

King of Dumbs

el rey de los tontos

Nivel: ♦ Intermedio

¿Regresión logística o red neuronal? Es la misma neurona

Una regresión logística es una red neuronal de una sola neurona. Qué añade el deep learning, cuándo compensa y cuándo basta lo clásico, en tres niveles.

Español Publicado 4 de octubre de 2026



Escanea para abrir la lección interactiva

<https://kingofdumbs.com/es/lecciones/regresion-logistica-o-red-neuronal>

Qué vas a aprender

- Regresión logística
- Redes neuronales
- Función sigmoide
- Gradiente que se desvanece
- Elegir entre machine learning clásico y deep learning

Símbolos que aparecen

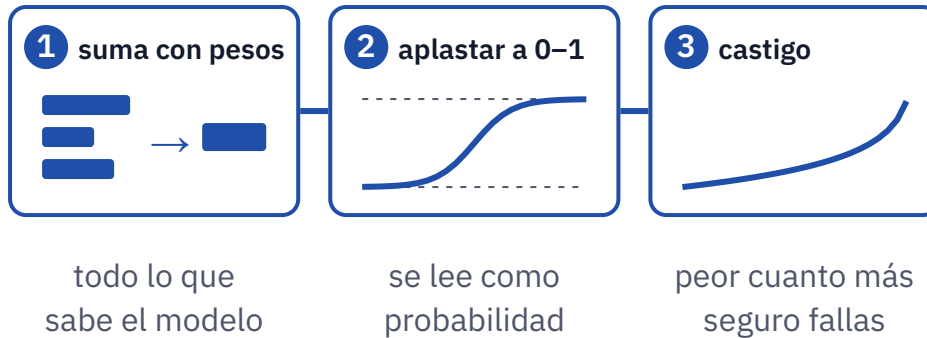
E_a subíndice: la etiqueta de abajo (♠) · Σ sumatorio: súmalos todos (♠) · x_i índice i: el elemento número i (♠) · Π productorio: multiplícalos todos (♠) · $\approx$ aproximadamente igual (♠) · $\sim$ del orden de / se reparte como (♠) · $\leq \geq$ menor o igual / mayor o igual (♠) · $|x|$ valor absoluto: sin signo (♠) · 10^{-3} exponente negativo: dividir (♠) · $h^{(1)}$ superíndice entre paréntesis: no es potencia (♠) · e el número e ($\approx 2,718$) (♠) · $\log$ logaritmo: ¿cuántas veces multiplico? (♦♠) · σ sigma: depende de la lección (♠) · λ lambda: un número que estira (♠) · δ delta minúscula: un error pequeño (♠) · $\|x\|$ norma: la longitud de la flecha (♠) · x^T traspuesta: tumbar la columna (♠) · $\in$ pertenece a (♠) · $\mathbb{R} \mathbb{Z}$ conjuntos de números (♠) · dE/dt derivada: el ritmo de cambio (♠) · ∂ derivada parcial: cambiar una sola cosa (♠) · ∇ gradiente: hacia dónde sube más (♠)

La primera vez que vi una red neuronal dibujada, con sus circulitos y sus flechas, pensé que era otra cosa que la estadística de toda la vida. Luego me tocó programar una regresión logística a mano y me hice la pregunta tonta de esta lección: **¿una regresión logística es una red neuronal?**

La respuesta corta es que sí. Una regresión logística es una red neuronal de **una sola neurona**. No se parecen ni son primas: son el mismo modelo con dos nombres. Si es la misma neurona, la pregunta interesante cambia: ¿qué añaden las redes grandes, y cuándo merece la pena pagarlas?

Elige tu carta: ♣ si empiezas de cero, ♦ si sabes qué es una función o una media, ♠ si quieres la notación. Puedes cambiarla en cualquier momento.

Una neurona, dos nombres



Una regresión logística tiene tres piezas, y conviene conocerlas porque cada una explica un fallo distinto.

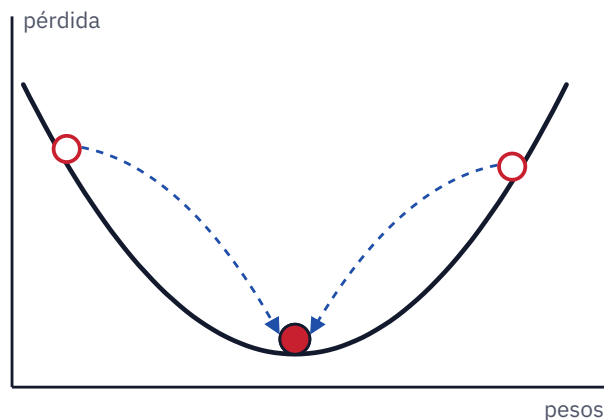
1. Una suma con pesos. Cada columna de tus datos se multiplica por un número, su peso, y todo se suma junto con un término fijo. Al resultado lo llamaremos la *puntuación*. Todo lo que el modelo sabe está ahí: si la señal está en una combinación de dos columnas o en una curva, el modelo no la ve a menos que se la des ya construida.

2. La sigmoide. Es una curva en forma de S que aplasta cualquier puntuación entre 0 y 1: una muy negativa da casi 0, una muy positiva casi 1, y cero da justo 0,5. No está elegida por estética: sale de suponer que la respuesta es una moneda trucada cuyo truco depende de una suma con pesos. Por eso lo que devuelve se puede leer como una probabilidad.

3. El castigo. Compara la probabilidad con la respuesta correcta y castiga mucho estar muy seguro y equivocarse. En la jerga se llama *pérdida*; esta en concreto, *log-loss* o entropía cruzada. Para aprender, el modelo mira hacia dónde baja el castigo y mueve los pesos un poco en esa dirección, una y otra vez: es el *descenso de gradiente*.

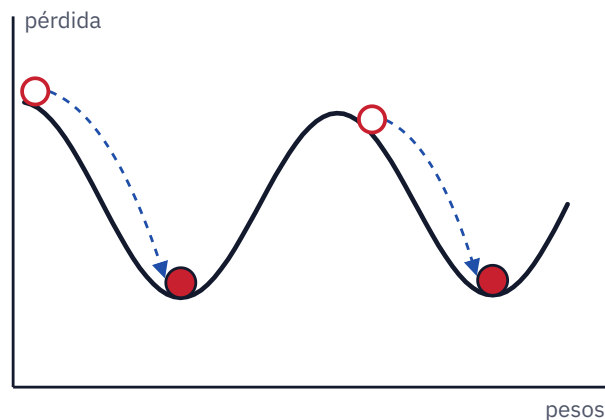
Y aquí está la propiedad que lo cambia todo: en la regresión logística, el castigo tiene **un solo fondo**. Cualquier forma de bajar llega al mismo sitio, así que la semilla aleatoria da igual. Una red con capas ocultas pierde esa garantía: tiene muchos valles, y dónde acabas depende de dónde empiezas.

Logística: un solo fondo



mismo sitio, empieces donde empieces

Red profunda: varios valles



depende de dónde empieces

Formas cualitativas, sin datos. Lo que importa es cuántos fondos hay.

Lo comprobé: la regresión logística de scikit-learn y una neurona entrenada a mano dan los mismos pesos hasta el tercer decimal. Un detalle práctico: muchas librerías añaden por defecto un freno que encoge los pesos, así que, si quieres leerlos, quítalo.

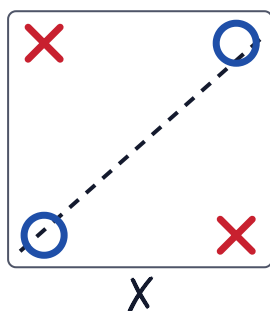
El Rey de los Tontos dice:

Yo me pasé una semana creyendo que tenía que elegir entre estadística y deep learning. Resulta que llevaba años haciendo deep learning de una neurona sin saberlo.

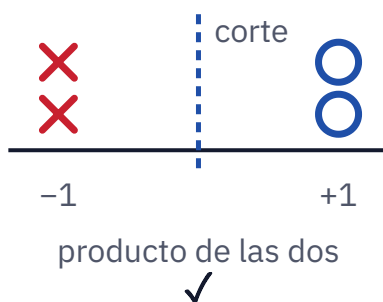


Quién fabrica las features

ninguna recta
los separa



con el producto
de las dos, sí



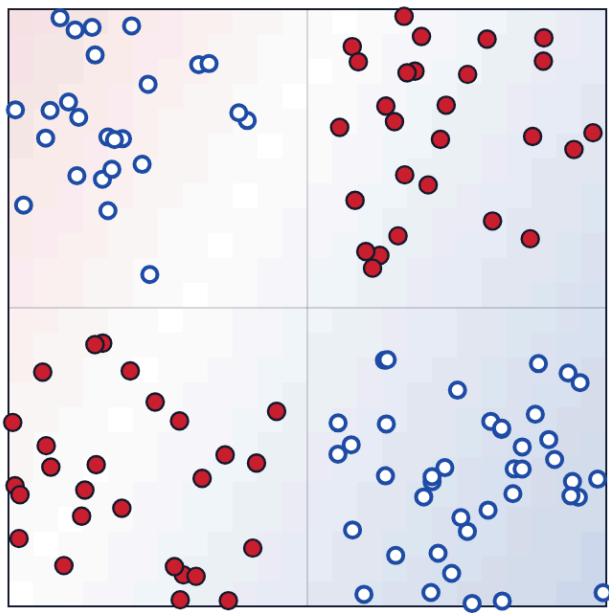
Si la neurona del final es la misma, lo que añade una red profunda es **lo que pasa antes de esa neurona**. Una red con diez capas que responde sí o no es, en su última capa, una regresión logística. La diferencia es que sus entradas no son tus columnas: son *features*, combinaciones nuevas que las **capas ocultas** han aprendido de los datos.

El ejemplo clásico es el XOR: cuatro puntos en las esquinas de un cuadrado, con la clase cambiada en cada diagonal. Una neurona solo sabe trazar una recta, y ninguna recta los separa. Pero si le das una columna nueva, **el producto de las dos columnas**, los puntos se separan solos. Esa columna nueva es una feature: o la fabricas tú a mano, o la fabrica una capa oculta.

Esto da la regla práctica más útil que conozco. Si tus datos son una tabla y las columnas ya significan algo, las features buenas existen y las capas ocultas tienen poco que aportar. Si son píxeles, audio o texto, ninguna suma con pesos de píxeles ve un gato: alguien tiene que fabricar las features. Pueden fabricarlas las capas ocultas con tus datos, cientos de árboles que suman su voto (el *boosting*, en tablas) o un modelo ya entrenado con datos de medio mundo, del que solo ajustas el final.

Un detalle práctico: en las librerías de deep learning, la red devuelve la puntuación sin aplastar, y la sigmoide se aplica dentro del castigo, donde las cuentas no se rompen con números enormes.

Pruébalo. El XOR y el anillo son los dos ejemplos clásicos de puntos que una recta no puede separar. Elige un conjunto y un modelo, y entrénalo:



● clase 1 ○ clase 0

Los puntos

El modelo

1 neurona (regresión logística) 3 parámetros

1 neurona + feature hecha a mano: $x_1 \cdot x_2$ 4 parámetros

Capa oculta de 8 + 1 neurona 33 parámetros

Pesos: $w = (-0,44; 0,24)$ · $b = -0,18$

Acierto: 43 % (decir siempre la clase más común: 55 %). Una neurona solo sabe trazar una recta, y ninguna recta separa un XOR. Acierta como una moneda al aire.



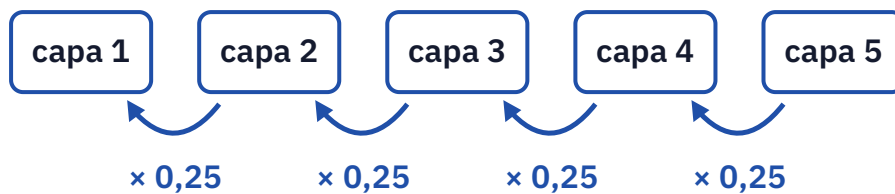
Widget interactivo. Pruébalo en la web: <https://kingofdumbs.com/es/lecciones/regresion-logistica-o-red-neuronal>

Lo que más me gusta de este experimento es la fila del medio. La misma neurona que fracasa con el XOR acierta en cuanto le das hecho a mano el producto de las dos coordenadas. La capa oculta llega al mismo sitio sin que nadie le diga qué buscar, a cambio de 33 números que ajustar en lugar de 4 y de que ya no puedas leer qué ha aprendido.



Por qué la sigmoide salió de las capas ocultas

El aviso del error va de la última capa a la primera



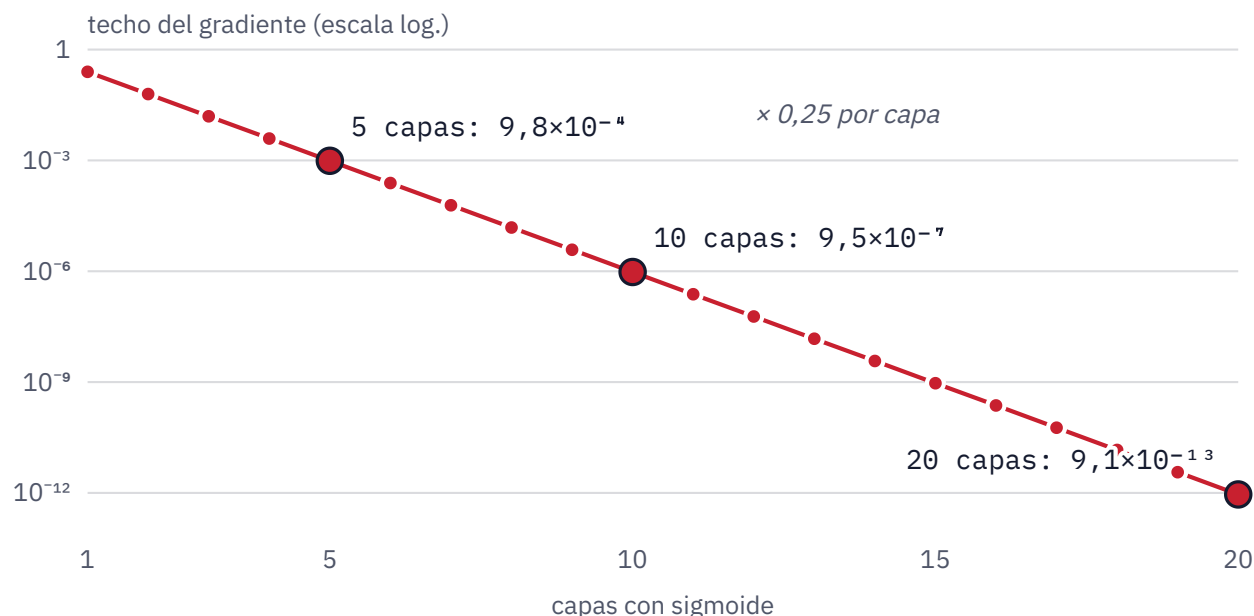
llega $\approx 0,001$

sale 1

error

Si una red es “muchas neuronas logísticas apiladas”, lo natural sería usar sigmoides en todas las capas. Así se hizo durante años, y salió mal.

El motivo está en la **pendiente** de la sigmoide. Como mucho vale 0,25, justo en el centro de la S, y en los extremos es casi cero, porque la curva ya está plana. Para entrenar una red, el aviso de cuánto y hacia dónde mover cada peso (el *gradiente*) se calcula de la última capa a la primera, multiplicando por la pendiente de cada capa: eso es la *retropropagación*. Con varias sigmoides seguidas, lo que llega a la primera capa tiene un techo de 0,25 multiplicado por sí mismo una vez por capa, y eso antes de contar lo que estiran o encogen los pesos.



Es aritmética, no una medida: 0,25 multiplicado por sí mismo una vez por capa.

Con 10 capas, la señal que llega al principio es menos de una millonésima. Las primeras capas casi no aprenden. Es el problema del *gradiente que se desvanece*, que Hochreiter describió en 1991. En la práctica fue igual de malo: Glorot y Bengio entrenaron en 2010 redes de 5 capas ocultas y el error de test fue del **82,61 % con sigmoides**, del 27,15 % con tangente hiperbólica y del 15,60 % con tangente hiperbólica y una inicialización mejor. En las capas ocultas, la sigmoide dejó su sitio a la **ReLU**, una función que deja pasar los positivos tal cual y pone a cero los negativos.

La sigmoide no desapareció: cambió de trabajo. Fuera de la cadena de capas ocultas no se multiplica por sí misma decenas de veces, y su rango de 0 a 1 es justo lo que hace falta para una probabilidad o para una puerta.

Dónde	Qué hace la sigmoide
Salida de un clasificador de sí o no	Convierte la última puntuación en una probabilidad. Es la regresión logística del final de la red
Clasificación con varias etiquetas a la vez	Una sigmoide independiente por clase: una foto puede ser “playa” y “atardecer”
Puertas de las redes LSTM y GRU	Decide qué fracción de la memoria pasa, de 0 a 1
SwiGLU, dentro de LLaMA y PaLM	Una rama, a través de una sigmoide, decide cuánto pasa de la otra

El Rey de los Tontos dice:

La sigmoide es como yo: la echaron de las capas ocultas y acabó de portera en todos los modelos de lenguaje.



Cuándo usar cada cosa

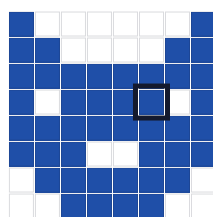
La decisión casi nunca va de “lo complejo que es el problema”. Va de qué forma tiene tu dato, cuántos ejemplos con la respuesta correcta tienes y quién tiene que entender cada decisión. Este árbol vale para los tres niveles:



Para predecir una etiqueta. Bajar de complejidad es barato; subir, caro: empieza por la izquierda.

edad	importe	visitas
34	120	3
51	15	9
27	300	1

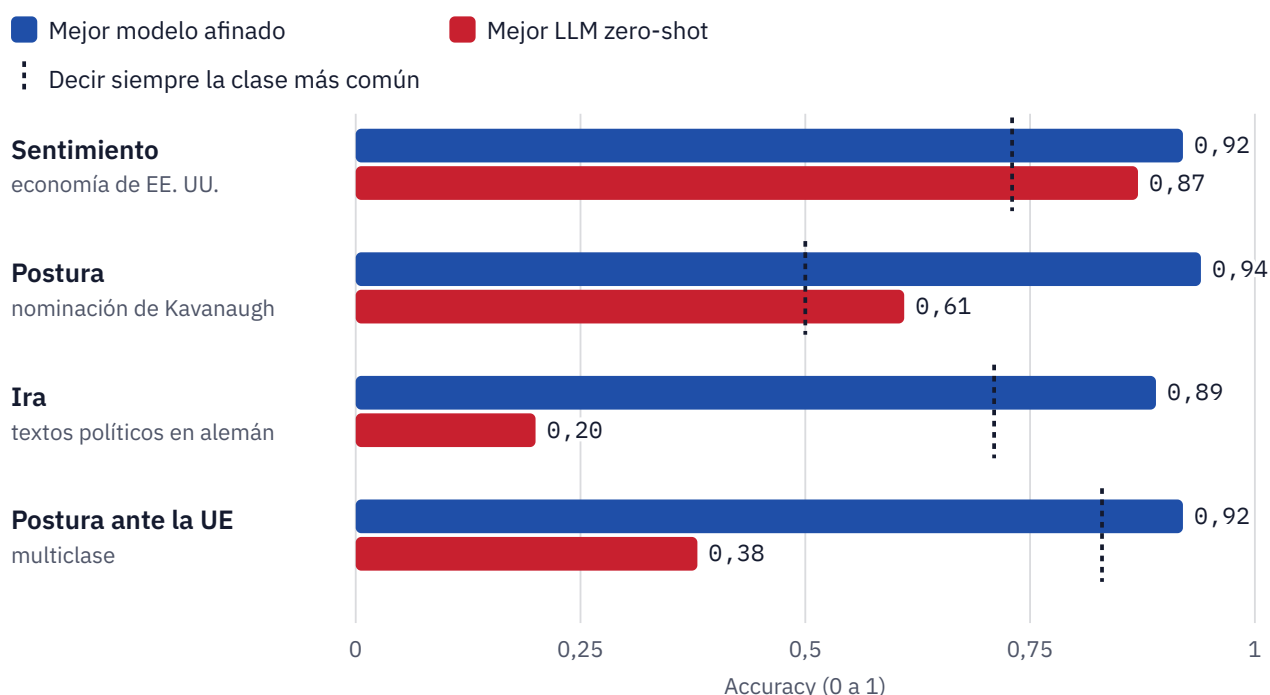
cada columna
ya significa algo



un píxel suelto
no significa nada

Tabla: lo clásico casi siempre basta. Christodoulou y su equipo revisaron 71 estudios de predicción clínica con 282 comparaciones entre regresión logística y machine learning. Midieron el acierto con el *AUC*, la probabilidad de que el modelo puntúe más alto un caso positivo que uno negativo. En las comparaciones con poco **riesgo de sesgo** (estudios bien hechos), la diferencia fue cero. La ventaja del machine learning solo aparecía en los estudios con mucho riesgo de sesgo. Leí el resumen del artículo, no el texto completo. En tablas grandes donde nadie tiene que entender las decisiones, los árboles que suman su voto suelen ganar a las redes, y la logística sigue siendo el punto de comparación con el que empezar.

Imágenes, audio o texto: hace falta aprender las features. Pero no siempre hace falta un modelo enorme. Bucher y Martini compararon en 2024 modelos pequeños **afinados** (ya entrenados y ajustados con unos cientos de ejemplos propios) con grandes modelos de lenguaje **sin ejemplos**, que solo reciben las instrucciones:



Accuracy de las Tablas 1 a 4 de Bucher y Martini (2024). La línea discontinua es lo que se acierta diciendo siempre la clase más común.

En dos de las cuatro tareas, el mejor modelo de lenguaje sin ejemplos queda **por debajo de decir siempre la clase más común**. Los modelos afinados rinden bien con entre 200 y 500 ejemplos etiquetados, según los autores. Un modelo de lenguaje sin ejemplos sirve para empezar el primer día y para etiquetar esos primeros cientos de ejemplos, pero no suele ser lo que quieres en producción.

Un último aviso: que un número salga entre 0 y 1 no lo convierte en una probabilidad de fiar. La logística sale bien **calibrada** de serie: si dice 80 %, acierta alrededor del 80 % de las veces. Las redes profundas modernas, no tanto, aunque se puede arreglar después con un solo ajuste.



Reto final

1. Una regresión logística y una neurona con sigmoide, entrenadas con el mismo castigo, ¿qué son?

- a. Primas: se parecen, pero dan resultados distintos
- b. El mismo modelo con dos nombres
- c. La neurona siempre acierta más, porque es deep learning

2. ¿Por qué dejó de usarse la sigmoide en las capas ocultas?

- a. Porque es lenta de calcular
- b. Porque el aviso del error se encoge en cada capa y las primeras casi no aprenden
- c. Porque los números entre 0 y 1 no sirven para nada

3. Tienes una tabla de 5.000 filas (edad, importe, visitas) y alguien tiene que entender cada decisión. ¿Por dónde empiezas?

- a. Un modelo de lenguaje con un prompt
- b. Una red profunda de diez capas
- c. Una regresión logística



En tres frases

1. Una regresión logística es una red neuronal de una sola neurona: la misma suma con pesos, la misma curva en S (la sigmoide) y el mismo castigo.
2. Lo que añade el deep learning son las capas de antes, que fabrican las features; la sigmoide salió de esas capas porque apaga el aviso del error, y hoy trabaja en la salida y en las puertas.
3. Si tus datos son una tabla con columnas que ya significan algo, empieza por la logística, que tarda minutos y te dice cuánto tiene que ganar lo complejo; si son píxeles, audio o texto, alguien tiene que fabricar las features.



Fuentes

- Regresión logística como modelo lineal generalizado y sigmoide como enlace canónico: [Stanford CS229, apuntes](#). Convexidad de la log-loss: Boyd y Vandenberghe, [Convex Optimization](#), §7.1.1.
- Valores por defecto de LogisticRegression (L2, C=1.0): [documentación de scikit-learn](#).
- Gradiente que se desvanece y derivada máxima de la sigmoide: Hochreiter, tesis de 1991 ([PDF](#), en alemán) y artículo de 1998 ([PDF](#)).
- Error de test con sigmoide, tanh y tanh con inicialización normalizada: Glorot y Bengio, *AISTATS* 2010, [Tabla 1](#).
- SwiGLU: Shazeer, [arXiv:2002.05202](#); su uso en [LLaMA](#) y [PaLM](#). Puertas de LSTM: [Hochreiter y Schmidhuber, 1997](#).
- Machine learning frente a regresión logística en predicción clínica: Christodoulou et al., *J Clin Epidemiol* 110, 12 (2019), [doi:10.1016/j.jclinepi.2019.02.004](#). Leído el resumen.
- Árboles frente a redes en datos tabulares: Grinsztajn et al., [arXiv:2207.08815](#).
- Modelos afinados frente a LLM sin ejemplos: Bucher y Martini, [arXiv:2406.08660](#), Tablas 1 a 4 y §5.5.
- Calibración de redes modernas: Guo et al., *ICML* 2017, [arXiv:1706.04599](#), Tabla 1.
- BCEWithLogitsLoss: [documentación de PyTorch](#).

Cómo está hecha esta lección: a partir de mis notas de machine learning, con ayuda de Claude para ordenarlas y redactar los tres niveles. Las cifras se comprobaron en las tablas originales de cada artículo el 4 de octubre de 2026. El laboratorio de la neurona entrena en tu navegador con descenso de gradiente, igual que el código del nivel ♠. Los dibujos son propios.

Soluciones

1. **b)** El mismo modelo con dos nombres. Eso es. Mismas tres piezas, mismos pesos al final.
2. **b)** Porque el aviso del error se encoge en cada capa y las primeras casi no aprenden. Exacto: como mucho 0,25 por capa. Con diez capas no llega ni un susurro.
3. **c)** Una regresión logística. Bien. Las columnas ya significan algo y la decisión se tiene que poder explicar: la logística es el punto de partida.

Lección de King of Dumbs, kingofdumbs.com. Versión para leer sin conexión; la original, con widgets que se tocan, está en: <https://kingofdumbs.com/es/lecciones/regresion-logistica-o-red-neuronal>

Actualizada el 4 de octubre de 2026.