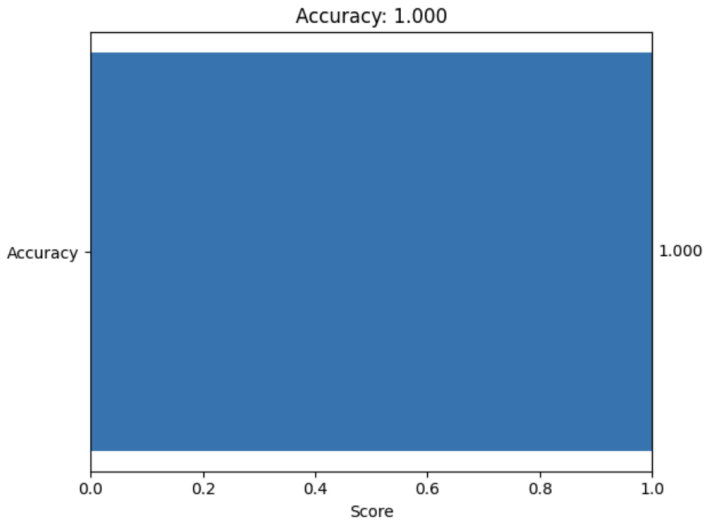
- **Árbol aprendido:** La raíz comienza en la característica odor, lo cual es lógico ya que el olor es un predictor muy fuerte en este dataset. Las ramas dividen de inmediato entre clases e/p de manera clara.

```
└── odor?
    ├── a → e [counts={'e': 400}]
    ├── c → p [counts={'p': 192}]
    ├── f → p [counts={'p': 2160}]
    ├── l → e [counts={'e': 400}]
    ├── m → p [counts={'p': 36}]
    ├── n
    │   ├── spore-print-color?
    │   │   ├── b → e [counts={'e': 48}]
    │   │   ├── h → e [counts={'e': 48}]
    │   │   ├── k → e [counts={'e': 1296}]
    │   │   ├── n → e [counts={'e': 1344}]
    │   │   ├── o → e [counts={'e': 48}]
    │   │   ├── r → p [counts={'p': 72}]
    │   │   ├── w
    │   │   │   ├── veil-color?
    │   │   │   │   ├── w
    │   │   │   │   │   ├── gill-size?
    │   │   │   │   │   │   ├── b → e [counts={'e': 528}]
    │   │   │   │   │   │   └── n
    │   │   │   │   │   │       └── gill-spacing?
    │   │   │   │   │   │           ├── c → p [counts={'p': 32}]
    │   │   │   │   │   │           └── w
    │   │   │   │   │   │               └── bruises?
    │   │   │   │   │   │                   ├── f → e [counts={'e': 48}]
    │   │   │   │   │   │                   └── t → p [counts={'p': 8}]
    │   │   │   │   └── y → p [counts={'p': 8}]
    │   │   └── y → e [counts={'e': 48}]
    ├── p → p [counts={'p': 256}]
    ├── s → p [counts={'p': 576}]
    └── y → p [counts={'p': 576}]
```

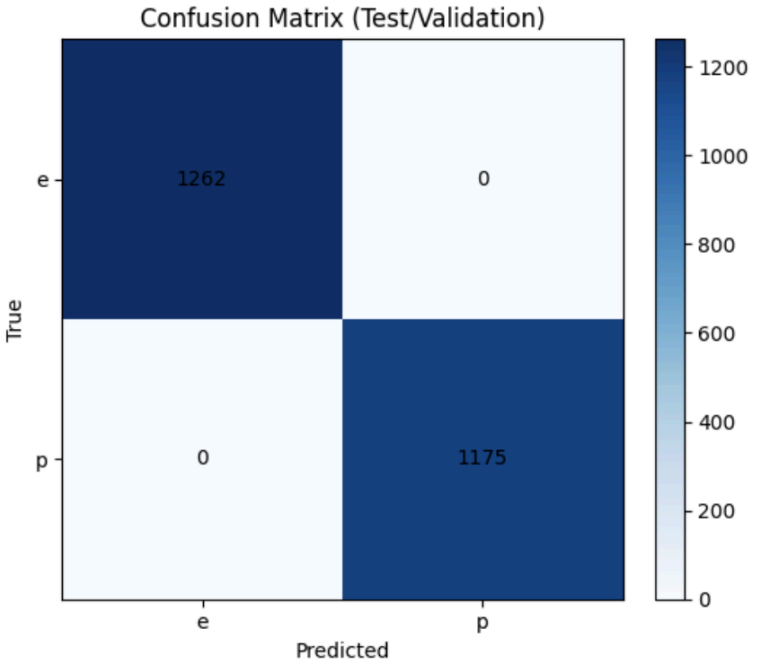
Interpretación: la importancia del olor confirma lo esperado: es un atributo determinante para saber si una seta es venenosa o comestible.

4.3.2 Validación (split 70/30, 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?
    ├── a → e [counts={'e': 268}]
    ├── c → p [counts={'p': 136}]
    ├── f → p [counts={'p': 1510}]
    ├── l → e [counts={'e': 273}]
    ├── m → p [counts={'p': 28}]
    ├── n
    │   ├── gill-size?
    │   │   ├── b
    │   │   │   ├── spore-print-color?
    │   │   │   │   ├── b → e [counts={'e': 36}]
    │   │   │   │   ├── k → e [counts={'e': 888}]
    │   │   │   │   ├── n → e [counts={'e': 911}]
    │   │   │   │   ├── o → e [counts={'e': 30}]
    │   │   │   │   ├── r → p [counts={'p': 45}]
    │   │   │   │   ├── w → e [counts={'e': 372}]
    │   │   │   │   └── y → e [counts={'e': 37}]
    │   │   └── n
    │   │       └── stalk-surface-above-ring?
    │   │           ├── f → e [counts={'e': 16}]
    │   │           ├── k → p [counts={'p': 23}]
    │   │           ├── s
    │   │           │   ├── bruises?
    │   │           │   │   ├── f → e [counts={'e': 115}]
    │   │           │   │   └── t → p [counts={'p': 6}]
    │   │           └── y → p [counts={'p': 6}]
    ├── p → p [counts={'p': 177}]
    ├── s → p [counts={'p': 406}]
    └── y → p [counts={'p': 404}]
```

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 de un dataset real y de mayor tamaño (8124 instancias), manteniendo exactitud perfecta tanto en entrenamiento como en validación. Esto valida que la implementación soporta datasets categóricos complejos.

- **Limitación:** La perfección en las métricas también señala que el dataset es muy "fácil" para este tipo de algoritmo (el atributo odor es casi determinista).

5. Conclusiones

En este proyecto implementé desde cero un árbol de decisión ID3 extendido (DecisionTreeID3Plus), con soporte para atributos numéricos y categóricos, criterios configurables y poda por error reducido. Al probarlo en datasets distintos, confirmé tanto su correcta implementación como sus limitaciones prácticas.

- **Tennis:** El modelo alcanzó 100% de exactitud en entrenamiento y validación, mostrando que la implementación es funcional y reproduce perfectamente reglas simples y limpias. Sin embargo, al ser un dataset muy pequeño, no representa un reto real para medir sobreajuste.

- **Academic Stress:** Los resultados fueron mixtos: en entrenamiento alcanzó un desempeño alto (94.3%), pero en validación bajó a 71.4%, con fuerte sesgo hacia la clase mayoritaria (Undergraduate). Esto evidenció que el algoritmo es sensible a desbalances de clase y que, en contextos reales y ruidosos, requiere técnicas adicionales como poda más agresiva o métodos de balanceo.

- **Mushrooms:** El algoritmo clasificó perfectamente (100% accuracy tanto en entrenamiento como en test). Esto valida que la implementación funciona en datasets más grandes y con muchas variables categóricas, aunque también deja claro que el dataset es "fácil" para ID3, pues cuenta con un atributo casi determinista.

5.1. Reflexión final

La experiencia de programar este clasificador desde cero me permitió entender en profundidad el funcionamiento de ID3 y sus extensiones. Demostré que el algoritmo es sólido y versátil, pero también confirmé sus debilidades: puede sobreajustar, es sensible a datos desbalanceados y depende mucho de la calidad de los atributos. A futuro, integraría técnicas como poda más sofisticada, manejo de pesos por clase o ensambles (Random Forest, Bagging) para mejorar robustez y generalización.