



SEMANA 5

Aprendizaje supervisado: regresión y  
clasificación con redes neuronales.  
Aprendizaje no supervisado: agrupamiento

Prof. Luis Torrejón



---

Al finalizar, el alumno conocerá sobre el funcionamiento de regresión y clasificación con redes neuronales basado en aprendizaje supervisado y revisaremos el agrupamiento de aprendizaje no supervisado.

---



# Recordemos - Tipos de clasificadores

## Bootstrapping

Esta técnica crea múltiples modelos a partir de subconjuntos aleatorios del conjunto de entrenamiento original, aprovechando la variabilidad presente en los datos.

## Bagging

El bagging (Bootstrap Aggregating) entrena una colección de clasificadores independientes en paralelo sobre muestras aleatorias del conjunto de entrenamiento, y luego combina sus predicciones.

## Boosting

A diferencia de Bootstrapping y Bagging, Boosting entrena los clasificadores de manera secuencial, prestando más atención a las muestras que son más difíciles de clasificar correctamente.

Cada uno de estos métodos tiene sus propias características, ventajas y desafíos, que exploraremos en detalle a lo largo de esta presentación.

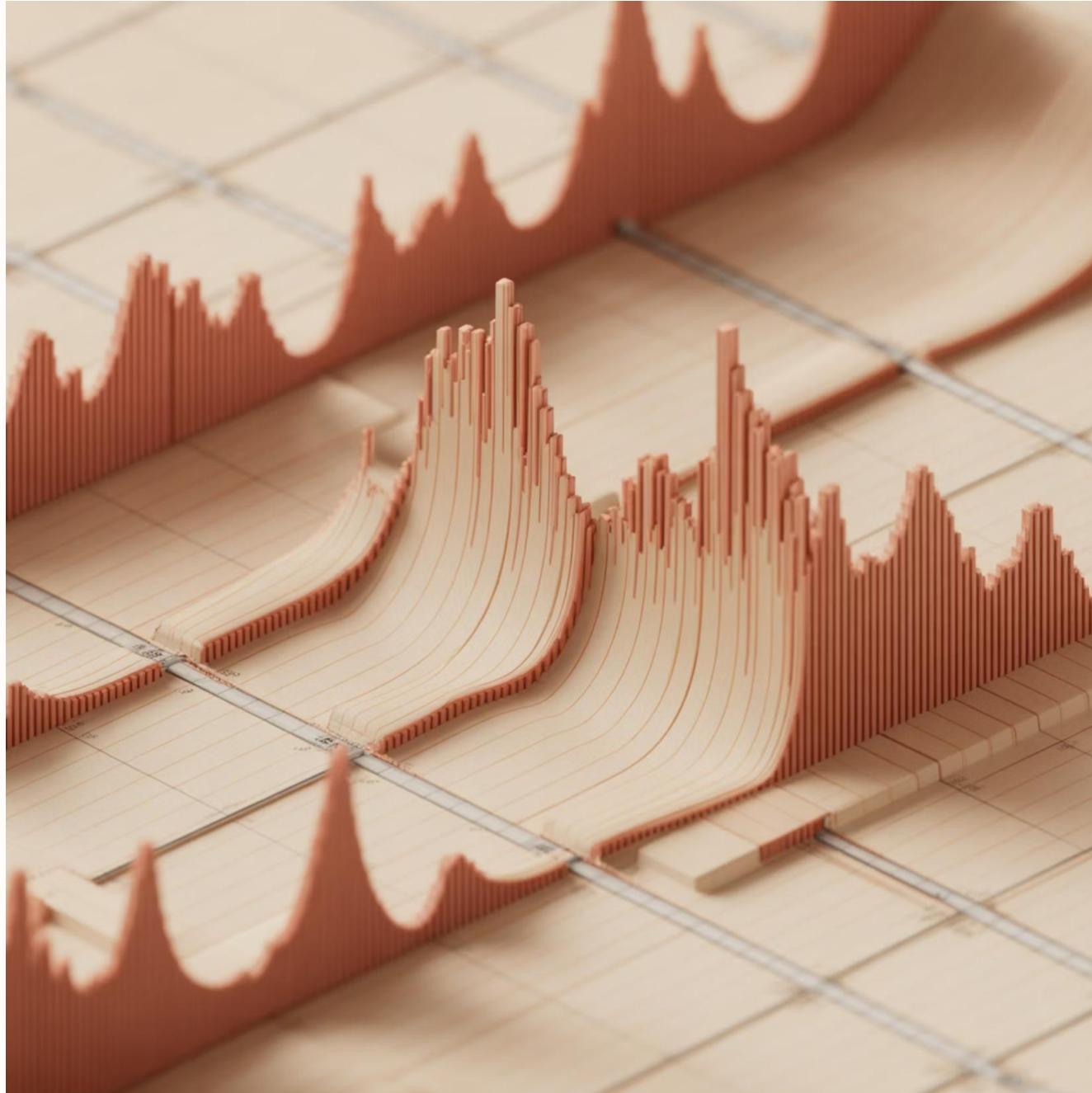




# Recordemos - Comparativa Bootstrapping, Bagging y Boosting

Técnica	Descripción	Fortalezas	Debilidades
<b>Bootstrapping</b>	Crea múltiples modelos a partir de muestras aleatorias con reemplazo del conjunto de entrenamiento original.	<ul style="list-style-type: none"><li>• Captura la variabilidad inherente a los datos</li><li>• Permite evaluar la incertidumbre de las predicciones</li><li>• Mejora la generalización del modelo</li></ul>	<ul style="list-style-type: none"><li>• Computacionalmente intensivo</li><li>• Depende de la calidad y representatividad del conjunto de entrenamiento</li><li>• Dificultad para interpretar y explicar los resultados</li></ul>
<b>Bagging</b>	Entrena múltiples clasificadores independientes en paralelo sobre diferentes muestras aleatorias y combina sus predicciones.	<ul style="list-style-type: none"><li>• Mejora la precisión y estabilidad de las predicciones</li><li>• Reduce la varianza y la sensibilidad a datos atípicos</li><li>• Flexibilidad para adaptarse a diversos algoritmos</li></ul>	<ul style="list-style-type: none"><li>• Mayor complejidad computacional</li><li>• Depende de la calidad y representatividad del conjunto de entrenamiento</li><li>• Dificultad para interpretar y explicar los resultados</li></ul>
<b>Boosting</b>	Construye una secuencia de clasificadores débiles, cada uno enfocado en corregir los errores del anterior.	<ul style="list-style-type: none"><li>• Mejora gradual y sistemática del rendimiento</li><li>• Mayor precisión en problemas complejos</li><li>• Flexibilidad para adaptarse a diversos algoritmos</li></ul>	<ul style="list-style-type: none"><li>• Sensibilidad a datos atípicos u observaciones ruidosas</li><li>• Mayor complejidad computacional</li><li>• Dificultad para interpretar y explicar el modelo final</li></ul>

# Recordemos - Máquinas de Vectores de Soporte (SVM)



## Support Vector Machines

Las **SVM** son algoritmos de aprendizaje supervisado utilizados principalmente para clasificación, aunque también aplican a regresión.

Desarrolladas por Vladimir Vapnik y Corinna Cortes en los años 90, las SVM fueron el método dominante en machine learning antes del auge del deep learning.

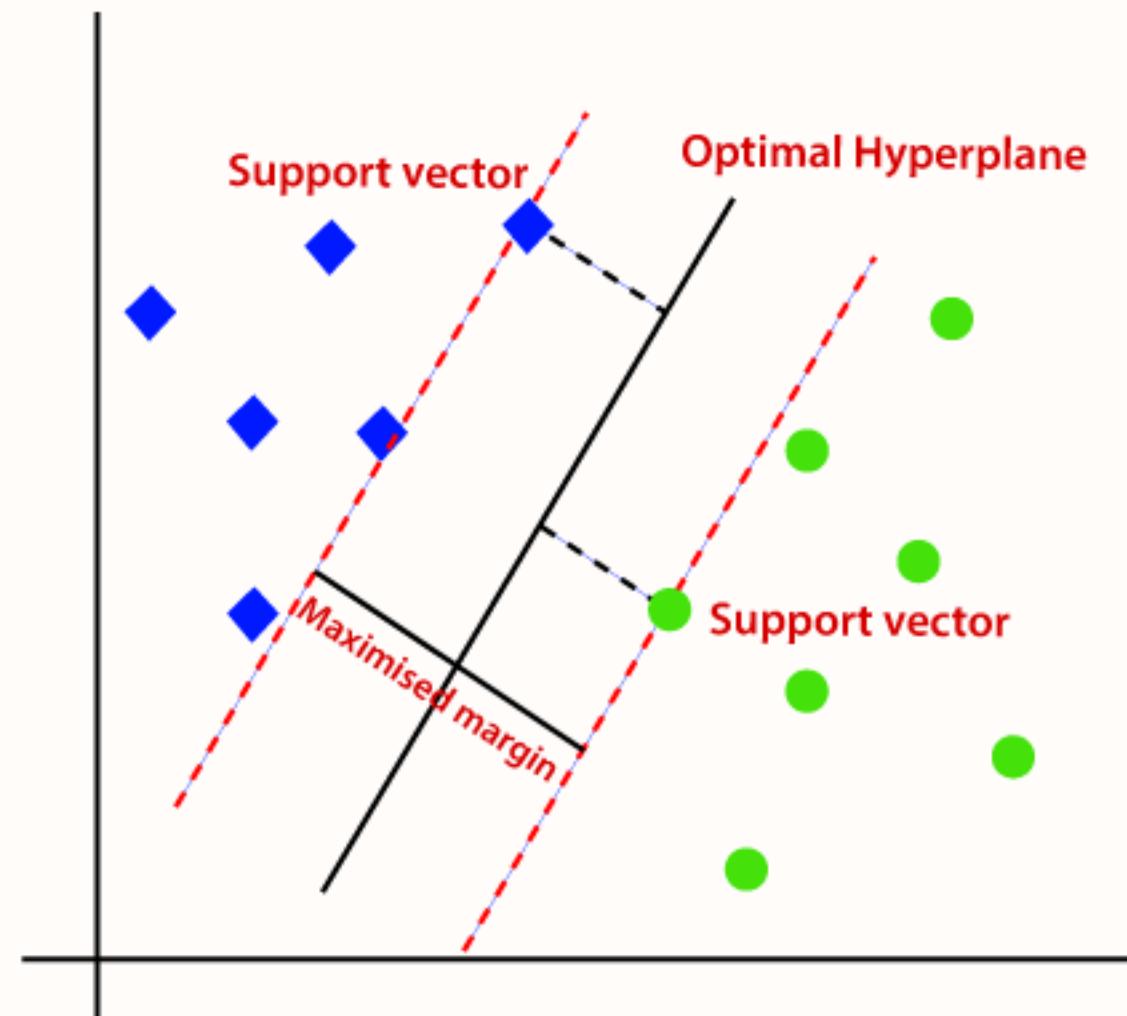
## Elegancia geométrica

Las SVM se basan en conceptos geométricos intuitivos: encontrar la mejor "línea divisoria" entre clases de datos.



# Recordemos - SVM – Definición

**SVM** es un modelo que busca el hiperplano óptimo que maximiza el margen entre las clases, creando una frontera de decisión clara



<https://www.javatpoint.com/machine-learning-support-vector-machine-algorithm>



# Recordemos - SVM – Ejemplos de uso de SVM

- ✓ Diagnóstico de cáncer
- ✓ Marketing para predecir la contratación de productos por clientes.
- ✓ Clasificación de textos





## TEMARIO

- 
- A blurry, low-light photograph of a modern building with large glass windows and a curved roofline. Some people are visible near the entrance.
- 1 — Aprendizaje supervisado: regresión y clasificación con redes neuronales.
  - 2 — Aprendizaje no supervisado: agrupamiento.



# Mapa Conceptual del Aprendizaje Automático



## Aprendizaje Supervisado

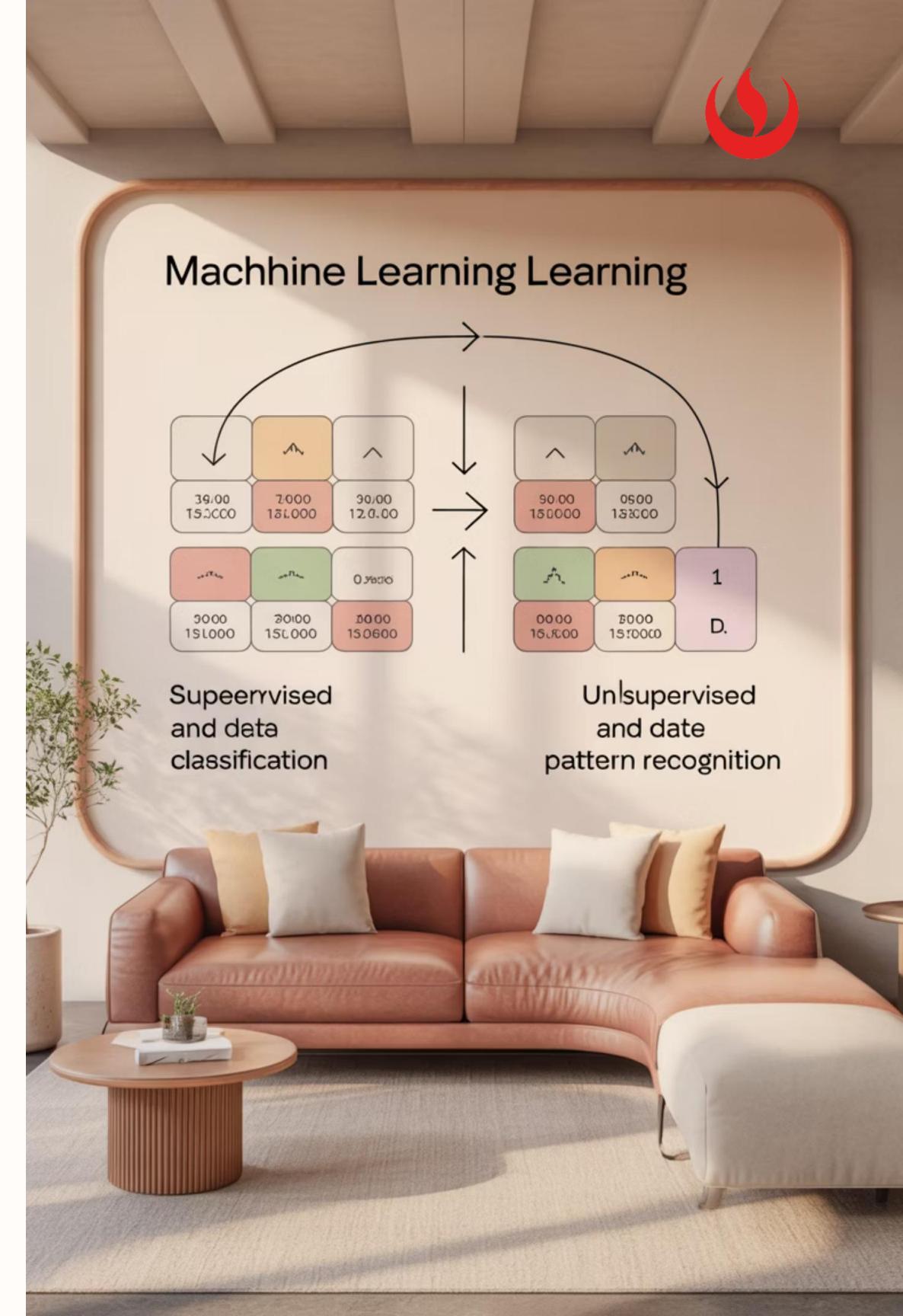
Requiere datos etiquetados donde conocemos la respuesta correcta. El modelo aprende la relación entre características (X) y etiquetas (Y).

- Datos de entrenamiento con respuestas conocidas
- El algoritmo aprende patrones supervisados
- Genera predicciones sobre nuevos datos

## Aprendizaje No Supervisado

Trabaja con datos sin etiquetar, buscando estructura y patrones inherentes en la información sin guía externa.

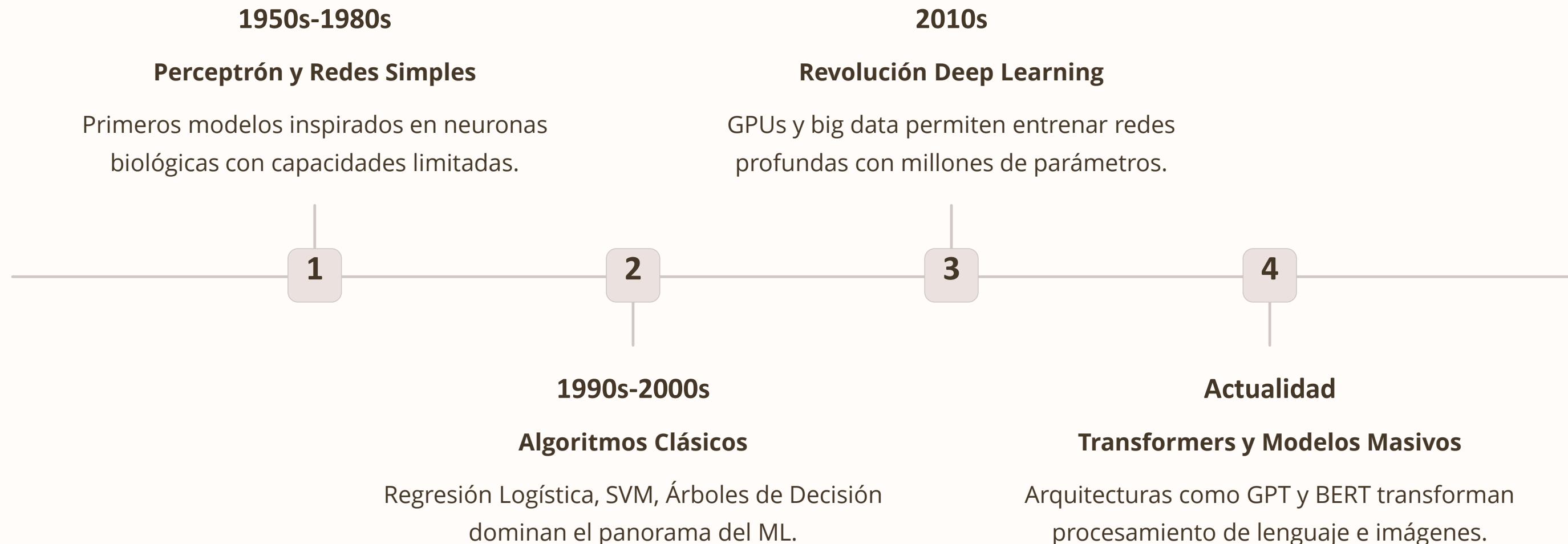
- No existen respuestas correctas predefinidas
- Identifica similitudes y agrupaciones naturales
- Descubre estructura oculta en los datos



# Evolución: De Modelos Clásicos a Deep Learning



El aprendizaje automático ha experimentado una transformación radical en las últimas décadas, pasando de algoritmos simples a arquitecturas neuronales profundas capaces de resolver problemas complejos.

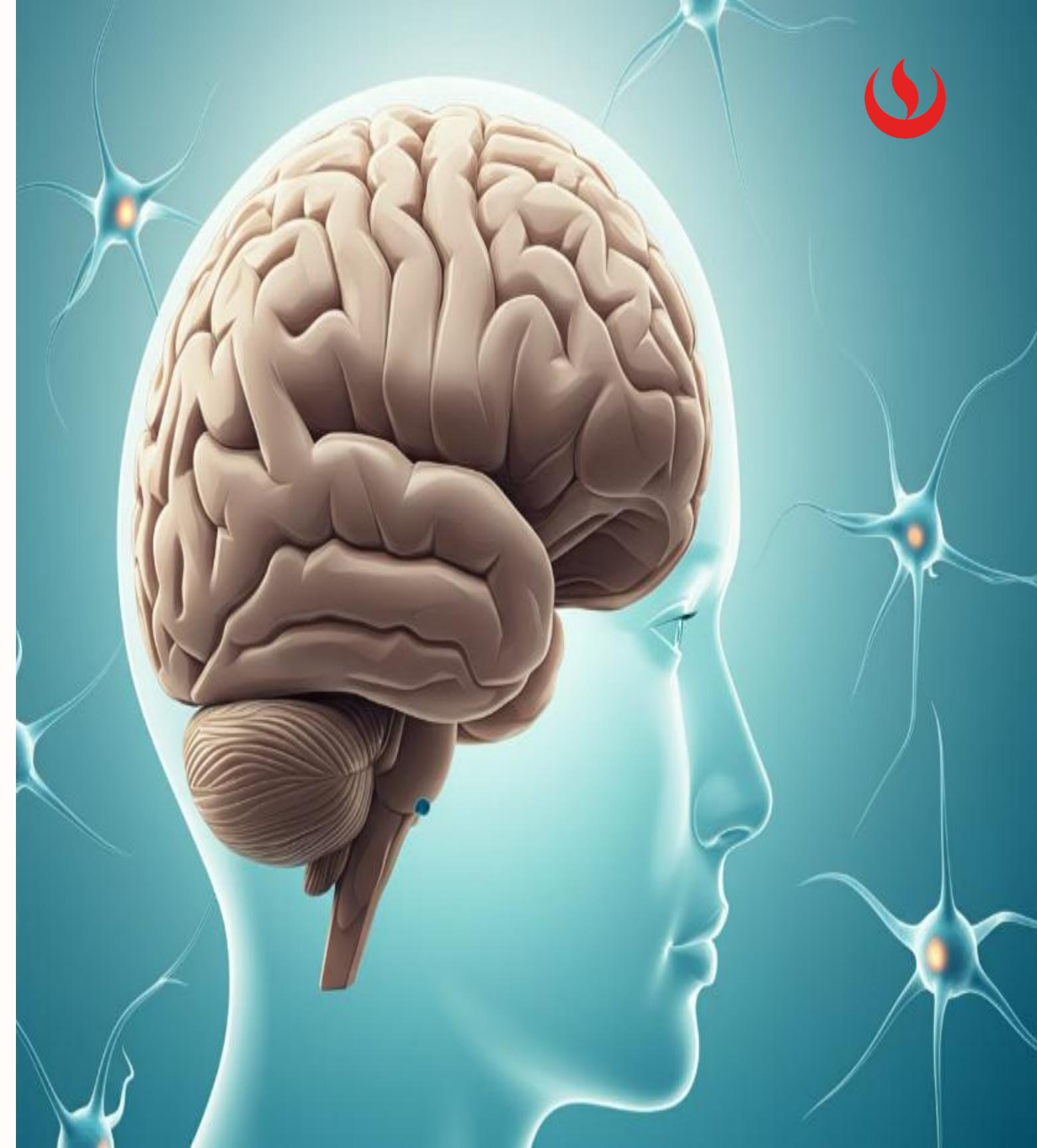




# Cerebro humano

El cerebro humano es un sistema complejo y fascinante, conformado por una intrincada red de células nerviosas conocidas como neuronas.

Estas neuronas se comunican entre sí a través de conexiones llamadas sinapsis, formando una red neural altamente dinámica y capaz de procesar una amplia gama de información.





# Comparación entre cerebro y computadoras

## Estructura

El cerebro humano se compone de billones de neuronas vinculadas, en cambio, las computadoras se fundamentan en circuitos electrónicos y chips. Cada uno posee una configuración esencialmente distinta para el procesamiento de la información.

## Paralelismo

El cerebro opera de forma altamente paralela, llevando a cabo diversas funciones al mismo tiempo. Las computadoras convencionales manejan la información de forma secuencial, pese a que los progresos en arquitecturas paralelas han potenciado su habilidad para paralelismo.

## Eficiencia energética

El cerebro es sumamente eficaz en la utilización de energía, consumiendo apenas 20 vatios, mientras que las computadoras actuales pueden necesitar cientos o miles de vatios para funcionar.



# Neuronas y conexiones sinápticas



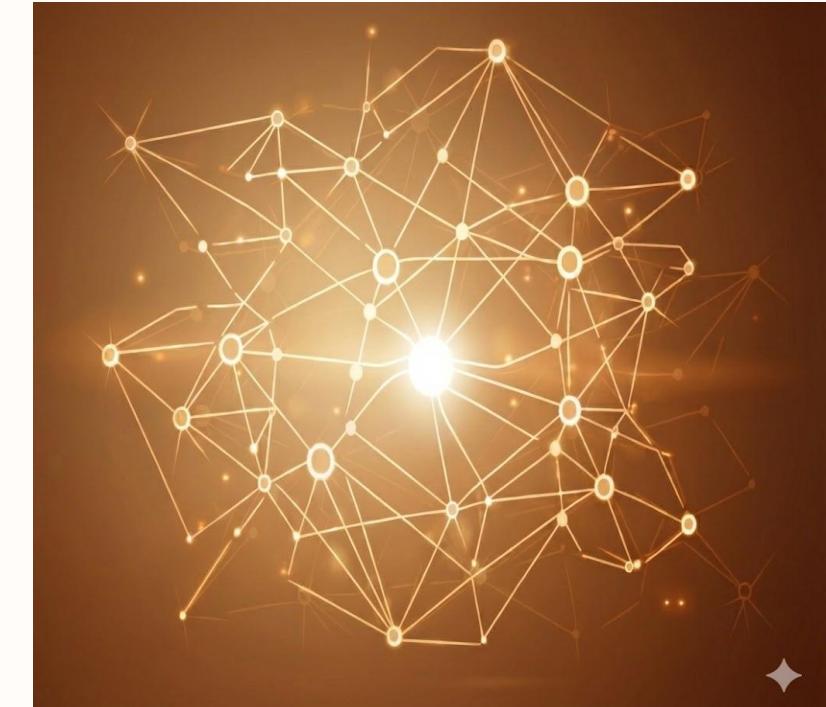
## Estructura de las Neuronas

Las neuronas son las unidades básicas del sistema nervioso, se componen de un soma o cuerpo celular, dendritas que reciben señales, y un axón que transmite impulsos a otras.



## Conexiones Sinápticas

Las conexiones entre neuronas, llamadas sinapsis, permiten la comunicación eléctrica y química entre ellas. En las sinapsis, las terminaciones del axón de una neurona liberan neurotransmisores que se unen a receptores en las dendritas de la neurona receptora. Este proceso desencadena o inhibe la propagación del impulso nervioso, permitiendo la transmisión de información entre neuronas.

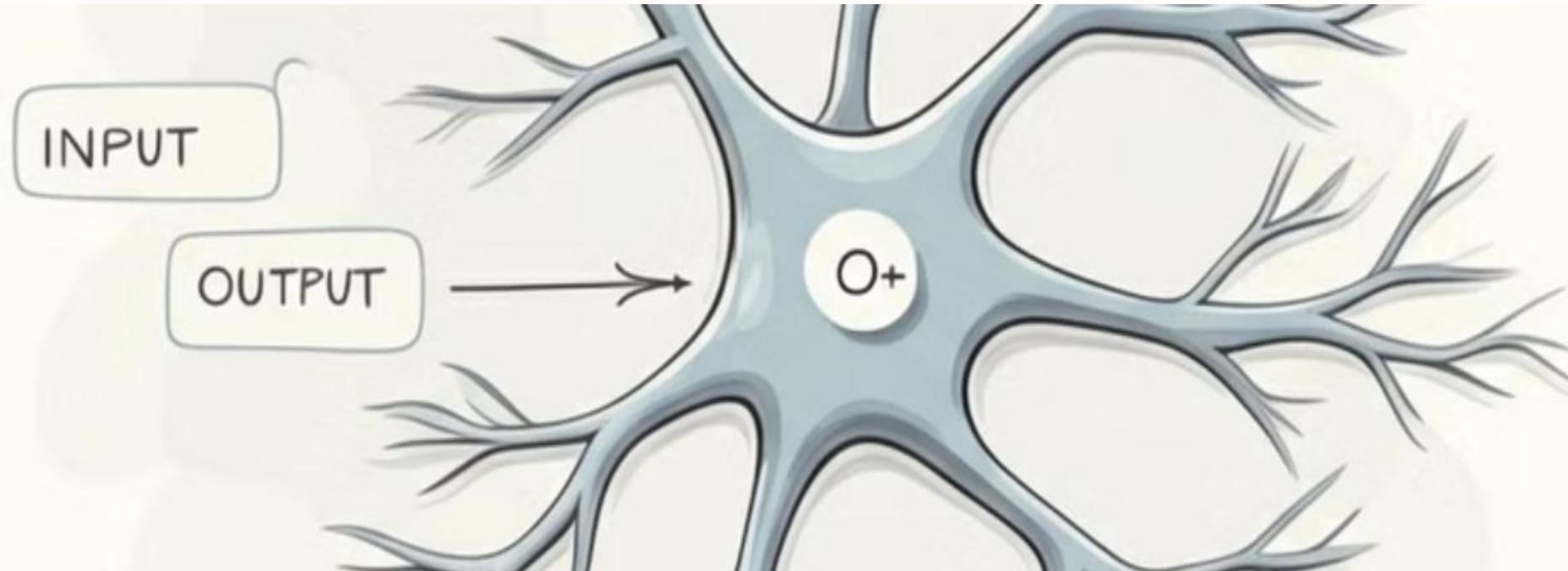


## Redes Neuronales

Las neuronas se organizan en complejas redes neuronales, donde cada una se conecta con miles de otras a través de sinapsis. Esta interconectividad permite procesos de aprendizaje, memoria y cognición en sistemas biológicos y artificiales inspirados en el cerebro.



# ¿Qué es una neurona artificial?



## Modelo computacional

Una neurona artificial es un modelo computacional inspirado en el funcionamiento de las neuronas biológicas del cerebro humano. Busca imitar la forma en que las neuronas procesan y transmiten información.



## Procesamiento de datos

Al igual que las neuronas biológicas, las neuronas artificiales reciben señales de entrada, las procesan y generan una señal de salida. Este proceso simula la capacidad de aprendizaje y adaptación del cerebro.

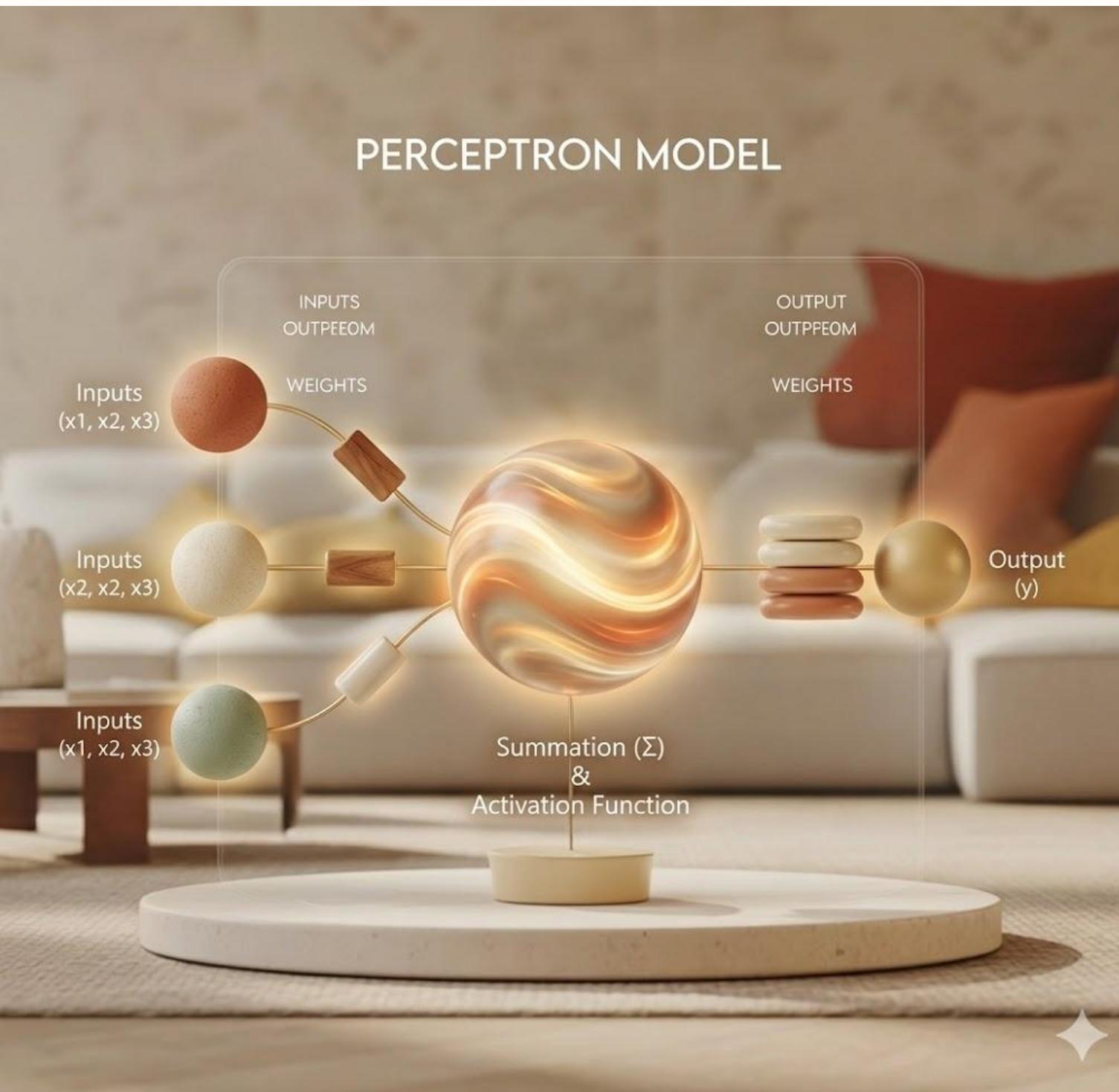


## Conexiones entre neuronas

Las neuronas artificiales se conectan entre sí formando una red neuronal, asemejándose a la estructura del cerebro. Estas conexiones permiten que la red aprenda y resuelva problemas complejos.



# El Perceptrón: La Unidad Fundamental



El perceptrón, propuesto por Frank Rosenblatt en 1958, es el bloque de construcción más básico de las redes neuronales. Aunque simple, su principio matemático sustenta todas las arquitecturas modernas.

## Limitaciones Históricas

El perceptrón simple solo puede resolver problemas linealmente separables (función XOR lo demostró imposible).

Esta limitación motivó el desarrollo de redes multicapa que pueden aprender patrones no lineales complejos.

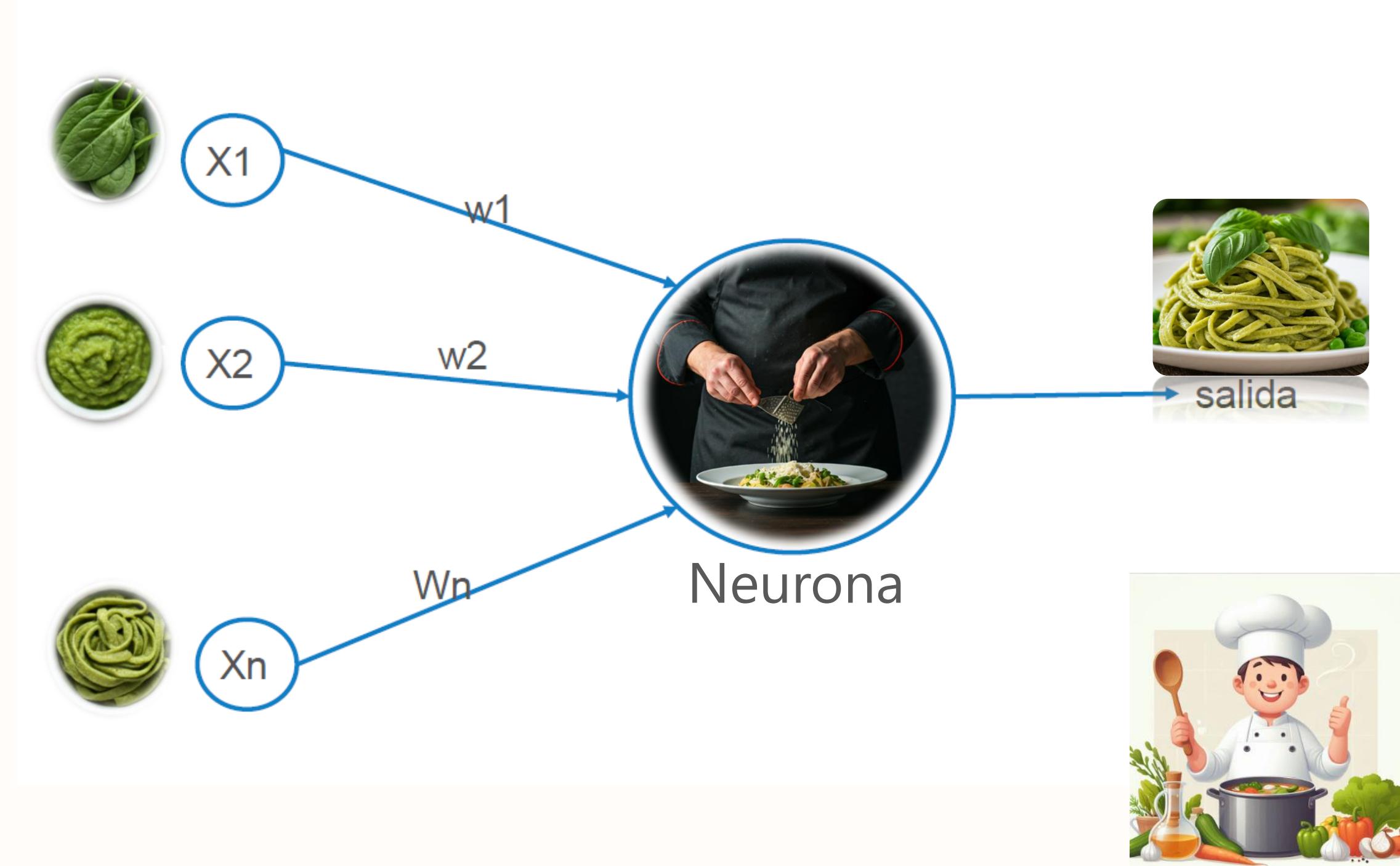
## Funcionamiento del Perceptrón

Recibe múltiples entradas numéricas, las multiplica por sus pesos correspondientes, suma los resultados junto con el sesgo, y aplica una función de activación para decidir si la neurona se "activa" o no.

Esta decisión binaria permite resolver problemas de clasificación lineal, sentando las bases para redes más complejas.



# ¿Qué es una neurona artificial?



$$\left\langle \begin{array}{c} 1^+ \\ m_x \end{array} \middle| \begin{array}{c} 3^- \\ -m_x \end{array} \right\rangle \equiv \left| \begin{array}{c} nai + 1 \\ n - i \end{array} \right\rangle \left| \begin{array}{c} -n_n \end{array} \right\rangle$$



## Estructura Matemática de una Neurona



### Entradas (x)

Vector de características o señales de entrada:  $x_1, x_2, \dots, x_n$

### Pesos (w)

Parámetros ajustables que determinan la importancia de cada entrada:  $w_1, w_2, \dots, w_n$



### Sesgo (b)

Término independiente que permite ajustar el umbral de activación de la neurona

### Salida

Resultado tras aplicar función de activación:  $y = f(\sum w_i x_i + b)$



# Conceptos básicos

## Inputs

- Datos de entrada

## Pesos

- **Ajuste de los pesos:** Durante el entrenamiento, los pesos se ajustan utilizando descenso del gradiente para minimizar el error.
- **Entrenamiento:** El ajuste de pesos se repite en iteraciones hasta que el modelo hace predicciones precisas.

## Función de Activación

- Determina cómo se transforma la salida de una neurona, introduciendo no linealidad en el modelo. Ejemplos: Sigmoide, ReLU, Softmax, entre otros.

## Nodo

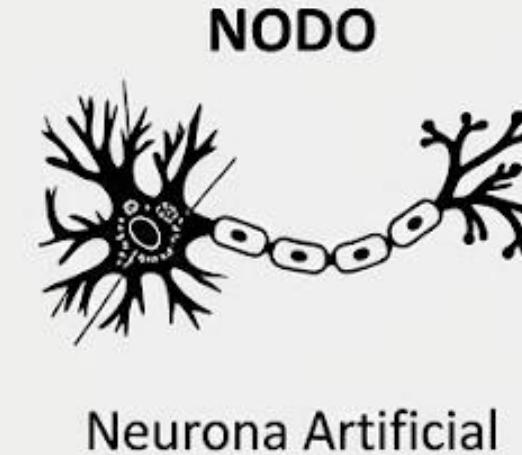
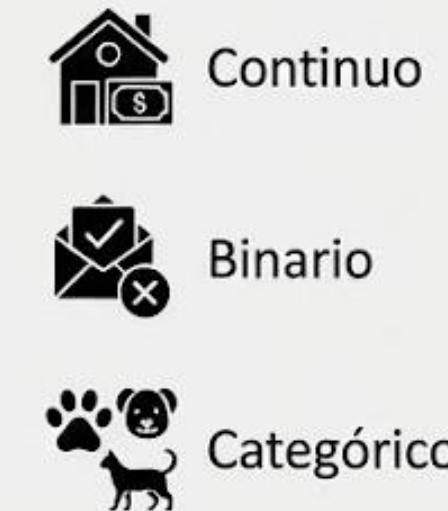
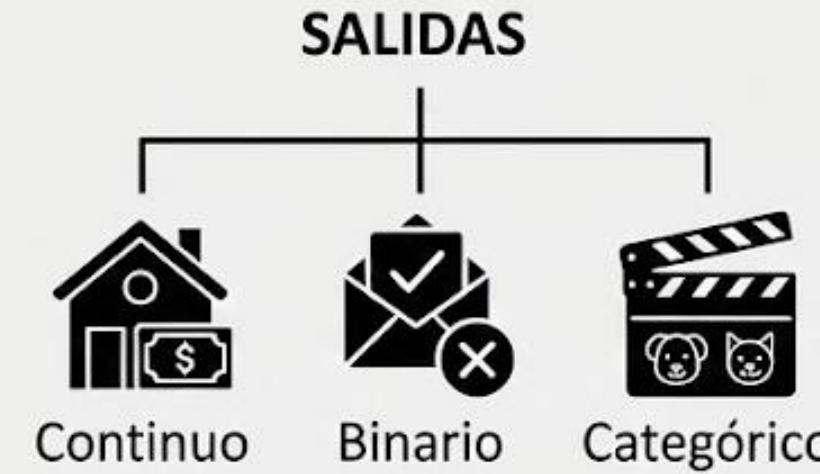
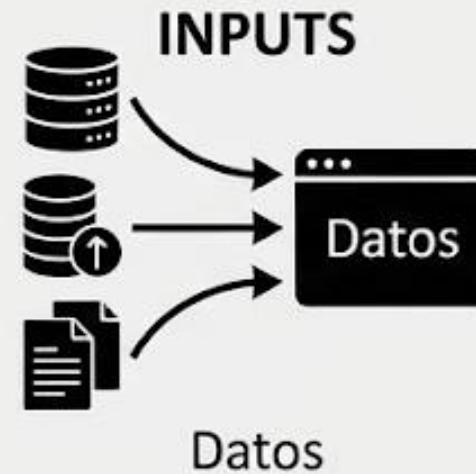
- Una unidad básica de procesamiento en una red neuronal artificial, conocido como neurona artificial.

## Valores de salida

- **Continuos:** Predecir el precio de una casa, la temperatura de una ciudad, o el ingreso anual de una persona.
- **Binarios:** Predecir si un correo es spam o no, si una persona será contratada o no, o si un cliente comprará o no.
- **Categóricos:** Género de una película (acción, comedia, drama), especie de un animal (perro, gato, loro).



# Conceptos básicos

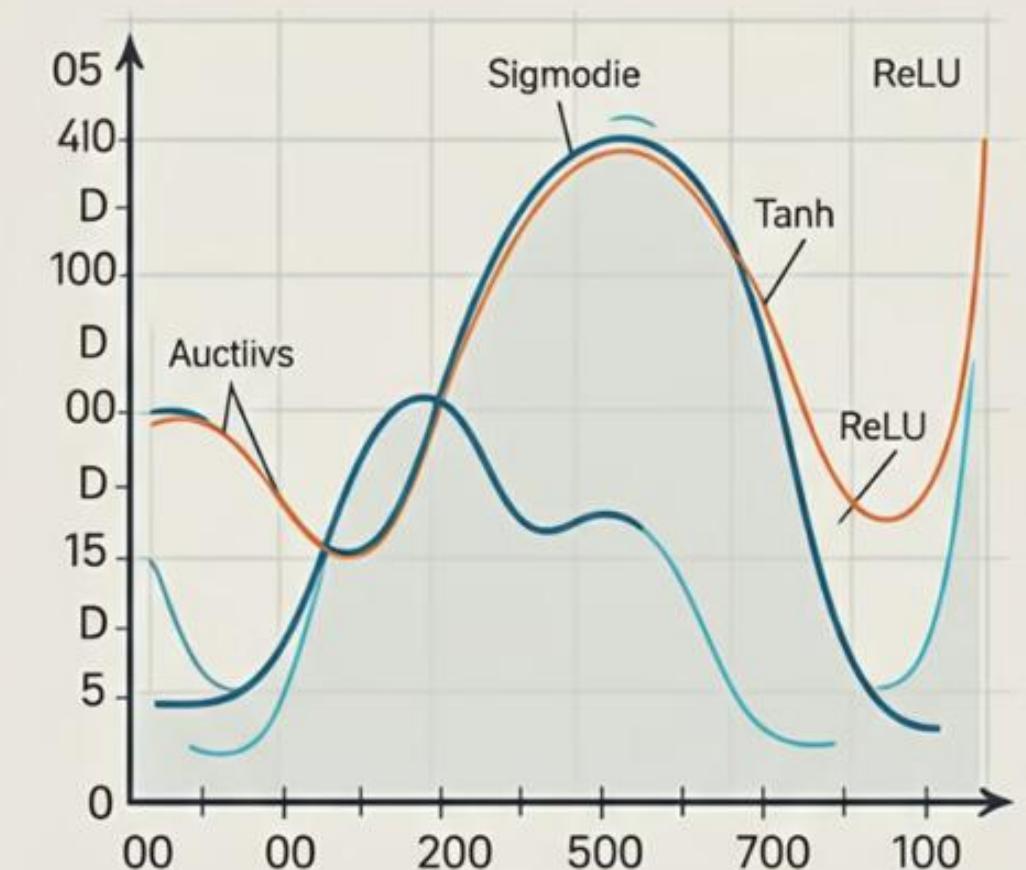


Neurona Artificial



# Conceptos básicos - Función de Activación

- Determina cómo se transforma la salida de una neurona, introduciendo no linealidad en el modelo. Ejemplos: Sigmoide, ReLU, Softmax, entre otros.
- Nos facilita transformar una salida lineal en algo ni lineal y así sucesivamente. Así se puede identificar mejor el problema.
- Es la forma de comunicar la mezcla de pesos y valores iniciales para la salida.





## Suma Ponderada

El corazón computacional de cada neurona es una operación de álgebra lineal que combina todas las entradas según su importancia relativa.

$$z = \sum_{i=1}^n w_i x_i + b = w_1 x_1 + w_2 x_2 + \dots + w_n x_n + b$$

Donde:

- $z$  es la suma ponderada (activación pre-función)
- $x_i$  representa cada valor de entrada
- $w_i$  son los pesos que amplifican o reducen cada entrada
- $b$  es el sesgo que desplaza el umbral de decisión
- $n$  es el número total de entradas a la neurona

Finalmente, se aplica la función de activación:  $y = f(z)$



# Funciones de Activación: Introduciendo No Linealidad

## ¿Por qué son vitales?

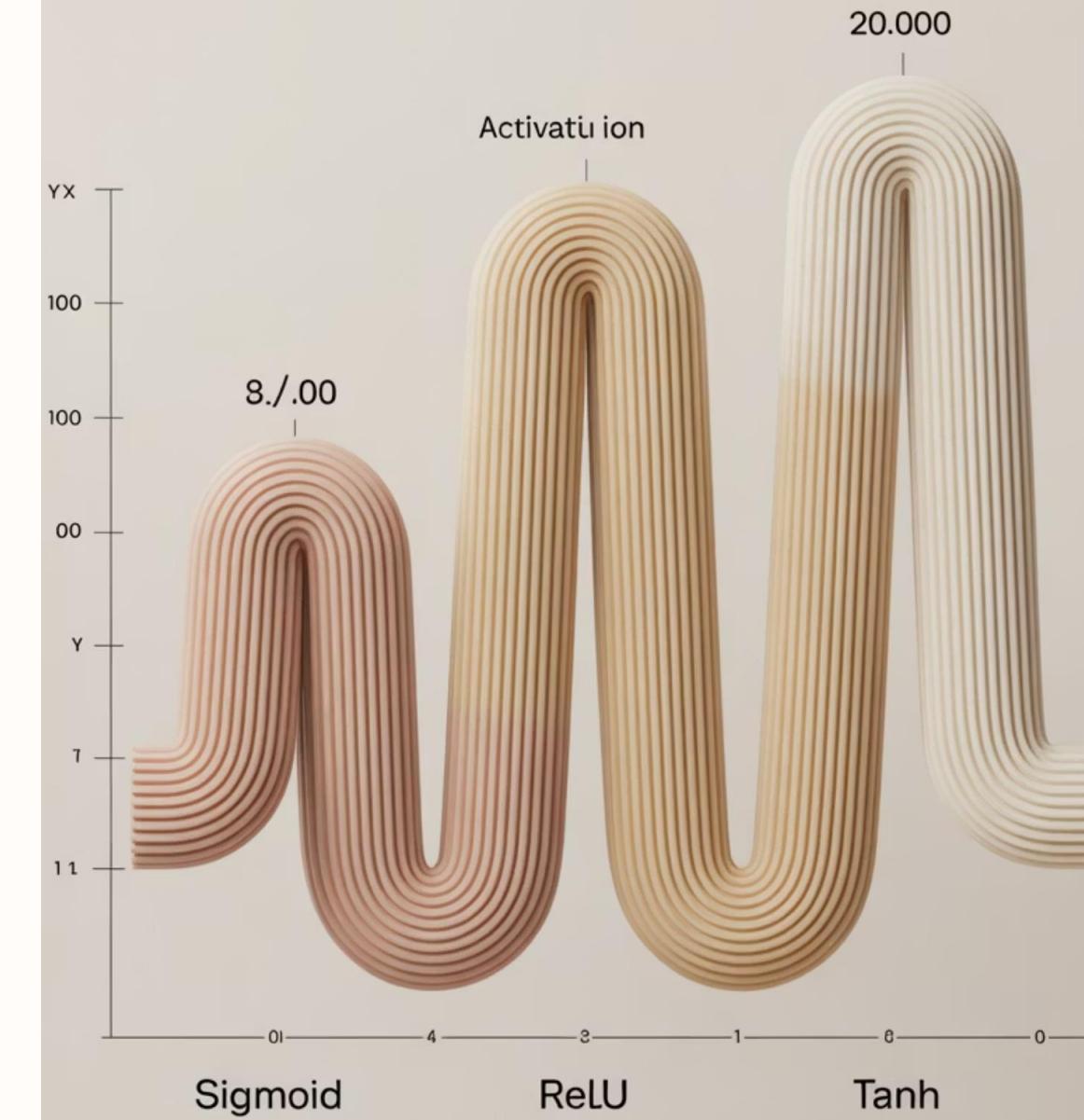
Permiten que la red aprenda relaciones complejas y no lineales entre entrada y salida, como reconocer rostros o predecir comportamientos.

## Efecto de Composición

Apilar capas con funciones de activación crea transformaciones progresivamente más abstractas de los datos originales.

## Diferenciabilidad

Deben ser derivables para permitir el cálculo de gradientes durante backpropagation, esencial para el aprendizaje.





# Función Sísmoide: La Clásica

## Fórmula Matemática

$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

Comprime cualquier valor de entrada al rango (0, 1), interpretable como probabilidad.

## Características

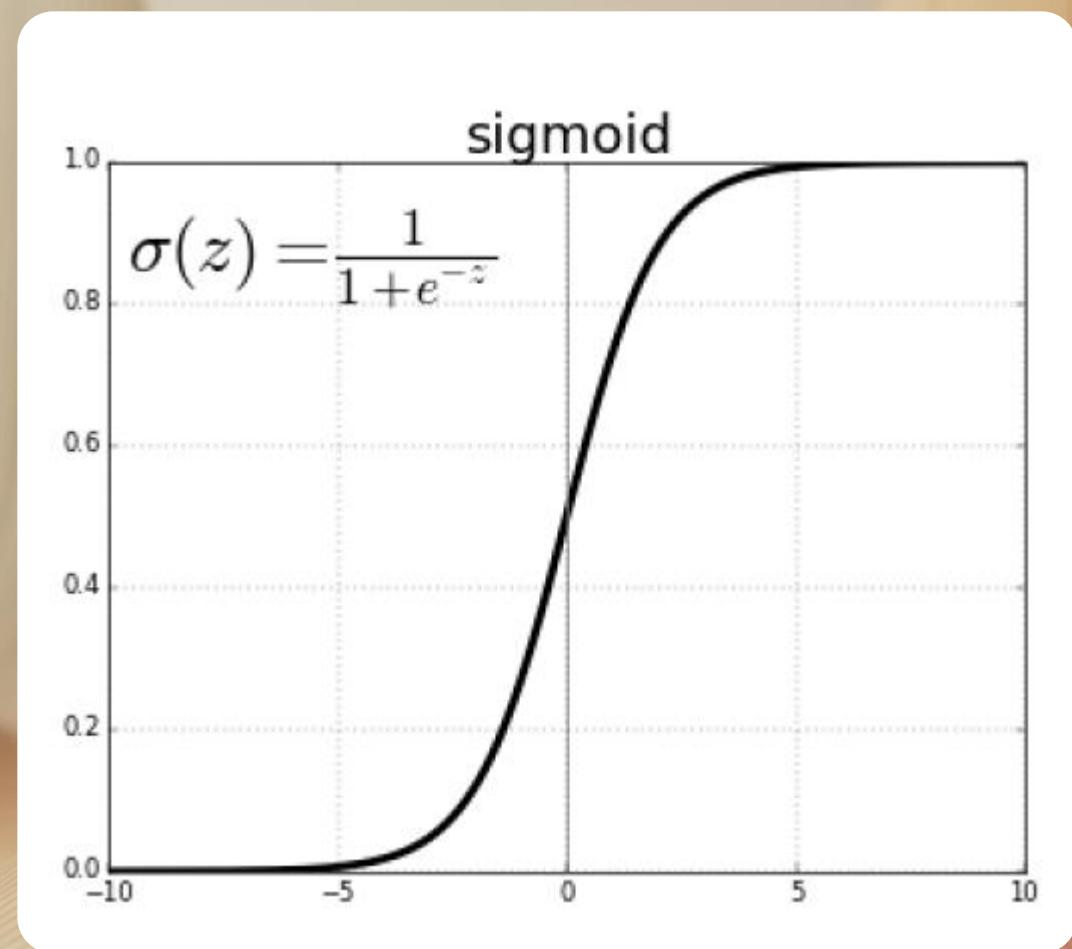
- Salida continua entre 0 y 1
- Forma de "S" suave y diferenciable
- Centro en 0.5 cuando  $z=0$
- Asintótica en los extremos

## Aplicaciones Principales

Ideal para clasificación binaria en la capa de salida, donde necesitamos interpretar el resultado como probabilidad de pertenencia a una clase.

## Desventajas

Sufre del problema del "gradiente desvaneciente" en redes profundas: cuando  $z$  es muy grande o pequeño, la derivada se acerca a cero, frenando el aprendizaje.



# ReLU: El Estándar Moderno



La función Rectified Linear Unit (ReLU) revolucionó el deep learning al resolver problemas de gradientes desvanecientes y acelerar significativamente el entrenamiento.

## Definición Matemática

$$f(z) = \max(0, z) = \begin{cases} z & \text{si } z > 0 \\ 0 & \text{si } z \leq 0 \end{cases}$$

Extremadamente simple: devuelve el valor de entrada si es positivo, cero en caso contrario.

## Ventajas Clave

- **Eficiencia computacional:** Solo requiere una comparación
- **Gradientes estables:** La derivada es 1 para valores positivos
- **Esparsidad:** Muchas neuronas se "apagan" (output = 0)
- **Convergencia rápida:** Entrenamientos 6x más veloces que sigmoide

## Variantes Populares

**Leaky ReLU:**  $f(z) = \max(0.01z, z)$  permite pequeños gradientes negativos

**ELU:** Exponencial para valores negativos, más suave

**Swish:**  $f(z) = z \cdot \sigma(z)$ , usado en arquitecturas estado del arte

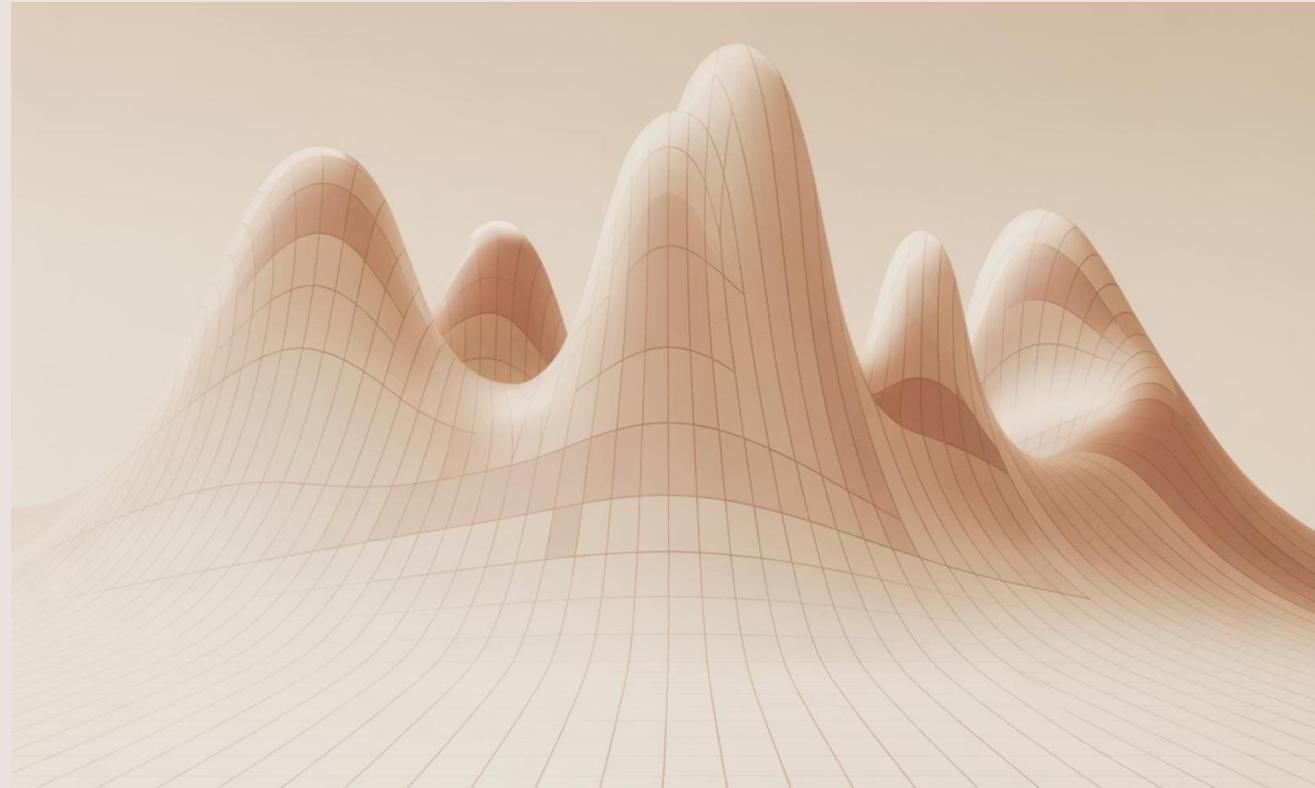


# Otras Funciones de Activación Importantes

## Tanh (Tangente Hiperbólica)

$$\tanh(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$$

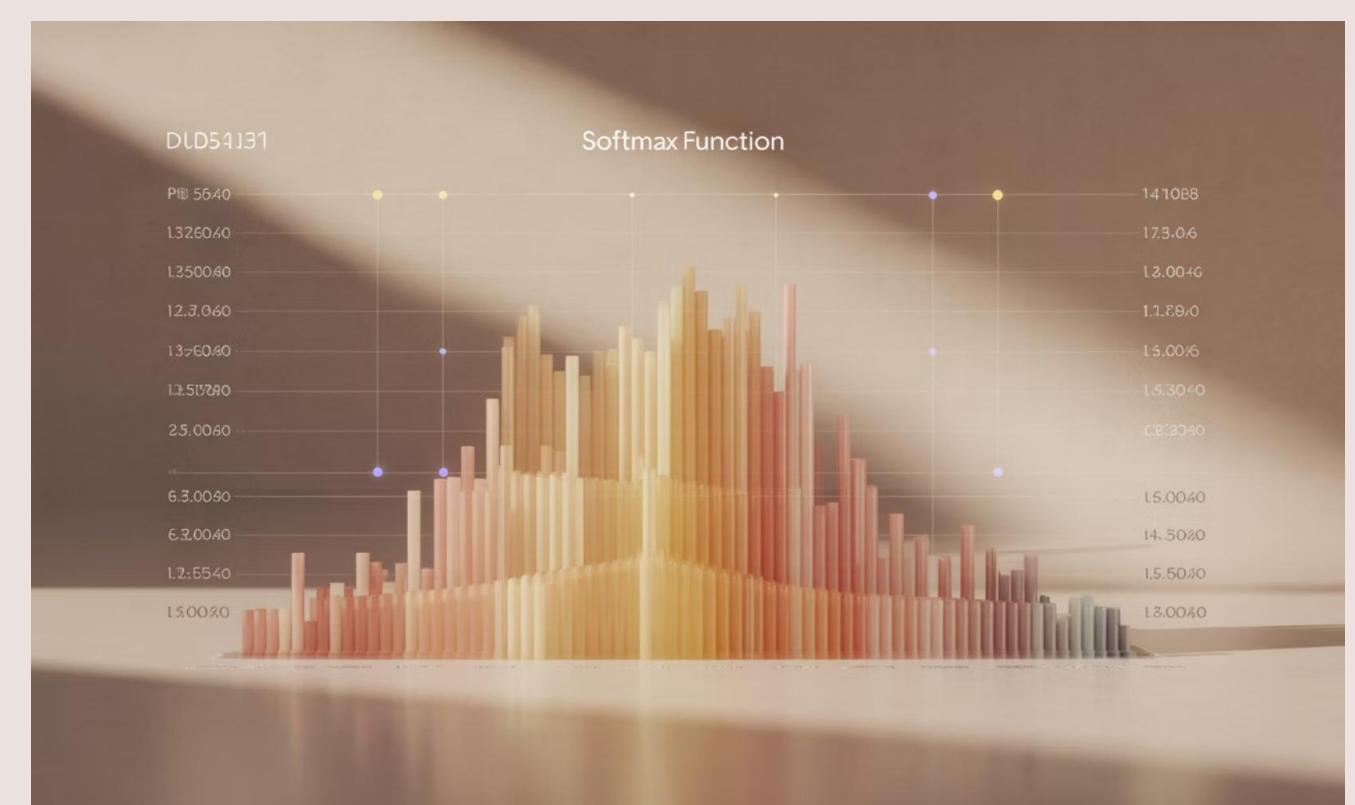
Similar a sigmoide pero centrada en cero, con rango (-1, 1). Útil cuando necesitamos salidas negativas y positivas simétricas.



## Softmax (Multiclasificación)

$$\text{softmax}(z_i) = \frac{e^{z_i}}{\sum_{j=1}^K e^{z_j}}$$

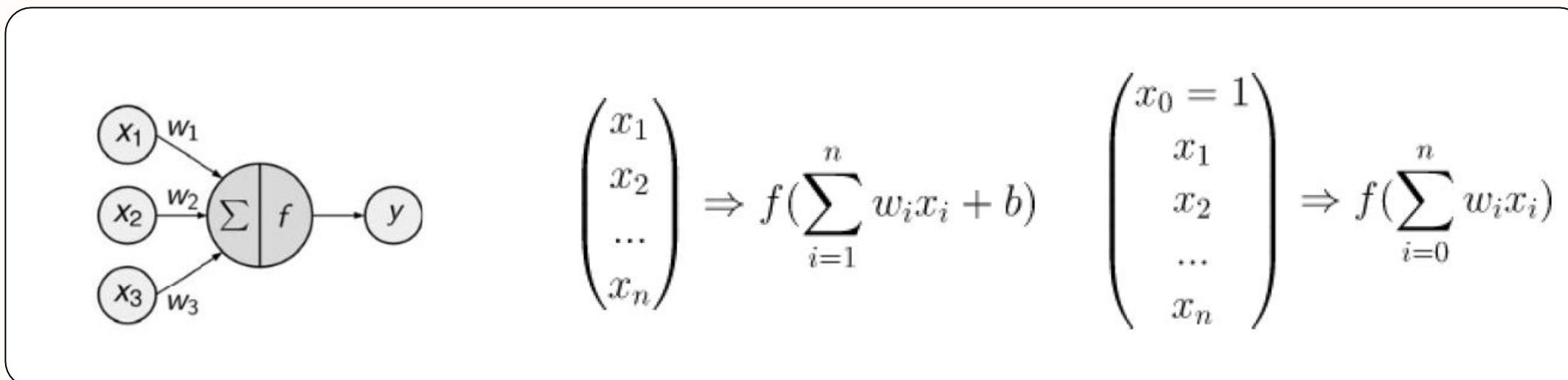
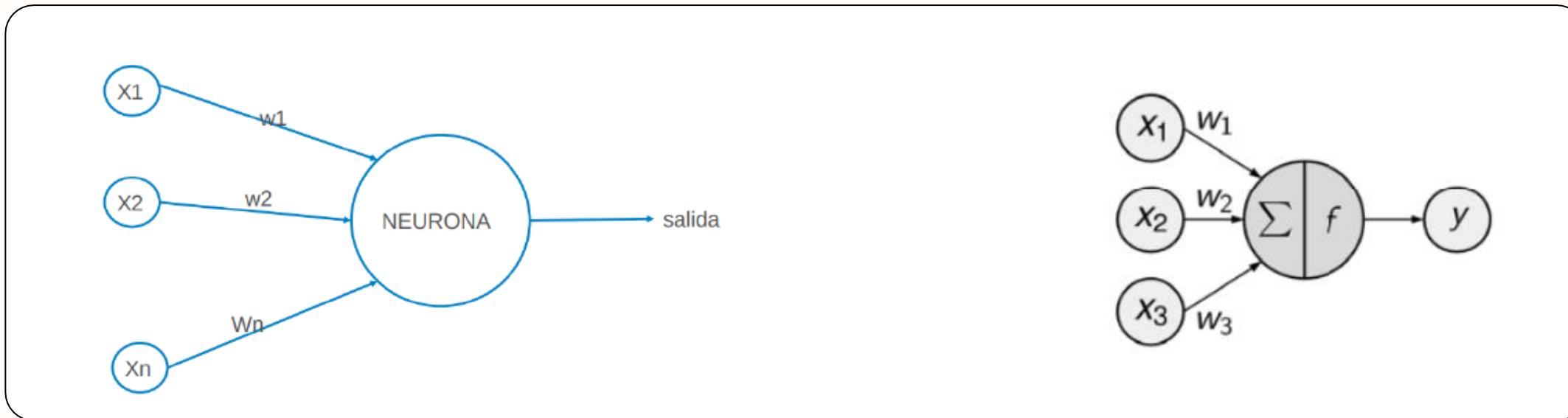
Convierte un vector de valores en una distribución de probabilidad. Esencial para clasificación con múltiples categorías mutuamente excluyentes.



La elección de la función de activación depende del problema: ReLU para capas ocultas generales, Softmax para clasificación multiclasificación, Sigmoide para clasificación binaria.

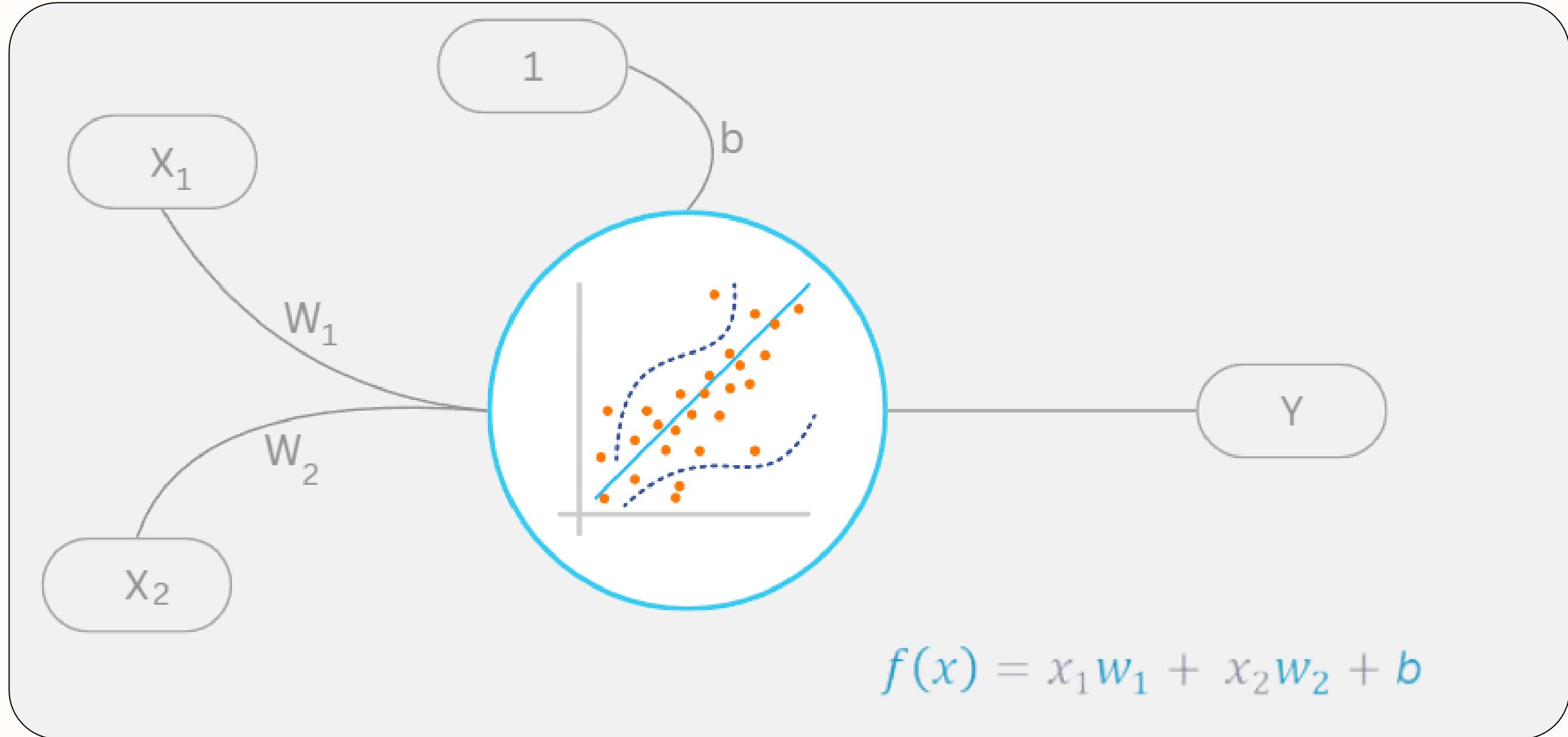


# Neurona Artificial





# Neurona Artificial





# Funciones de activación

Sigmoid



$$y = \frac{1}{1+e^{-x}}$$

Tanh



$$y = \tanh(x)$$

Step Function



$$y = \begin{cases} 0, & x < n \\ 1, & x \geq n \end{cases}$$

Softplus



$$y = \ln(1+e^x)$$

ReLU



$$y = \begin{cases} 0, & x < 0 \\ x, & x \geq 0 \end{cases}$$

Softsign



$$y = \frac{x}{(1+|x|)}$$

ELU



$$y = \begin{cases} \alpha(e^x - 1), & x < 0 \\ x, & x \geq 0 \end{cases}$$

Log of Sigmoid



$$y = \ln\left(\frac{1}{1+e^{-x}}\right)$$

Swish



$$y = \frac{x}{1+e^{-x}}$$

Sinc



$$y = \frac{\sin(x)}{x}$$

Leaky ReLU



$$y = \max(0.1x, x)$$

Mish



$$y = x(\tanh(\text{softplus}(x)))$$

# Redes Neuronales Artificiales

- Las RNA son sistemas de cómputo que imitan la estructura y el funcionamiento del cerebro.
  - Compuestas por nodos interconectados, similares a las neuronas que procesan y adaptan información a través del aprendizaje.
  - Las RNA aprenden patrones complejos sin necesidad de programación explícita.
  - Se usan en reconocimiento de imágenes, procesamiento del lenguaje natural y detección de fraudes, entre otros.
  - Ventajas: Alta capacidad para modelar relaciones no lineales y adaptabilidad a distintos problemas.
  - Desafíos: Necesitan muchos datos y recursos computacionales elevados para un entrenamiento efectivo.





# Topologías de Redes Neuronales



## Arquitectura de la Red

La topología de una red neuronal se refiere a la estructura y organización de las neuronas y sus conexiones dentro de la red.



## Conexiones entre Neuronas

Las neuronas se organizan en capas y se conectan entre sí a través de pesos sinápticos que transmiten la información.



## Capas de la Red

Las redes neuronales típicamente se componen de una capa de entrada, una o más capas ocultas, y una capa de salida.



## Dirección de la Información

La información puede fluir en una sola dirección (feedforward) o en múltiples direcciones (recurrente) dentro de la red neuronal.

# Topologías de Redes Neuronales



## Redes Simples (Shallow)

Una o dos capas ocultas con decenas de neuronas. Rápidas de entrenar y suficientes para muchos problemas tabulares.

- Entrenamiento rápido (minutos)
- Menor riesgo de sobreajuste
- Interpretabilidad más sencilla
- Limitadas en problemas complejos

## Redes Profundas (Deep)

Múltiples capas ocultas (10-200+) con cientos o miles de neuronas. Poder representacional masivo para datos complejos.

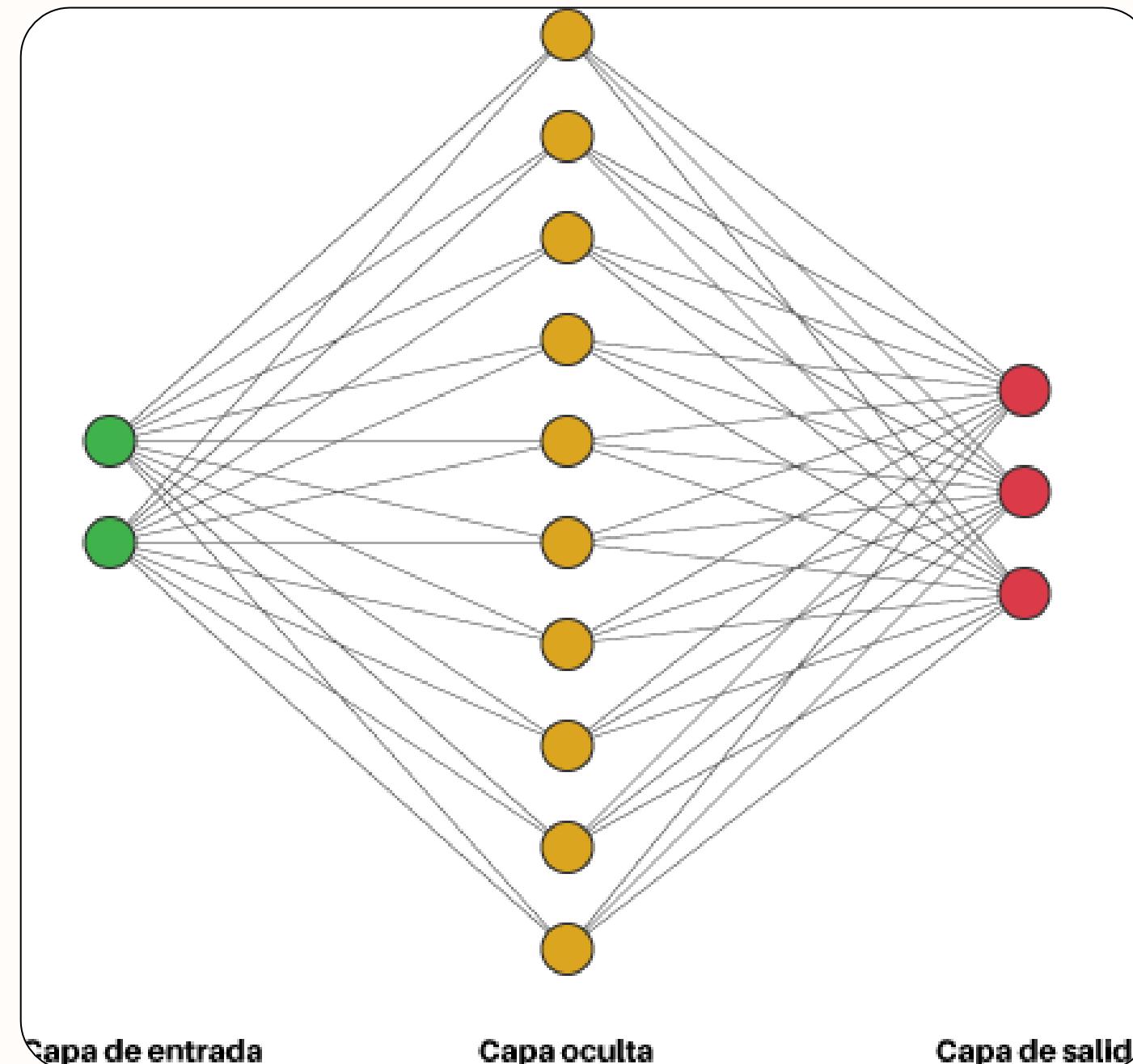
- Aprenden características jerárquicas
- Estado del arte en visión, NLP, audio
- Requieren grandes datasets
- Alto costo computacional (GPUs)

La elección depende de: cantidad de datos disponibles, complejidad del problema, recursos computacionales y tiempo de desarrollo.





# Redes neuronales simples

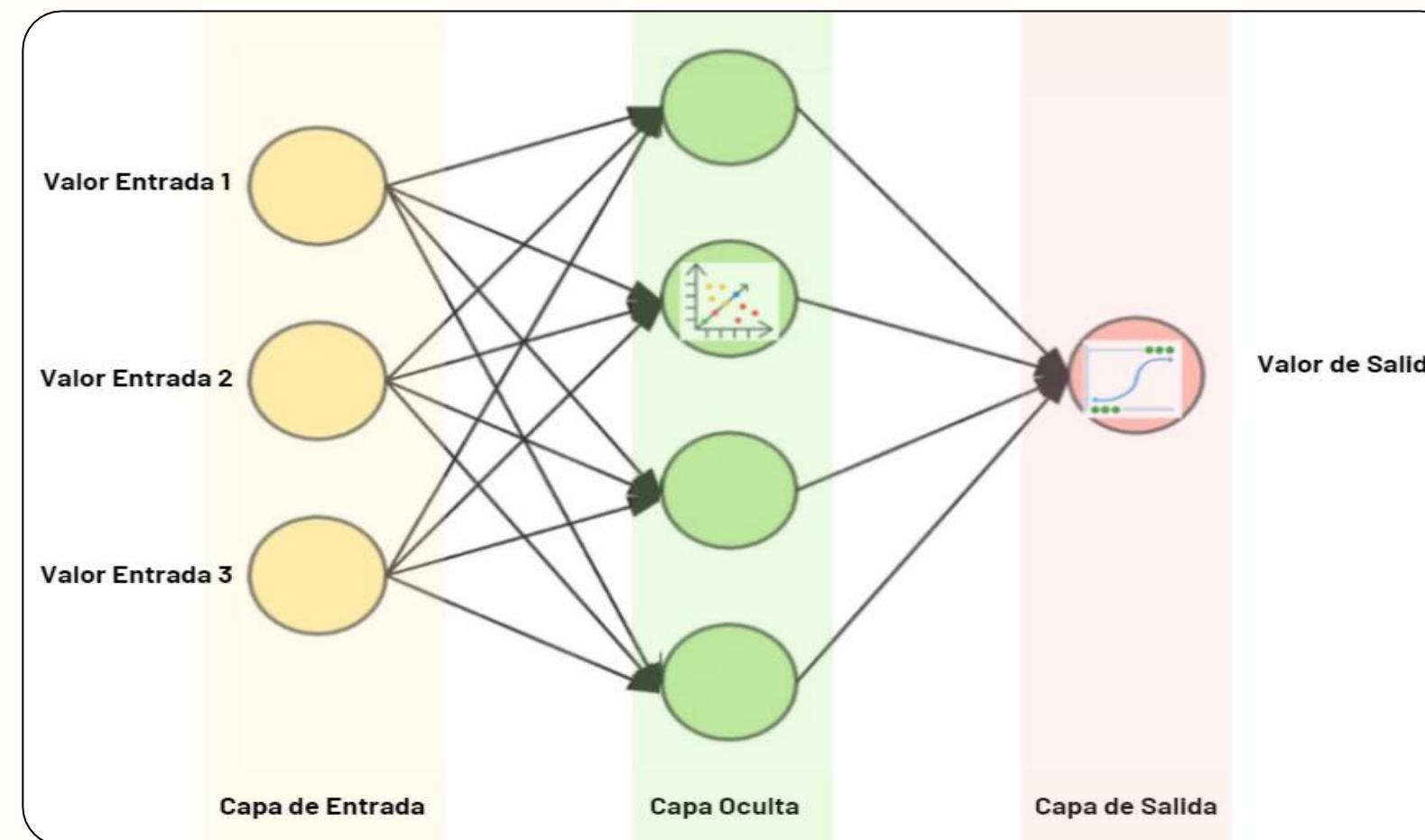


Fuente de la imagen: <https://codificandobits.com/blog/que-es-una-red-neuronal/>



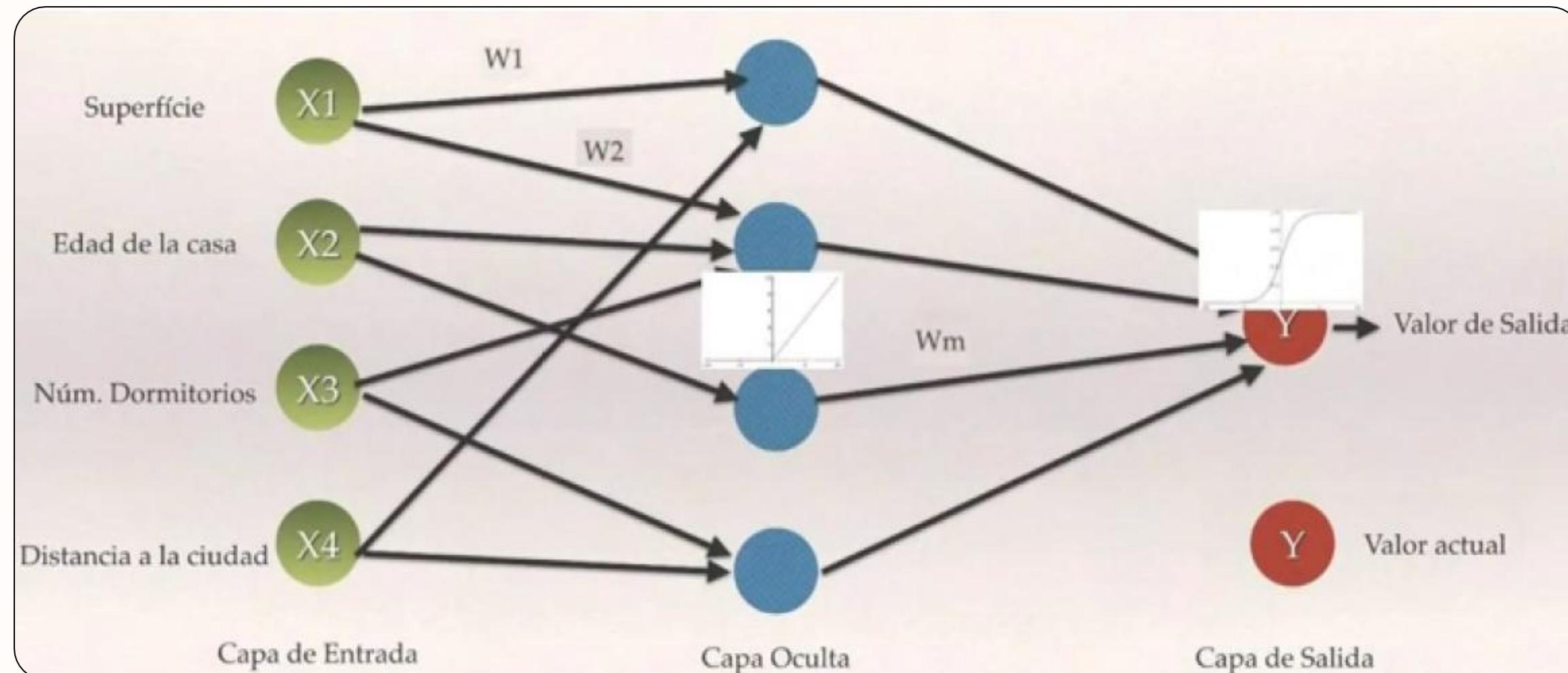
# Redes neuronales simples

- **La topología de la red** representa la cantidad de neuronas en el modelo, además de cómo están interconectadas (arquitectura).
- **La función de activación** convierte los datos introducidos en la señal que se propaga a través de la red.
- **El algoritmo de aprendizaje** que se utiliza para determinar los pesos.





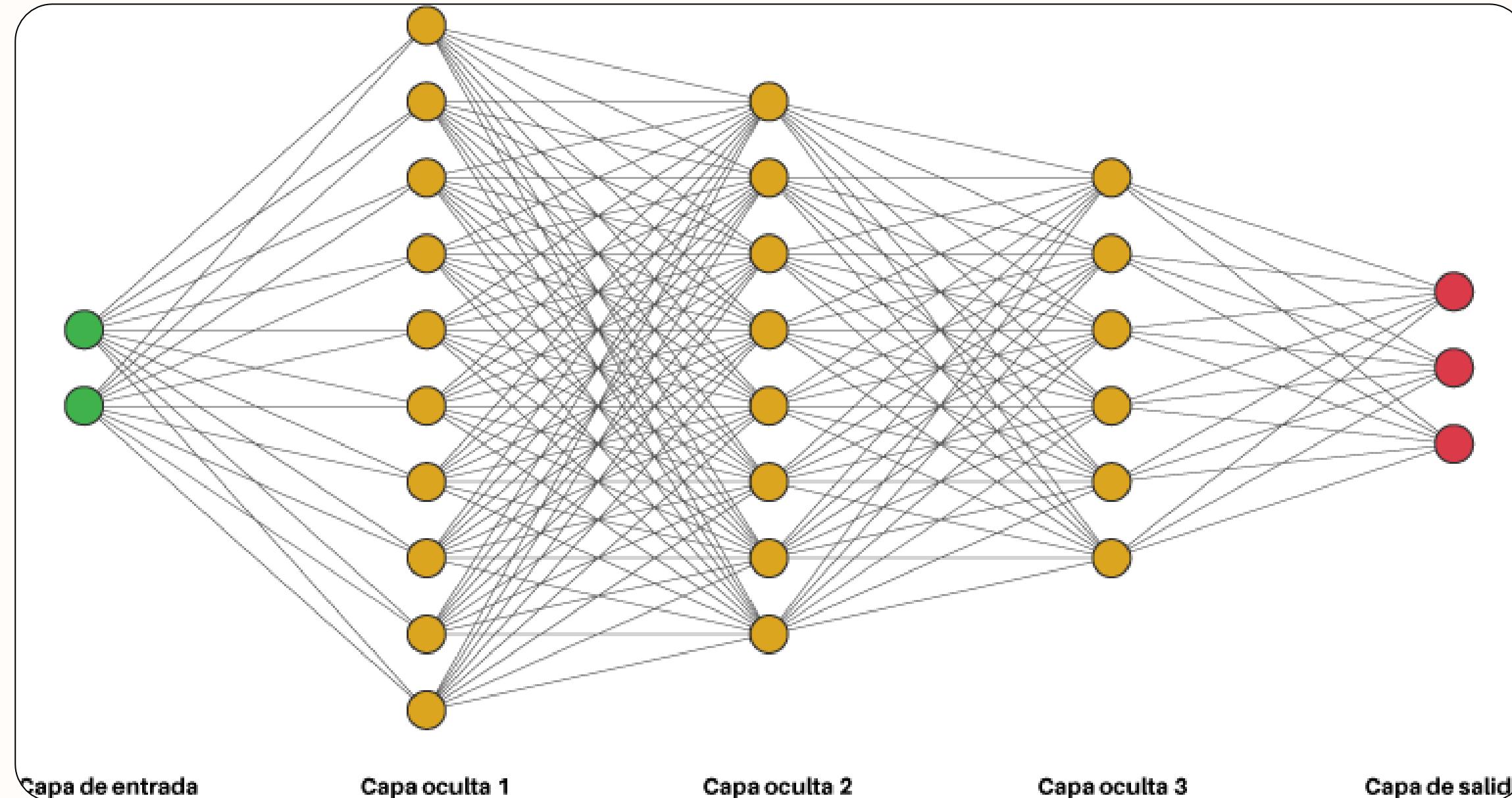
# Redes neuronales simples



Fuente de imagen: [https://github.com/jmartinezheras/2018-machinelearning-lectures-esa/tree/master/iartificial\\_net#Redes\\_neuronales](https://github.com/jmartinezheras/2018-machinelearning-lectures-esa/tree/master/iartificial_net#Redes_neuronales)



# Redes neuronales profundas

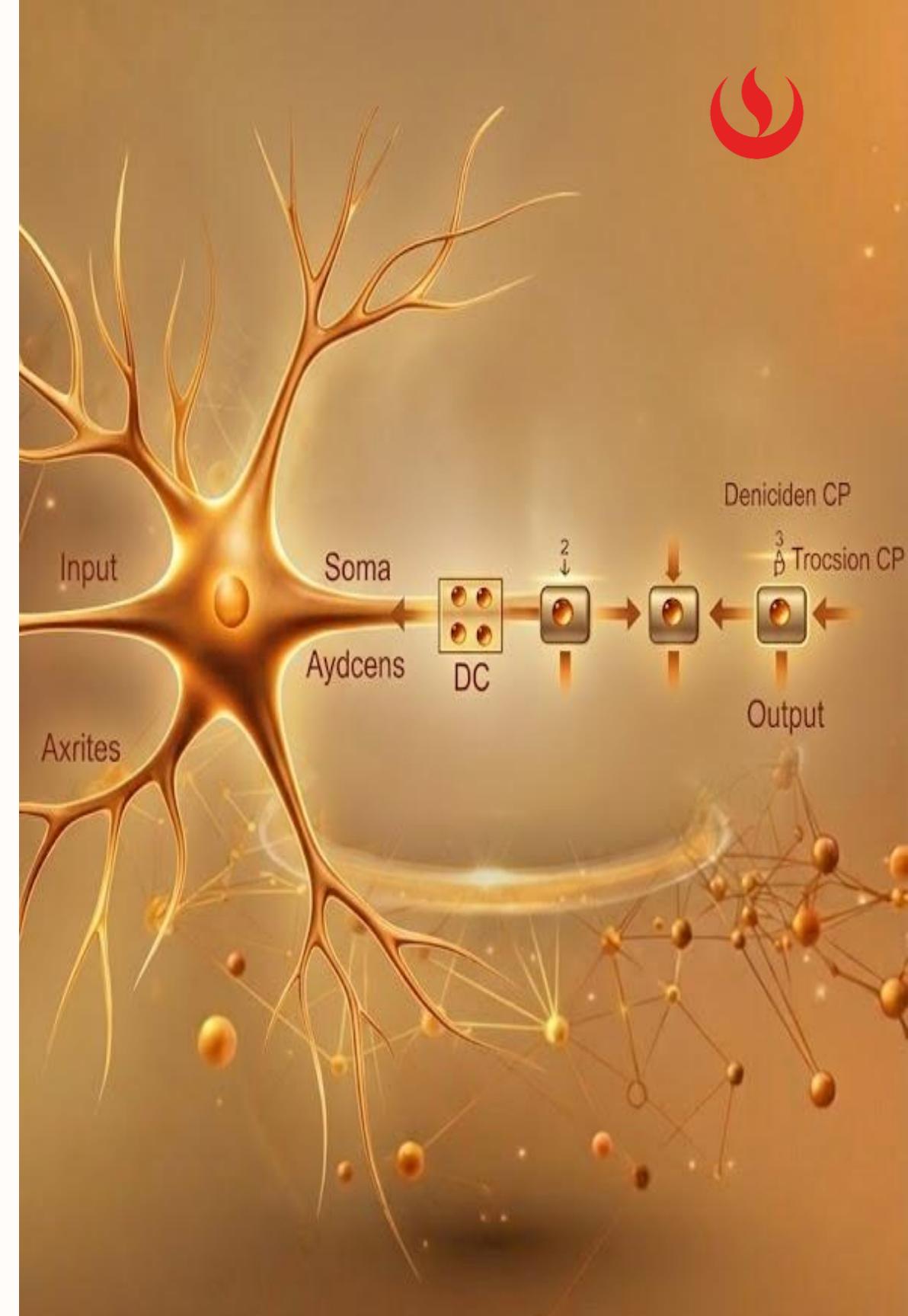


Fuente de la imagen: <https://codificandobits.com/blog/que-es-una-red-neuronal/>



# Cantidad de nodos por capa

- La cantidad de nodos de entrada se determina por la cantidad de características presentes en los datos de entrada.
- La cantidad de nodos de salida se establece por la cantidad de salidas a modelar o el número de niveles de la clase.
- No existe una norma que establezca el número de neuronas en cada capa.
- Por lo general, redes más sofisticadas con más conexiones facilitan el aprendizaje de problemas de mayor complejidad.
- La norma práctica consiste en iniciar con escasas neuronas y gradualmente incrementar su número.





# Algoritmo de propagación



- En una red neuronal, cada enlace entre las neuronas posee un peso vinculado. Estos pesos establecen el impacto de una neurona en otra y, en su totalidad, establecen la manera en que la red procesa los datos de entrada para conseguir una salida.
- El propósito del entrenamiento es modificar estos pesos de tal manera que la salida de la red (predicción) se alinee al máximo con la salida real (esperada) para una serie de ejemplos ya conocidos.



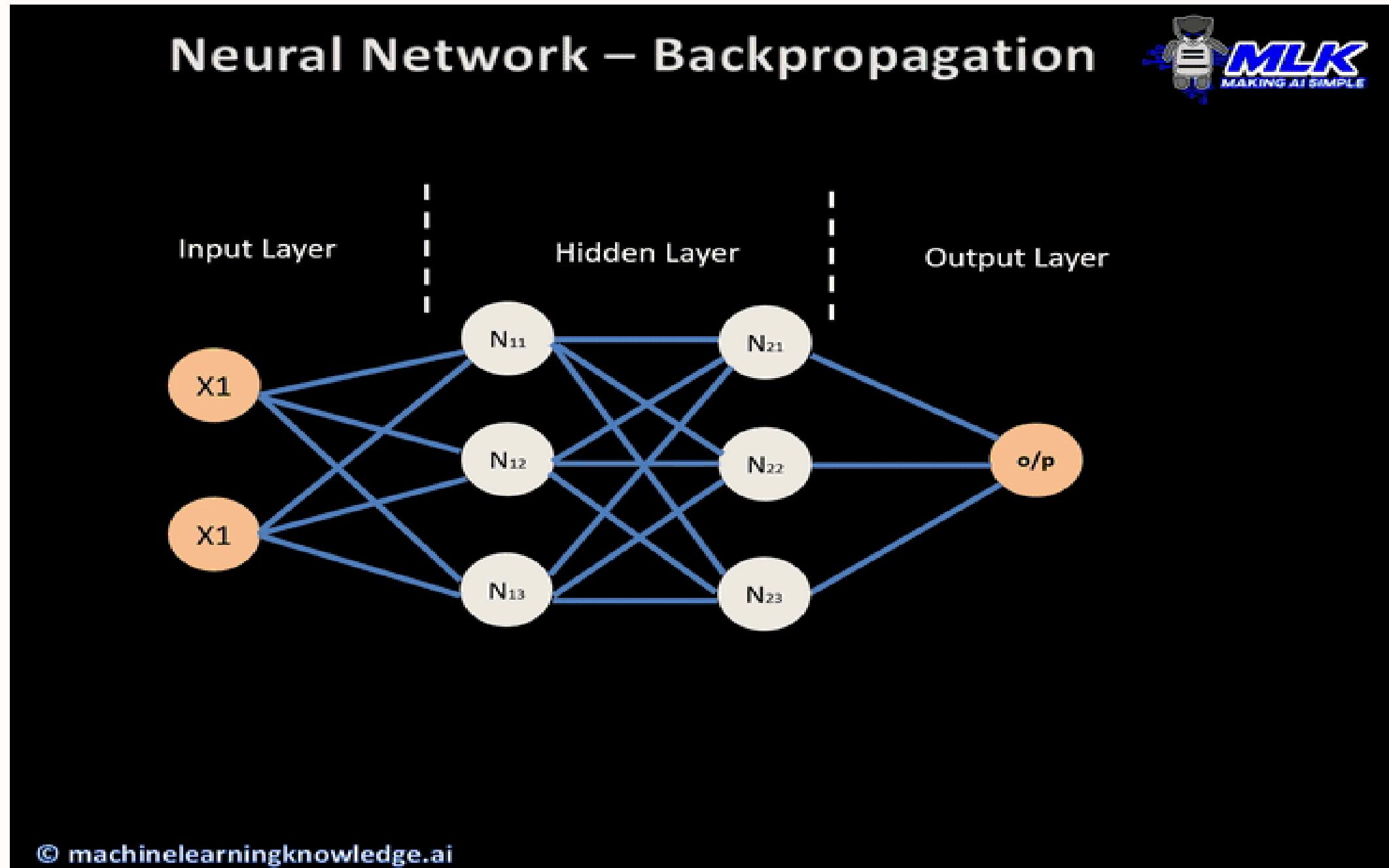
# Backpropagation

- Las redes neuronales no se planifican, sino que se capacitan.
- Debemos determinar el ajuste de pesos que ofrece resultados positivos para la red.
- El algoritmo backpropagation o de propagación inversa del gradiente es el más común.





# Backpropagation



Fuente: [https://images.datacamp.com/image/upload/v1703680173/image\\_8ee636b259.gif](https://images.datacamp.com/image/upload/v1703680173/image_8ee636b259.gif)

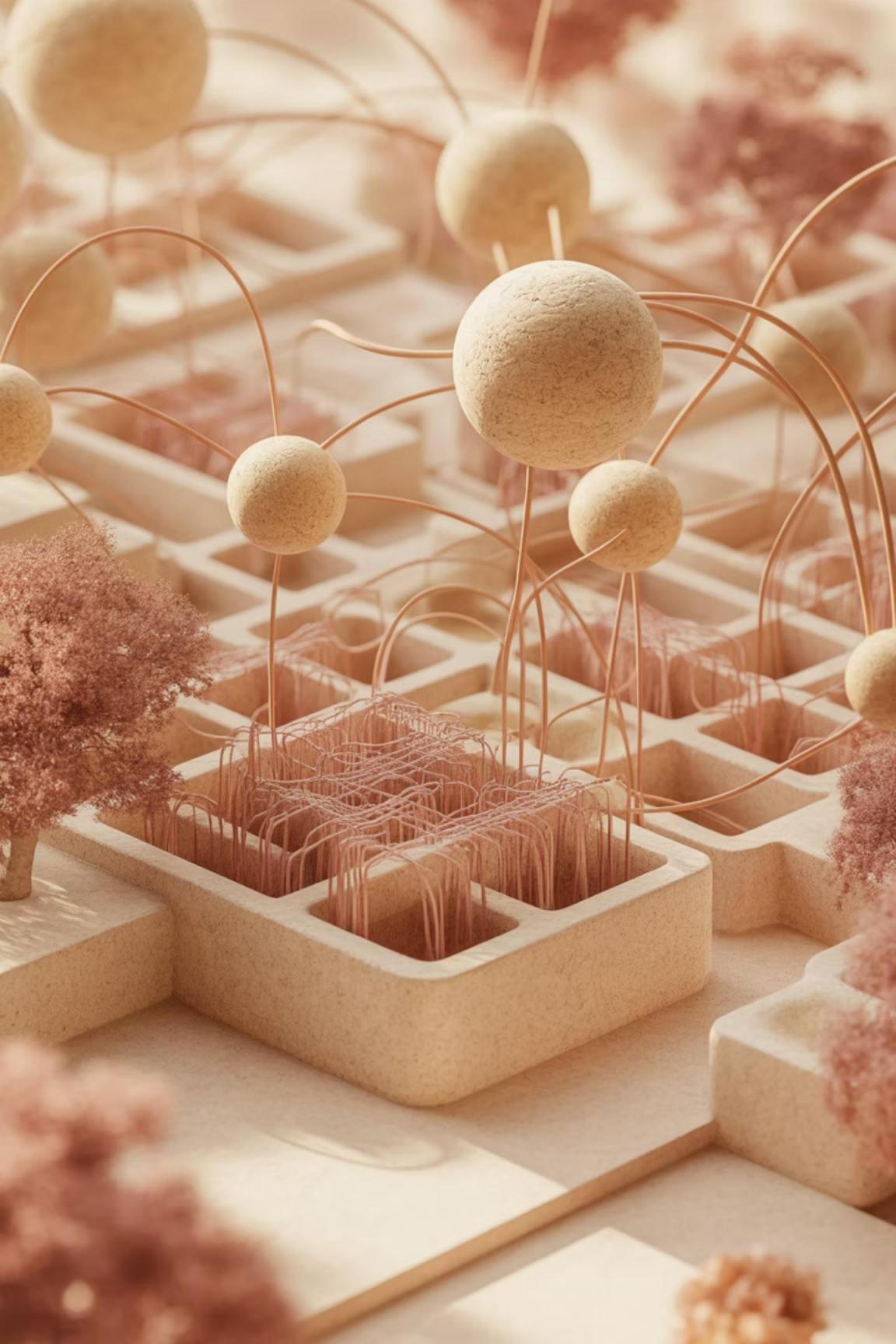




## TEMARIO

- 
- A photograph of a modern building with large glass windows and a dark facade. The sky is dark, suggesting it's nighttime. A few people are visible near the entrance.
- 1 — Aprendizaje supervisado: regresión y clasificación con redes neuronales.
  - 2 — Aprendizaje no supervisado: agrupamiento.





# Aprendizaje Supervisado: Clasificación

## Categorización de Datos

A diferencia de regresión, clasificación predice categorías discretas. La salida no es un número continuo, sino una etiqueta de clase.

### Clasificación Binaria

Dos clases mutuamente excluyentes: Spam vs No Spam, Fraude vs Legítimo, Enfermo vs Sano, Aprobado vs Rechazado.

**Salida:** 1 neurona con Sigmoide → probabilidad entre 0 y 1

### Clasificación Multiclas

Múltiples categorías donde cada dato pertenece a exactamente una: reconocimiento de dígitos (0-9), especies de plantas, tipos de clientes.

**Salida:** K neuronas con Softmax → distribución de probabilidad sobre K clases

### Clasificación Multilabel

Un dato puede pertenecer simultáneamente a varias categorías: etiquetas de películas (acción, comedia, drama), diagnósticos médicos múltiples.

**Salida:** K neuronas independientes con Sigmoide



## ¿Qué Hacemos Sin Etiquetas?

Hasta ahora hemos trabajado con aprendizaje supervisado: teníamos pares  $(X, Y)$  donde  $Y$  es la respuesta correcta. Pero la mayoría de datos en el mundo real no vienen etiquetados.

### El Desafío

Etiquetar datos es costoso, lento y a veces imposible:

- Segmentar millones de clientes manualmente: impracticable
- Identificar patrones desconocidos: ¿cómo etiquetar lo que no sabemos que existe?
- Datos médicos raros: no hay suficientes ejemplos etiquetados
- Detección de anomalías: los problemas nuevos no tienen precedente

### La Oportunidad

Aprendizaje no supervisado descubre estructura sin guía:

- **Clustering:** Agrupar datos similares automáticamente
- **Reducción de dimensionalidad:** Encontrar representaciones compactas (PCA, t-SNE)
- **Detección de anomalías:** Identificar outliers o casos raros
- **Sistemas de recomendación:** Encontrar usuarios o productos similares



# Aprendizaje no supervisado

El aprendizaje no supervisado es una técnica de aprendizaje automático que se utiliza para encontrar patrones ocultos y estructuras en datos sin etiquetar.

A diferencia del aprendizaje supervisado, donde se proporcionan ejemplos etiquetados para entrenar un modelo, en el aprendizaje no supervisado el objetivo es descubrir por sí mismo la estructura inherente de los datos.



# Clustering: Encontrar Estructura en el Caos



Clustering es el proceso de agrupar datos en conjuntos (clusters) donde miembros del mismo grupo son más similares entre sí que con miembros de otros grupos.

## Objetivo



Maximizar similitud intra-cluster (dentro del grupo) y minimizar similitud inter-cluster (entre grupos diferentes). Sin conocer las categorías de antemano.

## Aplicaciones

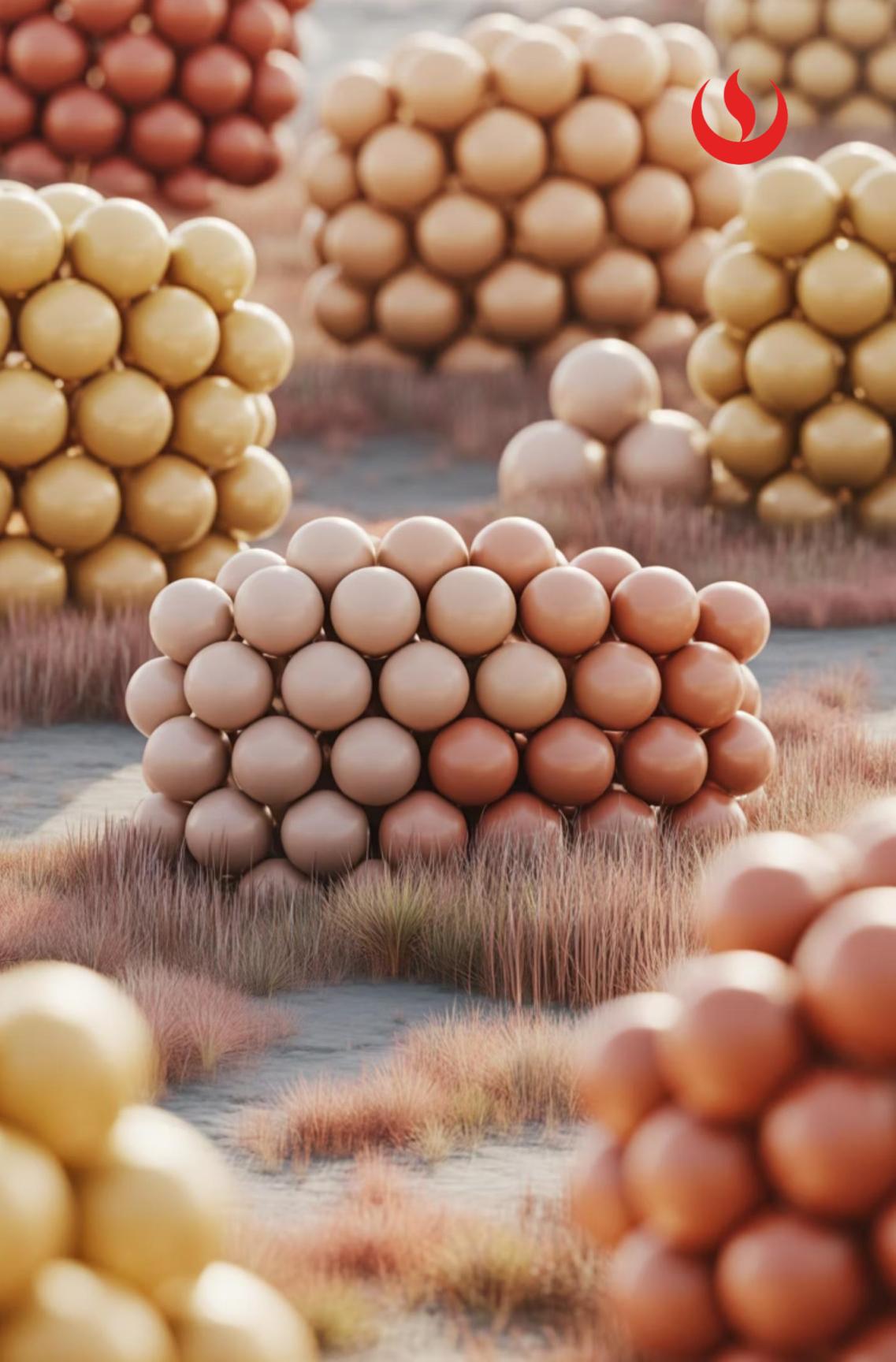


- Segmentación de clientes por comportamiento de compra
- Agrupación de documentos por tema
- Compresión de imágenes (reducir paleta de colores)
- Detección de comunidades en redes sociales
- Organización de resultados de búsqueda

## Algoritmos Principales



K-Means (particionamiento), DBSCAN (densidad), Hierarchical (jerárquico), Gaussian Mixture Models (probabilístico), Mean-Shift (centro de masa).





# ¿Qué se puede hacer?

Tenemos una base de datos con miles de registros de compras:

Fecha de venta	Producto	Categoría	Cantidad vendida	Precio unitario	Ciudad del cliente	País	Método de pago	Descuento aplicado	Canal de adquisición
5/01/2023	Camiseta blanca	Mujer	2	19.99 €	Madrid	España	Tarjeta de crédito	10%	Google Ads
10/01/2023	Jeans azules	Hombre	1	49.99 €	Barcelona	España	PayPal	0%	Búsqueda orgánica
15/01/2023	Vestido floral	Mujer	3	39.99 €	Valencia	España	Tarjeta de crédito	5%	Redes sociales
20/01/2023	Sudadera gris	Unisex	2	29.99 €	Sevilla	España	Contra reembolso	0%	Email marketing



Queremos entender mejor a nuestros clientes, a segmentar el mercado y a tomar decisiones más informadas para mejorar nuestro negocio. ¿Cómo lo hacemos?

# Clustering



## Descubrir Patrones Ocultos

El agrupamiento es una técnica de aprendizaje no supervisado que permite descubrir patrones y estructuras subyacentes en conjuntos de datos sin clasificación previa. Ayuda a identificar grupos o clústeres de elementos similares dentro de los datos.

## Segmentación y Clasificación

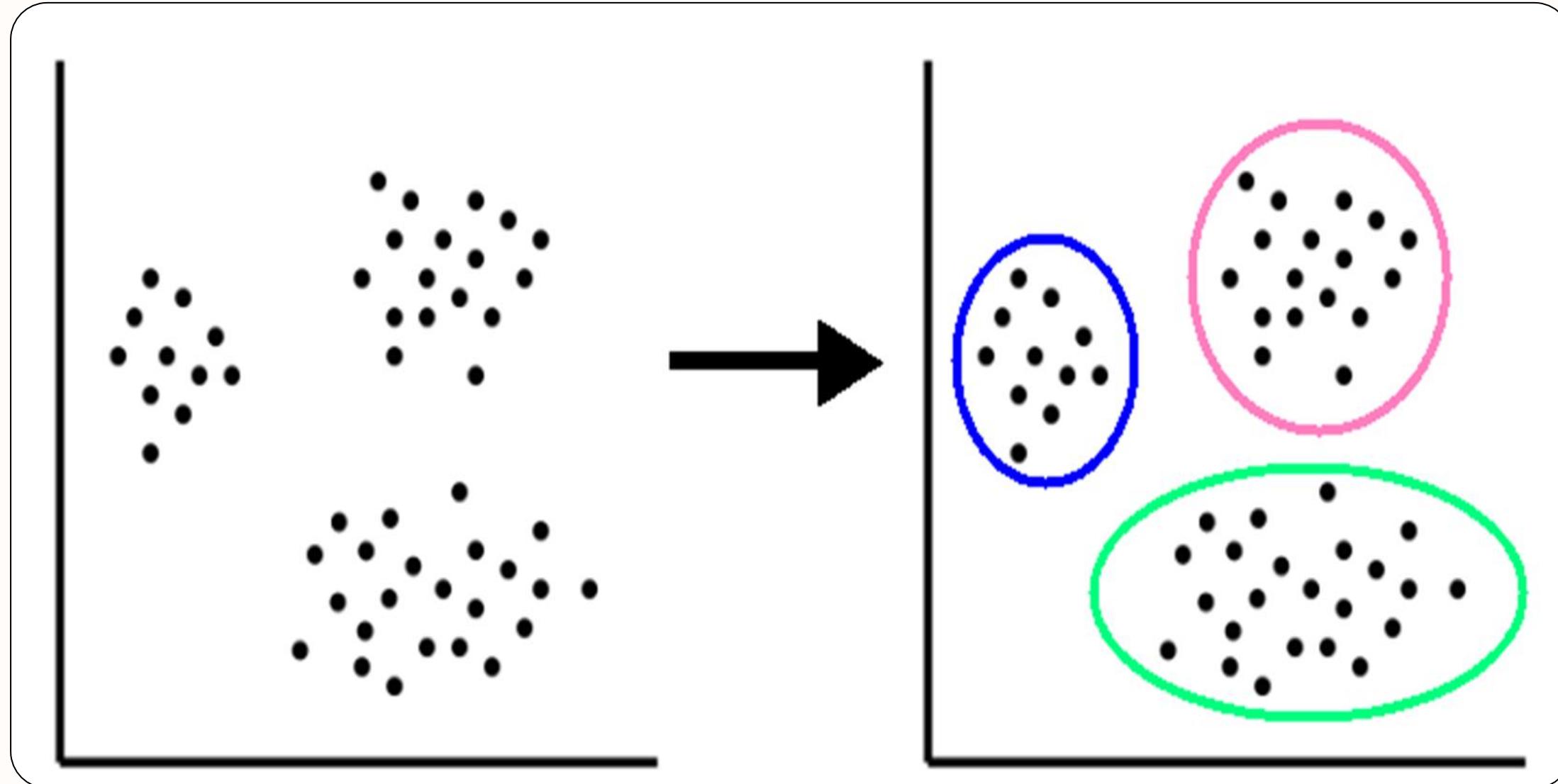
Al agrupar elementos similares, el clustering permite segmentar y clasificar datos de manera efectiva. Esto es útil para tareas como segmentación de mercados, detección de anomalías y recomendación de productos.

## Descubrimiento de Nuevos Conocimientos

El agrupamiento revela patrones y relaciones que pueden conducir al descubrimiento de nuevos conocimientos e insights valiosos dentro de los datos. Esto permite obtener una visión más profunda de los fenómenos subyacentes.

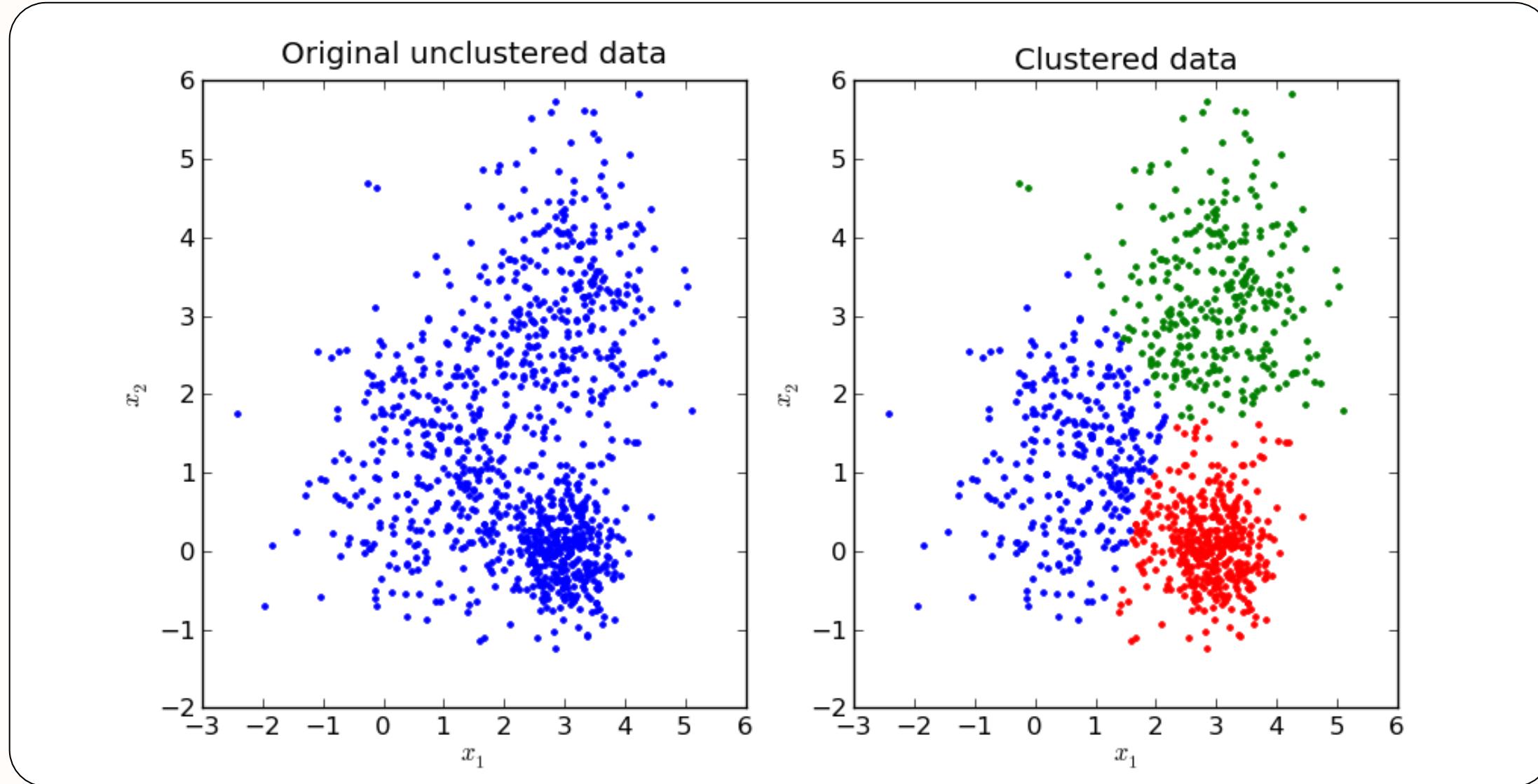


# Clustering



Fuente: <https://www.oreilly.com/api/v2/epubs/9781788393485/files/assets/cd2095b9-045f-48fc-86e2-e6df1f973034.png>

# Clustering



[https://pypr.sourceforge.net/\\_images/kmeans\\_2d.png](https://pypr.sourceforge.net/_images/kmeans_2d.png)



# Clustering - ¿Qué se puede hacer?

Tenemos una base de datos con miles de registros de compras

Fecha de venta	Producto	Categoría	Cantidad vendida	Precio unitario	Ciudad del cliente	País	Método de pago	Descuento aplicado	Canal de adquisición
5/01/2023	Camiseta blanca	Mujer	2	19.99 €	Madrid	España	Tarjeta de crédito	10%	Google Ads
10/01/2023	Jeans azules	Hombre	1	49.99 €	Barcelona	España	PayPal	0%	Búsqueda orgánica
15/01/2023	Vestido floral	Mujer	3	39.99 €	Valencia	España	Tarjeta de crédito	5%	Redes sociales
20/01/2023	Sudadera gris	Unisex	2	29.99 €	Sevilla	España	Contra reembolso	0%	Email marketing

En lugar de querer predecir una variable específica, queremos encontrar **patrones ocultos** en los datos que nos ayuden a entender mejor a nuestros clientes y sus comportamientos.

## Ejemplo: Segmentación de clientes basada en patrones de compra

**Objetivo:** Identificar grupos de clientes con comportamientos de compra similares para ofrecerles promociones y productos personalizados.

## Resultados:

- **Segmentación de clientes:** Obtenemos diferentes segmentos de clientes con características y necesidades distintas.
- **Personalización de ofertas:** Podemos crear campañas de marketing personalizadas para cada segmento, ofreciendo productos y promociones relevantes.
- **Optimización de inventario:** Ajustamos el inventario de acuerdo a la demanda de cada segmento.

# Clustering - Tipos de Agrupamiento

## Agrupamiento Jerárquico

Este enfoque construye una jerarquía de grupos, donde los grupos más pequeños se van fusionando progresivamente para formar grupos más grandes. Existen algoritmos aglomerativos y divisivos dentro del agrupamiento jerárquico.

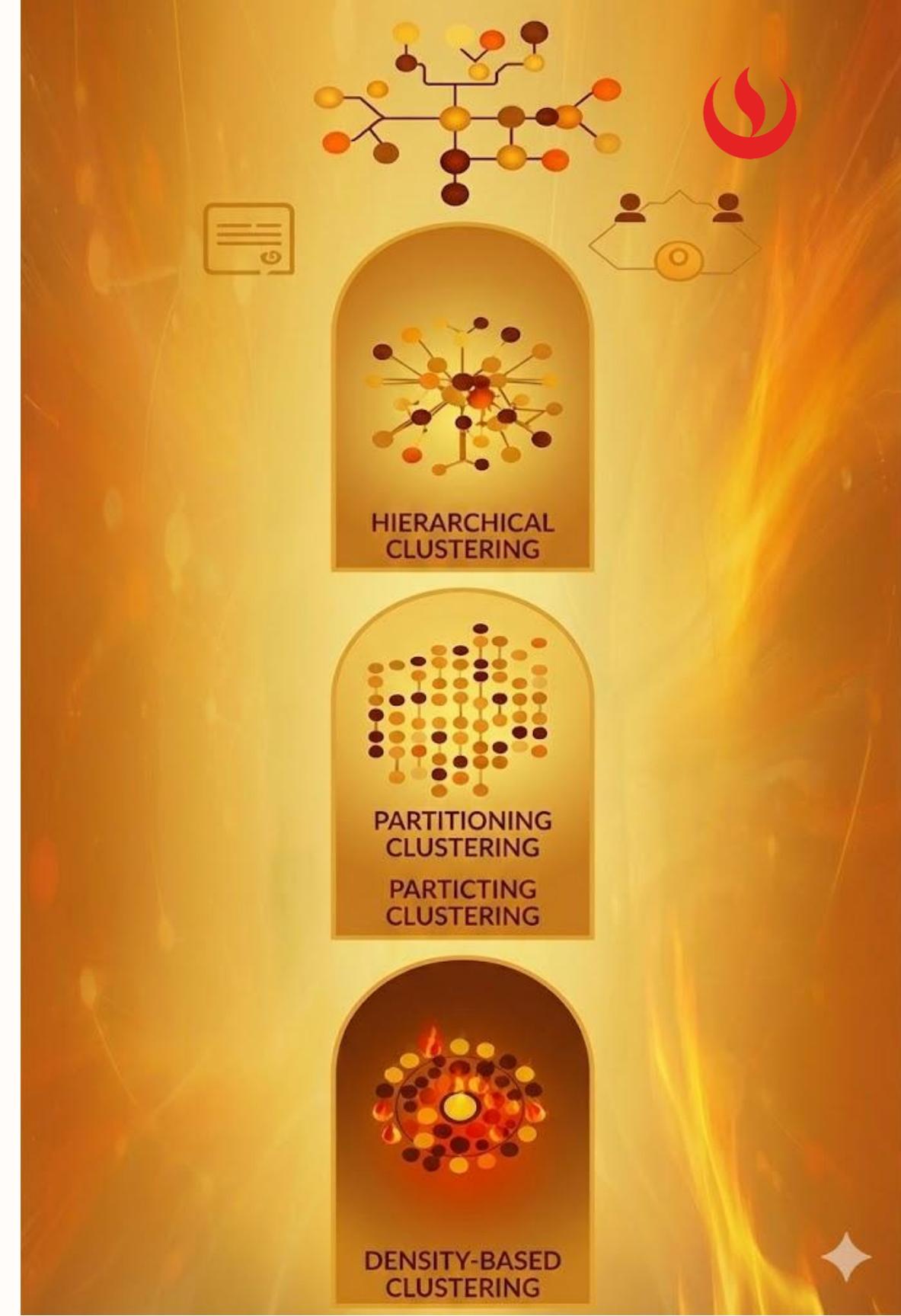
## Agrupamiento de Particionado y recolocación

Los métodos de particionado dividen los datos en un número predefinido de grupos, reasignando los objetos iterativamente hasta que se logre una partición óptima. Ejemplos incluyen el algoritmo k-means y el clustering probabilístico.

## Agrupamiento Basado en Densidad

Estos algoritmos identifican grupos densos en el espacio de características, separándolos de regiones de baja densidad. Ejemplos destacados son DBSCAN y OPTICS.

Métodos Basados en Rejillas  
Métodos Basados en Co-ocurrencia de Datos Categóricos  
Clustering Basado en Restricciones



# Clustering - Tipos de Agrupamiento



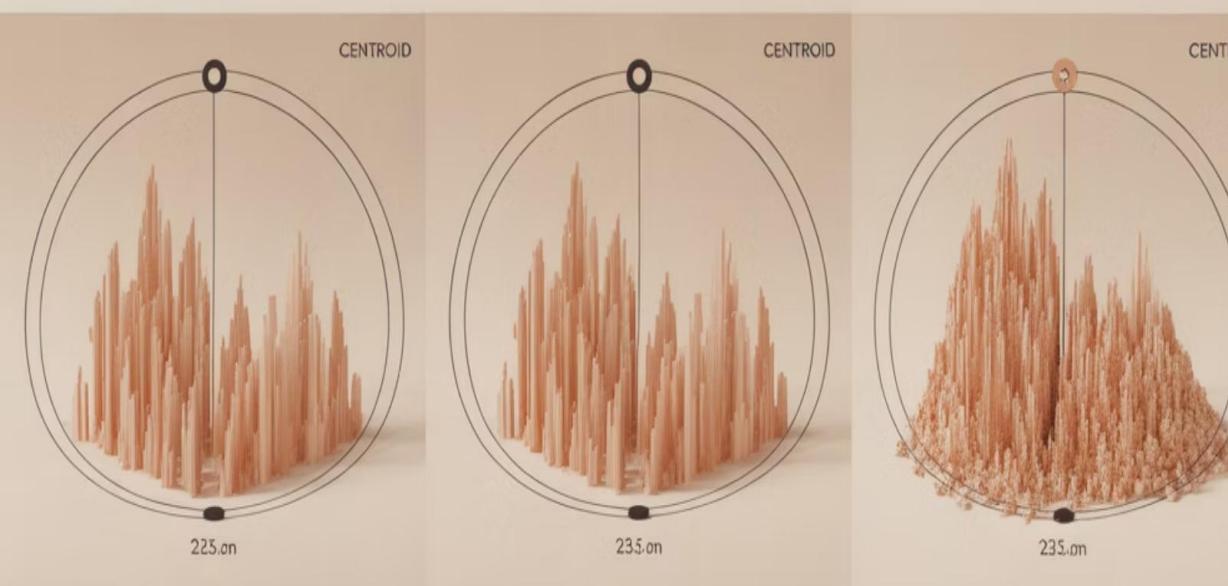
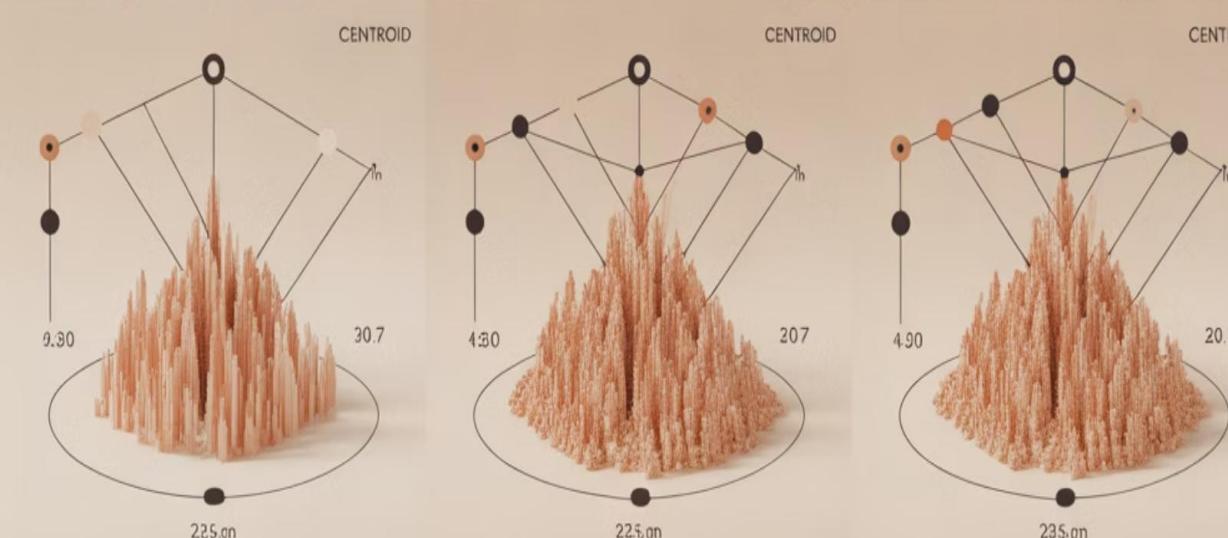
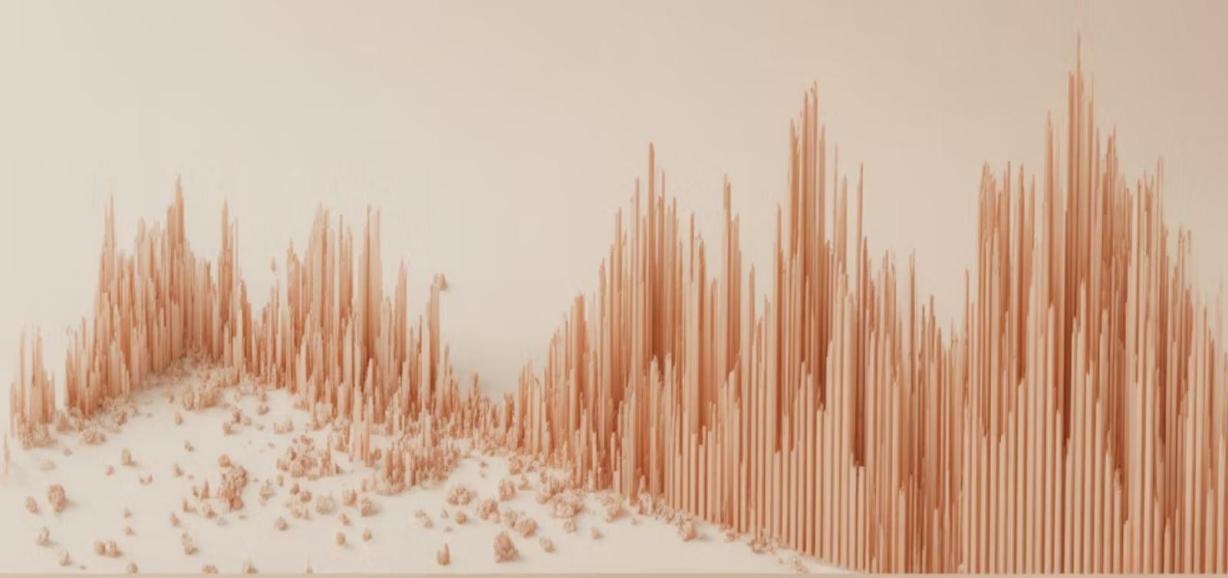
Métrica de Distancia	Características principales	Casos de uso típicos	Interpretación intuitiva
<b>Distancia Euclíadiana</b>	- Ideal para datos numéricos continuos. - Mide la distancia "en línea recta" entre dos puntos en un espacio euclidiano.	- Clustering k-means - Análisis de proximidad en datos geométricos - Recomendación de productos similares	Es como medir la distancia entre dos ciudades en un mapa plano.
<b>Distancia del Coseno</b>	- Ideal para vectores de texto y frecuencias. - Mide la similitud en la dirección de dos vectores, más que en su magnitud.	- Análisis de similitud de documentos - Procesamiento de lenguaje natural (NLP) - Recomendación de contenido similar	Es como medir qué tan similares son dos documentos en función de las palabras que comparten.
<b>Distancia de Manhattan</b>	- Adecuada para datos en cuadrículas o redes. - Suma las diferencias absolutas en cada dimensión.	- Planeación de rutas en ciudades - Análisis de redes de rutas - Problemas de optimización en redes	Es como medir la distancia que recorre un taxi en una ciudad con calles en cuadrícula.
<b>Distancia de Chebyshev</b>	- Útil cuando se quiere limitar la diferencia máxima en una dimensión. - Mide la diferencia máxima entre las coordenadas correspondientes de dos puntos.	- Juegos de tablero (ej: ajedrez) - Problemas de logística con restricciones	Es como encontrar la mayor diferencia entre dos puntos en cualquier dimensión, como encontrar la pieza más alejada en un tablero de ajedrez.

Algoritmo	Biblioteca	Parámetro para la métrica	Métricas Soportadas	Ejemplo de Uso en Python
<b>K-means</b>	sklearn	No configurable (usa Euclíadiana)	Solo Euclíadiana	<code>python from sklearn.cluster import KMeans; kmeans = KMeans(n_clusters=3); kmeans.fit(X)</code>
<b>K-medoids</b>	sklearn_extra	metric='manhattan', etc.	Euclíadiana, Manhattan, Chebyshev, Coseno, etc.	<code>python from sklearn_extra.cluster import KMedoids; kmmedoids = KMedoids(n_clusters=3, metric='manhattan'); kmmedoids.fit(X)</code>
<b>DBSCAN</b>	sklearn	metric='cosine', etc.	Euclíadiana, Manhattan, Chebyshev, Coseno, función personalizada	<code>python from sklearn.cluster import DBSCAN; dbscan = DBSCAN(eps=0.5, min_samples=5, metric='cosine'); dbscan.fit(X)</code>
<b>Clustering Jerárquico</b>	scipy	metric='chebyshev', etc.	Euclíadiana, Manhattan, Chebyshev, Coseno, Hamming, etc.	<code>python from scipy.cluster.hierarchy import linkage; Z = linkage(X, method='single', metric='chebyshev')</code>



# Algoritmo K-Means: Concepto de Centroídes

K-Means es el algoritmo de clustering más popular por su simplicidad y eficiencia. La idea: dividir datos en K grupos, cada uno representado por su centro (centroide).



## Fórmula de Distancia

Distancia Euclidiana (más común):

$$d(p, c) = \sqrt{\sum_{i=1}^n (p_i - c_i)^2}$$

Donde p es un punto de datos y c es un centroide.

## Actualización de Centroide

$$c_k = \frac{1}{|S_k|} \sum_{p \in S_k} p$$

El nuevo centroide es simplemente el promedio de todos los puntos  $p$  asignados al cluster  $S_k$ .

## Concepto Central

Un **centroide** es el punto promedio de todos los datos asignados a un cluster. Representa el "centro de masa" del grupo.



# Pasos del Algoritmo K-Means



## Inicialización

Seleccionar K puntos aleatorios como centroides iniciales. Alternativamente, usar K-Means++ para inicialización más inteligente que distribuye centroides alejados entre sí.



## Asignación a Clusters

Para cada punto de datos, calcular distancia a cada centroide y asignarlo al más cercano. Crear K grupos basados en estas asignaciones.



## Actualización de Centroides

Recalcular posición de cada centroide como el promedio (centro de masa) de todos los puntos actualmente asignados a ese cluster.



## Iteración

Repetir pasos 2-3 hasta que los centroides dejen de moverse significativamente (convergencia) o se alcance un número máximo de iteraciones.



## Resultado Final

K clusters estables con sus centroides finales. Cada punto pertenece al cluster de su centroide más cercano.

**Complejidad:**  $O(n \times K \times i \times d)$  donde n=puntos, K=clusters, i=iteraciones, d=dimensiones. Muy eficiente para datasets grandes.



# Algoritmo DBSCAN

Density-Based Spatial Clustering of Applications with Noise - Un enfoque radicalmente diferente que no requiere especificar K de antemano.

## Concepto de Densidad

DBSCAN agrupa puntos que están densamente empaquetados, y marca puntos en regiones de baja densidad como **ruido** (outliers).

Parámetros:

- **$\epsilon$  (epsilon):** Radio de vecindad
- **MinPts:** Mínimo de puntos para formar región densa

Tipos de puntos:

- **Core:**  $\geq$ MinPts vecinos en radio  $\epsilon$
- **Border:** Menos vecinos pero cerca de core
- **Noise:** No cumple criterios, outlier

## Ventajas sobre K-Means

- ✓ No requiere especificar K
- ✓ Descubre clusters de formas arbitrarias (no solo esféricas)
- ✓ Identifica automáticamente outliers
- ✓ Robusto a ruido

## Desventajas

- ✗ Sensible a elección de  $\epsilon$  y MinPts
- ✗ Problemas con densidades variables
- ✗ Menos eficiente en alta dimensionalidad
- ✗ Difícil para datos sin separación clara



# Caso de Uso: Segmentación de Clientes

Una de las aplicaciones más valiosas de clustering: dividir tu base de clientes en grupos homogéneos para personalizar estrategias de marketing, producto y atención.

## Datos de Entrada

### Variables RFM (Recency, Frequency, Monetary):

- Días desde última compra
- Número de compras (último año)
- Valor monetario total gastado

### VARIABLES DE COMPORTAMIENTO:

- Categorías de productos preferidas
- Canal preferido (online/tienda física)
- Uso de descuentos/promociones
- Engagement en emails (tasa apertura)

### VARIABLES DEMOGRÁFICAS:

- Edad, ubicación
- Tipo de cliente (B2B/B2C)

## Proceso de Segmentación

1. Limpiar datos (valores faltantes, outliers extremos)
2. Normalizar features (StandardScaler o MinMaxScaler)
3. Determinar K óptimo (método del codo → K=5)
4. Aplicar K-Means con K=5
5. Analizar características de cada cluster
6. Asignar nombres descriptivos a segmentos
7. Validar estabilidad con diferentes inicializaciones

**Herramientas:** Python (scikit-learn), R, o plataformas de CRM con ML integrado.



# Análisis de Resultados: Perfiles de Segmentos

Después de aplicar K-Means con K=5 a 50,000 clientes, emergieron perfiles claros que guían estrategia comercial.

## Champions (12%)

Compraron recientemente, frecuentemente y gastan mucho. Alto engagement.

- RFM: Alto en todo
- Ticket promedio: \$850
- Frecuencia: 8+ compras/año

**Acción:** Programa VIP, early access a productos

## Prometedores (23%)

Compradores recientes con potencial. Gasto medio-alto pero baja frecuencia.

- Recency: Alta
- Frequency: Baja (2-3/año)
- Monetary: Medio-alto

**Acción:** Campañas de frecuencia, recomendaciones personalizadas

## Fieles (18%)

Compran frecuentemente pero gastan poco cada vez. Muy engagement.

- Frecuencia: 10+ compras/año
- Ticket: \$180 promedio
- Productos: Básicos, recurrentes

**Acción:** Suscripciones, bundles, programa de puntos

## Durmientes (28%)

No compran hace 6+ meses. Gastaban medio-bajo.

- Recency: Baja (180+ días)
- Riesgo de churn alto

**Acción:** Campaña de reactivación agresiva, descuentos significativos

## Perdidos (19%)

Sin actividad hace 12+ meses. Bajo valor histórico.

- Última compra: >365 días
- LTV: Muy bajo

**Acción:** Win-back con oferta irresistible o descarte (costo-beneficio negativo)



# CONCLUSIONES



01

Las Redes neuronales pueden modelar relaciones no lineales complejas, mejorando la precisión en regresión y clasificación cuando los datos no siguen patrones simples.

02

Aprendizaje supervisado con redes neuronales requiere grandes volúmenes de datos y potencia computacional para entrenar modelos efectivos y evitar sobreajuste.

03

El agrupamiento no supervisado ayuda a descubrir patrones subyacentes en datos sin etiquetas, segmentando información en grupos similares para análisis más profundos.

04

La combinación de redes neuronales y agrupamiento puede mejorar la segmentación de datos y predecir resultados más precisos en tareas complejas como marketing o diagnóstico.



# Bibliografía

- Aurélien Géron (2024). Aprendizaje automático práctico con Scikit-Learn, Keras y TensorFlow, 3<sup>a</sup> edición, capítulo 9: Técnicas de aprendizaje no supervisado. O'Reilly Media, Inc.
- Scikit-learn. [https://scikit-learn.org/stable/modules/neural\\_networks\\_supervised.html](https://scikit-learn.org/stable/modules/neural_networks_supervised.html)
- Prosise, Jeff. Aprendizaje Automático Aplicado e IA para Ingenieros. O'Reilly Media, Inc. - Capítulo 9: Redes neuronales
- Kodali, T. (22 de enero de 2016). Hierarchical Clustering. <https://www.r-bloggers.com/2016/01/hierarchical-clustering-in-r-2/>