Fundado en la pandemia, hace 5 años.

Se estructura según semestres para introducir a el semillero.

**La fase inicial es:**

introducción a neuroinformática e IA

machine learning

deep learning

**La fase avanzada es:**

entrar a enseñar

entrar a sesiones de neuroinformática

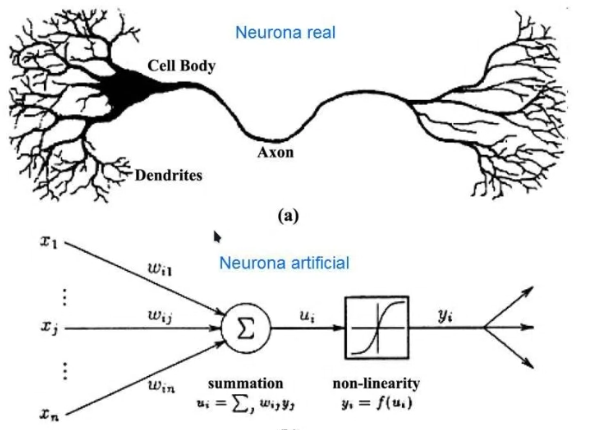
entrar a sesiones de laboratorio

Se suele utilizar el método de Feynman para enseñar y superar las falencias de conocimiento anteriores

**REDES NEURONALES**

son modelos computacionales inspirados en las neuronas reales, funcionan asi:

* Propagación hacia adelante: los datos fluyen desde la entrada hasta la salida, pasando por todas las capas de la red
* Función de activación: introduce la no linealidad (ej sigmoide) para modelar relaciones complejas
* Cálculo de la pérdida: mire el error entre la predicción de la red y el valor real esperado.
* Iteración: cuantas épocas y cuantas veces se usara la red durante su proceso de aprendizaje



**Dendrita:** entrada de la neurona

**Axion:** salida de la neurona

**Peso:** Es la importancia que la neurona le da a las neuronas de la capa anterior

**Bias:** Es como una personalidad de la neurona,evita el sobreentrenamiento y desvía la linealidad de los datos

**Obtención y procesamiento de los datos**

Hay que preparar la forma en cómo obtenemos y mandamos los datos a la red para que aprenda correctamente. Se debe eliminar el ruido de los datos para limpiar la entrada de la red

### **1. EEG (Electroencefalograma)**

* **Definición**: Técnica que mide la actividad eléctrica del cerebro a través de electrodos colocados en el cuero cabelludo.
* **¿Cómo se usa?**:
  + Se colocan electrodos en el cuero cabelludo.
  + Registra las ondas cerebrales generadas por la actividad neuronal.
* **¿Para qué se usa?**:
  + Diagnóstico de trastornos neurológicos como epilepsia.
  + Evaluación del sueño (trastornos del sueño, fase REM).
  + Investigación sobre la actividad cerebral y procesos cognitivos.

### **2. EOG (Electrooculograma)**

* **Definición**: Técnica que mide la actividad eléctrica de los movimientos oculares.
* **¿Cómo se usa?**:
  + Se colocan electrodos cerca de los ojos.
  + Detecta los cambios de potencial eléctrico relacionados con los movimientos del ojo.
* **¿Para qué se usa?**:
  + Estudio de los movimientos oculares (por ejemplo, en el sueño REM).
  + Diagnóstico de trastornos oculares (como nistagmo).
  + Control de dispositivos a través de los movimientos oculares (interfaz para personas con discapacidad).

### **3. EMG (Electromiograma)**

* **Definición**: Técnica que mide la actividad eléctrica de los músculos esqueléticos durante la contracción.
* **¿Cómo se usa?**:
  + Se colocan electrodos en la piel o directamente sobre los músculos.
  + Registra las señales eléctricas generadas por las fibras musculares al contraerse.
* **¿Para qué se usa?**:
  + Diagnóstico de trastornos musculares y nerviosos.
  + Evaluación de fatiga muscular y función muscular.
  + Rehabilitación y diseño de prótesis o dispositivos de asistencia.

La red tratada intenta reducir el ruido de entrada, para evitar errores en la lectura de resultados. Se le manda una muestra con ruido y una sin este, para que el modelo aprenda a obtener sobre la señal con ruido la parte importante, la señal sin ruido.

Los datos también hay que escalarlos a un rango pequeño, ya que los computadores calculan mejor con números pequeños que algo mucho más grande

A veces las señales tienen muchas características, pero a la hora de hacer estadística con estos datos, hay que tomar solamente las características relevantes para la evaluación a realizar

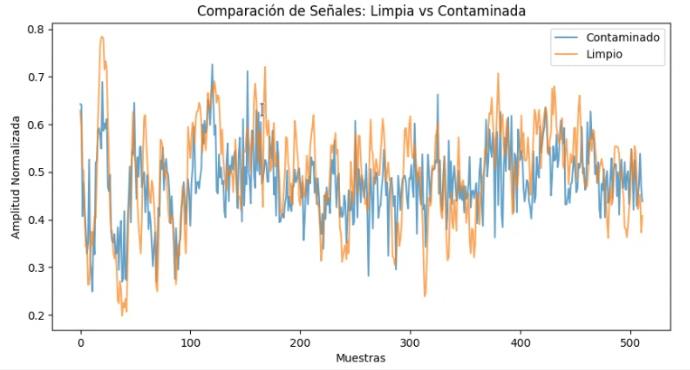
A la hora de entrenar el modelo no se deben usar todos los datos al tiempo al modelo, de esta manera el modelo simplemente se **aprende de memoria** estos datos y no **analiza** los datos, solo funcionara correctamente con estos datos de entrenamiento y no se generaliza al resto de muestras que se le puedan ingresar para probarlo.

(70-80 training / 20-30 testing) (70 training / 15 validation / 15 testing)

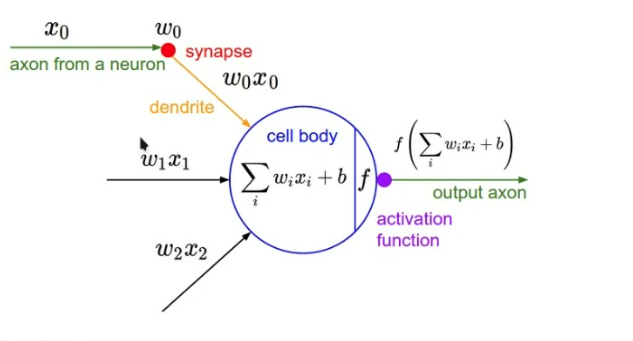
es lo que generalmente se usa para asegurarse de la buena generalización sobre los datos

**Hiper parámetro**

La red en sí tiene parámetros que no podemos modificar, como los pesos o los biases, pero los hiper parámetros si los puede modificar el usuario que maneja la red neuronal, como las épocas, learning rate, etc.



Podemos diferenciar entre ambas señales, la limpia parece ser más suave que la señal con ruido.



La neurona hace la sumatoria de la multiplicación de pesos y datos, después se le suma el bias, al final ese valor es la entrada de la funcion de activacion de la red, la cual es la salida de la neurona a todas las neuronas de la siguiente capa de neuronas.

**FUNCIONES DE ACTIVACIÓN**

En estos momentos la función **RELU** (rectified linear unit)es de las más utilizadas para la activación de las neuronas

La función **sigmoide** se utiliza para la clasificación binaria, obteniendo únicamente 0 o 1

**Tanh** es parecida a sigmoide, incluye valores negativos y por esto a veces es mejor usar sigmoide.

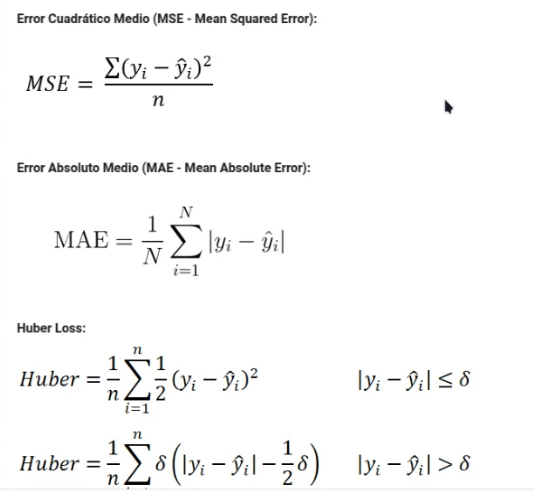
Debemos tener en cuenta las derivadas de las funciones de activación, ya que nos da una pista de cómo seguiría la función. Si el gradiente aumenta mucho, podría haber un problema de cancelación del resto de las neuronas y solo funcionaria una de estas, lo mismo ocurre si en algún momento la derivada es cero, esto inhabilita completamente la neurona y las que siguen.

Se usa una técnica para inhabilitar algunas neuronas, ya varias neuronas que están juntas, pueden hacerse “amigas” y no generalizar correctamente la función de la red neuronal

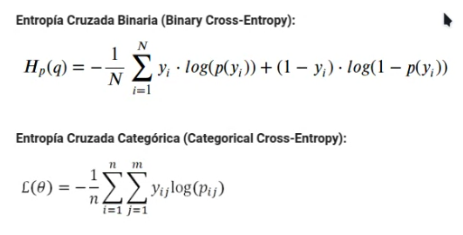
Esta es una arquitectura **une** con esto se le hace entrenar y empieza a aprender.

Una capa densa es una sola neurona en la capa y recibe todas las activaciones de la capa anterior. Se define su entrada de manera lineal. La salida de la red cambia su función de activación con respecto a las capas intermedias y la entrada

**FUNCIONES DE PÉRDIDA**



*La ‘ŷ’ es el valor obtenido y la ‘y’ es el valor esperado*



**OPTIMIZADORES**

Estos utilizan la derivada de la función de pérdida para llegar a un mínimo de la pérdida de datos modificando los pesos de las neuronas.

El gradiente descendente estocástico tiene un problema, puede ser muy lento ya que cada nueva iteración se reduce el tamaño de los pasos y no alcanza el mínimo correctamente Otro error posible es que se quede en una serie de puntos sin poder salir y encontrar un valor más cercano a el mínimo que intenta buscar.

No se debe dejar correr el algoritmo hasta siempre, debemos utilizar un limite para este, o usar el limite por defecto que este maneja.

Este maneja como parámetro un **learning\_rate**, que es el tamaño del paso que tomamos a la hora de buscar el mínimo de la función que estamos utilizando, si usamos un learning rate muy grande podemos saltarnos el mínimo global que estamos buscando en la función que estamos usando, si es muy pequeño puede tardar demasiado en hallar este mínimo.

A la hora de dar el salto en la función buscando el minimo, tomamos el learning\_rate y le multiplicamos un valor dado por el optimizador que nos dice el paso que daremos en la búsqueda del mínimo de la función.

**Métricas:** son maneras de evaluar la red neuronal en su desempeño mientras se entrena y valida los datos.

**Batch size:** nos dice el tamaño del lote que se va a utilizar para el entrenamiento por cada época

**Modelo:** maneja función de activación, función de pérdida y la métrica