



Manual de usuario

Akinator de Hollow Knight



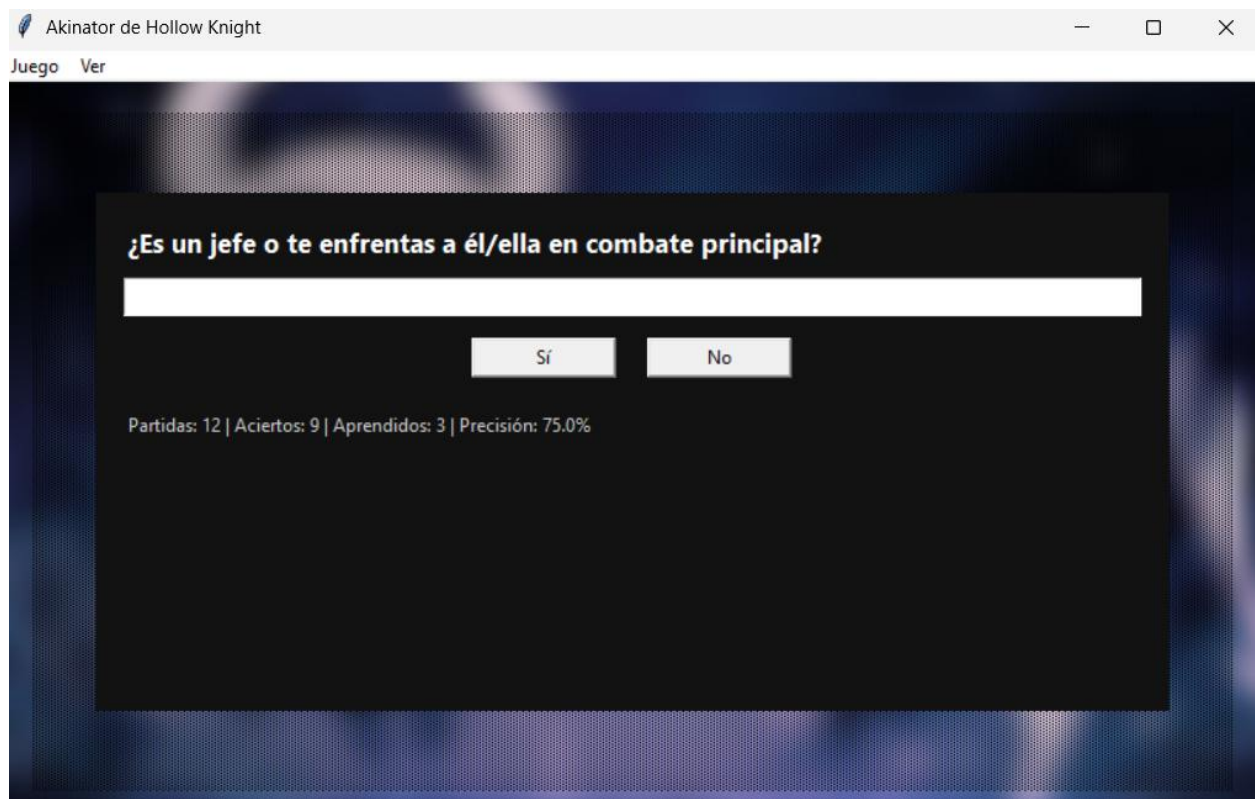
Gael Alexandro Silva Huacuja

Centro de Enseñanza Técnica Industrial (CETI) Plantel Colomos

24 de octubre de 2025

1. Introducción

El presente manual describe el funcionamiento y la teoría detrás del programa “Akinator de Hollow Knight”, una aplicación interactiva que utiliza un árbol de decisiones para adivinar personajes del videojuego Hollow Knight, mediante preguntas de tipo sí/no. Además, el sistema cuenta con la capacidad de aprendizaje, lo que le permite mejorar su base de conocimiento cada vez que no logra adivinar correctamente un personaje nuevo.



2. Funcionamiento del sistema

El sistema está implementado en Python utilizando la biblioteca Tkinter para la interfaz gráfica y Pillow para el manejo de la imagen de fondo difuminada. El programa basa su funcionamiento en un conjunto de reglas binarias almacenadas en un archivo JSON, que se actualiza dinámicamente cuando el usuario enseña un nuevo personaje.

Componentes principales:

- Árbol de conocimiento: estructura jerárquica de preguntas y respuestas (sí/no).

- Interfaz gráfica: permite la interacción visual con el usuario.
- Módulo de aprendizaje: agrega nuevos nodos (reglas y casos) al árbol.

3. Teoría: reglas, casos y encadenamiento hacia adelante

El sistema utiliza un enfoque basado en el razonamiento mediante reglas, donde cada nodo del árbol representa una regla o un caso particular. Este método forma parte de la inteligencia artificial simbólica, en la cual el conocimiento se representa explícitamente a través de proposiciones y condiciones. Reglas: cada pregunta del Akinator representa una regla del tipo “Si condición entonces resultado”.

Por ejemplo: “Si el personaje usa aguja e hilo, entonces podría ser Hornet.”

Casos: cada hoja del árbol representa un caso concreto o hipótesis final (por ejemplo, “Es Hornet”).

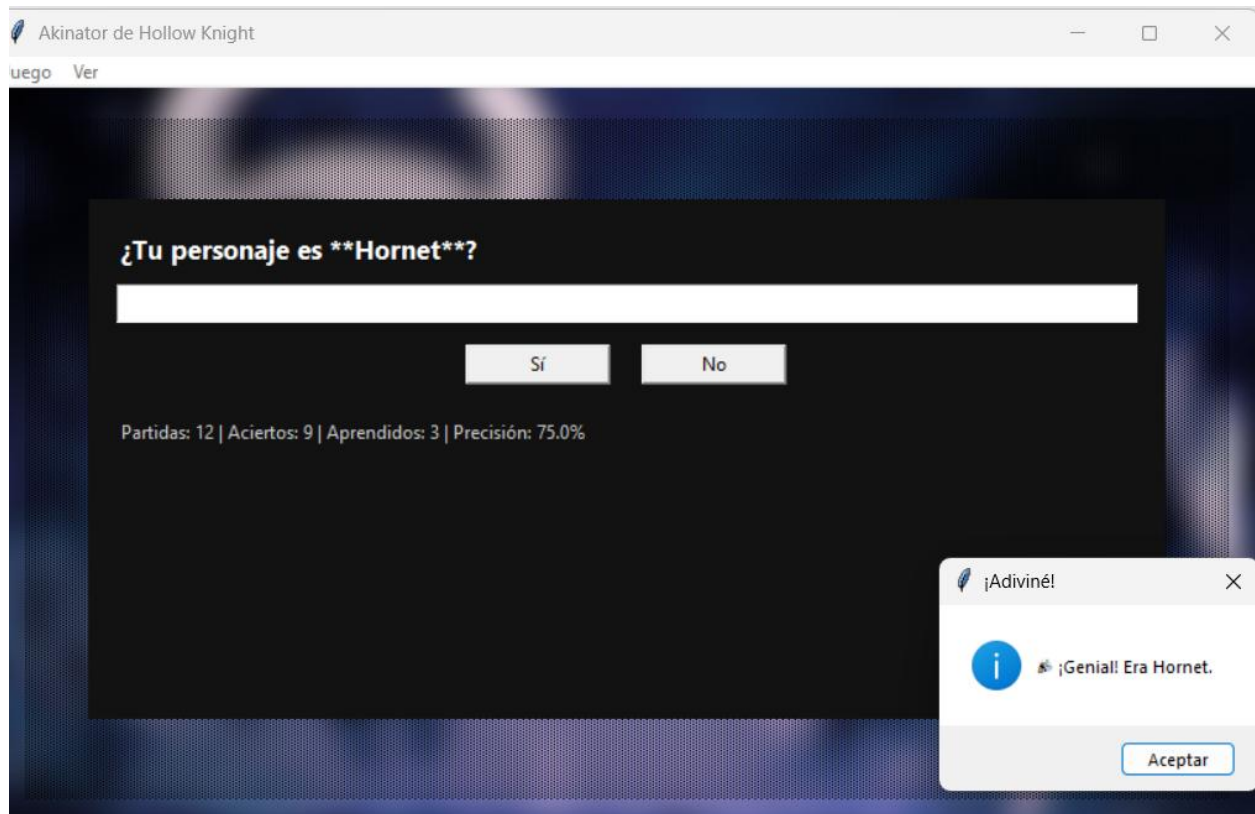
Estos casos se validan con la respuesta del usuario (“Sí” o “No”).

Encadenamiento hacia adelante: el proceso de razonamiento se inicia desde las preguntas generales (reglas de nivel superior) y avanza hacia las conclusiones más específicas (casos). A medida que el usuario responde, el sistema recorre el árbol tomando decisiones basadas en las respuestas. Si llega a una hipótesis incorrecta, el encadenamiento se interrumpe y se introduce un nuevo conocimiento (una nueva regla y un nuevo caso), expandiendo así la base de datos.

4. Ejemplo de funcionamiento

Supongamos que el usuario piensa en el personaje “Hornet”.

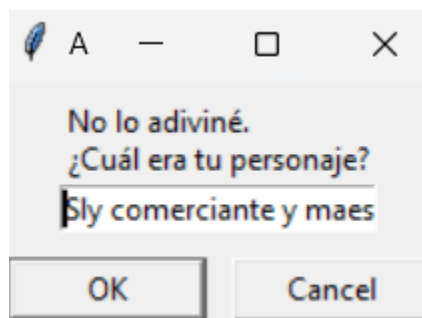
El programa inicia con la pregunta: “¿Es un jefe o te enfrentas a él/ella en combate principal?”. El usuario responde “Sí”, por lo que el sistema continúa por la rama de personajes jefes. Luego pregunta: “¿Usa una aguja e hilo en combate?”. Al responder “Sí”, el sistema concluye correctamente que se trata de Hornet.

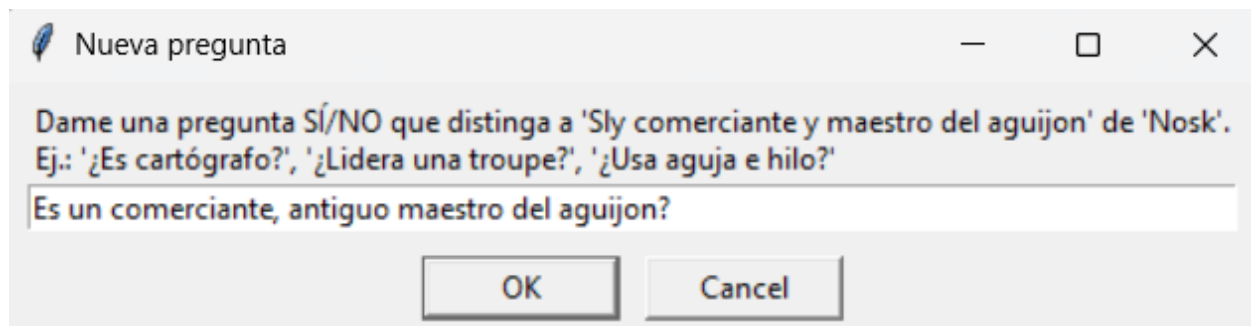


Si el sistema no hubiera logrado adivinar el personaje, habría pedido al usuario:

1. El nombre del nuevo personaje.
2. Una pregunta que lo distinga del personaje propuesto.
3. La respuesta correcta (sí/no) para el nuevo personaje.

Con esta información, el sistema agrega una nueva regla y un nuevo caso al árbol, aprendiendo automáticamente para futuras ejecuciones.



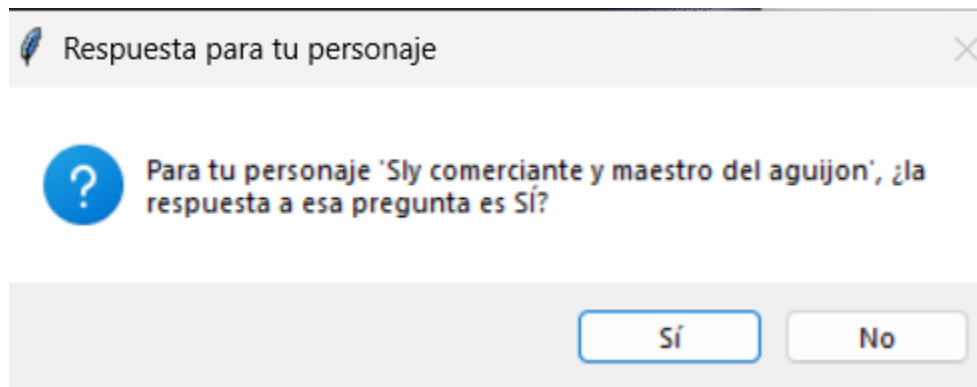


Nueva pregunta

Dame una pregunta SÍ/NO que distinga a 'Sly comerciante y maestro del agujon' de 'Nosk'.
Ej.: '¿Es cartógrafo?', '¿Lidera una troupe?', '¿Usa aguja e hilo?'

Es un comerciante, antiguo maestro del agujon?

OK Cancel



Respuesta para tu personaje

Para tu personaje 'Sly comerciante y maestro del agujon', ¿la respuesta a esa pregunta es SÍ?

Sí No

5. Estructura técnica

El conocimiento se almacena en formato JSON como un árbol de decisiones binario, donde cada nodo puede tener los campos “q” (pregunta), “yes” (rama afirmativa) y “no” (rama negativa). Las hojas contienen únicamente el campo “guess” con el nombre del personaje.

El archivo se actualiza automáticamente mediante el módulo de aprendizaje. La interfaz gráfica utiliza Tkinter y se adapta dinámicamente al tamaño de la ventana, manteniendo el fondo difuminado y los controles legibles sobre un panel oscuro.

6. Conclusión

El “Akinator de Hollow Knight” combina la lógica de inferencia basada en reglas con un aprendizaje incremental sencillo, logrando un sistema interactivo que mejora con cada uso. Este tipo de razonamiento basado en el encadenamiento hacia adelante es ampliamente utilizado en sistemas expertos y asistentes inteligentes, ya que permite simular el razonamiento humano a partir de un conjunto de reglas explícitas y casos particulares.