Ejercicio Realizado por ADAIR HERNANDEZ VALDIVIA

Neurona McCulloch y Pitts

(Ejercicio, 3 puntos posibles)

El modelo de McCulloch y Pitts, concebido por Warren McCulloch, neurocientífico, y Walter Pitts, lógico matemático, en 1943 [1], representa uno de los fundamentos teóricos de las redes neuronales y la inteligencia artificial. Este modelo es una simplificación abstracta de las neuronas biológicas, propuesta para entender cómo podrían las neuronas del cerebro generar patrones complejos de pensamiento a partir de operaciones simples.

Open in Colab

[1] McCulloch, W. S., & Pitts, W. (1990). A logical calculus of the ideas immanent in nervous activity. Bulletin of mathematical biology, 52, 99-115.

Para formalizar el modelo de McCulloch y Pitts, representaremos una neurona con la letra C y definiremos que la entrada de dicha neurona, denominada $input$, está determinada por una tupla (E, I), donde E es el conjunto de señales excitatorias e I es el conjunto de señales inhibitorias. Asimismo, estableceremos como restricciones que tanto la entrada como la salida sean variables binarias, es decir, $input, output \in 0, 1$. Adicionalmente, definiremos un umbral u que la neurona utiliza para determinar los casos de excitación..

Dado lo anterior la salida de C se calcula usando las siguientes reglas:

1. En caso de que alguna de las entradas inhibitorias esté activa la neurona no se exita, es decir,

   $C(E, I) = 0$ if $\exists \iota_i = 1; \iota_i \in I \backslash$

2. La neurona se exita si la integral de sus entradas exitatorias es igual o superior al umbral, es decir,

   $C(E, I) = 1$ if $\sum e_i \geq u; e \in I \backslash$

3. En cualquier otro caso la neurona permanece sin exitación.

In [1]:
```python
#importamos paquetes necesarios
import numpy as np
```

In [2]:
```python
# TODO: (2 puntos) Implemente la función del modelo M&P, no use funciones predefinidas de numpy
def neuronaMyP(E,I,u):
    # E: Exitatoria (vector de 0s y 1s)
    # I: Inhibitoria de entrada (vector de 0s y 1s)
    # u: Umbral (escalar)

    # Calcular si existe un i en I que sea 1 para que la salida sea 0
    for i in I:
        if i == 1:
            return 0
    # Si no existe un i en I que sea 1, entonces hacemos la suma de todos los elementos de E y comparamos con el umbral u
    suma = 0
    for e in E:
        suma += e

    if suma >= u:
        return 1
    else:
        return 0
```

In [3]:
```python
# suponga
E = [1]
I = [0]
u = 1

# Calcule la salida de la neurona y verfique si es correcto
salida = neuronaMyP(E, I, u)
print("La salida de la neurona es:", salida)
```

```
La salida de la neurona es: 1
```

In [4]:
```python
# TODO: (1 punto) Implemente un programa que reciba vectores arbitrarios de E, I y u y devuelva la salida de la neurona.

# Vectores de prueba
E = [1, 0, 1]
I = [0, 0, 0]
u = 2

def neuronaMyP(E,I,u):
    # E: Exitatoria (vector de 0s y 1s)
    # I: Inhibitoria de entrada (vector de 0s y 1s)
    # u: Umbral (escalar)

    # Calcular si existe un i en I que sea 1 para que la salida sea 0
    for i in I:
        if i == 1:
            return 0
    # Si no existe un i en I que sea 1, entonces hacemos la suma de todos los elementos de E y comparamos con el umbral u
    suma = 0
    for e in E:
        suma += e

    if suma >= u:
        return 1
    else:
        return 0

salida = neuronaMyP(E, I, u)
print("La salida de la neurona es:", salida)
```

```
La salida de la neurona es: 1
```