Clase 1. Introducción a Redes Neuronales (FeedForward)

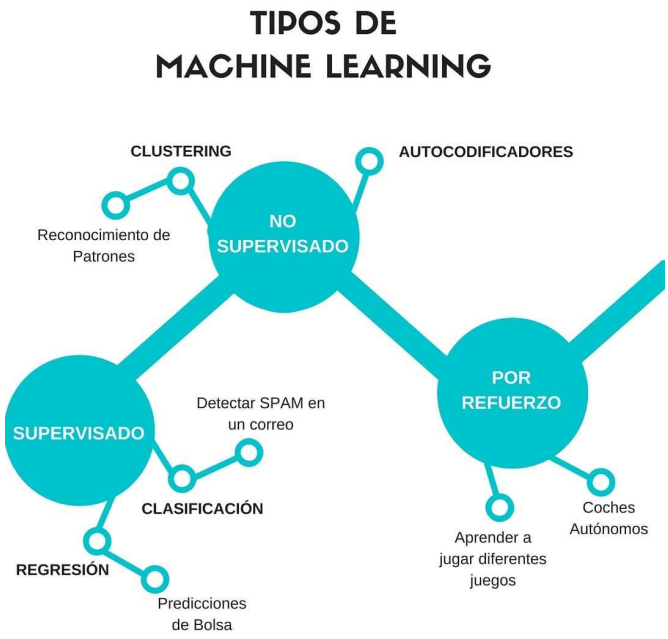
1. Introducción a AI

Inteligencia Artificial es un término muy ambiguo. Ha sido investigado en la filosofía desde los 20s y 30s, y en la computación desde sus orígenes, aunque tuvo su primera etapa en los 60s. Ha pasado por 3 periodos llamados AI Winters. En estos periodos, el hype que hubo alrededor de la investigación de AI murió y se dejó de hacer investigación de AI por años. En general, es un área con muchos campos: búsqueda, planeación, robótica, visión computacional y más. Aunque hay muchas definiciones, normalmente tiene que ver con una máquina que piensa, razona o actua “inteligentemente”. Entender qué significa hacer esto inteligentemente es algo complicado de entender. También es complicado diferenciar cosas que nos hacen humanos de inteligencia.

Por otro lado, **Machine Learning** es un campo más claro. Tiene que ver con que un programa aprenda a partir de datos o experiencia. Permite que las computadoras aprendan sin ser programadas explícitamente, es decir, que aprendan por su cuenta. Ha sido investigado desde los 70s, sido exitoso en los últimos 10 años, y tenido su auge en los últimos 5 años.

En los últimos 2-4 años, el área de Machine Learning más exitosa ha sido **Deep Learning**. Deep Learning es el enfoque de este curso. Aunque ha sido investigada desde los 80s, la capacidad de procesamiento y la acumulación de datos recién han permitido que funcione. Deep Learning tiene que ver con redes neuronales profundas.

2. Tipos de aprendizaje



Aprendizaje Supervisado

En el Aprendizaje Supervisado, utilizamos información (**dataset**) de entrenamiento. Por ejemplo, para saber si en una imagen tenemos un gato o un perro, entrenamos a nuestro modelo con miles de imágenes y le decimos cuáles son de perros y cuáles son de gatos (**labels**). Después de muchos ejemplos, dada una nueva imagen (sin labels), nuestro modelo podrá determinar si se trata de un gato o un perro. A este problema se le llama **clasificación**. Otro ejemplo es predecir un valor continuo. A partir del código postal, el tamaño y el número de cuartos de una casa, se puede predecir cuál será su valor. A este problema se le llama **regresión linear**. Lo que distingue el Aprendizaje Supervisado es que se entrena dando muchos ejemplos y, a partir de eso, puede generalizar para nuevos casos.

Aprendizaje No Supervisado

A diferencia de Aprendizaje Supervisado, aquí no tenemos un valor verdadero o label. Los modelos de Aprendizaje No Supervisado tienen el objetivo de comprender y abstraer los patrones de la información directamente. Por ejemplo, un modelo de Airbnb recibirá las ubicaciones de las casas, y determinará cómo separarlas en grupos. A este problema se le llama clustering. Aunque pueda sonar complejo, es muy parecido a cómo pensamos nosotros con nueva información. Por ejemplo, ¿cómo definimos nosotros cuáles son las constelaciones? A partir de observación de las estrellas, nosotros dedujimos ciertos patrones.

Aprendizaje por Refuerzo

En esta técnica nuestros modelos aprenden a partir de la experiencia. En un coche autónomo, cuando toma una mala decisión, se le “castiga”. A partir de sus premios y castigos, va aprendiendo a realizar su tarea mejor. Esta es una técnica de prueba y error, y de utilizar una función de premio que vaya optimizando. Esta es una de las técnicas más prometedoras porque no requieren grandes cantidades de datos. En esta técnica se está haciendo mucha investigación.

Lo interesante de Deep Learning es que permite crear redes neuronales que funcionan para estos 3 tipos de aprendizaje. Sus aplicaciones son muy extensas, como detección facial, predicción de acciones o subtitulado de imágenes.

3. Tipos de problemas de ML

Como mencionamos antes, hay diferentes tipos de problemas en Machine Learning como regresión lineal, clasificación, k-clustering y reducción de dimensiones. Por ahora nos vamos a enfocar en dos: regresión lineal y regresión logística.

- **Regresión lineal**

Como dice el nombre, el output es una línea, un valor continuo. Es decir, hay infinitas probabilidades. ¿Cuál será tu salario a partir de datos como dónde y qué estudiaste? ¿Cuánto valdrá tu casa a partir del número de cuartos?.

- **Regresión logística**

En este nos enfocaremos primero dado que tiene que ver mucho con clasificación. En la regresión logística tenemos posibilidades limitadas. ¿En este párrafo hay un sentimiento positivo o negativo? ¿En esta foto de nuestros clientes la persona está feliz o triste? ¿Este correo es spam? ¿Venderemos la casa por más de 200k?

Un ejemplo muy claro de regresión logística es el que hemos mencionado de aplicaciones de estudiantes. En este problema, a partir de un conjunto de features/entradas, determinamos si aceptamos a un aplicante.

Feature: Propiedad individual que se puede medir. Tú decides qué features definir. No es sencillo siempre. Hay una ingeniería llamada feature engineering que se trata de extraer, limpiar y definir features.

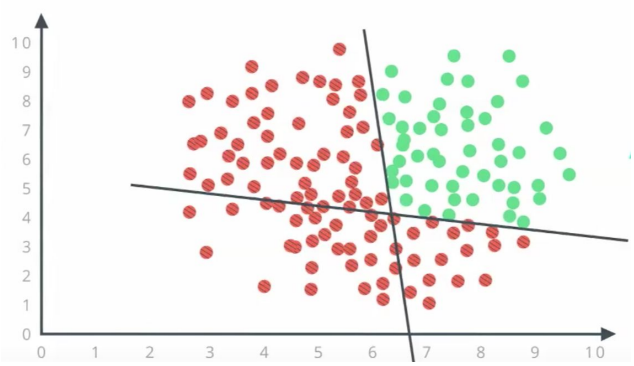
4. Evaluación y optimización de modelos

En el problema de los aplicantes, utilizamos una calificación de examen y el promedio del colegio para determinar si los aceptamos. Podemos crear un **modelo**, o una abstracción de la información (por decirlo con otras palabras, reglas que nos permitan describir un patrón o comportamiento), que nos permita distinguir a partir de los features si aceptamos o no a un aplicante. Según el tipo de información, con una línea podría ser suficiente. Otros problemas pueden requerir círculos, dos líneas o parábolas.

Algo importante para **evaluar** nuestro modelo es calcular el **error.** Nosotros como ingenieros debemos definir cómo calculamos el error. En el caso de clasificación, podemos simplemente contar el número de errores. Más adelante en el curso veremos diferentes funciones que nos permiten calcular el error.

Cuando entrenamos nuestro modelo y calculamos su error, es importante tener una manera de **optimizar** para reducir ese error. Un algoritmo de optimización muy común se llama **Gradient Descent**. Gradient Descent es muy parecido a bajar del pico de una montaña. Imagina que eres un ciego que está en una cima y quiere llegar al punto más bajo. ¿Cómo lo logras? A partir de la pendiente del punto en el que estás, vas dando pequeños pasos. El problema con Gradient Descent es que a veces llegar a mínimos locales en vez de mínimos globales. Hay variantes de GD que permiten prevenir esto.

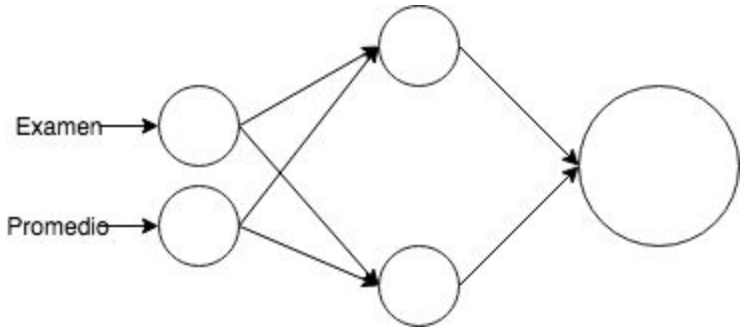
5. Redes Neuronales



Imagina que tu modelo debe ser descrito con dos líneas. Una red neuronal es una excelente manera de representarlo. Podemos hacer tres preguntas

- ¿El punto está arriba de la primera línea?
- ¿El punto está a la derecha de la otra línea?
- ¿La respuesta a las otras dos preguntas fue sí?

Si te das cuenta, la tercera pregunta es una combinación de las primeras dos. Esto también lo podemos representar de la siguiente manera:



Regresando a cómo funciona una neurona biológicamente,
Una neurona está compuesta por 3 componentes principales

- Nucleo
- Dendrita
- Axón

Equivalentemente, una neurona artificial tiene 3 componentes

- Input - recibe los datos (pulso eléctrico)
- Función de activación - la manera en la que procesa los datos
- Output - pulso eléctrico de salida

Ahora, ¿todos los features son igual de importantes? Volvamos al caso donde sólo tenemos dos features. ¿Queremos dar la misma importancia al examen de admisión que a la calificación del colegio? Probablemente no. Se le puede asignar un peso a cada feature, denotado por w. Podemos multiplicar el valor del feature por este peso, y eso nos va a dar un valor. Por ejemplo, digamos que el examen es más importante. Eso nos da la siguiente fórmula:

V = 0.5X1 + 2X2
Con esto podemos determinar un valor para cada alumno. Después, podemos decir a partir de qué valor aceptamos a los aplicantes. A este valor se le llama threshold.

Podemos generalizar el valor de una neurona con la siguiente fórmula

$$\sum_{i=1}^{m} w_i x_i$$

Lo bonito de Deep Learning es que la red neuronal va ajustando los pesos y el bias a partir de muchos ejemplos. Ahora nosotros lo hicimos de una manera algo aleatoria. La red neuronal los va ajustando lentamente para optimizar el objetivo. En teoría, una red neuronal puede aprender cualquier función.

6. Perceptron

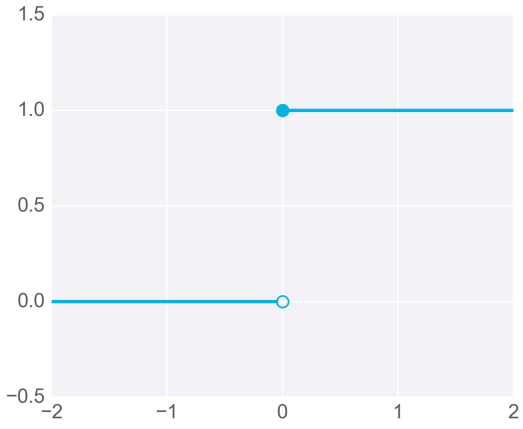
En un perceptrón, se asigna un límite. Si el valor de entrada de la neurona (**combinación linear**), es más alto que el threshold, se enciende la neurona (1).

$$\text{output} = \begin{cases} 0 & \text{if } \sum_j w_j x_j \leq \text{ threshold} \\ 1 & \text{if } \sum_j w_j x_j > \text{ threshold} \end{cases}$$

Para hacer la notación más amigable, podemos denotar la sumatoria como un producto punto de vectores, y, diciendo que bias = -threshold, tener el siguiente resultado:

$$\text{output} = \begin{cases} 0 & \text{if } w \cdot x + b \leq 0 \\ 1 & \text{if } w \cdot x + b > 0 \end{cases}$$

El perceptrón es muy sencillo. Tiene dos valores de salida: 0 y 1. Esto se representa facilmente con una función escalón. Nota que el 0 es inclusivo (el gráfico está mal)



7. Funciones de Activación

Mientras que en el perceptrón utilizamos la función escalón como función de activación, en una neurona podemos utilizar todo tipo de valores. Hay funciones que describen mejor otros tipos de comportamientos. Por ejemplo, la función sigmoide da valores entre 0 y 1 y eso la hace buena para describir probabilidades. Algo importante de notar es que la función debe ser derivable y continua.

$$S(x) = \frac{1}{1 + e^{-x}} = \frac{e^x}{e^x + 1}.$$

Recuerda que la entrada de la función de activación es la combinación linear. Es decir:

- Tenemos los pesos representados por W
- Tenemos las entradas representada por X
- Tenemos el bias representado por b
- La combinación linear es el producto punto de W y x, y se le suma b.
- Esta es la entrada de nuestra función de activación.

8. Redes Neuronales

Una red neuronal tiene 3 componentes

- Input layer - su tamaño debe ser el número de features.
    - En este caso, hablamos del promedio del colegio y de la calificación del examen.
- Hidden layer - aprende el comportamiento más complejo.
- Output layer - en este caso, su tamaño es el número de clases. Cada neurona representa una clase y la que se encienda más fuerte indica la clase elegida.