

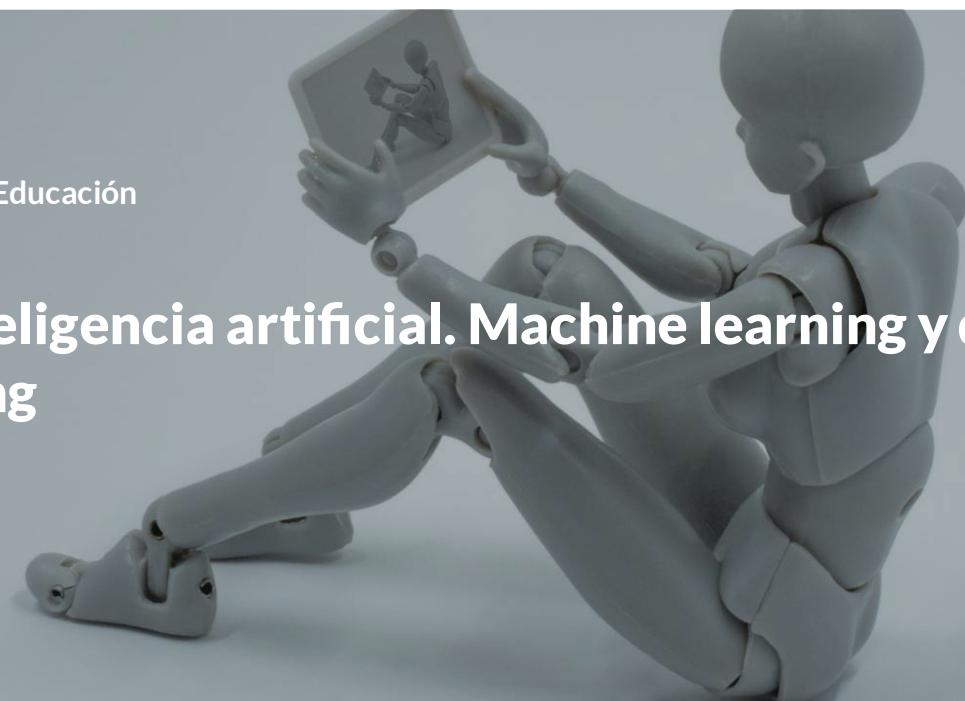
Fastbook 02

Big Data y Otras Técnicas Avanzadas Aplicadas a CRO

Inteligencia artificial. Machine learning y deep learning



02. Inteligencia artificial. Machine learning y deep learning



En este fastbook trataremos de explicar las diferencias entre machine learning y deep learning.

Adicionalmente, también mostraremos los principales ejemplos y tipos de cada una de estas disciplinas.

Así, dentro del machine learning profundizaremos en:

- Aprendizaje supervisado.
- Aprendizaje semisupervisado.
- Aprendizaje no supervisado.
- Aprendizaje por refuerzo.

Y continuaremos con la explicación de qué es una red neuronal (deep learning) y sus principales tipos.

El objetivo de este fastbook es que aprendas a diferenciar entre **inteligencia artificial, machine learning y deep learning**.

Autor: Eugenio Alamillos

— Inteligencia artificial vs. machine learning vs. deep learning

— Machine learning

— Deep learning

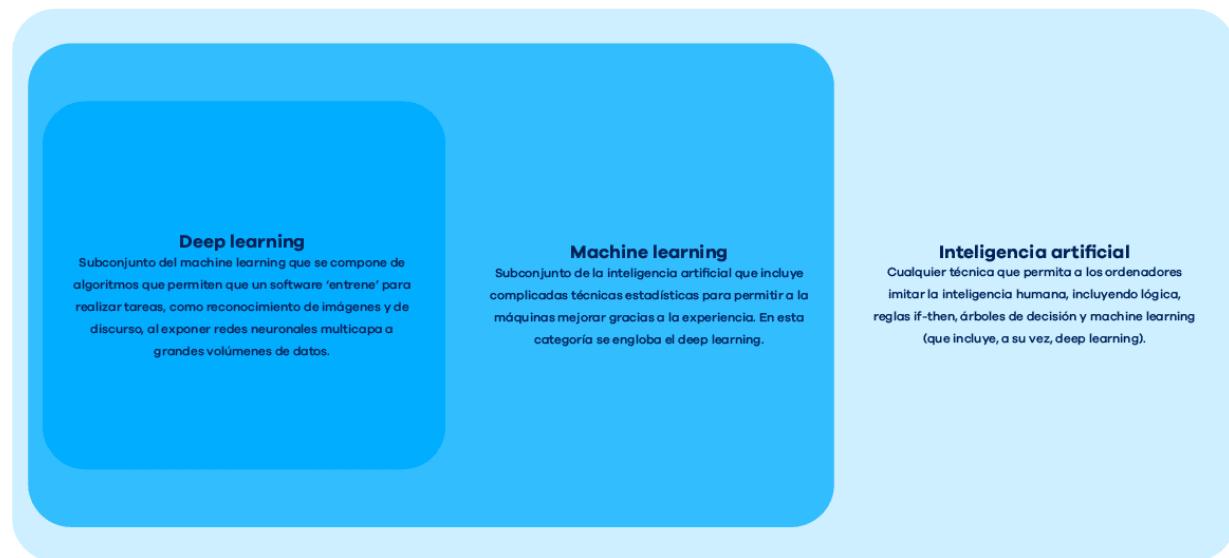
Lección 1 de 3

Inteligencia artificial vs. machine learning vs. deep learning

X Edix Educación

Como ya se habló en el fastbook anterior, hemos llegado a la conclusión de que inteligencia artificial no es únicamente redes neuronales ni machine learning. **La inteligencia artificial se divide en diferentes campos**, y uno de ellos es el machine learning o aprendizaje automático.

Al igual que sucede con la IA, el machine learning se divide a su vez en subcampos, entre los cuales figura el deep learning.



División de inteligencia artificial, machine learning y deep learning. Fuente: [What Is Machine Learning & Deep Learning? | by Claire D. Costa](#).

Machine learning

X Edix Educación

Como ya se ha explicado, en machine learning **los algoritmos son entrenados para encontrar patrones en grandes cantidades de datos**, con el objetivo de que sean capaces de **hacer predicciones o encontrar una solución** a un problema planteado.

El machine learning o aprendizaje automático otorga a las personas grandes beneficios hoy en día, por lo que su uso crece más rápido que nunca. Sin embargo, ha tenido que recorrer un largo camino hasta que ha logrado volverse popular.

El término *machine learning* (aprendizaje automático) fue acuñado por primera vez por el pionero de la inteligencia artificial y los juegos de ordenador Arthur Samuel, en 1959. Sin embargo, Samuel escribió el primer programa de aprendizaje informático cuando trabajaba en IBM en 1952. El programa era un juego de damas en el que el ordenador mejoraba cada vez que jugaba, ya que iba analizando qué jugadas componían una estrategia ganadora.

En 1957, Frank Rosenblatt creó la **primera red neuronal artificial**, también conocida como **perceptrón**, que fue diseñada para simular los procesos de pensamiento del cerebro humano.

En **1967**, se diseñó el algoritmo del ‘vecino más cercano’ (*k-nearest neighbor*), que marca el inicio del reconocimiento de patrones básicos mediante ordenadores.

Estos primeros descubrimientos fueron importantes, pero la falta de aplicaciones útiles y la limitada potencia de cálculo de la época provocaron un largo periodo de estancamiento en el aprendizaje automático y la IA hasta la década de **1980**.

Entre los **años 80 y principios de los 90**, el aprendizaje automático y la inteligencia artificial son prácticamente lo mismo. Es en los 90 cuando los investigadores comienzan a encontrar nuevas y más prácticas aplicaciones para las técnicas de resolución de problemas que habían creado.

El mirar más allá hizo que el machine learning abriera las puertas a nuevas aproximaciones que se basaban más en estadística y probabilidad que en el comportamiento humano y biológico.

Hoy en día, el machine learning está **integrado en un número importante de aplicaciones** y afecta a millones (quizá miles de millones) de personas cada día. La enorme cantidad de investigación sobre el aprendizaje automático dio como resultado el desarrollo de muchos enfoques nuevos, así como una variedad de nuevos casos de uso para el machine learning.

En realidad, las técnicas de machine learning pueden utilizarse en cualquier lugar en el que se necesite analizar una gran cantidad de datos, lo que es una necesidad común en los negocios.

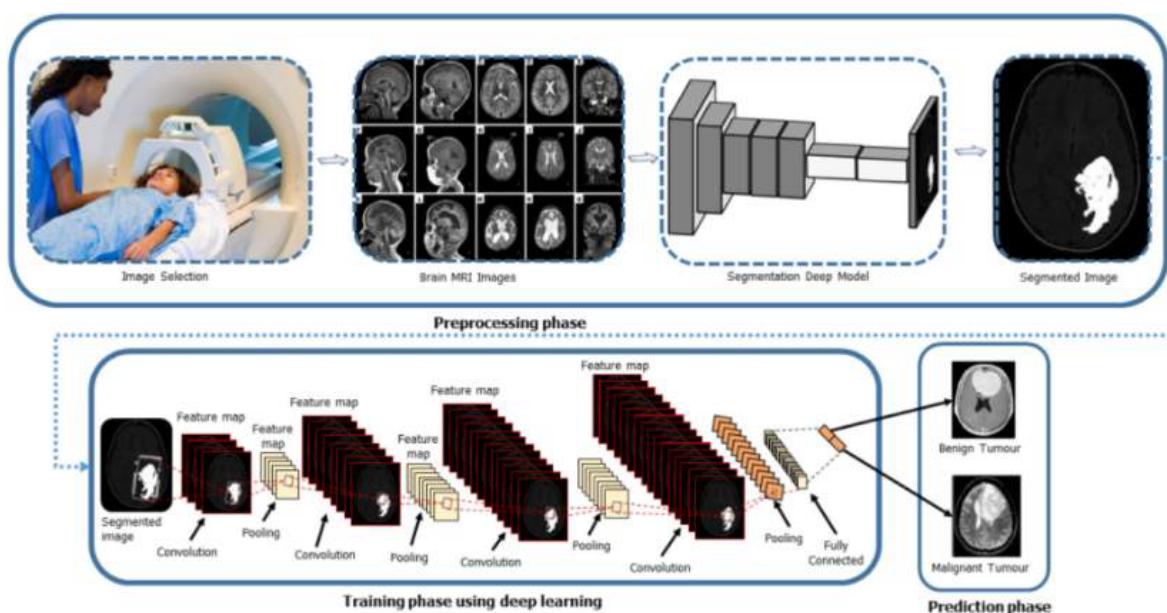
Las tres razones principales que han liderado la adopción en masa del machine learning en las aplicaciones empresariales y de investigación son:

- La capacidad de cálculo ha aumentado significativamente y es mucho más barata que en las últimas décadas.
- La expansión de internet ha hecho que la información sobre los beneficios y los casos de uso del machine learning se haya difundido.
- Las herramientas de machine learning de código abierto están cada vez más disponibles.

Las compañías tecnológicas dedicadas a la logística, la biomedicina y a casi cualquier otro tipo de industria están haciendo uso del machine learning en casi todas sus **aplicaciones**.

- **Google** usa el machine learning para entender qué es lo que los usuarios quieren buscar en su motor de búsqueda.
- **Waymo**, subsidiaria de Alphabet (padre de Google), emplea machine learning en sus vehículos para poder lograr así el nivel 5 de autonomía de un coche.
- Google usa también el machine learning para mejorar sus resultados al medir el porcentaje de acierto que tiene con los resultados que devuelve.

- Muchos clientes de **correo electrónico** utilizan machine learning para detectar qué correos son spam.
- Los **bancos** lo usan para detectar transacciones y comportamientos que pueden ser sospechosos o fraudulentos.
- Las organizaciones de **investigación médica** están utilizando el aprendizaje automático para analizar enormes cantidades de datos en un intento de identificar patrones en enfermedades y afecciones y mejorar la atención sanitaria. De hecho, a día de hoy, la detección temprana de tumores mediante computer vision es notablemente más alta que la realizada por un humano.



Framework generalizado basado en deep learning para la detección de tumores en el cerebro. Fuente: *Journal of Infection and Public Health*.

Dentro del campo del machine learning, podemos distinguir varios tipos de aprendizaje: supervisado, no supervisado, semisupervisado y aprendizaje por refuerzo.

Vamos a ver los tipos de aprendizaje uno a uno.

Aprendizaje supervisado



Ilustración 3: Aprendizaje supervisado. Fuente: Nvidia

Si estás aprendiendo a hacer algo nuevo bajo la supervisión de alguien, esta persona juzgará si lo estás haciendo del modo correcto.

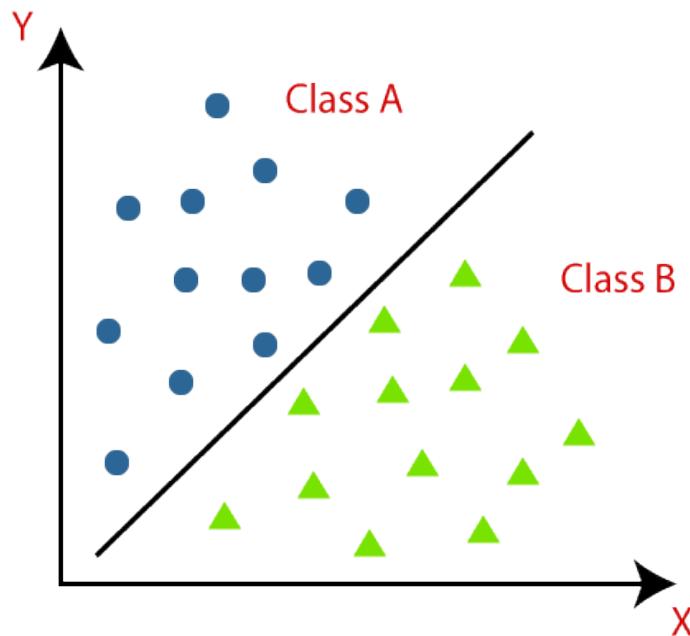
En esencia, esto es el aprendizaje supervisado, tener etiquetados todos los datos sabiendo *a priori* cuál va a ser la salida de estos, en otras palabras, cuál será el resultado.

Tener un conjunto de datos totalmente etiquetado significa que cada ejemplo del conjunto de datos de entrenamiento está **etiquetado con la respuesta que el algoritmo debería dar por sí mismo**. Así, un conjunto de datos etiquetados de imágenes de flores indicaría al modelo qué fotos son de rosas, claveles o tulipanes. Cuando se muestra una nueva imagen, el modelo la **compara con los ejemplos de entrenamiento** para predecir la etiqueta correcta.

Hay dos áreas principales en las que el aprendizaje supervisado es muy útil: los **problemas de clasificación** y los **problemas de regresión**.

Los **problemas de clasificación** piden al algoritmo que prediga un valor discreto, identificando los datos de entrada como miembros de una clase o grupo concreto.

Teniendo un conjunto de datos de entrenamiento de razas de perro (en el conjunto de entrenamiento ya tenemos cada foto etiquetada), sabemos desde un primer momento si se trata de un labrador, un mastín o un pastor alemán. El algoritmo dictaminará con un porcentaje de confidencia si cada foto nueva se trata de un labrador, un mastín o un pastor alemán.



Problema de clasificación. Fuente: *Javatpoint*

Los siguientes son algunos **algoritmos de clasificación**.

1

Regresión logística

Considera un escenario donde se necesite clasificar si un correo se trata de correo no deseado o no. Si se utiliza la regresión lineal para este problema, es necesario establecer un umbral a partir del cual se pueda realizar la clasificación. Digamos que, si la clase real es correo no deseado, el valor continuo predicho por el algoritmo es 0,4 y el valor del umbral es 0,5; el punto de datos se clasificará como no maligno, lo que puede acarrear graves consecuencias en tiempo real. Es decir, no sería un buen método para aplicar en los problemas de clasificación.

La regresión lineal no tiene límites, y esto hace que la regresión logística entre en escena. **Su valor oscila estrictamente entre 0 y 1.**

2

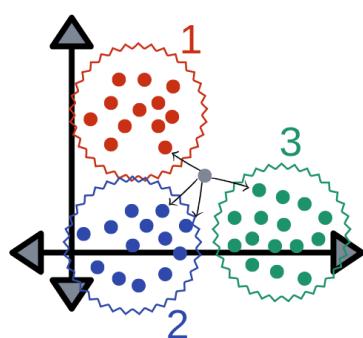
Naïve bayes

Un clasificador naïve bayes es un algoritmo que usa el teorema de Bayes para clasificar objetos.

3

KNN (*K vecinos más cercanos*)

KNN es un modelo que clasifica distintos *data points* basados en los puntos que son más similares a él. Utiliza los datos de prueba para hacer una suposición adecuada sobre cómo debe clasificarse un punto no clasificado.

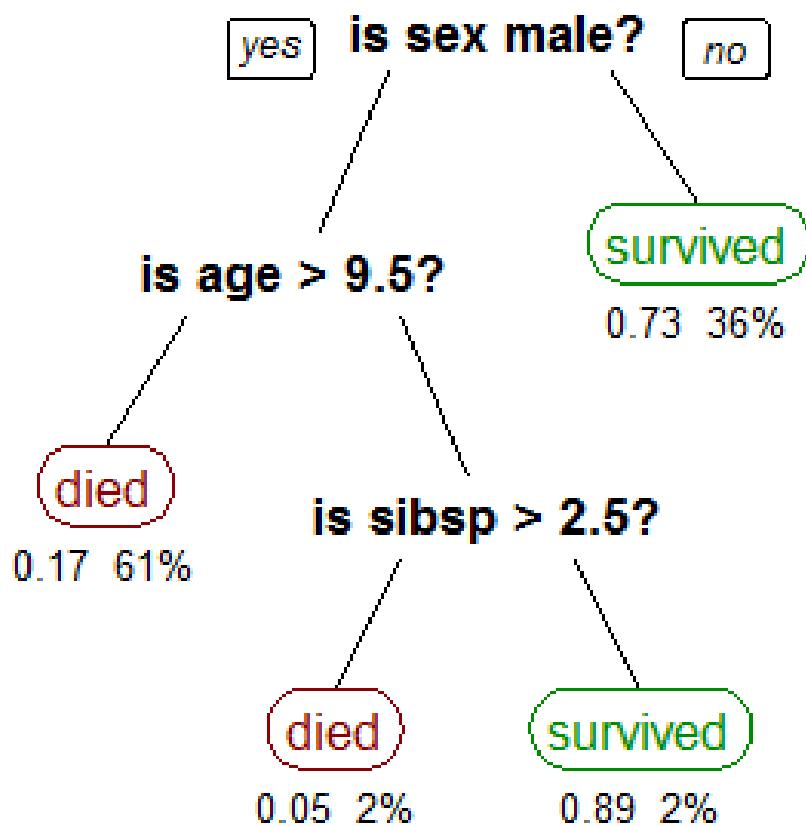


Representación de KNN. Fuente: Medium.

4

Árboles de clasificación

Un árbol de clasificación se puede usar para visualizar qué y por qué está tomando esas decisiones el algoritmo.

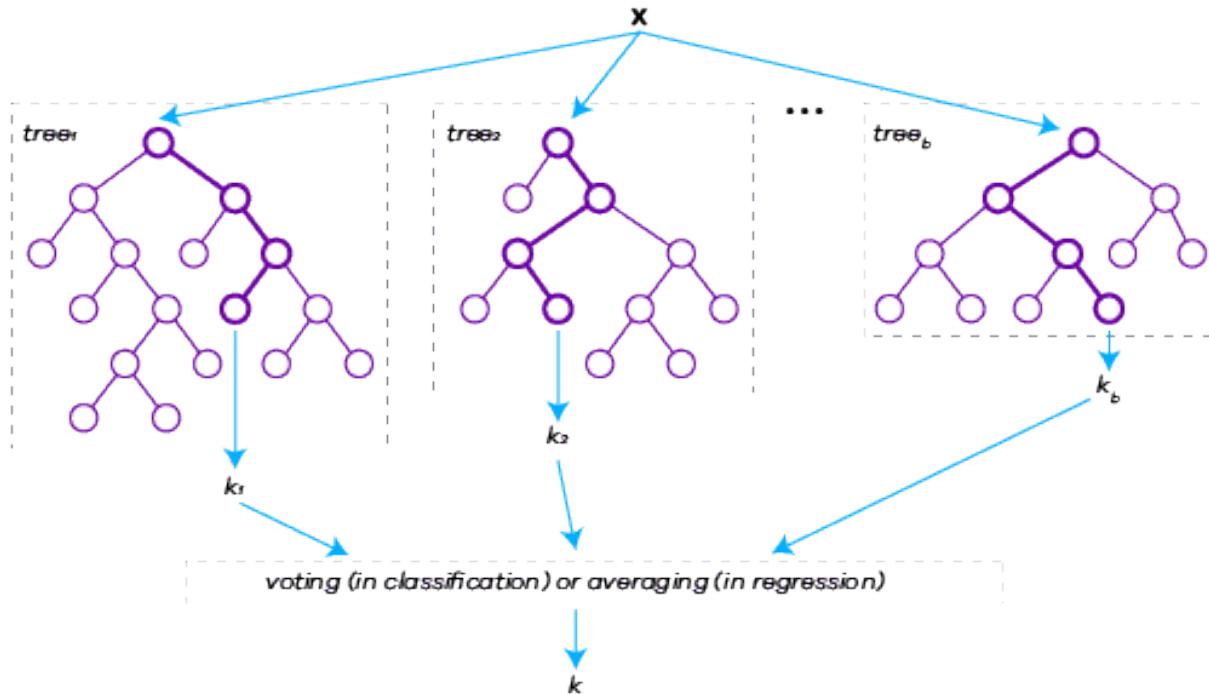


Árbol de clasificación. Fuente: TowarsDataScience.

5

Random forest

Random forest es un método de aprendizaje de conjunto para la clasificación, la regresión y otras tareas. Funciona mediante la construcción de una multitud de árboles de decisión en el momento del entrenamiento y la salida de la clase, que sería el modo de las clases (clasificación) o la predicción media / promedio (regresión) de los árboles individuales.



Random Forest. Fuente:ResearchGate.

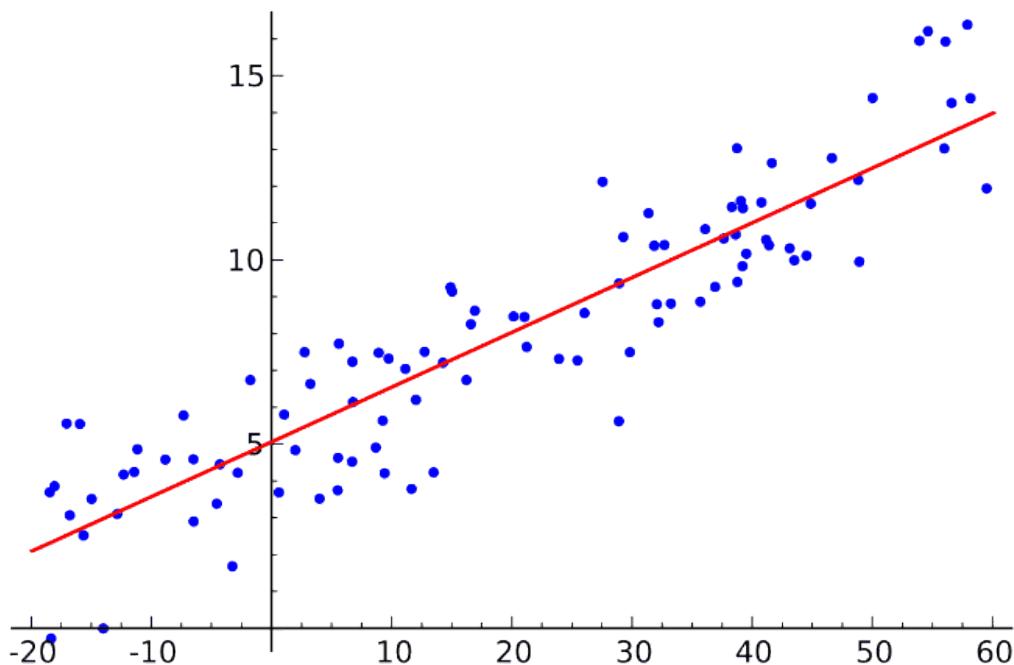
6

Support vector machine (SVM)

Esta es una de las técnicas de machine learning que todavía se utilizan en problemas de clasificación aplicados al big data, especialmente si se trata de un problema multidominio. El problema de las SVM es que son matemáticamente complejas y caras computacionalmente hablando.

Se trata de un algoritmo que aprende a asignar etiquetas mediante el ejemplo. En esencia, SVM es una entidad matemática, **un algoritmo para maximizar una particular función matemática con respecto a unos datos dados**.

Por otra parte, los problemas de regresión tienen en cuenta datos continuos. Un caso de uso, la regresión lineal: dado un valor particular de x , ¿cuál es el valor esperado de la variable y ?

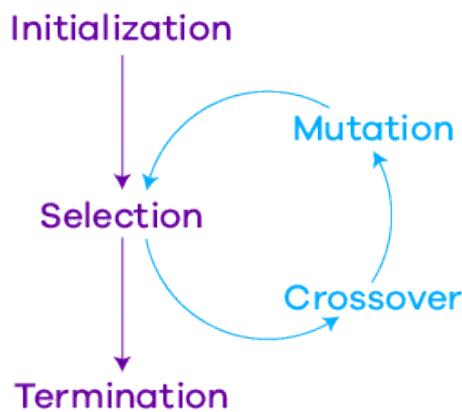


Regresión lineal. Fuente: Wikipedia.

7

Computación evolutiva

La computación evolutiva (CE) es un área de investigación dentro de la informática. Se inspira en los procesos naturales de la evolución natural. La metáfora fundamental de la CE relaciona el potencial de la evolución natural con un estilo de resolución de problemas particular, prueba y error.



Algoritmos evolutivos. Fuente: TowardsDataScience.

La idea de aplicar principios darwinianos surge en la década de 1940, mucho antes de la aparición de los primeros ordenadores. Los **algoritmos genéticos (AAGG)** presentan una analogía directa con el proceso evolutivo de selección natural presentado por Darwin. Explotan el proceso de la selección natural en función de una medida de adaptación o *fitness* para crear una población de soluciones de prueba que mejore con el tiempo.

A cada individuo le es asignado un valor de adaptación en función de cómo de buena es la solución que aporta al problema. Aquellos individuos que poseen un mejor valor de adaptación tienen una mayor probabilidad de poder reproducirse, esto se consigue al cruzarlos con otros individuos de la población. La descendencia de estos individuos comparte características de los padres, lo que da lugar a la incorporación de nuevos individuos a la población con una proporción mayor de individuos mejor adaptados. La **programación genética (PG)** aparece como una evolución de los AAGG existentes, manteniendo la base en la que se sustentan, el principio de la selección natural.

La novedad que introduce es la posibilidad de desarrollar de forma automática programas, es decir, algoritmos que, en función de una serie de entradas, generan una serie de salidas. La creación de estos programas se hace de forma totalmente aleatoria, no han sido diseñados previamente para que a partir de un problema propuesto generen una determinada solución, sino que esta solución se alcanza partiendo de un estado inicial dado y mediante mecanismos de selección. En este tipo de programación existe un problema fundamental, que es el problema del cierre. Se trata de la generación de nuevos individuos no válidos.

Para solventar este problema se hace uso de la **programación genética guiada por gramáticas (PGGG)**, que es una extensión de la PG y surge de la aplicación de una gramática de contexto (*context-free grammar*) dentro de un programa genético para asegurar la validez sintáctica de los individuos.

Aprendizaje semisupervisado

El aprendizaje semisupervisado es en realidad lo mismo que el aprendizaje supervisado, excepto que solo algunos datos del conjunto de entrenamiento están etiquetados.

El reconocimiento de imágenes es un buen ejemplo de aprendizaje semisupervisado. En este caso, podemos proporcionar al sistema varias imágenes etiquetadas que contengan objetos que deseamos identificar, y luego procesar muchas más imágenes no etiquetadas en el proceso de entrenamiento.

Aprendizaje no supervisado

En el aprendizaje no supervisado, el algoritmo no recibe ninguna etiqueta, por lo que **tiene que encontrar por sí mismo la estructura de los datos de entrada**.

El aprendizaje no supervisado puede ser un **objetivo en sí mismo** (descubrir patrones ocultos en los datos) o un **medio para alcanzar un fin** (aprendizaje de características).

En algunos problemas de reconocimiento de patrones, los datos de entrada se componen de un conjunto de vectores X sin ninguna correspondencia entre los valores del objetivo. El objetivo, como hemos mencionado, puede ser encontrar grupos de ejemplos con similares características dentro de los datos —aquí es donde se le llama **clúster**— o determinar cómo los datos están distribuidos en el espacio, conocido como **estimación de densidad**.

Podemos encontrar algunos **problemas en el aprendizaje no supervisado**, por ejemplo:

- Este tipo de aprendizaje es más costoso comparado con las tareas de aprendizaje supervisado.
- No podemos saber si los resultados son significativos, ya que no disponemos de etiquetas de respuesta.
- Tenemos que permitir que el experto vea los resultados (evaluación externa).
- Y definir una función objetivo sobre la agrupación (evaluación interna).

A pesar de ellos, el aprendizaje no supervisado **sigue siendo una opción** por los siguientes motivos:

- Anotar grandes conjuntos de datos es muy costoso, por lo que solo podemos etiquetar manualmente unos pocos ejemplos. Por ejemplo: Speech Recognition.
- Puede haber casos en los que no sepamos en cuántas o en qué clases están divididos los datos. Por ejemplo: la minería de datos.
- Es posible que queramos utilizar el clustering para conocer la estructura de los datos antes de diseñar un clasificador.

El aprendizaje supervisado se puede clasificar en **dos categorías: paramétrico y no paramétrico**.

1

Aprendizaje supervisado paramétrico

Se asume una distribución paramétrica de los datos. Supone que los datos de la muestra proceden de una población que sigue una distribución de probabilidad basada en un conjunto fijo de parámetros. Teóricamente, en una familia de distribuciones normales, todos los miembros tienen la misma forma y están parametrizados por la media y la desviación estándar. Esto significa que, si se conoce la media y la desviación estándar, y que la distribución es normal, se conoce la probabilidad de cualquier observación futura.

El aprendizaje paramétrico no supervisado implica la construcción de modelos de mezclas gaussianas (mezclas normales) y el uso del algoritmo de maximización de expectativas para predecir la clase de la muestra en cuestión. Este caso es mucho más difícil que el aprendizaje supervisado estándar, porque no hay etiquetas de respuesta disponibles y, por tanto, no hay una medida correcta de precisión disponible para comprobar el resultado.

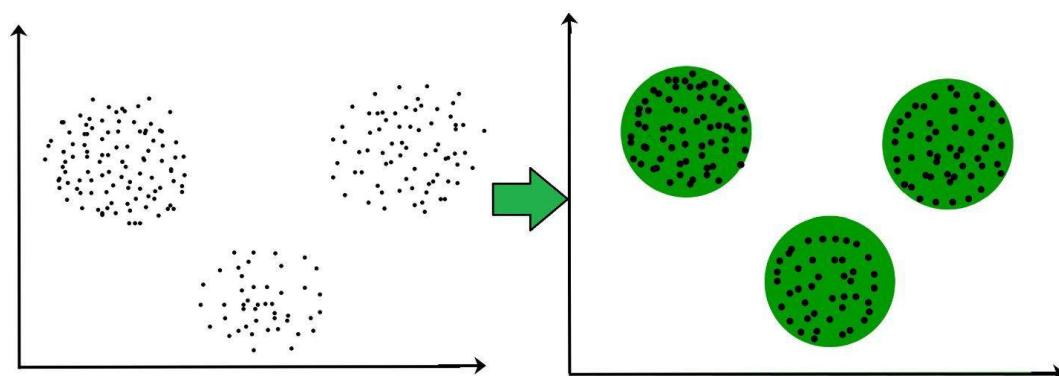
2

Aprendizaje no supervisado no paramétrico

En la versión no parametrizada del aprendizaje no supervisado, los datos se agrupan en clústeres, donde cada clúster dice algo (con suerte) sobre las categorías y clases presentes en los datos. Este método se suele utilizar para modelar y analizar datos con tamaños de muestra pequeños. A diferencia de los modelos paramétricos, los modelos no paramétricos no requieren que el modelador haga ninguna suposición sobre la distribución de la población, por lo que a veces se denominan **métodos sin distribución**.

Ahora bien, ¿qué es el **clustering**? Se puede considerar la parte más importante del aprendizaje no supervisado. Una definición general de clustering podría ser "el proceso de organizar objetos en grupos cuyos miembros son similares de alguna forma".

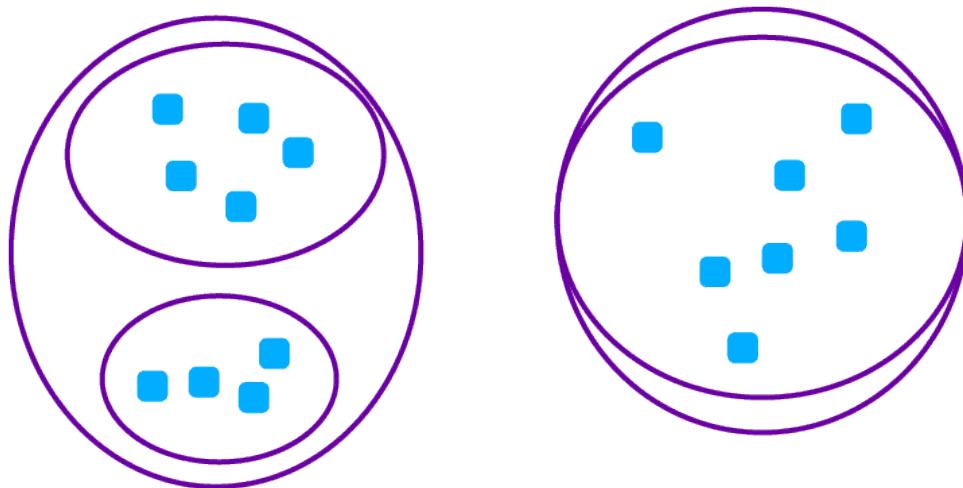
Un clúster es, por tanto, una colección de objetos que son similares entre sí, pero distintos a los objetos que pertenecen a otros clústeres.



Ejemplo de clustering. Fuente: GeeksforGeeks.

Se puede dar el caso de que haya más de una agrupación, entonces, ¿cómo sabemos qué constituye un buen clustering? No hay un criterio válido para todos, más bien depende de lo que se quiera conseguir con cada clustering.

En esta ilustración, ¿cómo sabemos cuál es la mejor solución? Para encontrar una solución particular, necesitamos definir las medidas de similitud para los clústeres.



Ejemplo de múltiples clústeres. Fuente: towardsdatascience.

Los cuatro algoritmos más utilizados para el clustering son:

- K-means.
- Fuzzy K-means.
- Hierarchical clustering.
- Mixture of Gaussians.

Aprendizaje por esfuerzo

Por otro lado, también tenemos el aprendizaje por refuerzo. El aprendizaje por refuerzo es el **entrenamiento de modelos de machine learning para tomar una secuencia de decisiones**.

El agente aprende a alcanzar un objetivo en un entorno incierto y potencialmente complejo. En el aprendizaje por refuerzo, el algoritmo se enfrenta a una situación similar a un juego. El ordenador emplea el método de ensayo y error para encontrar una solución al problema. Para conseguir que la máquina haga lo que el programador quiere, el algoritmo recibe recompensas o penalizaciones por las acciones que realiza. Su objetivo es maximizar la recompensa total.

Aunque el diseñador establece la política de recompensas —es decir, las reglas del juego—, no da al modelo ninguna pista o sugerencia sobre cómo resolverlo. Es el modelo el que **debe averiguar cómo realizar la tarea para maximizar la recompensa**, empezando por pruebas totalmente aleatorias y terminando con tácticas sofisticadas y habilidades sobrehumanas.

Al aprovechar el poder de la búsqueda y de muchos ensayos, el aprendizaje por refuerzo es actualmente **la forma más eficaz de insinuar la creatividad de las máquinas**.

El ejemplo más claro de la aplicación de este tipo de aprendizaje es el que utilizó Google para entrenar a su modelo **AlphaGo Zero**. El sistema comienza con una red neuronal que no sabe nada sobre el juego del Go. A continuación, juega partidas contra sí mismo, combinando esta red neuronal con un potente algoritmo de búsqueda. A medida que juega, la red neuronal se ajusta y actualiza para predecir las jugadas, así como el eventual ganador de las partidas.

A continuación, esta red neuronal actualizada se recombina con el algoritmo de búsqueda para crear una versión nueva y más potente de AlphaGo Zero, y el proceso comienza de nuevo. En cada iteración, el rendimiento del sistema mejora en una pequeña cantidad, y la calidad de las partidas autodidactas aumenta, dando lugar a redes neuronales cada vez más precisas y a versiones cada vez más fuertes de AlphaGo Zero.

AlphaGo es el primer programa informático que ha derrotado a un jugador humano profesional de Go y el primero en derrotar a un campeón del mundo de Go.

La historia completa la podemos encontrar perfectamente documentada en este enlace:
[AlphaGo](#).

Lección 3 de 3

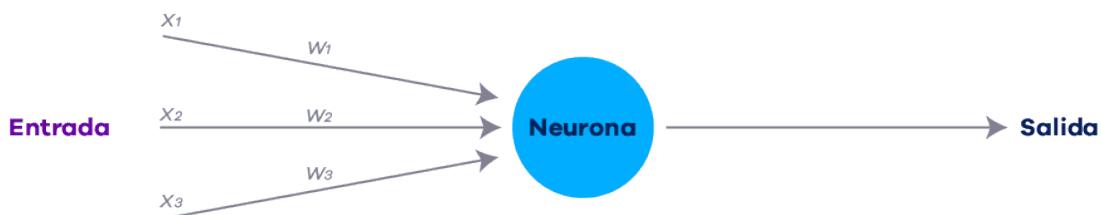
Deep learning

X Edix Educación

Dentro del machine learning encontramos el deep learning.

Deep learning hace referencia a la familia de algoritmos de machine learning que hacen **uso intensivo de las redes neuronales artificiales**.

Una **red neuronal artificial** es un modelo computacional basado en una red neuronal biológica, como las del cerebro humano. Utiliza una serie de funciones para procesar una señal o un archivo de entrada y traducirlo en varias etapas a la salida esperada. Este método se utiliza a menudo en el **reconocimiento de imágenes**, la **traducción de idiomas** y otras aplicaciones comunes hoy en día.



Ejemplo de red neuronal. Fuente: TowardsDataScience.

¿De qué se compone una red neuronal?

La entrada de la neurona es x , que tiene asociado un peso w . El peso es el parámetro intrínseco, el parámetro sobre el que el modelo tiene control para conseguir un mejor ajuste de la salida. Cuando pasamos una entrada a una neurona, la multiplicamos por su peso, lo que nos da:

$$x * w$$

El segundo elemento se llama **sesgo**. El sesgo está determinado únicamente por el valor b (bias). El sesgo añade un elemento de imprevisibilidad a nuestro modelo, que le ayuda a generalizar y le da la flexibilidad para adaptarse a diferentes entradas no vistas cuando se utilizan datos de prueba.

La combinación del sesgo y la entrada produce nuestra salida, lo que nos da la fórmula:

$$w*x + b = y$$

Esto debería resultar familiar como una modificación de la ecuación de una línea recta, $y=mx + c$. Las redes neuronales están **formadas por decenas, cientos o incluso miles de neuronas interconectadas**, cada una de las cuales ejecuta su propia regresión. Es esencialmente **una regresión con esteroides**.

Naturalmente, no podremos analizar la mayoría de los conjuntos de datos que encontramos en el mundo real utilizando una regresión tan simple como el ejemplo anterior. Es de esperar que haya muchas más entradas que se combinen para estimar la salida.

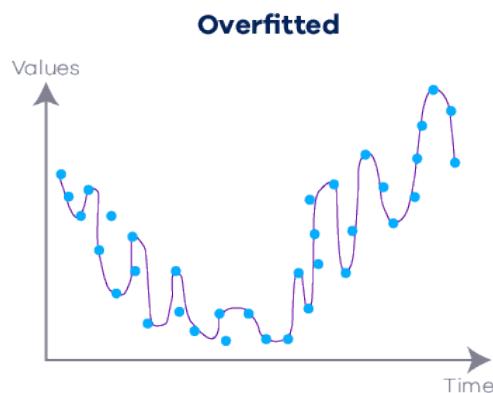
Las redes neuronales están **compuestas por capas**. Una capa en la que cada neurona está conectada con todas las neuronas de la capa siguiente se llama **dense layer**.

Para obtener una salida hay que aplicarle una **función de activación**. Una función de activación, en esencia, es una fórmula matemática para convertir los resultados que tenemos en la capa inmediatamente posterior a la salida en algo legible por el programa.

Una vez entrenado el modelo, **pueden darse tres situaciones**:

1

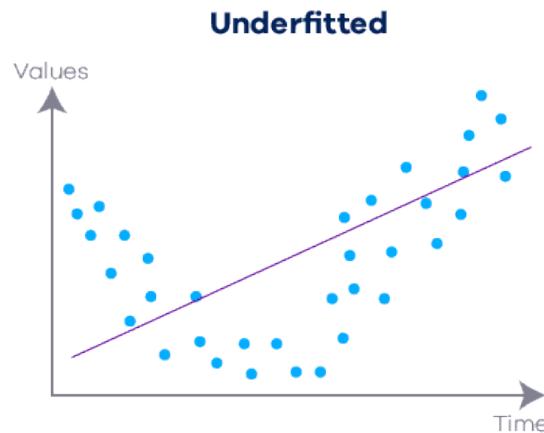
El modelo predice muy bien las muestras de entrenamiento, pero cuando le damos nuevas muestras, su predicción es mayormente errónea. Esto se llama **overfitting o sobreentrenamiento**. Nuestro modelo ha particularizado demasiado en el conjunto de entrenamiento y, por lo tanto, con nuevas muestras no es capaz de generalizar lo suficiente y otorga un resultado erróneo.



Overfitting. Fuente: Medium.

2

El modelo no ha generalizado lo suficiente tanto muestras de entrenamiento como en un conjunto nuevo. Esto se llama **underfitting**. Básicamente lo que hace es echar una moneda al aire.



Underfitting. Fuente: Medium.

3

El tercer caso es cuando el modelo es capaz de tener un buen porcentaje de elementos predichos en el entrenamiento y, cuando le enseñamos nuevos conjuntos, hace unas buenas predicciones (el porcentaje de acierto será siempre menor que en el entrenamiento). Ahí tenemos **un modelo robusto**. El modelo ha conseguido generalizar lo suficiente para hacer buenas predicciones con elementos nunca vistos

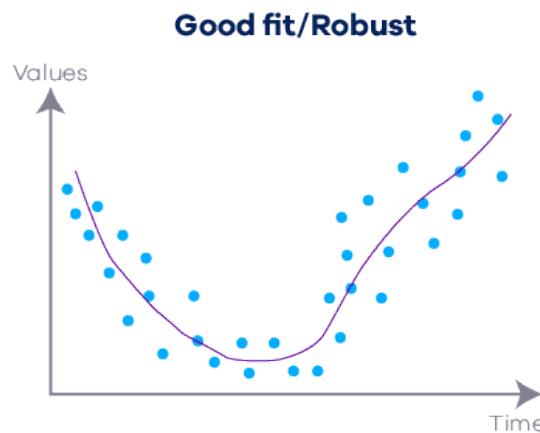
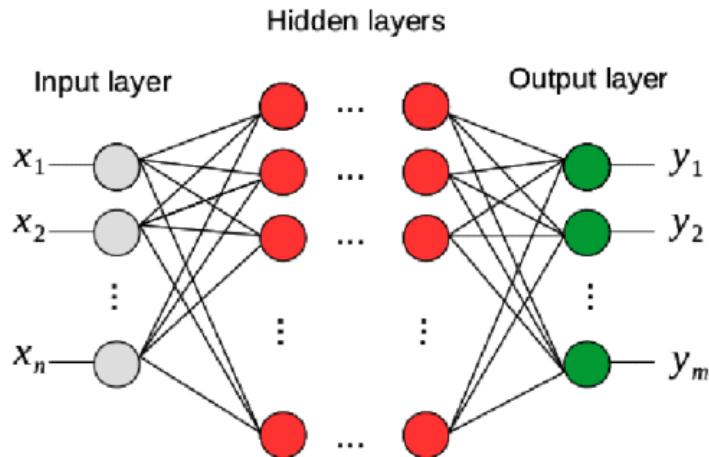


Ilustración 15: Good fit Fuente: Medium.

Esta ilustración corresponde a una **red neuronal multicapa**. Una red neuronal puede tener tantas capas ocultas (las que no son ni la entrada ni la salida) como el creador quiera.



Red neuronal multicapa. Fuente: ResearchGate.

Tipos de redes neuronales

Los tres tipos más famosos de redes neuronales son las redes neuronales artificiales (ANN o *artificial neural network*), las redes neuronales convolucionales (CNN o *convolutional neural network*) y las redes neuronales recurrentes (RNN o *recurrent neural network*).

1

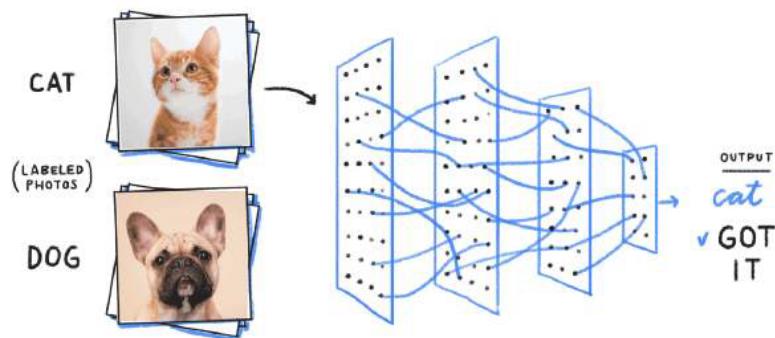
Artificial neural network

Las primeras son el ejemplo ya mostrado. Se trata de las más comunes. Son buenas para problemas como **clasificación de imágenes** o los **relacionados con datos numéricos**.

2

Convolutional neural network

Las redes neuronales convolucionales (CNN) **capturan las características espaciales de una imagen** (la disposición de los píxeles y la relación entre ellos). Nos ayudan a identificar un objeto, su ubicación y su relación con otros objetos en la misma imagen.

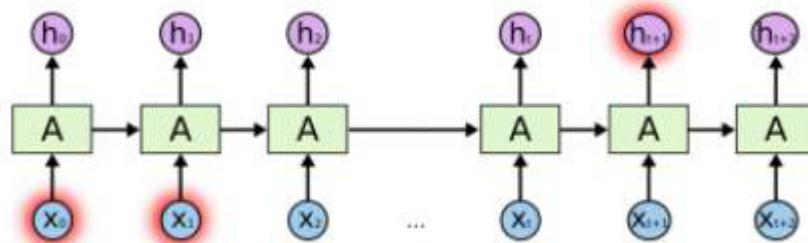


Ejemplo de red neuronal convolucional. Fuente: www.analyticsvidhya.com.

3

Recurrent neural network

Por último, tenemos las redes neuronales recurrentes (RNN). Estas son las redes neuronales que tienen ‘memoria’. Se suelen utilizar para **análisis de textos**. Las RNN capturan información secuencial presente en los datos de entrada, como por ejemplo, la dependencia entre palabras en un texto cuando estás haciendo predicciones.



Ejemplo de red neuronal recurrente. Fuente: www.analyticsvidhya.com.

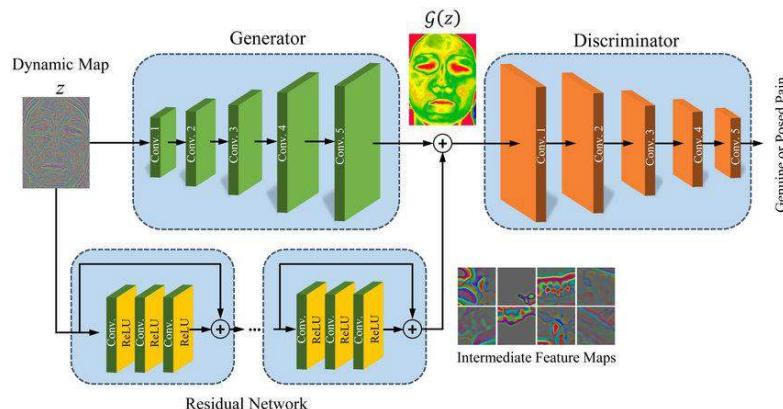
Como resumen, podemos utilizar la siguiente imagen para tener una idea más clara sobre qué red neuronal usar en qué situación.

	MLP	RNN	CNN
Data	Tabular data	Sequence data (Time Series,Text, Audio)	Image data
Recurrent connections	No	Yes	No
Parameter sharing	No	Yes	Yes
Spatial relationship	No	No	Yes
Vanishing & Exploding Gradient	Yes	Yes	Yes

Resumen redes neuronales. Fuente: www.analyticsvidhya.com.

Por último, aunque no es una red neuronal per se, las GAN (*generative adversarial network*) están siendo muy utilizadas para multitud de tareas. Generalmente se utilizan para aquellas en las que se requiere que la red neuronal **cree contenido**.

No son más que varias redes neuronales enfrentadas en las que una (el generador) juega a engañar a la otra (el discriminador). El generador se encarga de crear contenido y dárselo al discriminador, que este es el que decide si lo que ha creado el generador es algo que existe o ha sido creado por la red neuronal. Todo esto se hace iterativamente, dando feedback el discriminador al generador para que así este último mejore sus creaciones y logre engañar al discriminador, es decir, que crea que son ejemplos genuinos y no creados por la red neuronal.



Ejemplo de GAN. Fuente: ResearchGate.

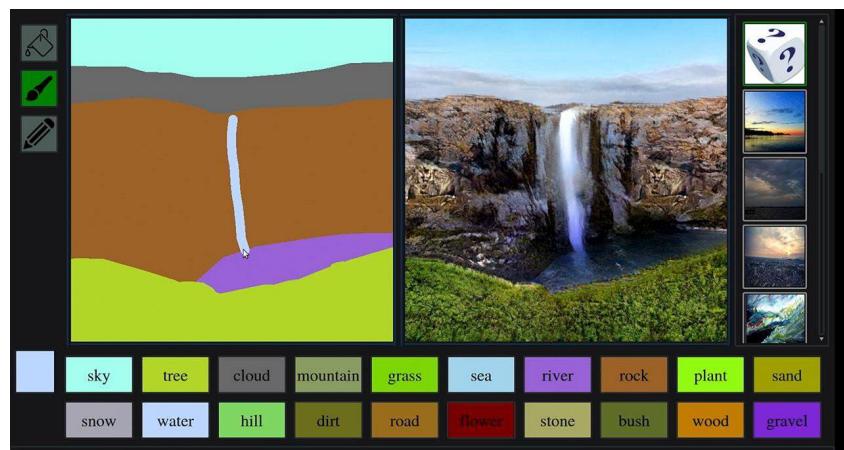
Nvidia tiene un gran recorrido en el uso de estas redes neuronales, y las aplica en infinidad de juegos (por ejemplo, [en el DLSS](#)). Aquí el ordenador funciona a una resolución menor, además, esta red neuronal se inventa los píxeles para poder mostrar la imagen en una resolución mucho mayor y así obtener un mejor rendimiento.

Fuente de la imagen: [pcmrace.com](#).



Otro ejemplo del uso de estas redes es la creación de **paisajes hiperrealistas**, dándole a la red como entrada un ejemplo similar al que podríamos hacer con Paint.

Fuente de la imagen: [Nvidia Blog](#).



Otro conocido ejemplo lo podemos encontrar en la siguiente dirección <https://thispersondoesnotexist.com/>. Las redes neuronales han conseguido crear imágenes de seres humanos que no existen. Todo creado por la GAN.

Fuente de la imagen: thispersondoesnotexist.com



Por ahora lo dejamos aquí, hay mucho contenido para procesar antes de descubrir en el próximo tema todas las aplicaciones realizadas con inteligencia artificial, algunas muy, muy curiosas...

¡Enhorabuena! Fastbook superado

edix

Creamos Digital Workers