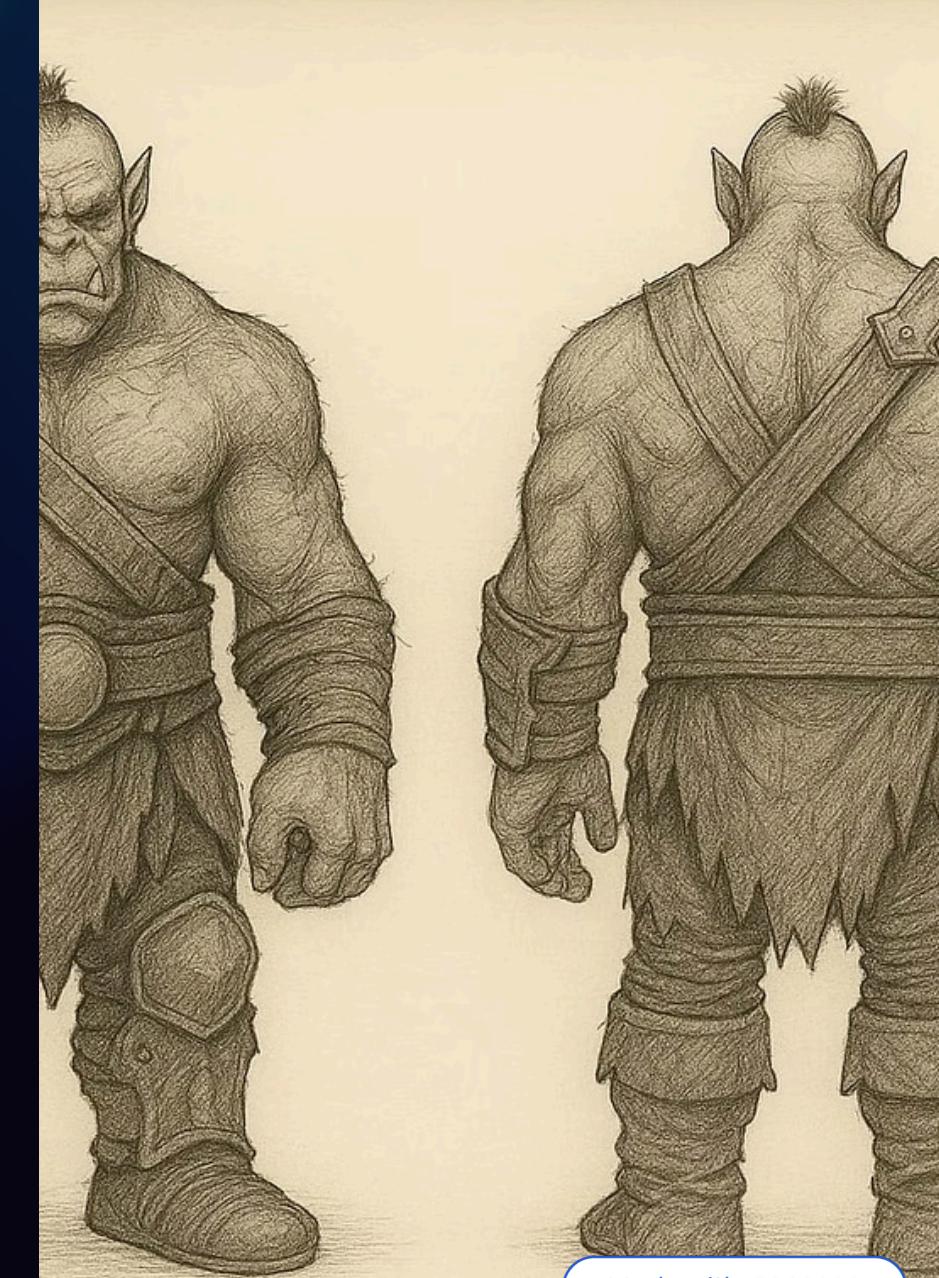


# IA Adaptativa en Videojuegos

Un prototipo de IA para comportamiento de enemigos, utilizando  
Árboles de Decisión en un contexto de juego de rol.



## **Gamificación del aprendizaje**

El proyecto aplica principios de gamificación para fomentar el interés, la participación activa y la retención del conocimiento. La gamificación es una metodología que traslada mecánicas de los videojuegos (puntos, niveles, progresión, recompensas, etc.) a contextos educativos y sociales. En este videojuego se incorpora gamificación al:

1. Usar la recolección de objetos y recetas como desafíos y recompensas.
2. Introducir una progresión del jugador mediante habilidades y profesiones ecológicas (ej. "Inventor", "Ingeniero Solar").
3. Dar retroalimentación inmediata en combate (IA reactiva).
4. Reforzar conceptos de energía limpia a través de mecánicas como el uso de maná solar que se recarga con un fractal solar.



# Arquitectura del Modelo de IA

1

## Tipo de Modelo

Se utiliza un **Árbol de Decisión** (**DecisionTreeClassifier**) por su simplicidad y capacidad de interpretación, ideal para un prototipo inicial.

2

## Tipo de Aprendizaje

Basado en **Aprendizaje Supervisado**, donde el modelo aprende de pares entrada-salida históricos.

3

## Variables Clave

**Independientes (Features):** vida\_jugador, vida\_enemigo.  
**Dependiente (Label):** acción (atacar, defender, esperar).

# Entrenamiento y Uso del Modelo

## Recolección de Datos

Los datos se recolectan automáticamente de partidas anteriores y se almacenan en `datos_partidas.csv`. Cada registro incluye:

- `vida_jugador` (vida del brujo)
- `vida_enemigo` (vida del orco)
- `acción` (decisión del enemigo: "atacar" o "esperar")

## Aplicación en Tiempo Real

La IA del enemigo utiliza el modelo entrenado para decidir su acción (atacar o esperar) en función de los puntos de vida actuales de ambos combatientes.

Si el modelo no está disponible (por falta de datos o errores), se elige una acción aleatoria para asegurar la jugabilidad.

ⓘ Aunque no hay limpieza de datos avanzada, el manejo de errores al cargar el CSV es robusto.

**Modelo usado: DecisionTreeClassifier de scikit-learn** Es un modelo de clasificación de aprendizaje automático que toma decisiones basadas en una estructura de tipo árbol. Imita un proceso lógico similar al humano: va haciendo preguntas "sí o no" para llegar a una decisión.

Por ejemplo: Si el enemigo está cerca → ¿sí? → ¿tengo poca vida? → ¿sí? → defender, si no → atacar.

Este modelo pertenece a la biblioteca scikit-learn, una de las más populares en Python para tareas de machine learning.

**Tipo de aprendizaje:** Aprendizaje supervisado El aprendizaje supervisado es una forma de entrenar a una IA usando datos etiquetados. Es decir, le damos ejemplos de entrada junto con la respuesta correcta (la etiqueta), para que el modelo aprenda a predecir por sí solo más adelante.

## Ejemplo

Vida jugador	Vida enemigo	Distancia	Acción esperada	20	50	Cerca	Defender	90	10	Lejos	Esperar	70	20	Cerca	Atacar
--------------	--------------	-----------	-----------------	----	----	-------	----------	----	----	-------	---------	----	----	-------	--------

El modelo aprende a decidir cuál acción tomar (atacar, defender o esperar) a partir de estos ejemplos.

**Entrenamiento:** El proceso de entrenamiento consiste en alimentar al modelo con estos datos de ejemplo para que:

Detecte patrones y relaciones entre las condiciones del juego (vida, distancia, etc.)

Aprenda qué acción es la más adecuada en cada situación.

Después del entrenamiento, el modelo puede predecir por sí mismo la mejor acción incluso en situaciones nuevas no vistas antes, usando lo aprendido.

# Comportamiento y Adaptación en el Juego



## Comportamiento Aprendido

El modelo permite que la IA del orco aprenda comportamientos específicos en distintos contextos. Por ejemplo, cómo reaccionar con poca vida.



## Toma de Decisiones Mejorada

La IA mejora continuamente sus decisiones a medida que acumula experiencia a través del entrenamiento incremental con cada partida jugada.



## Interacción de Combate

Tanto Gandalf como el Orco Maligno pueden Atacar o Defender. Al Defender, el daño recibido se reduce a la mitad. Gandalf también puede lanzar Hechizos, gastando maná.



### WIZARD



# Problema principal que soluciona

Falta de herramientas interactivas e inteligentes para enseñar habilidades estratégicas, toma de decisiones y conceptos educativos (como sostenibilidad energética), de forma atractiva y personalizada.

El código permite:

Simular situaciones de combate estratégico que requieren pensar antes de actuar.

Aplicar inteligencia artificial supervisada para adaptar el comportamiento del enemigo.

Crear una base para una experiencia educativa personalizada, en la que el sistema aprende del comportamiento del jugador y responde de forma coherente.



# Aplicaciones Futuras:

## Videojuegos



### IA Adaptativa

El modelo puede aprender del estilo de juego del usuario, ofreciendo respuestas más estratégicas y un desafío personalizado.



### NPCs Inteligentes

En juegos complejos, los personajes no jugables (NPCs) pueden tomar decisiones más realistas y contextuales.



### Juegos Educativos

Adaptación de la dificultad y contenido en función del progreso del jugador, haciendo el aprendizaje más efectivo y atractivo.

# Aplicaciones Futuras: Educación y Simulación



## Sistemas de Entrenamiento

Simuladores de combate, estrategia (militares), médicos o deportivos con oponentes de IA adaptativa.



## Gamificación del Aprendizaje

Retos educativos adaptados por IA en áreas como historia, ciencia o idiomas para estudiantes.



## Modelos de Comportamiento

Simulación de decisiones humanas o de agentes para modelos sociales o económicos, o entrenamiento de robots.

# Herramientas y Librerías Utilizadas

**scikit-learn**

Framework principal para el modelo de IA (DecisionTreeClassifier).

**pandas**

Gestión y manipulación eficiente de los datos de juego.

**random**

Generación de daño variable y decisiones aleatorias de fallback.

**ipywidgets**

Herramientas para la interfaz interactiva en Google Colab o entornos Jupyter.

**IPython.display**



# Expansión a Modelos Más Complejos

- **Redes Neuronales / Refuerzo:** Integrar modelos avanzados para una IA más sofisticada y capaz de auto-aprender.
- **Procesamiento de Lenguaje Natural (NLP):** Implementar para interacción por texto o lenguaje natural en el juego.
- **Percepción Visual:** Incorporar si el juego incluye procesamiento de imágenes o interacción con mapas visuales.

Este modelo es una base sólida para futuras expansiones y complejidades en la IA.

# Potenciales Mejoras

## Más Variables

Registrar maná, tipo de acción del jugador, turnos, estado emocional, etc., para decisiones más ricas.

## Aprendizaje por Refuerzo

Permitir que la IA aprenda a maximizar recompensas (victorias, daño evitado) para una estrategia óptima.

## Perfiles de Comportamiento

Crear estilos de enemigo adaptativos a la estrategia única de cada jugador.

## Visualización del Árbol

Renderizar el árbol de decisión para una comprensión profunda de la lógica de la IA.

# Conclusiones y Próximos Pasos

## Resultados Clave

- La IA demuestra capacidad para aprender comportamientos lógicos y adaptativos del historial de juego.
- Interacción positiva de jugadores con mecánicas simples que ocultan decisiones complejas.
- Gran potencial para gamificación educativa, premiando la sostenibilidad en el futuro.

## Próximos Pasos

Continuar la expansión con modelos más complejos y la integración de nuevas variables para una IA verdaderamente dinámica.

[Ver Código en Google Colab](#)

[Haz clic aquí](#)