

The background of the slide features a complex network diagram with numerous nodes and connecting lines, rendered in a light blue color against a dark blue background. The network is dense and spans the entire width and height of the slide.

# Fundamentos de **Deep Learning**

---

Sesión 3

---



# Entorno Python Para Redes Neuronales

## Keras

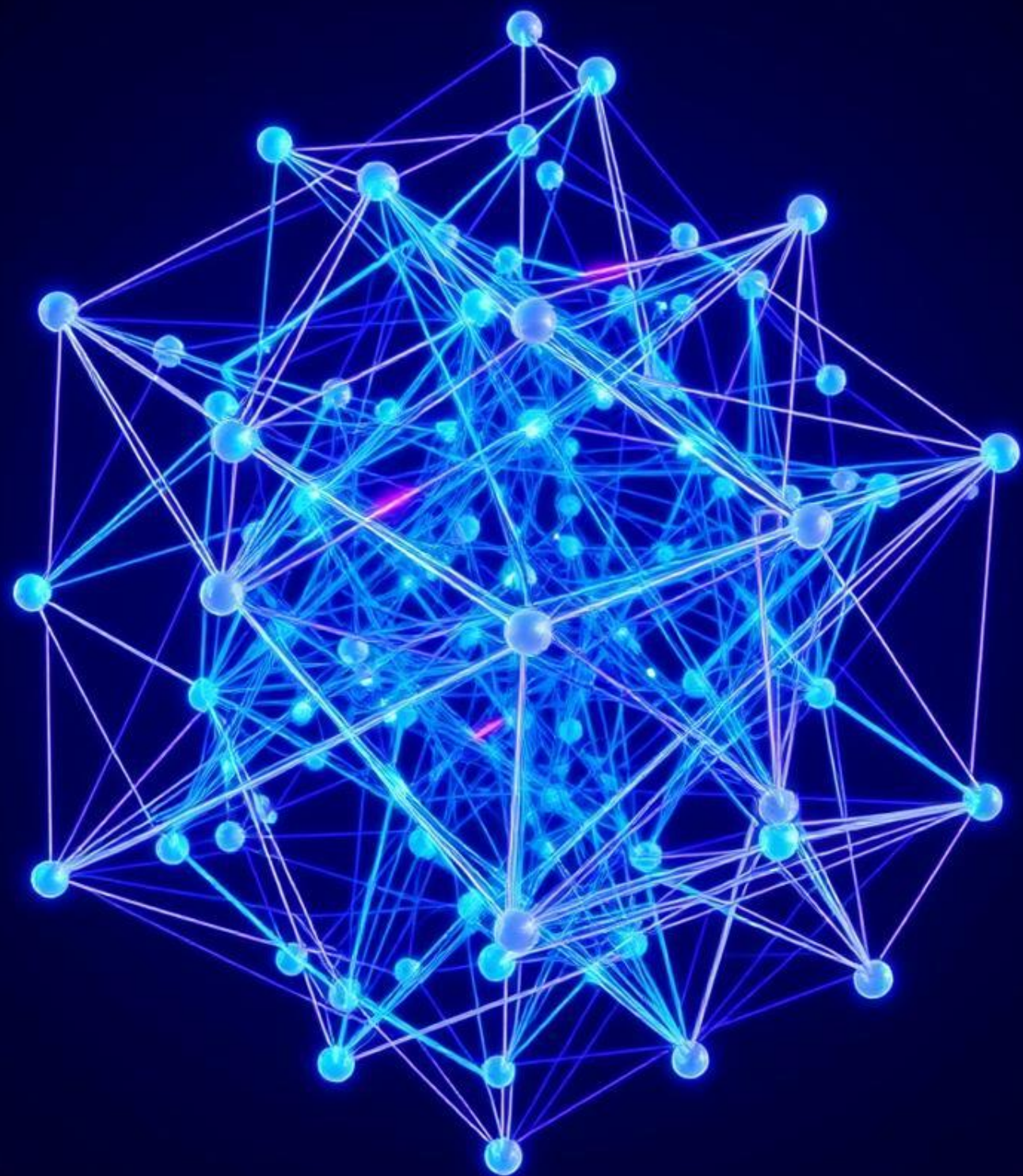
API de alto nivel para desarrollo rápido de modelos. Construida sobre TensorFlow, simple y fácil de usar.

## PyTorch

Librería flexible para control granular de modelos. Ampliamente usada en investigación y academia.



# ¿Qué es un Tensor?



## **Escalar**

Tensor de orden 0: un número único.



## **Vector**

Tensor de orden 1: una lista de números.



## **Matriz**

Tensor de orden 2: tabla bidimensional de números.



## **Imagen a color**

Tensor de orden 3: altura, ancho y canales de color.

# Diseño de RN para resolver un problema de Regresión

## Capa de entrada

Neuronas = características del dataset.

## Capas ocultas

Varias capas densas, cantidad depende del problema.

## Capa de salida

Una neurona, valor continuo.

## Función de pérdida

Mean Squared Error (MSE).

# Actividad Práctica Guiada

**Objetivo:** Resolver un problema de regresión usando una red neuronal. Utilizaremos Keras y datos de scikit-learn.

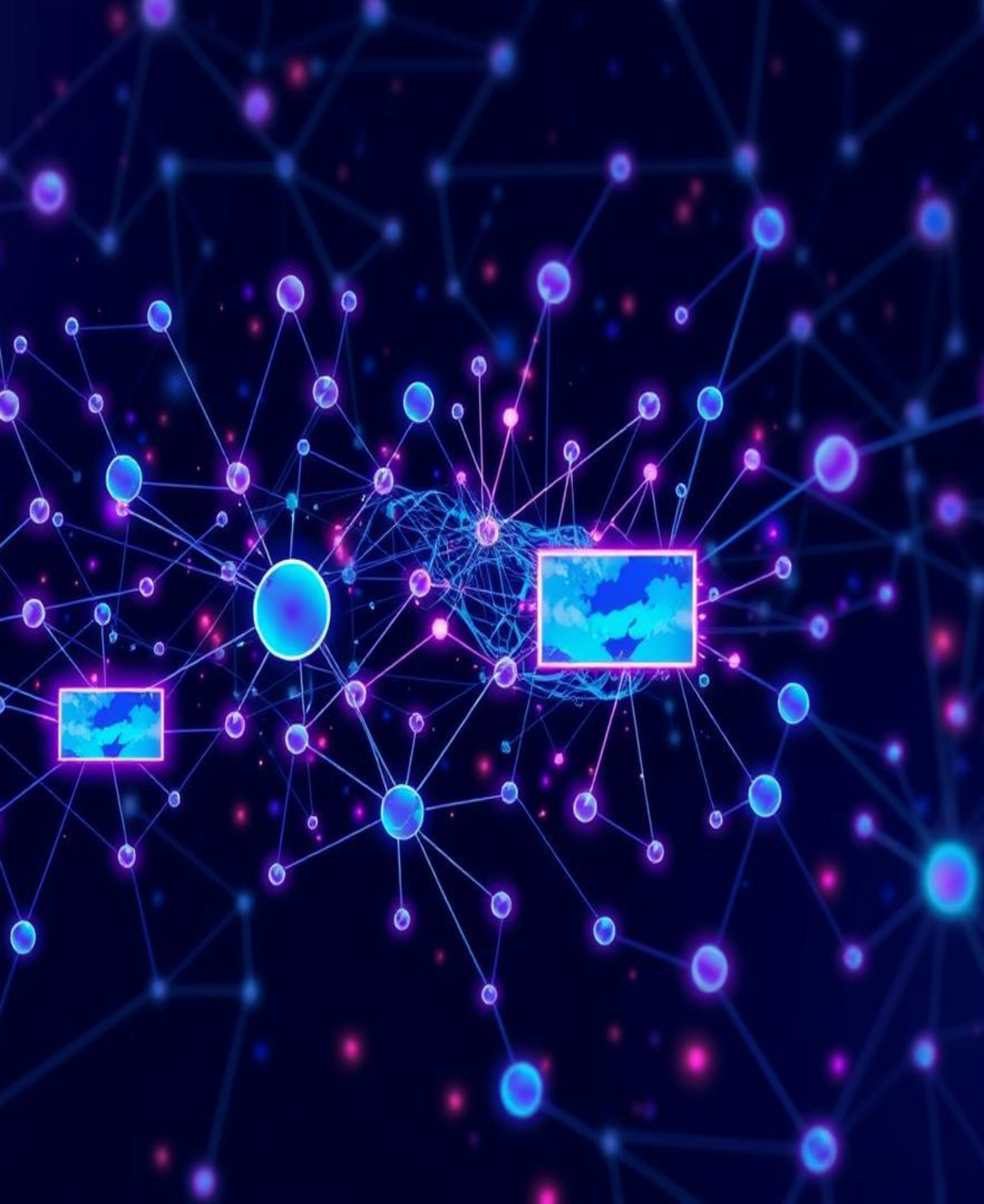
## Requisitos:

1. Instalar las librerías necesarias: tensorflow, scikit-learn, matplotlib
2. Cargar los datos del dataset de Boston Housing.
3. Definir la red neuronal usando Keras: capa de entrada, capa oculta y capa de salida.
4. Entrenar el modelo: configurar número de épocas y tamaño del batch.
5. Visualizar el rendimiento del modelo.
6. Realizar predicciones.



El detalle de la actividad se encuentra en la guía de estudio de la sesión.





# Clasificación con Redes Neuronales

## ⇒ **Capa de entrada**

Neuronas = características del dataset.

## ⇒ **Capas ocultas**

Múltiples capas con activaciones como ReLU.

## ⇒ **Capa de salida**

Neuronas = número de clases.

# Actividad Práctica Guiada

**Objetivo:** Resolver un problema de clasificación utilizando una red neuronal.

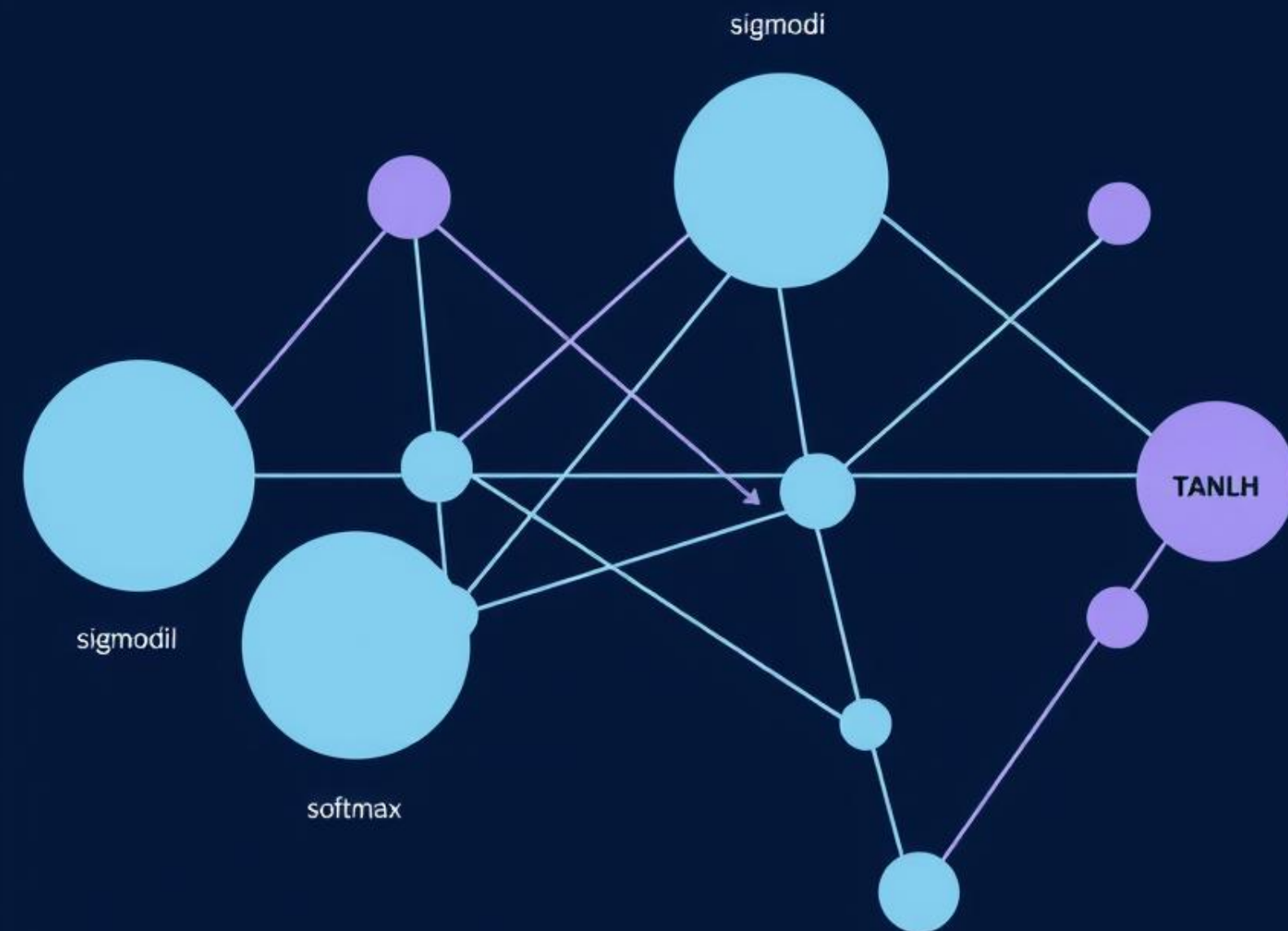
**Requisitos:**

1. Instalar las librerías necesarias: tensorflow, scikit-learn, matplotlib
2. Cargar los datos del dataset de Iris desde scikit-learn.
3. Definir la red neuronal: capa de entrada, una o más capas ocultas y capa de salida.
4. Entrenar el modelo.
5. Evaluar el modelo.
6. Visualizar el rendimiento del modelo.
7. Realizar predicciones.



El detalle de la actividad se encuentra en la guía de estudio de la sesión.

# Funciones de Activación



## ReLU

Simple y eficaz, resuelve gradientes vanishing.

## Sigmoid

Valores entre 0 y 1, clasificación binaria.

## Tanh

Rango entre -1 y 1, centra valores en 0.

## Softmax

Clasificación multiclase, convierte salidas en probabilidades.



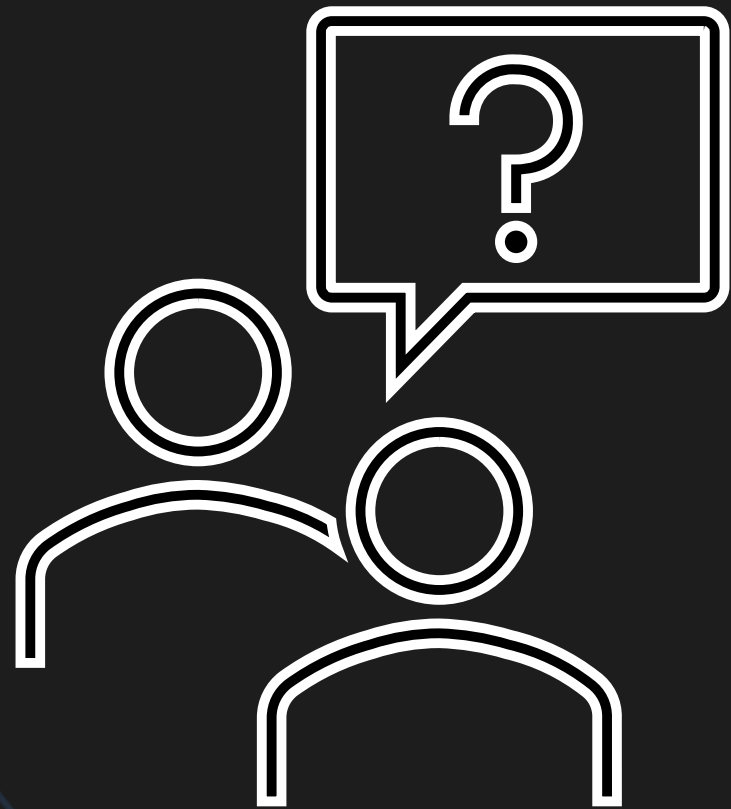
# Funciones para Activación

Función	Descripción	Uso
ReLU	Devuelve el valor si es positivo, 0 si es negativo.	General, redes profundas.
Sigmoid	Valores entre 0 y 1.	Clasificación binaria.
Tanh	Valores entre -1 y 1.	Centra valores en 0.
Softmax	Convierte salidas en probabilidades.	Clasificación multiclase.



# Preguntas

Sección de preguntas





The background of the slide features a complex network diagram with numerous nodes and connecting lines, rendered in a light blue color against a dark blue background. The nodes are small squares, and the lines are thin, creating a web-like structure that fills the entire slide.

# Fundamentos de **Deep Learning**

---

Continúe con las  
actividades

---