Perceptrón y lógica difusa

Perceptron and diffuse logic

Autor: Juan Camilo Varón Rodríguez
Facultad de Ingenierías, Universidad Tecnológica de Pereira, Pereira, Colombia
Correo-e: Juan.varon@utp.edu.co

***Resumen*— En la actualidad es normal que haya problemas de computación de dimensiones bastante grandes donde se hace necesario simular el funcionamiento de una neurona humana, este documento es un esbozo de la manera en cómo se simulan las neuronas ya mencionadas y la forma cómo se entrenan las mismas.**

***Palabras clave*— Aprendizaje, Perceptrón, Inteligencia Artificial, Entrenamiento, Neurona.**

***Abstract*— At present it is normal that there are problems of computation of quite large dimensions where it is necessary to simulate the functioning of a human neuron, this document is an outline of the way in which the aforementioned neurons are simulated and the way they are trained .**

***Key Word* — Learning, Perceptron, Artificial Intelligence, Training, Neuron.**

I. INTRODUCCIÓN

Con el paso del tiempo las necesidades en materia de complejidad computacional han aumentado en vista de que la dimensión de los problemas también, una herramienta para abordar los mismos es el perceptrón, ya que este supone hasta cierto punto una simulación del funcionamiento del cerebro para que sistemas informáticos logren pensar por sí mismos.

Hasta cierto punto, tenemos estudiado el cerebro humano y sabemos la forma en cómo se comporta la neurona, es decir, la manera en cómo obtiene información y se entrena en base a la misma para obtener resultados contundentes.

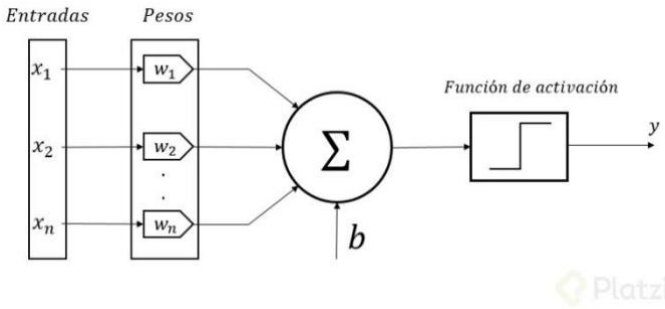
Es curiosa la manera en cómo es posible que se pueda simular una o muchas neuronas y se pueda aplicar conocimiento en dicha simulación para que en determinado momento un sistema esté en la capacidad de tomar decisiones.

II. CONTENIDO

¿Qué es el Perceptrón?

Para entrar en contexto, es fundamental comprender que el perceptrón es un modelo matemático capaz de simular el comportamiento de una neurona, desarrollado como modelo en 1943 por McCulloch y Pitts, un perceptrón está estructurado por los siguientes elementos: Entradas, Pesos, Bias y una función de activación.

En su forma gráfica un perceptrón se representa así:



Dónde sus entradas son x_n, Sus pesos son w_n, y su salida está representada por Y.

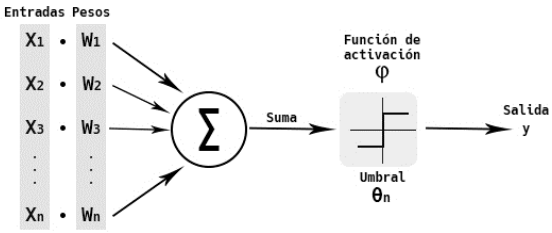
Entradas: Es toda la información que recibe el perceptrón.

Pesos: Estos son valores numéricos que le suman importancia a una entrada, sirven como factor que multiplica las entradas para determinar su influencia en los resultados.

Bias: Este parámetro solo algunos modelos lo tienen y a nivel de redes neuronales permite encontrar de manera sencilla la separación entre posibles salidas de la red neuronal.

Función de activación: Es la función matemática que determina el valor de salida una vez que se procesaron cada una de las entradas.

En forma sencilla y gráfica un perceptrón se representa así:

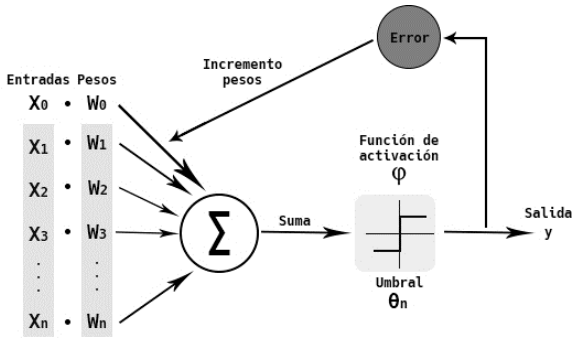


Este modelo además de su capacidad de decisión tiene la habilidad de aprender cuáles son los pesos adecuados para que una salida pueda ser correcta dependiendo el contexto del problema que se está abordando. A fin de cuentas, el perceptrón decide entre dos posibles valores, por lo que se considera una herramienta de clasificación.

Entrenamiento del Perceptrón

Como ya se mencionó anteriormente, el perceptrón se entrena para tener los pesos adecuados. ¿Y cómo sucede este proceso?

$$f(salida) = \begin{cases} 1 \; si \; salida \geq \theta \\ 0 \; si \; salida < \theta \end{cases}$$



Como se puede observar en la gráfica anterior, es posible que las salidas no sean satisfactorias, por lo que se hace necesario entrenar el perceptrón modificando el valor de los pesos.

Para dicho entrenamiento:

Paso 0: Inicialización
Inicializar los pesos sinápticos con números aleatorios pertenecientes al intervalo [-1,1].

Paso 1: (K-ésima iteración)
Calcular

$$y(k) = \operatorname{sgn}\left(\sum_{j=1}^{n+1} w_j x_j(k)\right)$$

Inicializando K en 1, y teniendo en cuenta que n = número de entradas del perceptrón.

Paso 2: Corrección de los pesos sinápticos

Si z(k)≠y(k):

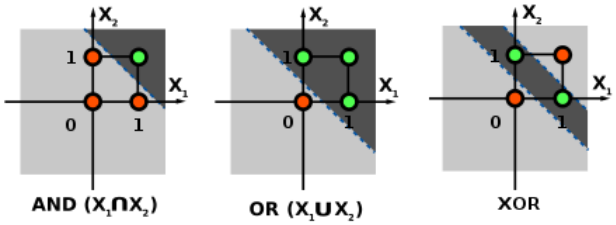
Modificar los pesos sinápticos según la expresión:

$$w_j(k+1) = w_j(k) + \eta\left[z(k) - y(k)\right]x_j(k), \quad j = 1,2,\ldots,n+1$$

Parar, la red ahora cuenta con pesos estables.

De otro modo, ir al Paso 1 con k = k + 1.

Sin embargo, cuando se da un conjunto de patrones de entrenamiento, en algunos casos el perceptrón no podría aprender a clasificarlos correctamente, esto en vista de que el algoritmo funciona siempre y cuando los patrones sean linealmente separables, reduciendo el campo de aplicaciones del perceptrón simple. A manera de ejemplo, podemos visualizar que un perceptrón podría ser útil para la función AND y OR, pero no para la función XOR.



III. CONCLUSIONES

El perceptrón es una herramienta de clasificación poderosa siempre y cuando los conjuntos y sus patrones sean linealmente separables.

Este modelo es una alternativa sencilla de entrenamiento fundamental para las alternativas que propone el Machine-Learning.

REFERENCIAS

[1]. Perceptrón: https://es.wikipedia.org/wiki/Perceptr%C3%B3n
[2]. Cómo entrenar a tu perceptrón: https://koldopina.com/como-entrenar-a-tu-perceptron/