****

**Inteligencia Artificial Aplicada a la Ingeniería**

**Trabajo Práctico Individual**

Noviembre | 2023

Cuerpo docente:

Dr. Ing. Enrique Gabriel Baquela

Ing. José Ernesto Valentini

Autores:

Policrite, Franco

Índice

[Introducción 3](#_Toc150533975)

[Desarrollo 3](#_Toc150533976)

[Ejecución del script en Python 6](#_Toc150533977)

[Conclusión 8](#_Toc150533978)

# Introducción

En el ámbito de la inteligencia artificial, los árboles de decisión son una herramienta esencial que desempeña un papel importante en la toma de decisiones en diversos campos, incluida la ingeniería industrial. Estos árboles son modelos predictivos que utilizan estructuras similares a árboles para describir y clasificar decisiones y sus posibles resultados.

El poder de los árboles de decisión para manejar conjuntos de datos heterogéneos y variables múltiples es uno de sus atractivos principales. Además, el uso de los árboles de decisión es super frecuente en la inteligencia artificial debido a su facilidad de interpretación y explicación.

Son modelos predictivos que ayudan a las personas a tomar decisiones basándose en condiciones y reglas lógicas. Los puedes representar como un conjunto de decisiones organizadas en una estructura similar en un árbol.

Cada nodo interno en un árbol de decisión representa una prueba sobre un atributo, cada rama representa el resultado de la prueba, y cada hoja representa la etiqueta de clase o el valor de salida. En otras palabras, tomando decisiones a medida que avanzas de la raíz hacia las hojas, se basa en las características de los datos.

Dividir el conjunto de datos en subconjuntos más pequeños en función de las características elegidas es un paso en el proceso de construcción de un árbol de decisión. Hasta que se cumple un criterio, como la pureza de las hojas o la profundidad máxima del árbol, este proceso se repite.

Los árboles de decisión son populares porque pueden manejar conjuntos de datos complejos y son fáciles de entender. Además, se pueden usar para problemas de regresión y clasificación.

En este trabajo práctico, abordaremos el tema de árboles de decisión desde un simple ejemplo obtenido, en el cual mediante un Script que se le solicito a ChatGPT(IA, s.f.), se puede observar e interpretar de mejor forma como es que el algoritmo de árboles de decisión funciona.

# Desarrollo

En primera parte, se investigó que son los árboles de decisión dentro de la inteligencia artificial. Se visualizaron videos para poder comprender mejor este algoritmo, en el cual todos trabajaban sobre **Scikit-learn** (Fabian Pedregosa, Gael Varoquaux, Alexandre Gramfort and Vincent Michel, s.f.),siendo esta una biblioteca de machine learning que trabaja con el lenguaje de programación Python,tambien es una herramienta introductoria y simple para el análisis de datos y modelado predictivo. Scikit-learn proporciona algoritmos fáciles de usar para usuarios principiantes.

Se paso a la segunda parte,en la cual le pedimos a ChatGPT (IA, s.f.) un script de:

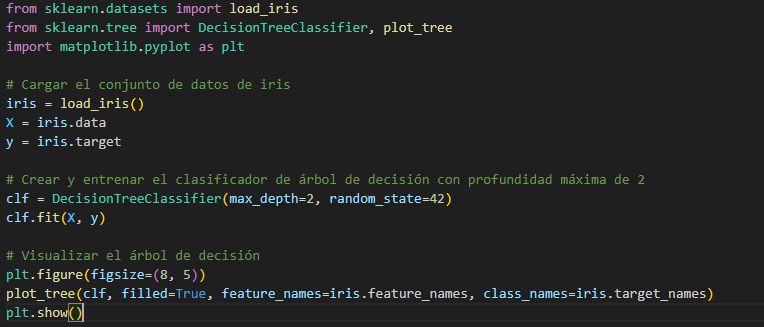
* <= 3 niveles (la raíz (nivel 0), los nodos de decisión en el segundo nivel (nivel 1), los nodos de hoja en el tercer nivel (nivel 2))
* <=3 nodos de decisión.
* Visualización Grafica.

ChatGPT (IA, s.f.), previo a esto nos solicitó tener instalado Python, se procedio a instalar Python.

Luego tambien nos solicitó instalar:

* scikit-learn (Fabian Pedregosa, Gael Varoquaux, Alexandre Gramfort and Vincent Michel, s.f.): pip install scikit-learn
* matplotlib (Se utiliza para crear visualizaciones del árbol de decisión):   
  pip install matplotlib

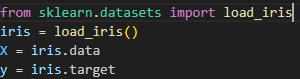
Habiendo instalado todo lo solicitado, ChatGPT (IA, s.f.) nos arrojó el siguiente Script:



El script proporcionado por ChatGPT (IA, s.f.) carga el conjunto de datos Flores Iris, entrena a un clasificador de árbol de decisión que tiene como máximo 3 niveles, para luego visualizarlo el árbol con `plot\_tree` de scikit-learn y `matplotlib`

Vamos a analizar profundamente que hace cada línea de código hecho que ha arrojado ChatGPT (IA, s.f.) :

* Carga el conjunto de datos de iris.



Este conjunto de datos es el utilizado para el entrenamiento del algoritmo.

Siendo Iris un conjunto de datos que contiene las características numéricas:

* + Longitud del pétalo (petal length): Longitud del pétalo de la flor en centímetros.
  + Ancho del pétalo (petal width): Ancho del pétalo de la flor en centímetros.
  + Longitud del sépalo (sepal length): Longitud del sépalo de la flor en centímetros.
  + Ancho del sépalo (sepal width): Ancho del sépalo de la flor en centímetros.

Cada fila contiene una “observación”, es decir, una especie, en el conjunto de datos , pueden ser tres clases o como en la materia dada llamamos datos del tipo categóricos:

1. Setosa
2. Versicolor
3. Virginica

Se carga la función load\_iris de Scikit-Learn (Fabian Pedregosa, Gael Varoquaux, Alexandre Gramfort and Vincent Michel, s.f.) para cargar el conjunto de datos de Flores Iris, recientemente mencionado. Las características numéricas son almacenadas dentro de la X y las etiquetas osea los datos categóricos, en las Y.

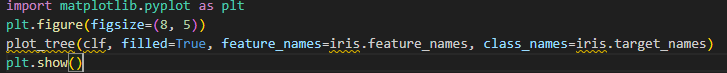
* Crear y entrenar el clasificador de árbol de decisión.



Importamos la clase `DecisionTreeClassifier` y creamos un clasificador de árbol de decisión con una profundidad de máxima de 2 (max\_depth=2) y una semilla aleatoria (random\_state=42) para que se generen datos pseudoaleatorios.

El clasificador que se genera ,es entrenado con el conjunto de datos ya mencionado, iris, es decir, el conjunto de datos con el que se **entrena es el mismo** con el que se predice.

* Grafica del árbol de decisión.



Se importa el módulo `matplotlib.pyplot` y se le asigna el nombre `plt`, luego se utiliza `plot\_tree` para poder ver el árbol de decisión en una pestaña de 8x5 (plt.figure(figsize=(8, 5)).

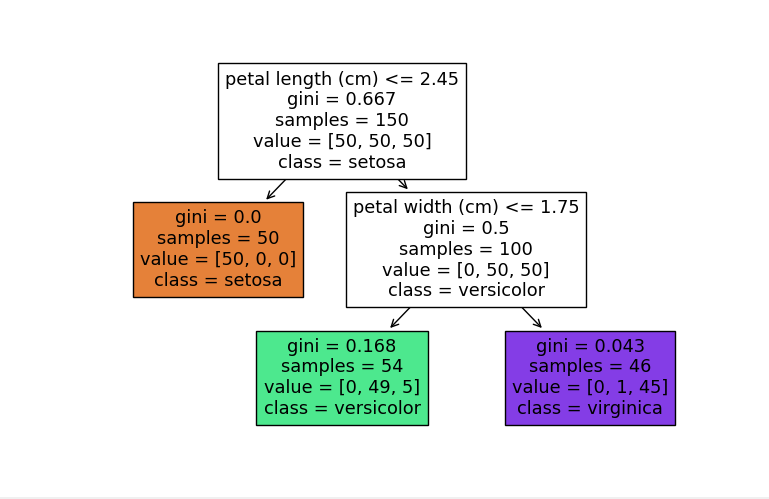
Luego se escribe las siguiente dos líneas:



* + **plot\_tree**: Es una función de la biblioteca **scikit-learn** que se utiliza para visualizar árboles de decisión.
  + **clf**: Es el clasificador de árbol de decisión que has entrenado previamente con tus datos.
  + **filled=True**: Este parámetro indica si se deben colorear los nodos del árbol. Cuando está configurado en **True**, los nodos se colorean según la clase mayoritaria de las muestras que llegan a ese nodo.
  + **feature\_names=iris.feature\_names**: Este parámetro proporciona los nombres de las características (atributos) del conjunto de datos, lo que mejora la interpretación de los nodos del árbol.
  + **class\_names=iris.target\_names**: Este parámetro proporciona los nombres de las clases o categorías a las que se están realizando las predicciones. En este caso, son las tres especies de flores del conjunto de datos Iris.
  + **plt.show()**: Es una función de la biblioteca **matplotlib** que se utiliza para mostrar la visualización del árbol de decisión generada por **plot\_tree**.

# Ejecución del script en Python

Se ejecuta el script en Python.



Explicación del contenido de los recuadros:

* Petal length (Longitud del pétalo): La elección de Petal Length como característica numérica la decisión la toma el algoritmo que la encuentra como una característica más discriminativa. Estas condiciones son determinadas automáticamente por el algoritmo de aprendizaje del árbol de decisión basándose en los datos de entrenamiento. Longitud del pétalo de la flor en centímetros
* Petal width (Ancho del pétalo): La elección de Petal width como característica numérica la decisión la toma el algoritmo que la encuentra como una característica más discriminativa. Estas condiciones son determinadas automáticamente por el algoritmo de aprendizaje del árbol de decisión basándose en los datos de entrenamiento. Ancho del pétalo de la flor en centímetros.
* Gini: La impureza de Gini es una medida de cuán impuros o mezclados son los datos de un conjunto. La impureza de Gini se mide en el contexto de un nodo en un árbol de decisión como la probabilidad de que una muestra seleccionada aleatoriamente sea incorrectamente clasificada. La impureza de Gini varía entre 0 (completamente puro) y 1 (completamente impuro o mezclado).
* Samples: cantidad de muestras (filas) que se encuentran en el nodo.Se puede observar en el primer recuadro que se cuentan con 150 datos , de los cuales 50 son clasificados como setosa y 100 como versicolor ( despues se categoriza si es versicolor o virgnica).
* Value: Es una lista que muestra la distribución de las clases en el nodo. Para un nodo hoja, esto indicaría la cantidad de muestras de cada clase en ese nodo. Para un nodo interno, muestra la distribución de clases antes de la división. Por ejemplo, en la hoja pintada de color verde, se puede visualizar que de los 54, 49 fueron bien categorizados ,5 no, eran virginica, esto hace que el GINI sea >=0. En cambio, por ejemplo, en la hoja pintada de naranja, da pureza total, debido a que las 50 muestras fueron bien categorizadas.

Explicación del funcionamiento del árbol de decisión:

Una vez generado el árbol de acuerdo el algoritmo es de fácil visualización e interpretación, se tienen 150 muestras, de las cuales, las que tienen un petal length <= 2,45 son categorizadas como Setosa, con un Gini de 0, es decir, son datos puros y se encuentran bien categorizados y predichos.

Luego, los datos que son >= 2,45 se los envía al otro nodo, en los cuales se pregunta si el petal width <= 1,75. Si asi lo es, se los clasifica como versicolor, esto lo predice de forma no tan pura, es decir, no lo predice de forma 100% correcta, ya que predijo 54 versicolor y terminaron siendo 49, por lo tanto, a mayor diferencia entre Samples y el valor de VALUE (en la class=versicolor) mayor va a ser el GINI, es decir, más impuro. Si el petal width>=1,75 se lo categoriza con la clase de virginica, con casi una pureza total, a excepción de un único dato.

# Conclusión

El trabajo practico individual de la materia de Inteligencia Artificial aplicada a la ingeniería, permitió abarcar nuevos conceptos, tales como la creación y visualización de un árbol de decisión utilizando un nuevo lenguaje de programación como Python. Esta herramienta mostro ser útil para la toma de decisiones, esto es valioso para un ingeniero industrial, donde la toma de decisiones se apoya en datos precisos y bien entendidos. La combinación de exploración de datos, modelado y visualización destaca la utilidad de estas técnicas para la toma de decisiones inteligentes y la optimización de procesos en entornos comerciales e industriales.