Redes Neuronales en Keras

Luis Norberto Zúñiga Morales

12 de octubre de 2022

Contenido

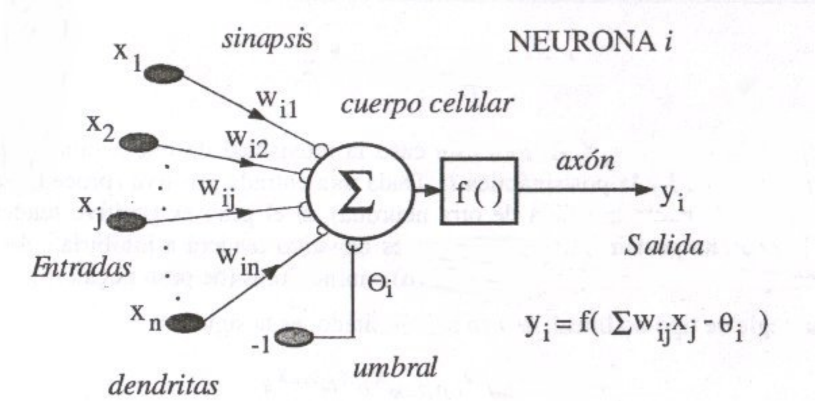


Figura: Neurona de McCulloch-Pitts [1], 1943.

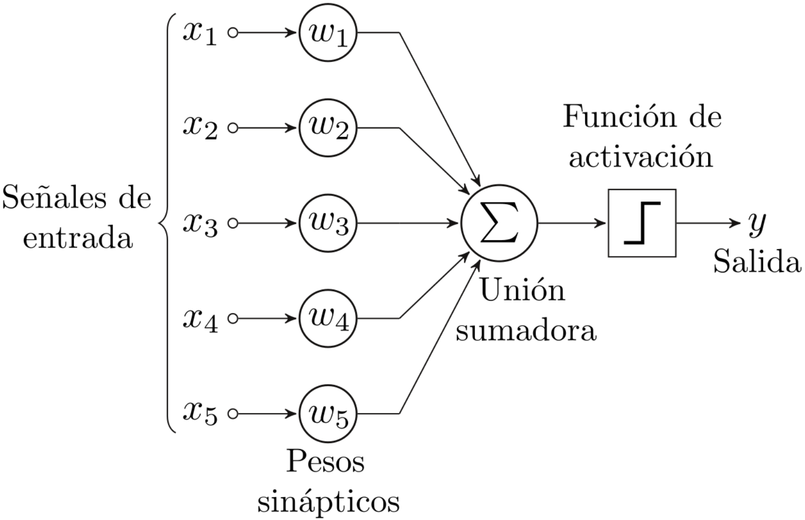


Figura: Perceptrón de Rosenblatt [2]. Fuente: Wikipedia.

- También llamado Treshold Logic Unit (TLU).
- Las entradas y salidas son números; cada conexión es un peso.
- Realiza un función de paso a la suma $z = \boldsymbol{x}^T \boldsymbol{w}$:

$$h_{\boldsymbol{w}}(\boldsymbol{x}) = \mathrm{step}(z)$$

- Las funciones de paso más comunes:

$$\mathrm{heaviside}(z) = \begin{cases} 0 & si \quad z < 0 \\ 1 & si \quad z \geq 0 \end{cases} \qquad \mathrm{sign}(z) = \begin{cases} -1 & si \quad z < 0 \\ 0 & si \quad z = 0 \\ +1 & si \quad z > 0 \end{cases}$$

- En solitario, sirve para clasificación binaria.

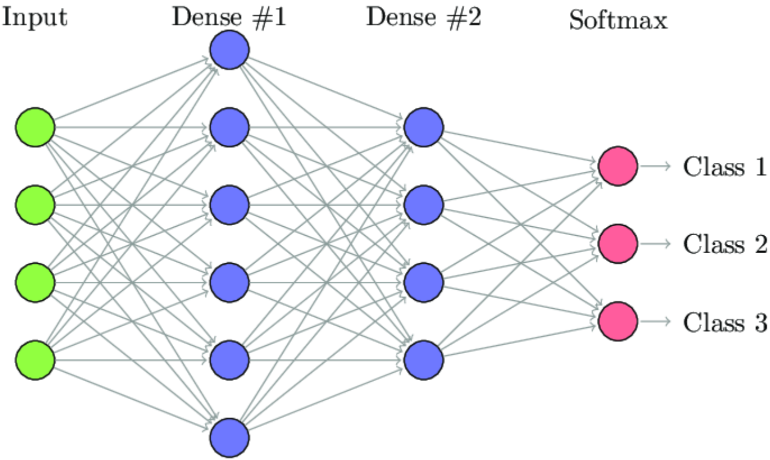


Figura: Red neuronal densa.

Para calcular las salidas de una capa de varias neuronas, se puede utilizar la fórmula:

$$h_{\boldsymbol{W},b}(\boldsymbol{X}) = \phi(\boldsymbol{X}\boldsymbol{W} + \boldsymbol{b}) \tag{1}$$

donde

- $\boldsymbol{X}$ representa la matriz de entrada de características, una fila por instancia y una columna por característica.
- $\boldsymbol{W}$ representa la matriz de pesos (enlaces entre las neuronas) a excepción de aquellos del bias. Una fila por neurona de entrada y una columna por neurona en la otra capa.
- $\boldsymbol{b}$ es el vector de pesos que conecta las neuronas de bias y las neuronas de la otra capa. Un bias por neurona artificial.
- ϕ es la función de activación: la función de paso y otras más.

- Originalmente se entrenaban con la regla de Hebb.
- La idea es reforzar la conexión cuando se realizar una predicción errónea.

$$w_{i,j}^{(\text{next})} = w_{i,j} + \eta(y_j - \hat{y}_j)x_i \tag{2}$$

  - $w_{i,j}$ es la conexión entre la i-ésima neurona de entrada y la j-ésima neurona de salida.
  - x_i es el i-ésimo valor de entrada de la instancia en turno.
  - $\hat{y}_j$ es la salida objetivo de la j-ésima neurona para la instancia x_i en turno.
  - y_j es la salida actual de la j-ésima neurona para la instancia x_i en turno.
  - η es la razón o tasa de aprendizaje.

Pregunta

La Ecuación 2

$$w_{i,j}^{(\text{next})} = w_{i,j} + \eta(y_j - \hat{y}_j)x_i$$

¿a qué se parece?

Pregunta

La Ecuación 2

$$w_{i,j}^{(\text{next})} = w_{i,j} + \eta(y_j - \hat{y}_j)x_i$$

¿a qué se parece?

Respuesta

¡Al gradiente descendiente estocástico!

Tarea

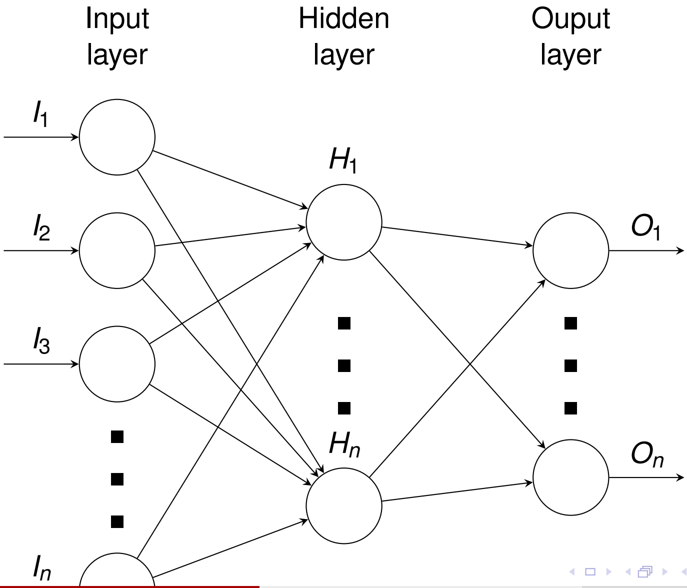
¿Qué dice el teorema de convergencia del perceptrón? ¿Cómo se demuestra?
Revisar el reporte técnico de Sompolinsky [4] para una explicación detallada.

- Se compone de una capa de entra (*input layer*).
- Una o más capas de TLUs llamadas capas ocultas (*hidden layers*).
- Una capa final de TLUs llamada capa da salida (*output layer*).
- Todas las capas, a excepción de la de salida, tienen una neurona de bias.

Ejercicio

Dibujen un MLP:

- Su capa de entrada.
- Una capa oculta.
- Una capa de salida.
- No olviden agregar el bias.


Input
layer
Hidden
layer
Ouput
layer
I_1
I_2
I_3
I_n
H_1
H_n
O_1
O_n

- Durante mucho tiempo, los investigadores sufrieron al momento de entrenar MLPs.
- En 1986, Rumelhart, Hinton y Williams [3] le dieron al mundo el regalo llamado *Backpropagation*.

- Recuerden la idea de usar gradiente descendiente es determinar la derivada de la función de pérdida con respecto a los pesos de la red.
- Esto es backpropagation.
- Asumiendo una neurona de salida, la función de error cuadrática se define como

$$E = L(t\hat{y}, y) \tag{3}$$

donde

  - L es el valor de pérdida entre la salida obtenida $\hat{y}$ y la salida objetivo y.

Algunas funciones de pérdida comunes:

Error Cuadrático Medio (ECM) - Regresión

$$\mathrm{ECM} = \frac{\sum_{i=1}^{n} (y_i - \hat{y}_i)^2}{n}$$

Error Absoluto Medio (EAM) - Regresión

$$\mathrm{ECM} = \frac{\sum_{i=1}^{n} |y_i - \hat{y}_i|}{n}$$

Cross Entropy Loss (CEL) - Clasificación

$$\mathrm{CEL} = -(y_i \log(\hat{y}_i) + (1 - y_i)\log(1 - \hat{y}_i))$$

Para cada neurona j, la salida o_j se define como

$$o_j = \phi\left(\sum_{k=1}^{n} w_{kj} o_k\right) \tag{4}$$

donde ϕ representa la función de activación.

Para cada neurona j, la salida o_j se define como

$$o_j = \phi\left(\sum_{k=1}^{n} w_{kj} o_k\right) \tag{4}$$

donde ϕ representa la función de activación.

Pregunta

¿Qué características debe tener ϕ?

Para cada neurona j, la salida o_j se define como

$$o_j = \phi\left(\sum_{k=1}^{n} w_{kj} o_k\right) \tag{4}$$

donde ϕ representa la función de activación.

Pregunta

¿Qué características debe tener ϕ?

Respuesta

¡Debe ser no lineal y diferenciable!

Pregunta

¿Cómo afecta esto a las funciones de decisión (de paso) que hablamos antes? (La función de Heaviside y la función signo)

Pregunta

¿Cómo afecta esto a las funciones de decisión (de paso) que hablamos antes? (La función de Heaviside y la función signo)

Respuesta

1. No son diferenciables en todos los puntos (ni son continuas).
2. Son planas (gradiente descendiente necesita gravedad para caer y funcionar).

Por lo anterior se proponen otras funciones de activación:

Función Logística Sigmoide

$$\sigma(z) = \frac{1}{1 + \exp -z} \tag{5}$$

Función Tangente Hiperbólica

$$\tanh z = 2\sigma(2z) - 1 \tag{6}$$

Rectified Linear Unit Function (ReLU)

$$\mathrm{ReLU}(z) = \text{máx}(0, z) \tag{7}$$

- Las redes neuronales permiten resolver problemas de regresión:
  - No se debe utilizar función de activación en las neuronas.
  - Es posible manipular los resultados con las funciones de activación si requieren que los resultados caigan en un rango en particular.
  - Función de pérdida: error cuadrático medio.
- Para el caso de clasificación:
  - Permiten trabajar fácilmente el caso de clasificación binaria multietiqueta.
  - Ya que se predicen distribuciones de probabilidad, se suele usar como función de pérdida entropía cruzada.

Implementación de MLPs en Keras

- Keras[1] es una API de alto nivel de Deep Learning que permite construir, entrenar, evaluar y ejecutar cualquier tipo de redes neuronales.
- Para realizar los cálculos requeridos para entrenar las redes neuronales, utiliza un backend computacional:
  - Tensorflow
  - Microsoft Cognitive Toolkit
  - Theano

[1] https://keras.io/

Implementación de MLPs en Keras

Vamos a Google Colab...

Ejercicio

1. Cargar el dataset MNIST-Fashion.
2. Construir una red neuronal con las siguientes características:
   - Una capa oculta de 128 neuronas con función de activación ReLU y dropout del 20 %.
   - Capa de salida de 10 neuronas con función de activación softmax.
   - 15 epochs y batch size de 1000.
3. Obtener el valor de accuracy.

Bibliografía Sugerida

[1] Warren S. McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5:115–133, 1943.

[2] Frank Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65 6:386–408, 1958.

[3] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning representations by back-propagating errors. *Nature*, 323:533–536, 1986.

[4] Haim Sompolinsky. Introduction: The perceptron. Technical report, MIT, 2013.