Curso base

INTELIGENCIA ARTIFICIAL DESDE CERO

LECCIÓN 22
Redes neuronales e IA moderna

Cómo comprender de manera intuitiva qué son las neuronas artificiales, las capas y el entrenamiento simple
para interpretar mejor la IA moderna y sus aplicaciones actuales

Lección	22	**Tema**	Redes neuronales e IA moderna
Nombre del participante	_____	**Fecha**	_____

Actividad central

Actividad principal: comprender la idea de neurona artificial, capas y entrenamiento, y analizar dos ejemplos guiados con redes neuronales pequeñas.
Producto esperado: esquema comprensible de una red neuronal y dos ejemplos guiados con interpretación básica.

Propósito de la guía. Comprender, sin fórmulas pesadas, cómo funcionan las redes neuronales, por qué aprenden a partir de datos y cómo se relacionan con la IA moderna que hoy aparece en asistentes, visión por computador, reconocimiento de voz y modelos generativos.

Idea central

Las redes neuronales son modelos inspirados de manera muy general en la idea de muchas unidades pequeñas que colaboran para tomar una decisión. No copian el cerebro humano real, pero sí aprovechan una intuición poderosa: varias señales simples, combinadas de manera adecuada, pueden producir respuestas mucho más complejas y útiles.

La IA moderna utiliza redes neuronales porque permiten aprender patrones no lineales, es decir, relaciones que no se describen bien con una sola regla sencilla. Gracias a ello han sido fundamentales en avances como clasificación de imágenes, detección de objetos, traducción automática, asistentes conversacionales y generación de texto.

¿Qué vamos a hacer en esta guía?

Primero construiremos una idea intuitiva de qué es una neurona artificial y cómo se organiza una red en capas. Después desarrollaremos un primer ejemplo guiado para reconocer patrones curvos con una red neuronal pequeña. Luego trabajaremos un segundo ejemplo guiado con datos tabulares sencillos. En ambos casos incorporaremos prompts de IA generativa para ayudar a comprender, interpretar y explicar resultados.

Vocabulario mínimo

Concepto	Explicación sencilla
Neurona artificial	Pequeña unidad que recibe varios valores de entrada y produce una salida.
Peso	Número que indica qué tan importante es cada entrada en la decisión final.
Sesgo	Ajuste adicional que da más flexibilidad a la respuesta del modelo.
Función de activación	Regla que transforma la suma interna y ayuda a capturar relaciones no lineales.
Capas	Conjuntos de neuronas: una de entrada, una o varias ocultas y una de salida.
Entrenamiento	Proceso de ajustar pesos y sesgos para que la red se equivoque menos.

¿Por qué las redes neuronales importan tanto?

Idea	¿Qué significa?	¿Por qué ayuda?
Muchas entradas a la vez	La red puede recibir varias características del problema.	Permite combinar señales diferentes.
Capas ocultas	La información se transforma paso a paso en el interior del modelo.	Ayuda a detectar patrones más complejos.
Ajuste de pesos	La red modifica su comportamiento cuando se equivoca.	Hace posible el aprendizaje a partir de ejemplos.
No linealidad	La red no se limita a una sola línea o regla rígida.	Captura fronteras curvas o relaciones difíciles.

Ejemplo guiado 1: reconocer patrones curvos con una red neuronal pequeña

Vamos a comenzar con una situación muy visual. Imaginemos dos grupos de puntos que no se separan bien con una sola línea recta. Un modelo demasiado simple puede fallar, pero una red neuronal pequeña puede adaptarse mejor a una frontera curva. Este ejemplo ayuda a entender por qué las redes neuronales son importantes en la IA moderna.

Antes de programar, conviene responder dos preguntas. Primera: *¿qué entra al modelo?* Entran las coordenadas de cada punto. Segunda: *¿qué sale del modelo?* Sale la clase a la que pertenece cada punto.

Paso	Qué haremos
Paso 1	Crear un conjunto de datos sencillo con una forma no lineal.
Paso 2	Separar los datos en entrenamiento y prueba.
Paso 3	Construir una red neuronal pequeña con capas ocultas.
Paso 4	Entrenar el modelo con los datos de entrenamiento.
Paso 5	Probar el modelo con datos nuevos y medir accuracy.

Código base del ejemplo 1

```
from sklearn.datasets import make_moons
from sklearn.model_selection import train_test_split
from sklearn.neural_network import MLPClassifier
from sklearn.metrics import accuracy_score

X, y = make_moons(n_samples=300, noise=0.20, random_state=42)

X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.25, random_state=42
)

modelo = MLPClassifier(
    hidden_layer_sizes=(8, 4),
    activation='relu',
    max_iter=2000,
    random_state=42
)

modelo.fit(X_train, y_train)
pred = modelo.predict(X_test)

print('Accuracy:', round(accuracy_score(y_test, pred), 2))
```

¿Cómo interpretar este ejemplo?

Lo importante aquí no es memorizar cada línea de código. Lo importante es reconocer el orden lógico. Primero se crean los datos. Después se separan entrenamiento y prueba. Luego

se define la red neuronal. Más tarde se entrena y finalmente se evalúa con datos que el modelo no vio durante el aprendizaje.
Si la accuracy es razonable, podemos decir que la red neuronal logró capturar parte del patrón curvo presente en los datos. No significa que sea perfecta, pero sí que aprendió una frontera más flexible que la de un modelo excesivamente simple.

Prompts de IA generativa para acompañar el ejemplo 1

Prompt	¿Para qué sirve?
Explícame con palabras de principiante qué hace una red neuronal en este ejemplo de `make_moons`.	Ayuda a traducir el código a una idea comprensible.
Descríbeme paso a paso qué representa cada bloque del código de `MLPClassifier`.	Permite identificar datos, modelo, entrenamiento y evaluación.
Compara un modelo lineal y una red neuronal pequeña en un problema con frontera curva.	Refuerza la intuición sobre la ventaja de la no linealidad.
Ayúdame a redactar una interpretación corta y prudente de la accuracy obtenida.	Sirve para escribir conclusiones claras y honestas.

Errores comunes en este primer acercamiento

Error frecuente	Consecuencia
Pensar que la red neuronal imita exactamente el cerebro humano.	Genera una idea equivocada del modelo.
Creer que más capas siempre significan mejor resultado.	Puede llevar a modelos innecesariamente complejos.
Mirar solo el puntaje final sin entender el proceso.	Se pierde la comprensión pedagógica del flujo.
Olvidar separar entrenamiento y prueba.	La evaluación deja de ser honesta.

Segundo ejemplo guiado: red neuronal pequeña con datos tabulares sencillos

Ahora llevemos la idea a un caso más cercano al trabajo con tablas. Supongamos que queremos predecir si un estudiante aprobará o no, a partir de horas de estudio, tareas entregadas y asistencia. En este ejemplo la forma de los datos es distinta, pero el flujo general sigue siendo el mismo: preparar, separar, entrenar y evaluar.

Lo interesante aquí es ver que una red neuronal también puede usarse con datos tabulares sencillos. Así el estudiante entiende que la red no pertenece solamente al mundo de las imágenes o del lenguaje, sino que también puede aparecer en problemas básicos de clasificación.

Variable	Tipo de información	Papel en el ejemplo
horas_estudio	Numérica	Describe cuánto tiempo dedica el estudiante al estudio.
tareas	Numérica	Resume cuántas tareas entrega.
asistencia	Numérica	Refleja constancia y presencia en clase.
aprueba	Etiqueta	Es la clase objetivo: 1 aprueba, 0 no aprueba.

Pasos del ejemplo 2

1. Crear la tabla pequeña con tres variables de entrada y una etiqueta final.
2. Separar `X` e `y` para dejar claro qué entra al modelo y qué debe aprender.
3. Dividir la tabla en entrenamiento y prueba.
4. Entrenar una red neuronal pequeña.
5. Generar predicciones e interpretar si el flujo tiene sentido.

Código base del ejemplo 2

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.neural_network import MLPClassifier
from sklearn.metrics import accuracy_score

datos = pd.DataFrame({
    'horas_estudio': [1, 2, 2, 3, 4, 4, 5, 6, 6, 7],
    'tareas': [0, 1, 1, 1, 2, 2, 2, 3, 3, 4],
    'asistencia': [60, 65, 70, 72, 80, 82, 85, 90, 92, 95],
    'aprueba': [0, 0, 0, 0, 1, 1, 1, 1, 1, 1]
})

X = datos[['horas_estudio', 'tareas', 'asistencia']]
y = datos['aprueba']

X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.30, random_state=42, stratify=y
)

modelo = MLPClassifier(hidden_layer_sizes=(6,), max_iter=2000, random_state
    =42)
modelo.fit(X_train, y_train)

pred = modelo.predict(X_test)
print('Accuracy:', round(accuracy_score(y_test, pred), 2))
print('Predicciones:', pred)
```

¿Qué aprendemos con el segundo ejemplo?

Aprendemos que una red neuronal no es magia. Sigue una secuencia ordenada parecida a la de otros modelos ya vistos en el curso. También vemos que la red toma varias entradas, las combina internamente y finalmente entrega una etiqueta de salida.
En esta etapa no importa alcanzar el mejor clasificador posible. Importa comprender la idea de red, reconocer que los pesos se ajustan durante el entrenamiento y entender por qué una red pequeña puede capturar relaciones más complejas que un modelo muy rígido.

Prompts de IA generativa para acompañar el ejemplo 2

Prompt	Utilidad pedagógica
Explícame este ejemplo de red neuronal con estudiantes usando lenguaje muy sencillo.	Traduce el caso técnico a una explicación accesible.
Dime qué significa cada columna de la tabla y cuál es la variable objetivo.	Refuerza la lectura correcta del conjunto de datos.
Ayúdame a comparar este ejemplo con un clasificador más simple como regresión logística.	Permite comprender semejanzas y diferencias.
Redáctame una conclusión breve sobre lo que sí aprendí de redes neuronales en esta guía.	Facilita cerrar la actividad con reflexión propia.

Comparación intuitiva entre los dos ejemplos

Aspecto	Ejemplo 1	Ejemplo 2
Problema	Separar patrones curvos	Clasificar aprobado / no aprobado
Tipo de datos	Puntos en el plano	Tabla con variables numéricas
Salida	Clase 0 o 1	Clase 0 o 1
Aprendizaje principal	Una red puede adaptarse a fronteras no lineales	Una red también sirve en tablas sencillas

Buenas prácticas para esta lección

Recomendación	¿Por qué importa?
Empezar con redes pequeñas	Ayuda a comprender el proceso sin abrumarse con complejidad innecesaria.
Diferenciar entrada, capas ocultas y salida	Permite entender el recorrido de la información.

Separar entrenamiento y prueba	Protege la evaluación y evita conclusiones engañosas.
Interpretar antes de complicar	La meta inicial es comprender la lógica del modelo.
Usar IA generativa como apoyo, no como reemplazo	Favorece el razonamiento propio y la escritura comprensible.

Producto esperado

El producto esperado en esta guía es un material corto y claro, presentado en notebook o en una hoja de trabajo, donde el estudiante explique con palabras sencillas qué es una red neuronal, identifique sus partes básicas y acompañe esa explicación con dos ejemplos guiados e interpretados.

Lista de verificación

Revisión	Criterio
☐	¿Se explicó con palabras sencillas qué es una red neuronal?
☐	¿Se diferenciaron entrada, capas ocultas y salida?
☐	¿Se mencionó la idea de ajustar pesos durante el entrenamiento?
☐	¿Se interpretó el primer ejemplo sin limitarse a copiar el código?
☐	¿Se comprendió que una red neuronal también puede trabajar con tablas sencillas?
☐	¿Se usó al menos un prompt de IA generativa para comprender o explicar mejor la lección?

Cierre

Trabajar con redes neuronales da un paso más hacia la IA moderna, pero sin abandonar la claridad pedagógica. En esta guía el objetivo no fue dominar todos los detalles técnicos, sino formar una intuición sólida: una red neuronal recibe entradas, combina información en capas, ajusta sus pesos cuando se equivoca y finalmente produce una predicción. Con esa base clara, será mucho más fácil comprender desarrollos posteriores de la inteligencia artificial contemporánea.