UNIVERSIDAD TECNOLOGICA NACIONAL

Trabajo Práctico Integrador

Árboles en Python

Alumnos

Jennifer Franco - Comisión 13 (jennyfranco31.jf@gmail.com)

Jonathan Franco - Comisión 13 (nahuelfranco7@icloud.com)

Tecnicatura Universitaria en Programación - Universidad Tecnológica Nacional.

Programación 1

Docente Titular

Ariel Enferrel

Docente Tutor

Franco Gonzalez

09 de Junio de 2025

Índice

1. Introducción

En esta investigación se aborda la implementación de un sistema de diagnóstico médico utilizando árboles de decisión binarios, aplicado a la detección de posibles casos de

COVID-19. A partir de una serie de preguntas con respuestas dicotómicas (sí/no), el árbol guía al usuario a través de un proceso lógico que finaliza en un diagnóstico tentativo, basado en síntomas frecuentes de esta enfermedad.

El objetivo es demostrar cómo los árboles de decisión pueden ser utilizados para tomar decisiones automatizadas en contextos reales. La implementación se realiza mediante la programación orientada a objetos, creando clases que representan los distintos nodos y ramas del árbol. Si bien se trata de una simplificación de los procesos diagnósticos médicos reales, este modelo permite comprender los fundamentos lógicos.

2. Marco Teórico

¿Qué es un árbol?

Un árbol es una estructura de datos compuesta por un conjunto finito de elementos llamados nodos, organizados de manera jerárquica. Su disposición recuerda a la de un árbol genealógico, existe un nodo principal o raíz, del cual se ramifican los demás nodos. Estas conexiones, conocidas como ramas, permiten enlazar los distintos elementos entre sí y representar relaciones de dependencia o jerarquía entre ellos (Universidad Politécnica del Instituto Politécnico Nacional, s.f.).

“Se define un árbol como una estructura de datos formada por varios objetos, (una cantidad finita), llamados nodos y varias líneas que conectan esos nodos, (la cantidad de líneas es finita), denominadas ramas” (Joyanes Aguilar & Zahonero Martínez, 2008).

Características de los árboles

Las más importantes son:

- ➢ Cualquier árbol no vacío tiene un nodo raíz que es único.
- ➢ Se dice que un nodo X es descendiente directo de un nodo Y, si el nodo Y apunta al nodo X, es muy común decir que X es hijo de Y.
- ➢ Decimos que un nodo X es antecesor directo de un nodo Y, cuando el nodo X apunta al nodo Y, es común decir que X es el padre de Y.
- ➢ Decimos que todos los nodos descendientes directos o hijos de un mismo nodo padre, son hermanos.
- ➢ Si nodo no tiene hijos o ramificaciones, es una hoja.
- ➢ Cualquier nodo que no es raíz, ni es hoja, es nodo interior
- ➢ El Grado de un nodo es el número de hijos de un nodo. El grado de un árbol es el grado máximo de cada uno de sus nodos.
- ➢ El nivel consta del número de líneas, enlaces o arcos que se deben recorrer para llegar a un nodo en particular. La raíz tiene nivel 1 por definición.

- La altura del árbol es el máximo nivel de cada uno de los nodos del árbol (Universidad Politécnica del Instituto Politécnico Nacional, s.f.).

¿Que es un árbol binario?

Un árbol binario es un tipo de estructura de datos de árbol donde cada nodo puede tener un máximo de dos nodos secundarios, un nodo secundario izquierdo y un nodo secundario derecho (W3Schools, s.f.).

Tipos de árboles binarios

- Árbol binario completo: Cada nodo tiene 0 o 2 hijos.
- Árbol binario lleno: Todos los niveles, excepto posiblemente el último, están completamente llenos, y todos los nodos están lo más a la izquierda posible.
- Árbol binario balanceado: La diferencia de altura entre subárboles izquierdo y derecho es mínima, lo que mejora la eficiencia de ciertas operaciones.

¿Qué es un árbol decisión?

Los árboles de decisión es una de las técnicas de aprendizaje inductivo supervisado no paramétrico, su principal uso es la predicción, especialmente dentro del ámbito de la inteligencia artificial, donde a partir de una base de datos se construyen diagramas de construcción lógica, muy similares a los sistemas de predicción basados en reglas, que sirven para representar y categorizar una serie de condiciones que ocurren en forma repetitiva para la solución de un problema (Ávila Torres, 2011).

Propiedades de los Árboles de decisión.

Una de las ventajas principales de los árboles de decisión es su capacidad para organizar los datos de manera estructurada y eficiente, debido a que los árboles son construidos a partir de la evaluación del primer nodo (raíz) y de acuerdo a su evaluación o valor tomado se va descendiendo en las ramas hasta llegar al final del camino (hojas del árbol) (Ávila Torres, 2011).

El conocimiento obtenido durante el proceso de aprendizaje inductivo se representa mediante un árbol. El nodo principal o raíz es el atributo a partir del cual se inicia el proceso de clasificación; los nodos internos corresponden a cada una de las preguntas acerca del atributo en particular del problema. Cada posible respuesta a los cuestionamientos se representa mediante un nodo hijo. Las ramas que salen de cada uno de estos nodos se encuentran etiquetadas con los posibles valores del atributo. Los nodos finales o nodos hoja corresponden a una decisión, la cual coincide con una de las variables clase del problema a resolver (Barrientos Martínez et al., 2009).

¿Cuáles son sus aplicaciones prácticas?

Los árboles de decisión se utilizan en múltiples campos, entre ellos:

- Medicina: Para apoyar diagnósticos clínicos a partir de síntomas y resultados de estudios.
- Marketing: Para segmentar clientes y predecir comportamientos de compra.
- Finanzas: Para evaluar riesgos crediticios y fraudes.

- ➢ Sistemas expertos: Para simular procesos de toma de decisiones humanas.

Impacto y relevancia en la medicina

Los árboles de decisión han demostrado ser una herramienta valiosa en el campo de la medicina, principalmente como soporte para diagnósticos clínicos y la toma de decisiones médicas. Su estructura lógica y fácil interpretación permite a los profesionales de la salud analizar síntomas y signos clínicos de forma ordenada para llegar a diagnósticos certeros o descartar enfermedades con rapidez.

Diagnósticos clínicos y apoyo en decisiones médicas

Los árboles de decisión permiten modelar procesos complejos de diagnóstico médico mediante una serie de preguntas categóricas, que conducen a conclusiones claras basadas en la evidencia recogida. Esto ayuda a estandarizar y agilizar la evaluación clínica, especialmente en entornos con alta demanda o falta de especialistas. Al representar visualmente la relación entre síntomas, resultados de exámenes y posibles diagnósticos, se favorece la comprensión y comunicación entre profesionales de salud y pacientes. Además, los árboles de decisión son útiles para identificar patrones de síntomas y signos que pueden no ser evidentes a simple vista, ayudando a detectar casos probables, atípicos o de gravedad variable, como en el diagnóstico de enfermedades infecciosas, enfermedades crónicas o condiciones complejas con múltiples factores.

Casos de éxito y estudios relevantes

Existen varios estudios que han confirmado la eficacia de los árboles de decisión para apoyar diagnósticos médicos:

- ➢ COVID-19: Durante la pandemia, se han desarrollado sistemas de diagnóstico basados en árboles de decisión que evalúan síntomas clave (fiebre, tos seca, dificultad respiratoria, pérdida del olfato) para clasificar el riesgo y recomendar la acción clínica inmediata, mejorando la detección temprana y optimizando el uso de recursos hospitalarios.
- ➢ Cáncer: En oncología, árboles de decisión han sido empleados para determinar la probabilidad de malignidad en tumores, ayudando a decidir entre procedimientos invasivos o monitoreo, contribuyendo a una atención más personalizada.
- ➢ Enfermedades cardiovasculares: Modelos basados en árboles han facilitado la predicción del riesgo cardiovascular mediante el análisis de variables como presión arterial, colesterol y hábitos de vida, permitiendo intervenciones preventivas oportunas.

¿Cuáles son los beneficios de su implementación?

Entre los principales beneficios de los árboles de decisión se encuentran:

- ➢ Claridad e interpretación visual: La estructura de los árboles de decisión facilita su comprensión, ya que se representan gráficamente mediante nodos y ramas que ilustran el proceso de toma de decisiones de forma lógica y ordenada.

- ➢ Manejo de distintos tipos de variables: Pueden trabajar tanto con datos categóricos como numéricos, lo que los hace adecuados para una amplia variedad de problemas.
- ➢ Versatilidad: Son aplicables tanto a tareas de clasificación como de regresión, lo que amplía su utilidad en diferentes contextos de análisis y predicción.
- ➢ Poca necesidad de preprocesamiento: A diferencia de otros algoritmos, los árboles de decisión no requieren normalización ni estandarización de los datos.
- ➢ Identificación de variables relevantes: Permiten visualizar claramente qué atributos tienen mayor influencia en el resultado final, lo que aporta valor en la interpretación de datos.

¿Cuáles son las limitaciones de su implementación?

Algunas de las limitaciones de la implementación de árboles de decisión son:

- ➢ Sobreajuste (overfitting): Es común que los árboles se ajusten demasiado a los datos de entrenamiento, lo que disminuye su capacidad para generalizar a nuevos datos.
- ➢ Sensibilidad a cambios en los datos: Pequeñas modificaciones en la base de datos pueden generar árboles muy distintos, lo que afecta la estabilidad del modelo.
- ➢ Baja precisión en casos complejos: En problemas con múltiples relaciones entre variables, un árbol de decisión simple puede no ser suficiente para alcanzar un nivel óptimo de precisión.
- ➢ Crecimiento excesivo: Si no se controla la profundidad del árbol, puede volverse muy grande, lo que impacta negativamente en el rendimiento y aumenta el uso de recursos computacionales.

3. Caso Práctico

Implementamos un sistema interactivo en Python que pregunta al usuario sobre síntomas relacionados con COVID-19 y, según las respuestas, deriva a un diagnóstico tentativo. Se usó programación orientada a objetos, definiendo clases Nodo y ArbolDeDecision.

Implementación del sistema

Se definieron dos clases:

1. Nodo: Representa cada elemento del árbol de decisión, ya sea una pregunta o un diagnóstico final. Cada nodo almacena:

    a. texto: la pregunta o el diagnóstico a mostrar.
    b. hijo_si y hijo_no: referencias a los nodos hijos según la respuesta del usuario ("sí" o "no").

2. Arbol Decision: Construye y gestiona el árbol binario de decisión, además de implementar el método interactivo de diagnóstico.

El árbol está estructurado de manera jerárquica, debido a que comienza con la pregunta inicial acerca de la fiebre. A partir de allí, el árbol ramifica hacia preguntas más específicas

como pérdida del olfato, tos seca, dificultad para respirar y dolor de garganta. Cada camino conduce finalmente a un nodo hoja que contiene un diagnóstico tentativo (compatible con COVID-19 moderado/severo, posible COVID-19, poco probable, etc.).

Además, se incorporó la biblioteca Colorama para dar color a los mensajes en consola, facilitando la identificación de los diagnósticos finales (rojo para casos graves, amarillo para casos moderados y verde para casos leves o poco probables).

A continuación se muestra el código completo:

```
# Importar la librería Colorama para darle color a la consola
from colorama import init, Fore, Style

#Definimos clase Nodo: representa cada pregunta o diagnóstico final del árbol de decisión
class Nodo:
    def __init__(self, texto, hijo_si=None, hijo_no=None):
        self.texto = texto          #texto de la pregunta diagnostico
        self.hijo_si = hijo_si      #rama del arbol en caso de la respuesta "si"
        self.hijo_no = hijo_no      #rama del arbol en caso de la respuesta "no"

#Definimos la clase ArbolDeDecision: implementa el arbol binario de decision
class ArbolDeDecision:
    def __init__(self):

        #Diagnósticos finales (nodos hoja)
        diag_covid = Nodo(f'{Fore.RED}Diagnóstico:{Style.RESET_ALL} ' + 'Caso compatible
con COVID-19 moderado o severo. Consulte al médico.')
        diag_posible_covid = Nodo(f'{Fore.YELLOW}Diagnóstico:{Style.RESET_ALL} ' +
'Posible caso de COVID-19. Controle síntomas.')
        diag_covid_poco_probable =Nodo(f'{Fore.YELLOW}Diagnóstico:{Style.RESET_ALL} '
+ 'Poco probable que sea COVID-19.')
        diag_no_concluyente = Nodo(f'{Fore.YELLOW}Diagnóstico:{Style.RESET_ALL} ' +
'Síntomas no concluyentes. Controle temperatura y síntomas.')
        diag_posible_resfriado = Nodo(f'{Fore.GREEN}Diagnóstico:{Style.RESET_ALL} ' +
'Posible resfriado común. Siga controlando.')

        # Preguntas (nodos intermedios)
        nodo_tos = Nodo('¿Tiene tos seca?', diag_posible_covid, diag_no_concluyente)
        nodo_respiratorio = Nodo('¿Tiene dificultad para respirar?', diag_covid,
diag_posible_covid)
        nodo_olfato = Nodo('¿Perdió el olfato?', nodo_respiratorio, nodo_tos)
        nodo_garganta = Nodo('¿Tiene dolor de garganta?', diag_posible_resfriado,
diag_covid_poco_probable)

        nodo_fiebre = Nodo('¿Tiene fiebre?', nodo_olfato, nodo_garganta)

        # Nodo Raiz del árbol
        self.raiz = nodo_fiebre
```

```
    def iniciar_diag (self):
        nodo_actual = self.raiz #Comienza desde la raiz del arbol
        #Mientras el Nodo tenga hijos continuara haciendo preguntas
        while nodo_actual.hijo_si or nodo_actual.hijo_no:
            respuesta = input(nodo_actual.texto +  " (si/no): ").strip().lower()
            #Validar las respuestas del usuario
            while respuesta not in ("si", "no"):
                print("Por favor, responda 'sí' o 'no'.")
                respuesta = input(nodo_actual.texto + " (si/no): ").strip().lower()

            #Avanza al siguiente Nodo segun la respuesta
            if respuesta == "si":
                nodo_actual = nodo_actual.hijo_si
            else:
                nodo_actual = nodo_actual.hijo_no

        #Imprime el diagnóstico final (Nodo hoja)
        print("\n" + nodo_actual.texto)

#Programa principal
print(f"{Fore.CYAN}Sistema Simplificado de Diagnóstico COVID-19\n{Style.RESET_ALL}")
arbol = ArbolDeDecision()
arbol.iniciar_diag()
print(f"\n{Fore.GREEN}Gracias por utilizar el sistema. ¡Cuídese!{Style.RESET_ALL}")
```

4. Metodología Utilizada

Para llevar a cabo el trabajo práctico integrador se siguieron los siguientes pasos:

1. **Selección del tema:** Se eligió trabajar con la estructura de datos de árboles binarios, específicamente en un árbol de decisión binario aplicado a un sistema de diagnóstico médico.

2. **Investigación teórica:** Se consultaron fuentes como el artículo de Barrientos Martínez et al. (2009), W3Schools (s.f.), Joyanes Aguilar y Zahonero Martínez (2008), y otros recursos especializados en estructuras de datos y árboles de decisión para comprender los conceptos de árboles binarios, nodos y su uso en el diagnóstico médico.

3. **Diseño del árbol de decisión:** Se definió la estructura lógica del árbol como un árbol de decisión binario, donde cada nodo representa una pregunta que puede tener dos respuestas posibles (sí/no). Según la respuesta, el árbol guía al usuario hacia la siguiente pregunta o hacia un diagnóstico tentativo.

4. **Implementación del código:** Se desarrollaron las clases Nodo y ArbolDeDecision en Python, aplicando principios de programación orientada a objetos.

5. **Pruebas de funcionalidad:** Se realizaron ejecuciones del programa para verificar que el flujo de preguntas y respuestas funcione correctamente y que los diagnósticos finales sean coherentes.

6. **Documentación y README:** Se elaboró un documento con el marco teórico, la metodología, el código y las conclusiones. Además, se creó un README detallado para el repositorio de Git.

7. **Video tutorial:** Se planificó la grabación de un video explicativo mostrando el uso del programa y reflexionando sobre lo aprendido.

Trabajo colaborativo

El trabajo se realizó de forma conjunta entre los dos integrantes del grupo, quienes participaron activamente en todas las etapas, desde la investigación hasta el desarrollo del código y la documentación final. Compartiendo y analizando conjuntamente los resultados obtenidos, favoreciendo el aprendizaje colaborativo.

Herramientas utilizadas

Se utilizó el lenguaje de programación Python, el editor de código Visual Studio Code, y se gestionó el repositorio de código en GitHub para control de versiones y trabajo colaborativo.

5. Resultados Obtenidos

Durante la implementación del proyecto se logró desarrollar un sistema funcional de diagnóstico médico simplificado, basado en árboles de decisión binarios. El sistema permitió:

- ➢ **Guiar al usuario** a través de una serie de preguntas simples de “sí” o “no” relacionadas con síntomas comunes de COVID-19.

- ➢ **Emitir un diagnóstico** según las combinaciones de síntomas ingresadas, simulando un proceso básico de triaje.

- ➢ **Validar la lógica del árbol**, confirmando que cada nodo dirige correctamente a su siguiente rama según la respuesta.

- ➢ **Comprobar la utilidad de la estructura jerárquica** para representar decisiones condicionales encadenadas de forma clara.

- ➢ **Ejecutar con éxito en consola**, mostrando mensajes con estilos de color usando la librería colorama

El árbol fue capaz de emitir diferentes diagnósticos como “posible COVID-19”, “síntomas no concluyentes” o “posible resfriado”, demostrando su funcionalidad en distintos escenarios.

6. Conclusiones

Este trabajo nos permitió profundizar en la implementación de árboles de decisión binarios y comprender su uso en la programación de sistemas de ayuda a la toma de decisiones. Aplicamos conceptos de programación orientada a objetos, como clases y métodos, y reforzamos el uso de atributos y estructuras jerárquicas. También aprendimos a validar entradas del usuario y organizar el código de forma clara y modular en Python.

Si bien el programa desarrollado cumple con su objetivo básico, se identificaron posibles mejoras que permitirían escalar el proyecto a una solución más robusta y flexible. Entre ellas:

- ➢ Incorporar datos personalizados como nombre, edad o antecedentes médicos, para hacer el diagnóstico más preciso y adaptado al paciente.
- ➢ Permitir que el sistema analice múltiples síntomas en paralelo, en lugar de seguir una lógica estrictamente secuencial.
- ➢ Incluir niveles de gravedad o riesgo estimado basados en combinaciones de síntomas y factores de riesgo.
- ➢ Utilizar técnicas de aprendizaje automático con árboles de decisión entrenados sobre conjuntos de datos reales, para mejorar la precisión diagnóstica.

Para ello, se podrían incorporar las siguientes librerías especializadas:

- ➢ **Pandas** y **NumPy**: para el manejo de datos tabulares de síntomas, pacientes y diagnósticos.
- ➢ **DecisionTreeClassifier** de *Scikit-learn*: para construir y entrenar modelos predictivos a partir de bases de datos médicas.
- ➢ **Confusion Matrix**: para evaluar la precisión del sistema con métricas cuantitativas (exactitud, sensibilidad, especificidad).
- ➢ **Pydotplus** y **Tree**: para representar visualmente los árboles generados automáticamente por el algoritmo de clasificación.
- ➢ **Seaborn** y **Matplotlib**: para generar gráficas sobre la distribución de síntomas o el rendimiento del modelo.
- ➢ **StringIO**: para manipulación eficiente de texto en memoria.

Estas mejoras permitirán evolucionar el sistema desde una herramienta didáctica a una aplicación inteligente, útil en contextos reales, con capacidad de aprendizaje y análisis avanzado.

7. Bibliografía

Barrientos Martínez, R. E., Cruz Ramírez, N., Acosta Mesa, H. G., Rabatte Suárez, I., Gogeascoechea Trejo, M. del C., Pavón León, P., & Blázquez Morales, S. L. (2009). Árboles de decisión como herramienta en el diagnóstico médico. *Revista Médica de la Universidad Veracruzana*, *9*(2). http://www.soporte.uv.mx/rm/num_anteriores/revmedica_vol9_num2/articulos/arboles.pdf

Joyanes Aguilar, L., & Zahonero Martínez, I. (2008). *Estructura de datos en Java*. Madrid, España: McGraw Hill Interamericana.

Russell, S., & Norvig, P. (2003). *Artificial Intelligence: A Modern Approach* (2.ª ed.). Upper Saddle River, NJ: Prentice Hall/Pearson Education.

Universidad Politécnica del Instituto Politécnico Nacional. (s.f.). *Árboles, explicación y ejemplos*. https://www.sites.upiicsa.ipn.mx/estudiantes/academia_de_informatica/estructura_y_rd/docs/u3/%C3%81rboles,%20explicaci%C3%B3n%20y%20ejemplos.pdf

W3Schools. (s.f.). *Data Structures and Algorithms - Binary Trees*. https://www.w3schools.com/dsa/dsa_data_binarytrees.php

García, C. A., & Mendoza, J. E. (2013). Árboles de decisión en la predicción del rendimiento académico. *Ingeniería Industrial*, *34*(3), 232–243. https://www.redalyc.org/pdf/849/84922625018.pdf

Téllez Treviño, M. A., & Díaz Martínez, R. (2023). Árboles de decisión para la clasificación de datos. *Revista Mexicana de Ciencias Informáticas*, *12*(1), 1–10. https://www.redalyc.org/journal/6738/673870841002/html/

8. Anexos

Representación gráfica del árbol de decisión

A continuación, se presenta un diagrama elaborado en Canva que representa el árbol de decisión implementado en el proyecto. Este esquema facilita la comprensión de la estructura lógica del árbol, incluyendo preguntas intermedias y diagnósticos finales, y permite visualizar cómo se deriva un diagnóstico tentativo de COVID-19 a partir de las respuestas del usuario.

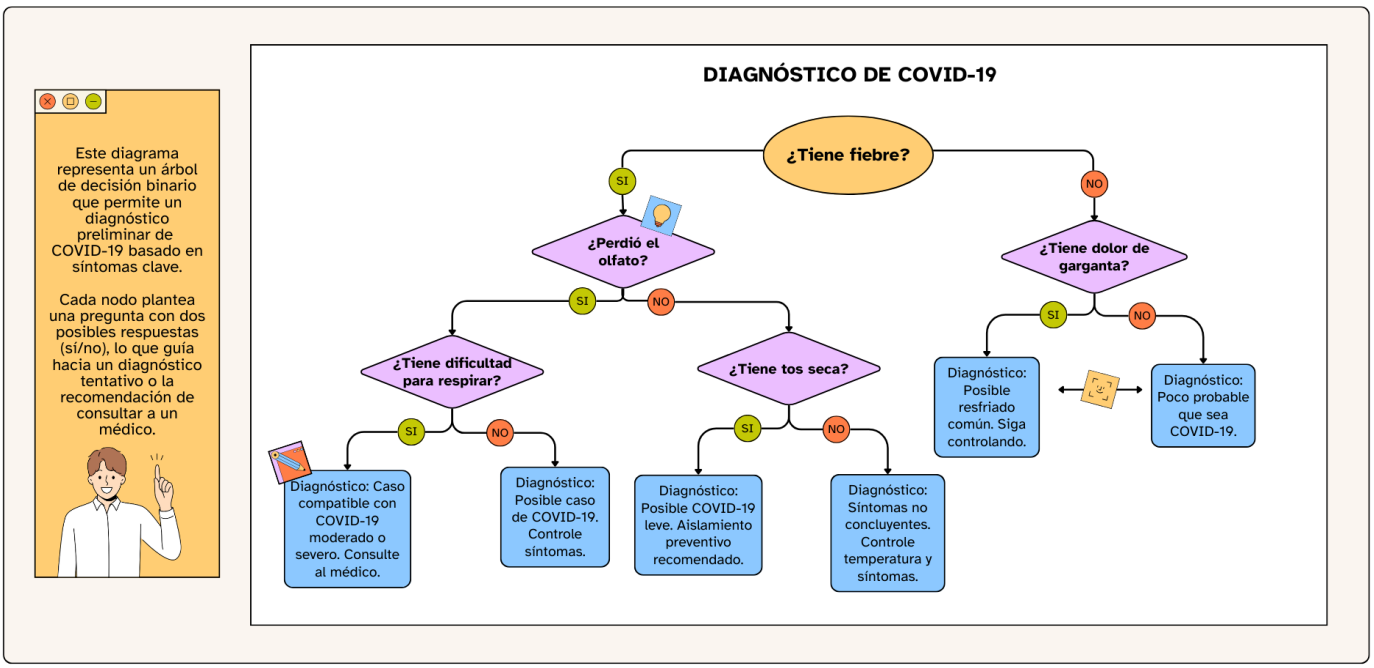

Figura 1. Esquema del árbol de decisión para el diagnóstico tentativo de COVID-19. Elaboración propia en Canva.

Capturas de pantalla del código funcionando en consola

```
TERMINAL
PS C:\Users\Jennifer\Desktop\UTN-programacIon> & C:/Users/Jennifer/AppData/L
ogramacion/arboles.py
Sistema Simplificado de Diagnóstico COVID-19

¿Tiene fiebre? (si/no): si
¿Perdió el olfato? (si/no): noo
Por favor, responda 'sí' o 'no'.
¿Perdió el olfato? (si/no): NO
¿Tiene tos seca? (si/no): si

Diagnóstico: Posible caso de COVID-19. Controle síntomas.

Gracias por utilizar el sistema. ¡Cuídese!
```

```
PS C:\Users\Jennifer\Desktop\UTN-programacIon> & C:/Users/Jennifer/AppData/L
ogramacion/arboles.py
Sistema Simplificado de Diagnóstico COVID-19

¿Tiene fiebre? (si/no): nO
¿Tiene dolor de garganta? (si/no): SI

Diagnóstico: Posible resfriado común. Siga controlando.

Gracias por utilizar el sistema. ¡Cuídese!
```

```
PS C:\Users\Jennifer\Desktop\UTN-programacIon> & C:/Users/Jennifer/AppData/
ogramacion/arboles.py
Sistema Simplificado de Diagnóstico COVID-19

¿Tiene fiebre? (si/no): NO
¿Tiene dolor de garganta? (si/no): No

Diagnóstico: Poco probable que sea COVID-19.

Gracias por utilizar el sistema. ¡Cuídese!
```

```
PS C:\Users\Jennifer\Desktop\UTN-programacIon> & C:/Users/Jennifer/AppData/Local/Prog
ogramacion/arboles.py
Sistema Simplificado de Diagnóstico COVID-19

¿Tiene fiebre? (si/no): si
¿Perdió el olfato? (si/no): si
¿Tiene dificultad para respirar? (si/no): si

Diagnóstico: Caso compatible con COVID-19 moderado o severo. Consulte al médico.

Gracias por utilizar el sistema. ¡Cuídese!
```

Video explicativo

https://youtu.be/5WyVNpVQBYQ

Repositorio GitHub

https://github.com/JenniferFranco/tp-integrador-programacion.git