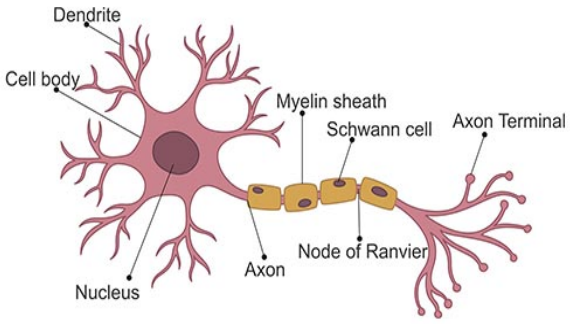
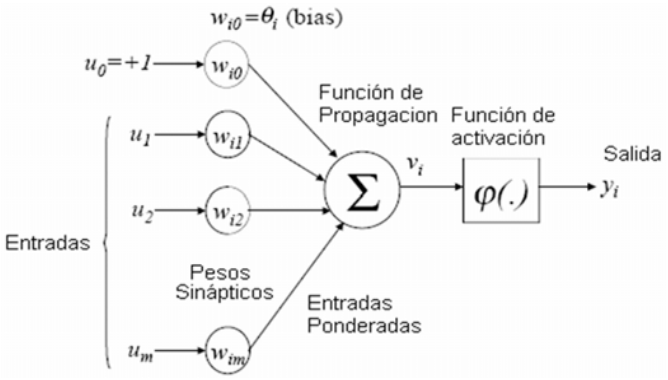
INTELIGENCIA ARTIFICIAL

TP 4. INTRODUCCIÓN A LAS REDES NEURONALES

Breve revisión.

En el campo de la inteligencia artificial, Frank Rosenblatt publicó el primer concepto de regla de aprendizaje del perceptrón, basado en el modelo de la neurona MCP, en “The perceptron: A perceiving and Recognizing Automaton, F. Rosenblatt Cornell Aeronautical Laboratory, 1957.
Con esta regla del perceptrón, Rosenblatt propuso un algoritmo que podía automáticamente aprender los coeficientes de pesos óptimos que luego se multiplican con las características de entrada para tomar la decisión de si una neurona se activa o no. En el contexto del aprendizaje supervisado y la clasificación, un algoritmo como este podría utilizarse para predecir si una muestra pertenece a una clase o a otra.
A continuación se presenta una representación visual del modelo matemático propuesto visto en teoría





Definición formal de una neurona artificial

De un modo más formal, para simplificar, podemos situar las ideas de las neuronas artificiales en el contexto de una tarea de clasificación binaria donde hacemos referencia a nuestras dos clases como 1 (clase positiva) y -1 (clase negativa). Además, podemos definir una función de decisión ($\phi(z)$) que toma una combinación lineal de determinados valores de entrada de x y un vector de peso correspondiente w, donde$z$ es la denominada entrada de red, se calcula $z = w_1 x_1 + \dots + w_m x_m$.

$$w = \begin{bmatrix} w_1 \\ . \\ . \\ . \\ w_n \end{bmatrix} \quad x = \begin{bmatrix} x_1 \\ . \\ . \\ . \\ x_m \end{bmatrix}$$

Ahora, si la entrada de la red de una muestra concreta es mayor que un umbral definido θ, predecimos la clase 1, de otro modo procedimos -1. En el algoritmo del perceptrón, la función de decisión $\phi(.)$ es una variante de una función escalón unitario:

$$\phi(z) = \begin{cases} 1 & if\ z \geq 0 \\ -1 & otherwise \end{cases}$$

Matricialmente, podemos escribir $z = \sum_{j=0}^{m} x_j w_j = w^t x$, $\phi(z) = \phi(w^t x)$

BioIng. Hadad, Alejandro.
Lic. Díaz, Santiago R.
Cátedra Inteligencia Artificial - LIC. en sistemas de información

La regla de aprendizaje del perceptrón

1. Iniciar los pesos a 0 o números aleatorios más pequeños.
2. Para cada muestra $x^{(i)}$
    a. Calcular el valor de salida $\hat{y}$
    b. Actualizar los pesos según la regla de aprendizaje del perceptrón vista en teoría, ósea $w_j := w_j + \Delta w_j$, donde $\Delta w_j = \eta(y^{(i)} - \hat{y}^{(i)})x_j^{(i)}$

El factor η es la tasa de aprendizaje (normalmente, varía entre 0.0 y 1.0), $y^{(i)}$es **la etiqueta de clase verdadera** de la muestra de entrenamiento i, y $\hat{y}^{(i)}$ **es la etiqueta de clase predicha.**

Antes de trabajar con el algoritmo en Python, se sugiere estudiar los siguientes ejemplos manuales para entender correctamente la técnica:

Si el perceptrón predice correctamente la clase, la diferencia entre el valor predicho y el valor real es 0, entonces los pesos no cambian. Ej. si la clase predicha es 1, cuando la clase es 1, el cálculo de Δw_j es :

$$\Delta w_j = \eta(y^{(i)} - \hat{y}^{(i)})x_j^{(i)} = \eta(1 - 1)x_j^{(i)} = \eta(0)x_j^{(i)} = 0.$$

Sin embargo, si la predicción no es correcta, los pesos se verán empujados hacia la dirección de la clase de destino según corresponda, Ej:

$\Delta w_j = \eta(y^{(i)} - \hat{y}^{(i)})x_j^{(i)} = \eta(1 - (-1))x_j^{(i)} = 2\eta x_j^{(i)}$, en este caso, los pesos cambian según $w_j := w_j + 2\eta x_j^{(i)}$

Es importante observar que la ***convergencia del perceptrón*** sólo será garantizada si las dos clases son linealmente separables y si el rango de aprendizaje es suficientemente pequeño, de otro modo, el perceptrón nunca dejará de actualizar sus pesos.

Actividades

Todo debe ser entregado en Colab Notebook.

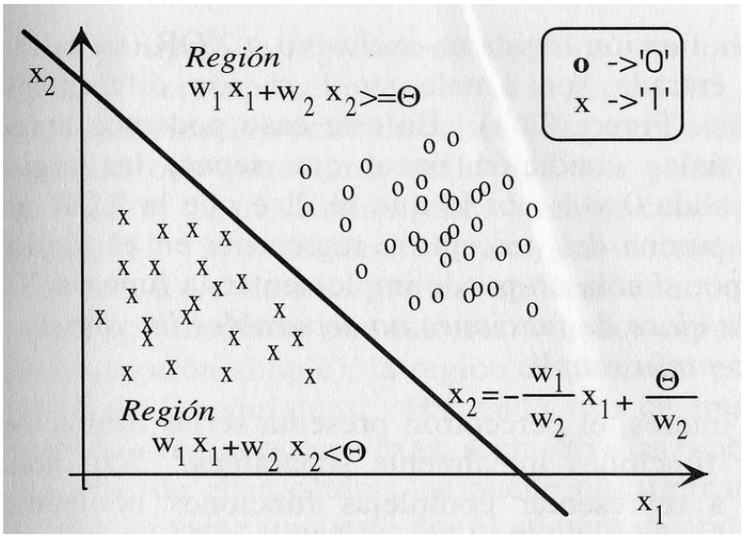
1) Considerar el conjunto de datos *Iris-dataset* (Ver código abajo para crear DF), y seleccione aquellas filas que se correspondan con solo los tipos de flores del tipo "***Setosa y Versicolor***". Por razones de visualización sólo tendremos en consideración las características "longitud de pétalo" y "longitud del sépalo".
Entrenar un perceptrón para poder separar ambas clases y realice el gráfico de la recta generada por el perceptrón.
Trabaje con un conjunto de datos de entrenamiento (training) y pruebas(testing) según su criterio.

Código para traer conjunto de datos Iris a pandas DF

```
import pandas as pd
df = pd.read_csv('https://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.data',
header=None)
```

2) Trabaje con diferentes valores de tasa de aprendizaje, al menos **deben ser 10 diferentes valores**. ¿Qué efecto tiene el cambio de la tasa de aprendizaje(η) sobre la modificación de los pesos?

3) Escriba las ecuaciones correspondientes para la recta y las regiones de decisión, consulte los resultados del script python. ¿Cuál es el efecto que se obtiene cambiando el umbral?



4) Dada la afirmación ***"el algoritmo sólo convergerá hacia un óptimo si los datos de entrenamiento son linealmente separables y la tasa de aprendizaje es suficientemente pequeña."*** Muestre que bajo datos de entrenamiento linealmente no separables el algoritmo no converge, utilice Colab y explique el paso a paso.

Bibliografía consultada

Python Machine Learning - Aprendizaje Automático y Aprendizaje profundo con Python - Scikit-Learn y TensorFlow. Sebastian Raschka - Vahd Mirjalili, Segunda Edición, Año 2019.

Machine Learning in the AWS Cloud. Add Intelligence to Applications with Amazon SageMaker and Amazon Rekognition. Abhiskey Mishra. 2019.

Tema 8. Redes Neuronales - Scientific Figure on ResearchGate. Available from: https://www.researchgate.net/figure/Figura-7-Region-de-decision-correspondiente-a-un-perceptron-simple-con-dos-neuronas-de_fig4_268291232 [accessed 26 Aug, 2021]

Introduction: The Perceptron, Haim Sompolinsky, MIT - Massachusetts Institute of Technology October 4, 2013

Articulo: Understanding Long Short-Term Memory Recurrent Neural Networks. 2019