

Investigación en Inteligencia Artificial

Soluciones empresariales de inteligencia artificial

Índice

Esquema	3
Ideas clave	4
6.1. Introducción y objetivos	4
6.2. La inteligencia artificial en la empresa	4
6.3. Herramientas comerciales para inteligencia artificial	13
6.4. Tecnologías comerciales de inteligencia artificial	17
6.5. El futuro de la inteligencia artificial	28
A fondo	30
Test	31

Esquema

Aplicación de la inteligencia artificial al ámbito empresarial	
Tipos de datos	Visualización
Google	Marketing, ventas, CRM
Facebook	Automoción
DeepMind	<i>Business intelligence</i>
OpenAI	ChatBots
Baidu	Plataformas
Microsoft Research	Ciberseguridad
Apple	Fintech
IBM	Salud y sanidad
AIBrain	Internet de las cosas
Amazon	Robótica
Netflix	Narrativa
	Visión artificial
	Biología y medio ambiente

6.1. Introducción y objetivos

En este tema se transmite una visión amplia de los principales proveedores y soluciones basadas en *machine learning* disponibles en el mercado. Paralelamente, el alumno recibirá información sobre el estado del arte de este tipo de soluciones en las diferentes industrias.

Por último, se pretende que el alumno sea capaz de detectar los ámbitos en los que los principales actores del área están centrando sus esfuerzos.

6.2. La inteligencia artificial en la empresa

Las **soluciones basadas en inteligencia artificial están a nuestro alrededor a día de hoy más de lo que creemos**. Según un estudio realizado por Microsoft en quince países europeos, se estima que el porcentaje de empresas a nivel europeo que usan o están en desarrollo de proyectos de inteligencia artificial actualmente es del 28 %. Y se estima que un 51 % de ellas están planificando su implantación en el futuro. Por lo tanto, la inteligencia artificial está empezando a penetrar poco a poco en todo el tejido empresarial, tanto en las grandes empresas como en las pequeñas y medianas. Según el mismo estudio el 89 % creen que la IA (inteligencia artificial) generará beneficios comerciales al optimizar las operaciones de sus compañías y el 74 % espera que la IA sea clave para atraer a más clientes.

Informe del estado de la inteligencia artificial en las empresas europeas. Puedes acceder a este recurso pinchando en el siguiente enlace.

<https://pulse.microsoft.com/es-es/business-leadership-es-es/na/fa1-articial-intelligence-report-at-a-glance/>

Aunque son datos extraídos de este informe en concreto pueden no ser al 100 % representativos de la realidad. Sí que **denotan una tendencia clara a que el peso de la IA en el futuro será muy grande en el ámbito empresarial**. Pero hoy, ¿cuál sería la penetración de la IA en las empresas actuales? Podríamos dar muchos ejemplos desde medición de cosechas usando internet de las cosas y *machine learning*, hasta asistentes virtuales en diferentes compañías, o el uso de la IA para mejorar la obtención de imágenes en las cámaras de fotos o en la calidad de imagen de nuestros televisores. Pero nos vamos a centrar principalmente en los grandes actores que muchas veces están detrás de estas implementaciones concretas de IA en las empresas. Siempre teniendo en cuenta que es un entorno muy dinámico y cambiante y las tendencias pueden cambiar en muy poco tiempo. La lista de actores y herramientas que enunciamos, por tanto, en este tema no pretende ser una lista exhaustiva pero sí incluir empresas y soluciones más importantes y que deberían ser conocidas por todos los profesionales del sector para ser tenidas en cuenta.

Líderes en investigación y productos basados en inteligencia artificial

Comenzaremos desglosando algunas de las compañías que se están posicionando como líderes en temas de investigación en inteligencia artificial. Destacaremos a Google, Facebook, Deepmind, OpenAI, Baidu, Microsoft Research. Muchos de los productos que estas empresas ofrecen están basados en la inteligencia artificial, aprendizaje automático o subdominio del área.

Alphabet

Alphabet es la **matriz de conglomerados de Google que engloba a una gran cantidad de empresas bajo el mismo paraguas corporativo**. Muchas de ellas relacionadas directa o indirectamente con la IA. Google es una de las empresas más conocidas del mundo y también una de las más prolíficas en lo que se refiere a la generación de soluciones basadas en inteligencia artificial. Su funcionamiento interno, orientación hacia el mundo digital y la gran cantidad de datos de los que dispone facilita el desarrollo de este tipo de productos.

Una de las herramientas más destacadas de Google en el campo de la inteligencia artificial es **Google Brain**. Este proyecto combina técnicas de aprendizaje automático (haciendo un uso más intensivo de las técnicas de *deep learning*) con el desarrollo de sistemas computacionales escalables y eficientes. Su objetivo es **contribuir a que las máquinas puedan aprender por sí mismas a realizar tareas cada vez más complejas**. Y todo ello haciendo uso eficaz y eficiente de los recursos computacionales puestos a su disposición. Sus investigadores han publicado varios resultados relacionados con la detección de imágenes, traducción automática, criptografía, etc. Son habituales las colaboraciones de esta ramificación de Google con la comunidad académica y con otras empresas del área.

Alguno de los proyectos más interesantes que ha generado Google Brain es Deep Dream, un sistema que genera arte usando *machine learning*.

Deep Dream o cómo generar arte con *machine learning*. Puedes acceder a este recurso pinchando en el siguiente enlace.

<https://deeplearninggenerator.com/>

Otro de los productos destacados que ha generado Google Brain ha sido TensorFlow. **TensorFlow** es un *framework* que permite, esencialmente, el diseño y programación de redes de neuronas artificiales de distintas tipologías y

complejidades, con soporte para compilar álgebra lineal, *open source* y que cada vez tiene mayor número de usos y algoritmos implementados.

TensorFlow una herramienta *open source* que permite ejecutar modelos de *deep learning* usando la GPU. Puedes acceder a este recurso pinchando en el siguiente enlace.

<https://www.tensorflow.org>

DeepMind es una de las empresas más activas en los últimos años en el campo de la IA y con los resultados más impresionantes. Es otra de las patas de Alphabet, tras ser adquirida por Google en el año 2014. La empresa está ubicada en Londres y son los creadores de **AlphaGo** y **AlphaStar**, ambos proyectos los primeros capaces de ganar a los mejores jugadores de *Go* y del videojuego de estrategia en tiempo real *Starcraft 2*.

Sus investigadores son realmente prolíficos y respetados. Realizan contribuciones a diversos campos dentro de la inteligencia artificial, campos como *deep reinforcement learning*, redes neuronales bayesianas, robótica, *transfer learning*, etc. Se atreven con los temas más diversos llegando incluso a abarcar la relación entre ecología e inteligencia.

DeepMind, creadores de AlphaGo y AlphaStar. Puedes acceder a este recurso pinchando en el siguiente enlace.

<https://deepmind.com/>

Microsoft Research

Es la **división de Microsoft dedicada a la investigación en general y en IA en particular**. Actualmente es una de las empresas más activas en esta tecnología junto con IBM y Alphabet, y se está potenciando su peso en ella a través de diferentes servicios agrupados en lo que denominan Azure AI. Su objetivo es **tener**

diferentes plantillas para crear proyectos de forma rápida. Microsoft ofrece módulos para diferentes soluciones con muchas de las funcionalidades necesarias ya implementadas. A esto lo llama **Azure Cognitive Services**, que es una completa familia de servicios de inteligencia artificial y API (application programming interface) cognitivas que le ayudan a crear aplicaciones inteligentes. También dispone de **Azure Machine Learning** que es un servicio de aprendizaje automático para empresas, que permite aprender modelos usando Azure. También existen otros servicios como chatbots o plataformas de análisis de datos usando Apache Spark.

Microsoft Azure Machine Learning y Cognitive Services son los dos pilares principales de la IA de Microsoft destinada para empresas. Puedes acceder a este recurso pinchando en el siguiente enlace.

<https://azure.microsoft.com/es-es/services/machine-learning/>
<https://azure.microsoft.com/es-es/services/cognitive-services/>

Facebook

Facebook es otro de los grandes actores de esta industria. A lo largo de su historia ha producido conocimientos diversos relacionados con el procesamiento de texto, gestión de recursos computacionales, evaluación de máquinas inteligentes, etc. La división de investigación de la compañía se denomina **Facebook AI Research (FAIR)**.

Una de las aportaciones más destacadas de Facebook a la comunidad es el *framework Pytorch*, al que se puede considerar competidor de TensorFlow como recurso para generar modelos basados en *deep learning*.

Facebook y su librería *open source* Pytorch. Puedes acceder a este recurso pinchando en el siguiente enlace.

<http://pytorch.org>

OpenAI

OpenAI es una **pequeña compañía sin ánimo de lucro liderada por conocidos actores del mundo de la inteligencia artificial y de la que es fundador Elon Musk entre otros.** Tiene el objetivo de lograr una inteligencia artificial ligada a las necesidades reales de la sociedad. Suele liberar los resultados de sus investigaciones y patentes. Uno de sus productos más conocidos es **Universe**, una plataforma de *software* para medir y entrenar soluciones basadas en inteligencia artificial. Open AI ha tenido importantes éxitos en robótica, logrando vencer a jugadores humanos del videojuego *Dota 2* así como la creación de una IA que es capaz de jugar a juegos retro (Retro Contest)

Open AI y su proyecto Retro Contest donde una IA aprende a jugar a juegos retro usando aprendizaje por refuerzo profundo. Puedes acceder a este recurso pinchando en el siguiente enlace.

<https://openai.com/blog/retro-contest/>

Baidu

Baidu, el motor de búsqueda por excelencia en el idioma chino, es uno de los grandes gigantes del mercado. Esta compañía nacida en Pekín **lidera el mercado chino abarcando una gran diversidad de productos** que van desde asistentas para el hogar a coches autónomos, pasando por distintas soluciones de ayudas a discapacitados.

Apple

Apple es una de las marcas más conocidas del panorama empresarial. Como compañía innovadora no podría permanecer ajena a la innovación basada en inteligencia artificial. En este ámbito, uno de sus productos más conocidos es **Siri**. Esta aplicación basada en procesamiento del lenguaje natural es capaz de

interpretar las instrucciones (orales) del usuario para realizar diversas acciones como programar una alarma o realizar una llamada. Apple también trabaja intensamente en técnicas de reconocimiento facial u optimización del *hardware* de los diferentes dispositivos móviles para ejecución óptima de algoritmos basados en inteligencia artificial.

IBM

Sin duda, **Watson** es uno de los productos estrella de IBM (relacionado además con la inteligencia artificial). Watson es una solución orientada a dar respuesta a preguntas formuladas mediante lenguaje natural. Es capaz de acumular, ordenar y clasificar información proveniente de diversas fuentes de información. En su diseño y desarrollo se aplican técnicas y metodologías procedentes de varios subdominios, procesamiento del lenguaje natural, representación del conocimiento, razonamiento automático, aprendizaje automático, etc. Está diseñado para usar IA a gran escala (similar a AWS, Azure AI, Google Cloud AI) y tiene diferentes divisiones especializadas en características concretas como Watson for business (la más importante para lo que ocupa este tema) Watson assistant o for medical donde ha tenido éxitos importantes.

IBM Watson, la solución de IA y sistemas cognitivos de IBM. Puedes acceder a este recurso pinchando en el siguiente enlace.

<https://www.ibm.com/watson/>

AIBrain

AIBrain es una empresa californiana **centrada en el desarrollo de soluciones basadas en inteligencia artificial para la resolución de problemáticas de diverso tipo**. Destacan sus soluciones para telefonía móvil de última generación y robótica. AIBrain está especializada en el desarrollo de soluciones cognitivas y con capacidad autónoma de aprendizaje.

Amazon

Amazon es uno de los grandes líderes tecnológicos mundiales. Sus algoritmos de recomendación son foco de atención continua por parte de la comunidad especializada y la competencia. Actualmente, **está centrado en ofrecer servicios relacionados con la inteligencia artificial a través de su oferta de servicios en la nube**. También prevé extender servicios de consultoría digital en este ámbito. Algunos de sus principales productos son:

- ▶ Amazon SageMaker: entrenamiento de modelos de *machine learning* (integración con Jupyter Notebook).
- ▶ Amazon Recognition: reconocimiento de imágenes.
- ▶ Amazon Polly: reconocimiento de voz.
- ▶ Amazon Lex: asistentes virtuales (Alexa).

También tiene una tienda que permite comprar modelos ya entrenados.

Amazon web services for machine learning. Puedes acceder a este recurso pinchando en el siguiente enlace.

<https://aws.amazon.com/es/machine-learning/>

Netflix

A la hora de hablar de algoritmos de recomendación, si existe una empresa que venga a la mente de los especialistas en este ámbito es Netflix. Netflix es una **empresa estadounidense de entretenimiento que oferta series, películas y documentales a través de una plataforma de streaming**. Los algoritmos de recomendación implementados por Netflix ayudan al usuario a explorar la amplia diversidad de oferta existente sugiriendo productos acordes a sus preferencias y gustos.



Figura 1. Recomendaciones en Netflix.

Fuente: <https://medium.com/netflix-techblog/artwork-personalization-c589f074ad76>

Otros actores importantes

Como se comentaba inicialmente, aunque se ha intentado dibujar un espectro amplio de actores en el ámbito de la inteligencia artificial identificando a los principales agentes, el listado no es exhaustivo. Un mapeo absolutamente detallado de este mercado no es objetivo de esta unidad, simplemente se pretende que el alumno esté informado sobre algunas de las que están consideradas como compañías de referencia que están desarrollando productos basados en inteligencia artificial.

El listado anterior puede enriquecerse mencionando a otras compañías que, aunque quizás no hayan alcanzado un grado de innovación tan destacado como las anteriores, se posicionan como **empresas punteras** que están realizando valiosas contribuciones al estado de la disciplina.

En el campo financiero, **la banca de inversión es gran consumidora y productora de soluciones basadas en inteligencia artificial**. Siendo más concretos y dentro del sector bancario español, destaca el **caso de BBVA**, que ha sido premiado por varias iniciativas realizadas en el campo del aprendizaje automático y ha sido reconocido

por el Massachusetts Institute of Technology (MIT) como una de las entidades más innovadoras de la industria.

La compañía norteamericana **Salesforce** está ampliando su equipo tanto mediante la contratación externa como mediante la adquisición de pequeñas compañías innovadoras en el área. El objetivo de sus proyectos va desde mejorar la eficiencia de los procesos hasta mejorar las herramientas de ayuda automatizada para sus clientes.

Spotify, la empresa sueca que centra su actividad en la distribución de contenido musical vía *streaming*, es otra de las compañías innovadoras especialmente en el campo del aprendizaje automático y sistemas de recomendación. Debido al gran volumen de datos y usuarios también se posiciona como una compañía innovadora en lo que se refiere al tratamiento de grandes volúmenes de datos y optimización de los sistemas de información.

Por último, no podríamos dejar de mencionar a **Twitter**. La compañía norteamericana ha invertido ingentes cantidades económicas en la adquisición de varias compañías especializadas en inteligencia artificial. Su ámbito de actuación es amplio: desde proponer al usuario los tuits más relevantes en función de sus gustos, hasta mejorar sus algoritmos de recomendación de anuncios publicitarios.

6.3. Herramientas comerciales para inteligencia artificial

En este apartado pretendemos destacar las herramientas *software* más importantes que podemos utilizar para IA. De algunas ya hemos hecho mención en el apartado anterior, pero aquí nos centraremos en sus principales bondades y añadimos algunas otras que no hemos citado anteriormente.

TensorFlow

Es una **biblioteca de código abierto para aprendizaje automático** y que dispone de multitud de modelos de redes profundas. Está desarrollado por Google para satisfacer sus necesidades de sistemas capaces de construir y entrenar redes neuronales para detectar y descifrar patrones y correlaciones, análogos al aprendizaje y razonamiento usados por los humanos. Esta biblioteca se centra en *deep learning*, pero ha ido incluyendo otros algoritmos con el tiempo. Está integrada con los servidores de Google, con lo cual podemos de forma fácil lanzar los algoritmos en la nube de Google. Así que una de sus principales bondades es que permite un fácil despliegue de las aplicaciones. También tiene acceso a las GPU (*graphics processing unit*) instaladas a nivel local, con lo que podemos aprovecharnos de las capacidades de computación de estas tarjetas para acelerar los cálculos. Tiene una interfaz de uso en Python, aunque muchos de los algoritmos internamente están desarrollados en C++.

Scikit-learn

Scikit-learn **lidera las librerías de *machine learning* en Python** al estar construido a su vez sobre las mejores librerías para matemáticas y ciencias (NumPy, SciPy y Matplotlib). La librería está disponible bajo licencia BSD, por lo tanto, es completamente abierto y reutilizable e implementa un gran conjunto de algoritmos de *machine learning*. Casi cualquier algoritmo conocido está disponible en la librería y no se centra solo en *deep learning* como TensorFlow. Es además muy fácil de usar e incorpora algunos *datasets* para poder hacer pruebas con los diferentes algoritmos.

Universe

Universe es una API *open source* de OpenAI. Su principal reclamo es que es totalmente accesible al público. Sirve para medir y entrenar algoritmos de *machine learning* de propósito general. Y proporciona una sencilla interfaz de comunicación con la plataforma Gym de OpenAI (<https://github.com/openai/gym>) Dicha plataforma es un conjunto de herramientas para desarrollar y comparar algoritmos de aprendizaje de refuerzo.

Universe permite a cualquiera entrenar y evaluar agentes de IA en un rango extremadamente amplio de entornos complejos en tiempo real, haciendo posible que cualquier programa existente se convierta en un entorno válido para OpenAI Gym, sin necesidad de un acceso a dicho entorno. Lo consigue empaquetando el programa en un contenedor Docker, y presentando la IA con la misma interfaz que usa un humano, es decir, enviando eventos de teclado y ratón, y recibiendo píxeles de pantalla: <https://github.com/openai/universe>

H2O

Plataforma de aprendizaje automático de código abierto para el análisis predictivo de *big data*, scoring de datos y modelado de datos. Tiene diferentes integraciones con otras librerías como Spark y es muy usada para análisis de datos. Aseguran que buscan una IA responsable y fomentan las buenas prácticas favoreciendo cosas como la explicabilidad, la comprensibilidad o la IA centrada en las personas.

Apache Spark

Apache Spark es un framework de programación para procesamiento de datos distribuidos, tiene muchos algoritmos y tipos de datos útiles (basado en Apache Hadoop). Diseñado para ser rápido y escalable. Aunque Java es el primer lenguaje

para trabajar con ella, los usuarios de Python pueden conectar con NumPy. Está destinada a *big data* principalmente.

Accord.NET

Librería para *machine learning* y procesamiento de señales hecha para .NET, es una extensión del proyecto AForget.NET. Accord incluye un conjunto de librerías para procesamiento de audio y flujo de imágenes (vídeos). Este algoritmo puede ser utilizado para detección de rostros o para seguimiento del movimiento de objetos.

Keras

Biblioteca de red neuronal de código abierto, escrita en Python, que admite redes recurrentes y redes convolucionales. Está construido sobre TensorFlow y su objetivo es hacerlo más usable. Presume de ser usado en algunas de las principales organizaciones científicas de todo el mundo como el CERN, NASA, NIH. Lo cual la hace una librería con mucho apoyo a nivel académico.

NGC

La plataforma acelerada por GPU **permite a los científicos de datos administrar datos**, así como también diseñar y capacitar modelos de redes neuronales. Está desarrollado por Nvidia para su uso en sus propias gpus para abstraer a los desarrolladores de la complejidad de acceso a la misma.

Caffe

Marco de aprendizaje profundo de código abierto que brinda asistencia para GPU y CPU y que está desarrollado en la universidad de Berkeley en la división de AI Research. Perfecto para investigaciones académicas y aplicaciones industriales.

Presume de poder configurarse sin programar mediante ficheros de configuración, así como ser muy extensible y rápido.

6.4. Tecnologías comerciales de inteligencia artificial

En este apartado pretendemos destacar **otras soluciones comerciales basadas en inteligencia artificial**, y que son menos conocidas que los productos ofrecidos por las grandes compañías o, al menos, menos conocidas comercialmente que las comentadas en el apartado anterior. Enumeraremos ante todo productos ofrecidos por *startups* o pequeñas empresas. Dividiremos las distintas propuestas teniendo en cuenta la industria en la que operan. De nuevo, no se pretende crear un listado exhaustivo sino, simplemente, destacar casos de uso que se han considerado relevantes. Según el caso, se detallará de forma individual cada compañía o por el contrario se proporcionará una descripción genérica del sector enumerando destacadas compañías de este.

Marketing, ventas y CRM

En este apartado, se hace referencia a compañías que principalmente operan en el sector de la publicidad, gestión de ventas, *marketing* o *customer relationship management* (CRM).

- ▶ DrawBridge: compañía californiana fundada en el año 2011. Permite conectar a personas operando a través de dispositivos diferentes para ofrecer anuncios personalizados.
- ▶ Appier: compañía con sede central en Taipéi (Taiwán). Su producto estrella es CrossXAI y, al igual que el caso anterior, uno de sus objetivos es enlazar los dispositivos asociados al mismo usuario.

- ▶ Persado: compañía con localizaciones en Estados Unidos y Europa, que centra su trabajo en la personalización del lenguaje empleado en las campañas comerciales con el fin de obtener la mejor reacción por parte del potencial cliente.
- ▶ InsideSales.com: compañía estadounidense especializada en la creación de distintas soluciones tecnológicas con la misión de apoyar la actividad de los equipos de venta comercial.
- ▶ BloomReach: ofrece el producto BloomReach DXP, consistente en una plataforma abierta e inteligente focalizada en el desarrollo, personalización y pruebas de experiencias digitales, ayudando a sus clientes a identificar en tiempo real la estrategia digital más adecuada a cada momento. Está presente en EE. UU., Europa y Asia.

Tecnología para el sector automovilístico

En lo que respecta a compañías centradas en la producción de soluciones tecnológicas para el sector de la automoción, podemos citar las siguientes:

- ▶ Nutoromy: empresa nacida en el seno del MIT, centrada en el diseño de coches autónomos.
- ▶ Nauto: compañía radicada en Silicon Valley centrada en mejorar la seguridad en el transporte rodado. Sus soluciones suelen basarse en el empleo de cámaras y análisis de imágenes para evitar accidentes de tráfico.

Finalizamos este apartado destacando el trabajo de la compañía automovilística **Tesla**. Esta compañía norteamericana está en la lista de las empresas llamadas a liderar la producción de los futuros coches autónomos.



Figura 2. Coche autónomo Tesla.

Fuente: <https://www.tesla.com/autopilot?redirect=no>

Tecnología de *business intelligence* y analítica genérica

Los sistemas de *business intelligence* se apoyan en la tecnología para analizar la información de que dispone la compañía con vistas a apoyar la toma de decisiones. Se listan a continuación algunas de las compañías destacadas:

- ▶ Logz.io: compañía israelí que se dedica a analizar el inmenso volumen de información generado por los sistemas informacionales (*logs*). A partir de dichos análisis se ofrecen soluciones de cara a mejorar la eficiencia de los procesos y evitar problemas en los sistemas.
- ▶ CrowdFlower: conocida compañía ubicada en San Francisco (EE. UU.). Ofrece soluciones que permiten enriquecer con información adicional los datos a nuestra disposición para dotarlos de mayores posibilidades.
- ▶ RapidMiner: compañía con sedes en Europa y EE. UU. RapidMiner ofrece una plataforma del mismo nombre que permite ejecutar el flujo completo de análisis de datos y aplicación de técnicas de aprendizaje automático, *deep learning*,

análisis de texto o análisis predictivo con fines varios. Esta solución se ofrece a varias industrias y sectores.

- ▶ Tamr: compañía con delegaciones en EE. UU., Europa y Asia, centrada en lograr la integración de datos procedentes de varias fuentes para dotar así al análisis de mayor potencia.
- ▶ Versive: pequeña compañía afincada en Seattle desde el año 2012. Versive tiene en su foco la automatización de procesos de negocio, captura de conocimiento experto para la toma de decisiones y el desarrollo de modelos basados en aprendizaje automático.
- ▶ Datarobot: compañía radicada en Boston que ofrece a sus clientes una plataforma que permite cargar los datos del cliente y aplicar algoritmos de aprendizaje automático obteniendo modelos analíticos que pueden aplicarse en el día a día.
- ▶ Paxata: empresa californiana nacida en el año 2012. Paxata ofrece un servicio que permite agregar, limpiar y transformar datos procedentes de diversas fuentes con el fin de facilitar el tratamiento posterior de estos para la aplicación de técnicas de aprendizaje automático.
- ▶ Trifacta: compañía establecida en San Francisco con un objetivo parecido a Paxata. Esta empresa ofrece un producto que permite explorar, transformar y enriquecer los datos para su posterior análisis.
- ▶ Dataminr: se trata de una empresa neoyorquina centrada en la monitorización de redes sociales y otras fuentes de datos con el fin de proporcionar las alertas adecuadas en tiempo real a negocios de diversos sectores.

Soluciones conversacionales, *bots*, lenguaje natural

En el mundo de los asistentes personales que intentan interactuar con el usuario empleando técnicas de procesamiento del lenguaje natural encontramos diversas opciones.

Mindmeld, Nice, Nuance, OpenText, Mobvoi y X.ai son conocidos casos de este ámbito. Mobvoi está centrada en el idioma chino. Nuance y OpenText abarcan

también el procesamiento de información documental y en formatos variados. Por el contrario, X.ai se adentra en el terreno de los gestores personales.

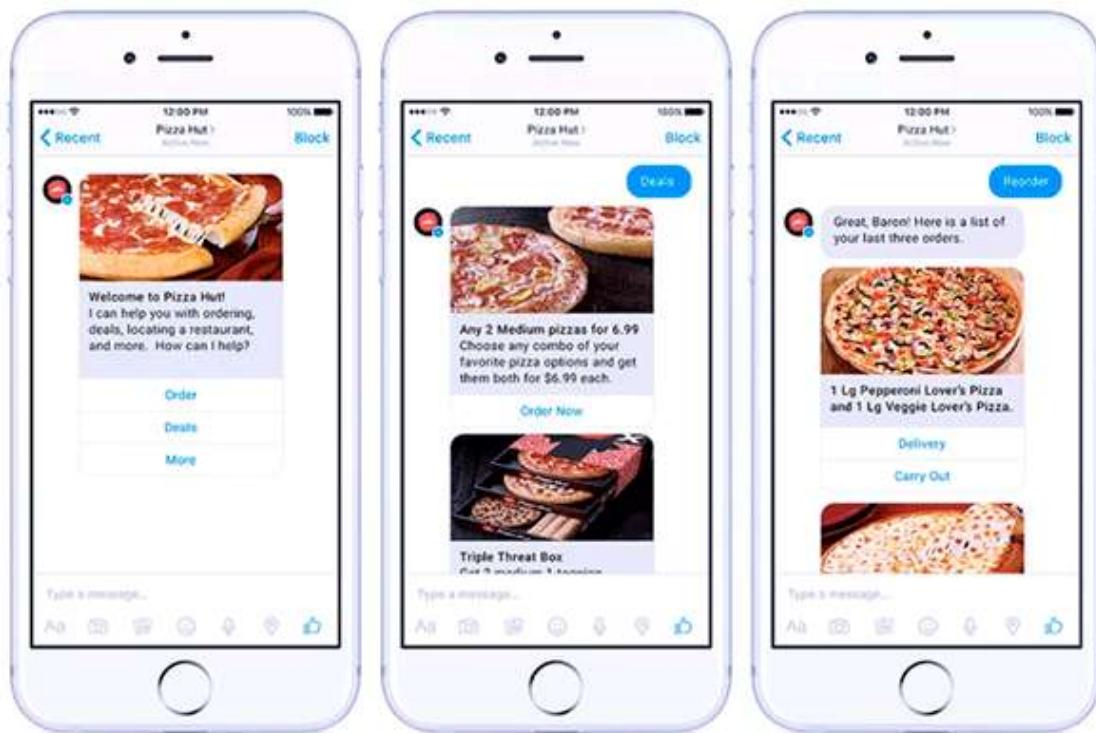


Figura 3. Ejemplo chatbot para la adquisición de una pizza.

Fuente: <https://www.theverge.com/2016/7/13/12170682/pizza-hut-chatbot-catch-up>

Plataformas y soluciones de *hardware*

Diversas empresas, entre ellas algunas de las ya mencionadas, **se dedican a proporcionar algoritmos, herramientas, datos, utilidades y acceso a sistemas e infraestructuras de *hardware* de distinto tipo** con el fin de que otras compañías (sus clientes) puedan emplear estos recursos para analizar sus datos, añadir fuentes de datos adicionales y desarrollar modelos de distintos tipos que aporten valor a las empresas usuarias. Entre las compañías que ofrecen este tipo de servicios encontramos a Amazon, Fractal Analytics, Google, H2O.ai, Microsoft, SAS, Skytree, etc.

De forma más concreta, debemos citar también a empresas que participan en la fabricación y diseño de componentes de *hardware* centrados en ejecutar

aplicaciones que requieren alta capacidad de cómputo, muchos de ellos pensados para entrenar algoritmos de aprendizaje automático. Un ejemplo clásico de este tipo de piezas son las tarjetas gráficas (GPU por sus siglas en inglés: Graphics Processing Unit) de altas prestaciones. Ejemplos de compañías que abarcan este tipo de productos son Alluviate, Cray, Google, IBM, Intel y, como no, Nvidia.

Dentro de las soluciones basadas en reglas y razonamiento lógico para la gestión de procesos de negocio y apoyo a la toma de decisiones, destacan las soluciones propuestas por compañías como Advanced Systems Concepts, Informatica, Maana, Pegasystems o UiPath.

Un sector que genera una cifra de negocio importante es el **sector de la biometría**, centrado en la identificación de las personas basándose en atributos diferenciales. Empresas destacadas del ámbito son: 3VR, Affectiva, Agnitio, FaceFirst, Sensory, Synqera o Tahzoo.

Core AI-Aplicaciones genéricas

Existen compañías difíciles de etiquetar dentro de un sector concreto debido a la diversidad de servicios que ofrecen. Se proporciona a continuación un listado de las compañías susceptibles de ser agrupadas en esta categoría.

Numenta

Empresa afincada en California que cuenta entre sus filas con expertos muy reconocidos en el ámbito de la inteligencia artificial. Produce tanto soluciones propietarias como soluciones abiertas a la comunidad. Sus productos abarcan diversos casos de uso, desde análisis de valores bursátiles hasta procesamiento del lenguaje natural.

Cognitivescale

Reconocida compañía estadounidense que ofrece soluciones en todos los ámbitos bajo el concepto de inteligencia aumentada. Su concepto principal es crear productos no orientados a reemplazar el personal humano sino a dotarlo de herramientas que le permitan hacer mejor su trabajo.

Affectiva

Esta compañía ramifica de trabajos previos realizados en el MIT (EE. UU.). Su foco principal es la medición y reconocimiento de las emociones humanas para su explotación con fines comerciales.

Sentient Technologies

Compañía norteamericana que ha generado gran interés entre fondos de inversión varios. Actúa principalmente en dos ámbitos que no parecen tener relación directa, el comercio electrónico y operaciones de inversión en los mercados financieros. Sentient centra su actividad en la creación de sistemas de recomendación que tienen en cuenta el perfil del usuario para proponer la mejor recomendación o producto. Dicho producto varía dependiendo del sector, puede ser un producto financiero o no.

Digital Reasoning

Esta compañía estadounidense centra su actividad principal en la aplicación de la inteligencia artificial para apoyar la labor de servicios de inteligencia varios e instituciones financieras.

Voyager Labs

Compañía con sedes en EE. UU. e Israel. Se trata de una compañía pionera en soluciones basadas en inteligencia cognitiva. Aunque opera en diversos sectores, es quizás en el sector comercial (incluyendo comercio electrónico) donde sus soluciones están más extendidas.

Ciberseguridad

El sector de la seguridad informática o ciberseguridad está llamado a experimentar una gran transformación en los próximos años. El contexto político y tecnológico obliga a ello. Las **estructuras informáticas son, desde hace ya muchos años, parte esencial de la infraestructura de un país**. La inteligencia artificial es un pilar esencial para proteger de forma eficiente dichos recursos. Compañías conocidas que operan en este ámbito son Sift Science, Darktrace o Cylance entre otras.

Fintech

Fintech (Financial Technology) **hace referencia a un paradigma de negocio centrado en la aplicación de la tecnología para ofrecer y gestionar productos financieros de forma innovadora**, ágil y rompiendo las barreras y los procedimientos estandarizados típicos de las grandes corporaciones bancarias. Son múltiples y diversas las opciones en este ámbito. Algunas que se pueden citar a modo de ejemplo son: Alphasense, Kensho, Due.com, SoFi, LendUp, LendFriend, WePay, Lending Club, etc.

Salud y sanidad

Las posibilidades que se abren en el sector sanitario en relación con la explotación de grandes volúmenes de datos y aplicación de técnicas de inteligencia artificial son infinitas. Los focos principales de actuación son la personalización de los

tratamientos clínicos, soporte para la toma de decisiones, optimización de los procesos hospitalarios, análisis predictivo de la evolución de la enfermedad, diagnóstico por imagen, soporte clínico virtual, etc. Algunas de las compañías que están liderando este sector son: Zebra Medical Vision, Babylon Health, Benevolent.ai o Icarbonx.

Internet de las cosas

El internet de las cosas (Internet of Things o IoT) hace referencia al **ecosistema que se genera con la interconexión de dispositivos electrónicos varios a través de las redes de comunicaciones**. Dicha interconexión permite integrar fuentes de datos diversas, contextualizar la información y tomar decisiones en tiempo real.

Este paradigma se puede aplicar a una gran diversidad de ámbitos, desde la gestión de las ciudades impulsando el concepto de «ciudad inteligente», hasta la agricultura, pasando por sectores tan diversos como la gestión integral de fábricas, gestión y optimización de los dispositivos del hogar, eficiencia energética, logística y un largo etcétera. Compañías con aportaciones interesantes en este ámbito son Verdigris Technologies y Sight Machine.

Robótica

El sector industrial y empresarial asociado al mundo de la robótica está ofreciendo importantes avances en los últimos años. A modo de ejemplo, se ha hecho referencia con anterioridad a Sophia, producto de la compañía Hanson Robotics.

La **industria contempla el uso de robots en multitud de contextos y ámbitos**, asistencia en el hogar, asistencia a personas enfermas o con algún tipo de discapacidad, producción industrial, exploración espacial, actuación ante emergencias (como en un accidente nuclear), intervención policial o militar,

entretenimiento, etc. Algunas empresas destacadas en este ámbito son: Rokid, Ubtechrobotics, Anki, Vicarious...

Narrativa y análisis semántico

Otro de los campos en auge dentro de la inteligencia artificial, **intenta transformar los datos en una narración con sentido para comunicar de forma automática ideas, noticias o hechos relevantes.** Este tipo de soluciones se aplican en diversos sectores, algunos ejemplos son el sector editorial, periódicos, noticias..., pero también puede aplicarse al ámbito empresarial transmitiendo historias fáciles de entender y compartir en lugar de una avalancha de datos de complicada interpretación.

Narrative Science es una de las compañías más activas en este campo.

Cambridge Semantic es otra de las compañías que podemos citar en este sector. Esta compañía norteamericana está focalizada en añadir a la capa de datos una capa adicional de conocimiento que permite establecer relaciones entre datos y dota de mayor valor a estos.



Figura 4. Página web de Narrative Science.

Fuente: <https://narrativescience.com/>

Visión artificial

La visión artificial **hace referencia a la disciplina científica que desarrolla metodologías y algoritmos que permiten adquirir, procesar, analizar e identificar imágenes de distinto tipo con fines varios.** Este tipo de conceptos pueden aplicarse a gran diversidad de campos, tales como seguridad, diagnóstico por imagen, coches autónomos, robótica, etc. A las empresas que hayan podido ser citadas en algún apartado anterior podemos añadir otras como Chronocam, Orbital insight, Clarifai o Captricity.

Biología, agricultura y medioambiente

Finalizamos este apartado haciendo referencia a casos de usos particulares que quizá resulten menos conocidos para la comunidad no especializada. **Las técnicas de inteligencia artificial, y más concretamente aprendizaje automático, pueden ser aplicadas al ámbito de la biología y análisis de moléculas,** virus o bacterias para conseguir procesos más eficientes en la fabricación de medicamentos, fertilizantes o productos empleados en nuestra vida cotidiana. **Zymergen** es una compañía estadounidense volcada en el desarrollo de este tipo de productos.

Haciendo hincapié en el sector de la agricultura, citaremos a **Blue River Technology**. Esta compañía emplea robótica y visión artificial para controlar y optimizar la gestión de explotaciones agrícolas. Al tratarse de un campo de actuación particular se ha creído oportuno incluir a Blue River en este apartado en lugar de otros previos.

6.5. El futuro de la inteligencia artificial

Llegados a este punto, tenemos una visión amplia de sectores en los que opera la inteligencia artificial y tipos de soluciones que se están ofreciendo. Resta ahora centrarnos en las posibilidades de futuro.

*Antes de nada, y a modo de «renuncia de responsabilidad» (*disclaimer*), debemos insistir en que no es posible determinar con certeza (y en opinión del autor ni siquiera con una precisión elevada) el futuro de la inteligencia artificial.*

Cualquier descubrimiento puntual puede abrir un espectro totalmente inesperado o generar una nueva línea de productos. Fuera de este apartado queda el contexto regulatorio y legal que, especialmente en Europa, puede influir en el desarrollo de nuevos productos basados en inteligencia artificial en general y aprendizaje automático en particular.

Respecto a los sectores donde la industria centrará sus esfuerzos de cara al futuro, sin duda, la **generación de vehículos autónomos** de todo tipo será uno de los grandes caballos de batalla para las próximas décadas.

Otro interesante campo de acción es la traducción automática y en tiempo real. Google ha lanzado los auriculares inalámbricos Pixel Buds, producto que pretende convertirse en un referente de la **traducción instantánea**.

La fusión de *big data*, internet de las cosas e inteligencia artificial permitirá evolucionar la gestión de las ciudades, permitiendo una gestión ágil de los recursos y espacios públicos en función de la actividad diaria de la ciudad.

Como se ha comentado en el apartado anterior, también se abre un abanico sin fin de posibilidades en el ámbito sanitario. Lograr una atención clínica auténticamente

personalizada al paciente es un reto por afrontar. La **medicina personalizada** considerará datos comportamentales, genéticos, sanitarios, etc., con el fin de proporcionar el mejor servicio que optimice el sistema preventivo, minimice la posibilidad de reincidencia y maximice las probabilidades de sanación.

La educación no puede permanecer ajena a la evolución de la inteligencia artificial, disciplina que posibilita mejorar brindar una auténtica **educación personalizada** para todos, de forma que pueda conseguir un aprendizaje efectivo ajustado a su perfil.

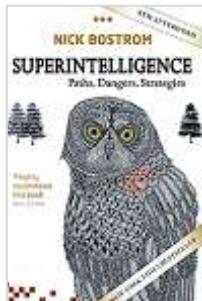
El sector comercial con sistemas de recomendación que se ajustan a nuestras preferencias, **supermercados inteligentes** que nos facilitarán la compra empleando el mínimo de personal, integración de todo tipo de dispositivo en la cadena de atención al cliente..., sufrirá una transformación difícil de predecir.

La industria financiera y aseguradora, el sector energético, minería, exploración espacial, seguridad, agricultura, ecología y medio ambiente, logística, etc., no hay campo del dominio humano que no se vaya a ver afectado por la evolución que marcará la inteligencia artificial.

A fondo

Superintelligence: Paths, dangers, strategies

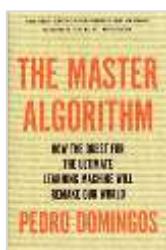
Bostrom, N. (2014). *Superintelligence: Paths, dangers, strategies*. Oxford: Oxford University Press.



Libro de referencia en el área que permite entender mejor el significado de la inteligencia y las distintas posibilidades de evolución que surgen.

The master algorithm

Domingos, P. (2014). *The master algorithm. How the quest for the ultimate learning machine will remake our world*. United Kingdom: Penguin.



Obra esencial para comprender los cambios venideros en la sociedad debido a los avances que marcará la inteligencia artificial.

1. ¿Qué es Alphabet?

- A. Una solución de Google para la traducción automática.
- B. Es la matriz a la cual pertenece Google o DeepMond, uno de los principales actores dentro del campo de la IA.
- C. Una empresa de Microsoft que destaca por sus contribuciones a la inteligencia artificial.
- D. Un alfabeto de lenguas antiguas que ha resultado muy útil para el desarrollo de los traductores automáticos.

2. ¿Qué es TensorFlow?

- A. Un *framework* que permite entrenar redes neuronales y modelos varios de *deep learning*.
- B. Un *software* para la realización de cálculo matricial.
- C. Una metodología para el desarrollo de redes neuronales.
- D. Una empresa subsidiaria de Google.

3. ¿Qué es Pytorch?

- A. Un *software* para aprender a programar en Python.
- B. Un *framework* basado en Python que facilita la limpieza de datos.
- C. Un *framework* que permite el desarrollo de modelos de *deep learning*.
- D. La división de Facebook encargada de la investigación en Python.

4. ¿Qué es AlphaGo?

- A. La división de investigación en *deep learning* de Microsoft.
- B. El primer programa informático basado en inteligencia artificial que ha conseguido derrotar al campeón mundial de *Go*.
- C. Está considerado como el programa informático basado en inteligencia artificial que mejor juega al *Go*, aunque nunca ha conseguido derrotar al campeón mundial.
- D. El programa informático más avanzado para jugar al ajedrez.

5. En cuanto a librerías de IA, señala la afirmación más correcta:

- A. Scikit-learn es una librería escrita en Python con multitud de modelos de IA y *machine learning*.
- B. Acord.net es una librería para *machine learning* y procesamiento de señales hecha para .Net.
- C. Universe es la librería *open source* de OpenAI.
- D. Todas las anteriores.

6. Una de las esencias fundamentales de Watson es:

- A. Que dispone de una interfaz gráfica sencilla y amigable que permite generar informes interactivos.
- B. Dar respuesta a preguntas formuladas mediante lenguaje natural.
- C. Que facilita a los programadores el desarrollo de soluciones usando algoritmos genéticos.
- D. Que facilita a los ingenieros *big data* la distribución de los datos entre servidores.

- 7.** Los sistemas de recomendación consideran el perfil del usuario para ofrecerles productos personalizados. Una de las compañías más avanzadas en este sentido, puesto que es una parte esencial de su actividad, es:
- A. El Corte Inglés.
 - B. IBM.
 - C. Microsoft.
 - D. Netflix.
- 8.** La medicina del futuro se caracterizará por:
- A. Tratamientos personalizados ajustados a las características del paciente.
 - B. Los robots harán innecesarios los médicos.
 - C. La creación de vacunas para todas las enfermedades.
 - D. La publicación en abierto de todos los datos clínicos de todos los pacientes para mejorar los resultados de investigación.
- 9.** Selecciona la respuesta más correcta. Retos que la inteligencia artificial afrontará en el futuro son:
- A. Coches autónomos.
 - B. Traducción instantánea.
 - C. Contribuir a una educación personalizada.
 - D. Todas las anteriores.
- 10.** El internet de las cosas...
- A. Puede resultar de gran valor para las «ciudades inteligentes».
 - B. Puede resultar muy útil en agricultura para un mejor control de las plantaciones.
 - C. Puede resultar muy útil en las fábricas para monitorizar toda la actividad.
 - D. Todas las anteriores.

Investigación en Inteligencia Artificial

Gestión de proyectos de inteligencia artificial

Índice

Esquema	3
Ideas clave	4
7.1. Introducción y objetivos	4
7.2. Conceptos generales de ingeniería del <i>software</i>	4
7.3 Ciclo de vida del proyecto de inteligencia artificial	16
7.4. Despliegue de soluciones de IA	19
7.5. Perfiles de participantes en proyectos de inteligencia artificial	23
7.6. Referencias bibliográficas	27
A fondo	28
Test	30

Esquema

Gestión de proyectos de inteligencia artificial en el ámbito empresarial

Proyecto

Un proyecto es un esfuerzo temporal emprendido para crear un único

Gestión de proyectos

Aplicación de conocimientos, habilidades, herramientas y técnicas para proyectar actividades destinadas a satisfacer las necesidades y expectativas de los interesados de un proyecto

Ciclo de vida del software

Marco de referencia que contiene los procesos, las actividades y las tareas involucradas en el desarrollo, la explotación y el mantenimiento de un producto de software abarcando la vida del sistema desde la definición de requisitos hasta la finalización de su uso

Metodologías ágiles

Permiten adaptar la forma de trabajo a las condiciones del proyecto, consiguiendo flexibilidad e inmediatez en la respuesta para amoldar el proyecto y su desarrollo a las circunstancias específicas del entorno

Conocimientos base y vocabulario

Método científico
CRISP-DM
Data Science Process
Entorno de análisis

Líneas base del proyecto
Perfiles involucrados

7.1. Introducción y objetivos

La gestión de proyectos de inteligencia artificial presenta algunas particularidades concretas. Sin embargo, todas las metodologías que se aplican al desarrollo ágil de *software* se pueden incorporar a esta disciplina aportando un valor notable. En esta unidad nos planteamos los objetivos siguientes:

- ▶ Comprender el ciclo de vida de un proyecto de inteligencia artificial.
- ▶ Saber diferenciar entre un entorno de análisis y un entorno de producción.
- ▶ Ser capaz de reconocer a grandes rasgos qué implica la gestión de un proyecto de inteligencia artificial

7.2. Conceptos generales de ingeniería del *software*

Un **proyecto** es un esfuerzo temporal emprendido para crear un único producto o servicio. Se debe **distinguir claramente los proyectos de las operaciones**. Hacer una nueva carretera es un proyecto. Mantener una carretera en las condiciones adecuadas para que sea transitable forma parte de las operaciones de mantenimiento. Las operaciones finalizan cuando finaliza la vida del producto o servicio en cuestión. El proyecto finaliza cuando se ha creado el producto o servicio y, normalmente, entregado al cliente.

La **gestión de proyectos** es la aplicación de conocimientos, habilidades, herramientas y técnicas para proyectar actividades destinadas a satisfacer las necesidades y expectativas de los interesados (*stakeholders*) de un proyecto. Entre estos interesados

podemos encontrar, además de los propios clientes o usuarios, a otros actores como: directivos, servicios jurídicos de la compañía, departamento de seguridad, departamento de marketing, entidades reguladoras, legisladores, gobiernos, grupos de presión, etc.

¿Qué tipo de producto o servicio se espera de un proyecto de inteligencia artificial? Quizá estemos pensando en un coche autónomo, un robot, etc. Todos esos productos tienen algo en común: *software*. El *software* es el producto más habitual que un proyecto de estas características genera. Por supuesto, en muchas ocasiones el *software* viene acompañado de hardware o algún tipo de componente mecánico que ejecuta las órdenes emitidas por el *software*. Dado el ámbito y alcance del presente máster, nos centraremos en el *software*, que será el vehículo en el que se expresarán nuestros algoritmos y modelos.

Así pues, vamos a definir cuál es el ciclo de vida de un proyecto *software* y a continuación veremos las diferencias entre este y el ciclo de vida de un proyecto de inteligencia artificial. Como veremos, hay muchas ideas similares, pero también importantes diferencias.

El ciclo de vida del *software*

Se considera ciclo de vida de un producto *software* a la **evolución del producto desde su concepción hasta su retirada**. Las actividades del ciclo de vida deben ser separadas en fases.

Siendo más explícito, el ciclo de vida del *software* es el marco de referencia que contiene los procesos, las actividades y las tareas involucradas en el desarrollo, la explotación y el mantenimiento de un producto de *software*, abarcando la vida del sistema desde la definición de requisitos hasta la finalización de su uso.

Los modelos de ciclo de vida han evolucionado desde los modelos de una fase de 1968. Esta evolución se ha dirigido a mitigar los riesgos y a conseguir una mejor adaptación del producto a lo que realmente se espera de él. En los inicios, el **ciclo de vida en cascada era la principal opción**. Este ciclo de vida determinaba claramente una serie de fases que, de una forma u otra, se han mantenido más o menos comunes en la mayoría de los modelos posteriores.

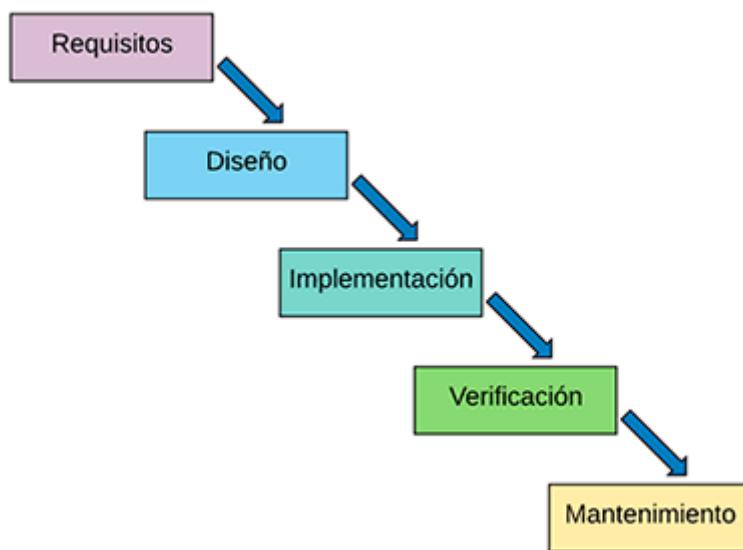


Figura 1. Ciclo de vida en cascada.

Fuente: <https://openclassrooms.com/courses/gestionas-tu-proyecto-de-desarrollo/en-que-consiste-el-modelo-en-cascada>

Pero tal cual se definió, este modelo presentaba serios problemas ya que la fase de verificación tenía lugar al final del proceso. En caso de detectar errores en la implementación, en el diseño o en la propia toma de requisitos (que determina los objetivos a conseguir) el camino a desandar implica unos costes más que considerables en tiempo y dinero.

El **modelo del ciclo de vida en espiral** desarrollado por Boehm en 1986 introdujo elementos importantes al incorporar una completa gestión del riesgo y control de la calidad. El desarrollo iterativo permite avanzar gradualmente hacia el producto final, mitigando los riesgos existentes y adelantando la detección de problemas y sus soluciones lo antes posible.

Este modelo **da participación continua al cliente, evaluando el producto de forma constante**. De esta forma, los errores se pueden detectar de forma anticipada. Las distintas iteraciones permiten generar diferentes prototipos del producto final. Por lo que este modelo exige equipos muy preparados y un seguimiento continuo y detallado.



Figura 2. Ciclo de vida en espiral.

Fuente: <https://upload.wikimedia.org/wikipedia/commons/thumb/3/39/ModeloEspiral.svg/359px-ModeloEspiral.svg.png?>

El **modelo del ciclo de vida iterativo e incremental** es la base de los modelos más usados a dia de hoy. Con este modelo se persigue que el cliente reciba de forma continua algo de valor. Según se avanza en el número de iteraciones el producto final incorpora más funcionalidades y se convierte en un producto más elaborado y complejo. Este modelo permite acometer el producto en base a pequeños entregables que hacen más realizable la tarea.

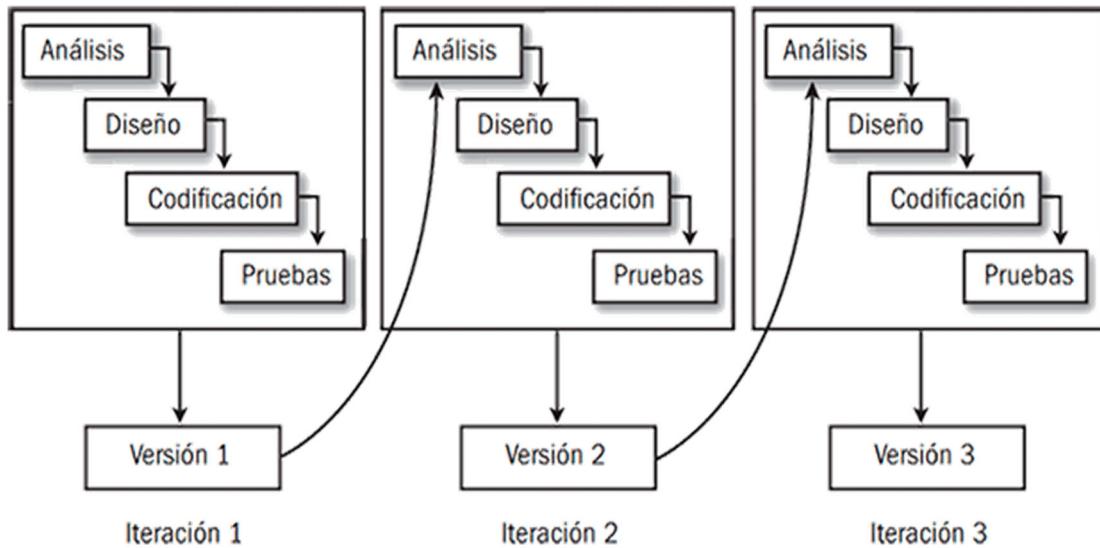


Figura 3. Ciclo de vida iterativo e incremental.

Fuente: http://3.bp.blogspot.com/-6HiwEBD3k1A/T5YoY36dtZI/AAAAAAAEEk/EQ_Y5eVPo7Y/s1600/iterativo.png

Antes de iniciar el proyecto no solo debe estar claro el ciclo de vida escogido sino también la metodología seleccionada.

El principal problema del modelo iterativo en espiral es que la carga de trabajo no es proporcional en todas las iteraciones. Por lo que en la práctica se usan modelos corregidos como el que se muestra en la imagen

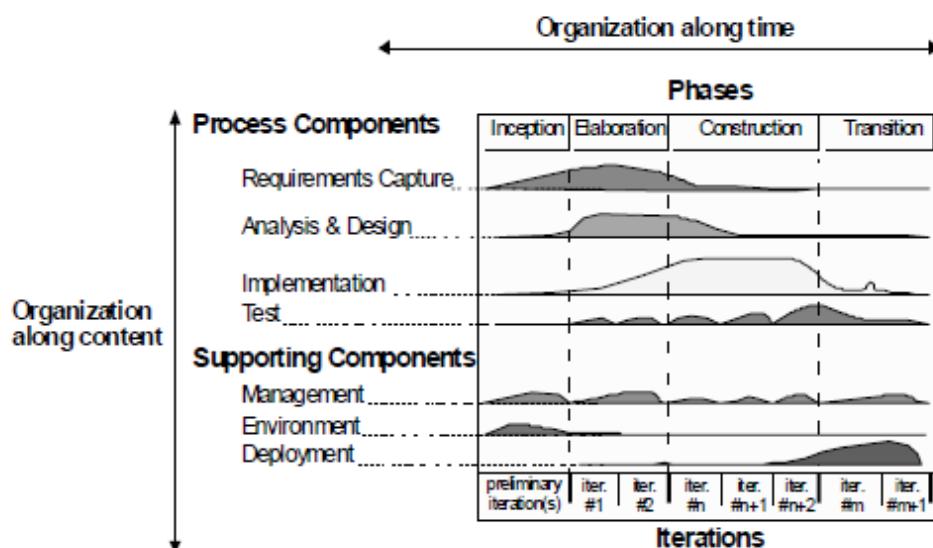


Figura 4. Diferentes cargas en las diferentes iteraciones.

De esto surge el denominado **modelo Proceso Unificado**. Este es el modelo culmen de la ingeniería del *software* clásica. Un modelo formal, muy documentado y que mantenerlo en funcionamiento requiere de multitud de recursos. Su principal ventaja es que es muy verificable y está muy documentado, lo que le hace ideal para proyectos de alto riesgo y que necesitan una verificación y un control de calidad exhaustivo. Su principal inconveniente es su excesiva sobrecarga para llevarla a cabo. Actualmente este modelo solo se usa para entornos muy concretos como proyectos de alto riesgo, para las grandes administraciones, para seguridad del estado, sistemas críticos, etc.

Metodologías de desarrollo de *software*

Una metodología es un **conjunto de procedimientos que permiten producir y mantener un producto software**. La metodología define cuáles son las fases del ciclo de vida del *software* de las que se va a ocupar, definiendo para cada fase las actividades, tareas, los objetivos correspondientes y los participantes involucrados en cada tarea.

Para llevar a cabo las tareas, **las metodologías utilizan técnicas que describen cómo llevarlas a cabo**. Las técnicas suelen ser gráficas con apoyos textuales formales, lo que mejora una descripción sencilla, detallada y comprensible para el usuario. Las reglas de notación y sintaxis de cada técnica se dan con la guía de la metodología. Existen muchas y diversas metodologías: MÉTRICA 3, Structured Systems Analysis and Design Methodology (SSADM), Rational Unified Process (RUP), Extreme Programming (XP), PRojects IN Controlled Environments 2 (PRINCE2), International Project Management Association (IPMA), la norma ISO 21500:2012, etc.

Por ejemplo, un paso crucial en cualquier proyecto es la **definición de requisitos**. Es esta una tarea nada trivial donde es esencial detectar objetivos y las suposiciones de partida de forma precisa y rigurosa. Es muy habitual en este contexto la imagen siguiente explicando el tipo de confusiones a las que puede dar lugar no manejar un lenguaje y alcance compartido del proyecto.

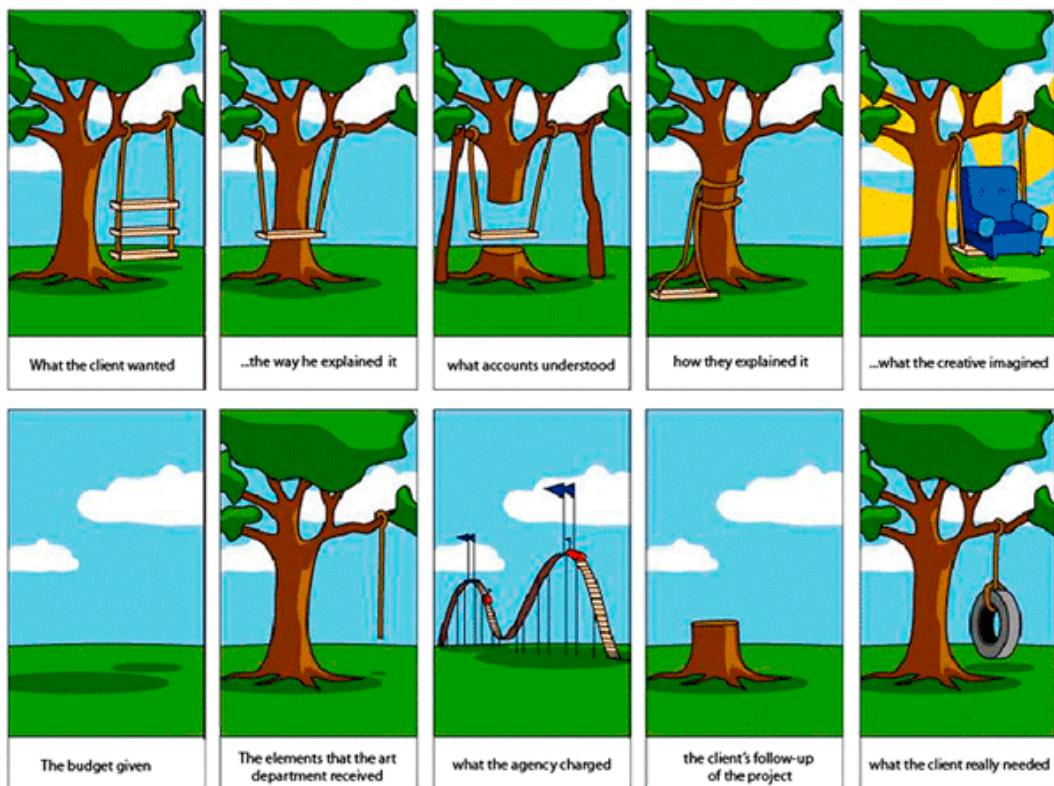


Figura 5. Requisitos de un proyecto de *software*, confusiones.

Fuente: <https://tavovalencia.files.wordpress.com/2010/08/creative-process.png>

Sin duda alguna, una de las instituciones que más ha aportado a la estandarización de la práctica de dirección de proyectos es el **Project Management Institute (PMI)**. Se trata de una asociación profesional sin fines de lucro a nivel mundial, que tiene como misión promover el desarrollo del conocimiento y competencias básicas para el ejercicio profesional de la dirección de proyectos. Ofrece varias certificaciones siendo la certificación Project Management Professional (PMP) una de las más solicitadas. Los conocimientos que requiere esta certificación están recogidos en el Project Management Body of Knowledge (PMBOK).

El PMBOK no se trata de una metodología en sí, sino de una guía de estándares internacionales para que los usuarios puedan adaptar a cada caso y contexto particular los procesos reconocidos como buenas prácticas por el PMI. El PMBOK es el marco de referencia formal para desarrollar proyectos y alcanzar los objetivos. En septiembre de 2017 se publicó la sexta versión de este, actualmente vigente.

A través de siguiente enlace podremos acceder a una explicación gráfica de los distintos procesos: <http://rvarg.as/pmbok6en>

Gestión ágil de proyectos

Las metodologías ágiles **permiten adaptar la forma de trabajo a las condiciones del proyecto**, consiguiendo flexibilidad e inmediatez en la respuesta para amoldar el proyecto y su desarrollo a las circunstancias específicas del entorno. Se persigue de esta forma reducir costes e incrementar la productividad, al avisar rápidamente de los errores y problemas encontrados. Este proceso surge como un rebote frente a los modelos tradicionales donde cada vez se tendía a introducir más y más documentación y más y más procesos que dificultaban y alargaban enormemente los proyectos. Entre las metodologías ágiles más aplicadas hoy en día tenemos Scrum, Kanban, Scrumban y Extreme Programming (XP). Se comentará muy brevemente Scrum y Kanban por ser metodologías de uso habitual en los proyectos que nos ocupan.

Las metodologías ágiles **suelen ser útiles en proyectos donde los requisitos no están claros o son muy cambiantes**. En este tipo de proyectos, el prototipado y la investigación continua marcan el camino a seguir. Estas metodologías ágiles son la clave del éxito de las *startups*. Poder desarrollar rápido hace que una *startup* tenga una ventana de oportunidad dentro de su nicho de mercado, saliendo antes que la competencia y pudiendo captar clientes y posicionarse en el mercado. Los modelos ágiles facilitan la salida de un producto con la funcionalidad mínima y que se puede ir mejorando e incrementando. Sin embargo, con los modelos tradicionales el

desarrollo se retrasa demasiado y es mucho mas complicado adelantarte al mercado. El auge de este tipo de compañías es una de las principales claves del éxito de estas metodologías.

Las metodologías ágiles han desplazado a las tradicionales en muchos entornos de desarrollo, pero eso no significa que estas no tengan cabida. Se siguen utilizando en entornos donde se necesita un control exhaustivo de los procesos y donde el resultado del *software* es crítico como por ejemplo en *software* para el ejército, para sistemas de control críticos o para grandes proyectos en la administración.

Scrum

Scrum fue desarrollada por Ikujiro Nonaka e Hirotaka Takeuchi a principios de los 80, analizando cómo desarrollaban los nuevos productos destacadas empresas de manufactura tecnológica como Fuji-Xerox, Canon, Honda, etc. El término Scrum proviene de la figura del rugby del mismo nombre y se escogió por sus implicaciones a la hora de referirse al trabajo en equipo.



Figura 6. Scrum de rugby.

Fuente:

https://upload.wikimedia.org/wikipedia/commons/thumb/5/5b/2014_Women%27s_Six_Nations_Champions_hip - France Italy %2851%29.jpg/300px-2014_Women%27s_Six_Nations_Championship_-France_Italy_%2851%29.jpg?

Scrum se caracteriza por la planificación en base a iteraciones (*sprints*), con una duración de entre dos y cuatro semanas. Cada *sprint* genera un entregable para el cliente. Las necesidades o requisitos del cliente se anotan en una «pila» de

producto de forma priorizada. Esta lista será creada y gestionada por el cliente con la ayuda del equipo.

La lista de los trabajos que debe realizar el equipo durante el *sprint* se añade a la «pila» del *sprint* y se selecciona una persona responsable de su ejecución. Así el proyecto se descompone en unidades más pequeñas y gestionables.

En ocasiones, la «pila» del *sprint* acumula un número excesivo de tareas para que el equipo pueda abarcárlas con eficiencia. Kanban es otra metodología que trata de solucionar esos excesos.

En Scrum **las tareas a realizar se denominan historias**. Estas historias tienen unos puntos de esfuerzo o puntos de historia. Estos puntos que se van colocando de forma manual por los propios encargados de realizarlas denotan el esfuerzo que considera el responsable que van a tener llevarlas a cabo. Este esfuerzo se va recalculando en función del número de historias que el responsable es capaz de resolver. Por ejemplo, en una semana un ingeniero ha podido resolver tareas por valor de 5 puntos de esfuerzo en media. De forma que si se le asignan tareas que él estima que tendrían 10 puntos de esfuerzo debería tardar dos semanas en realizar. Si esto no es así y las realiza en una semana, incrementará el numero de puntos de esfuerzo que es capaz de resolver. De esta forma se regula de forma dinámica la gestión del trabajo y el esfuerzo y con unas cuantas iteraciones, el equipo es capaz de planificar mucho mejor cuándo tendrá la próxima iteración y con qué contenido.

La figura del **Scrum Máster** es una figura capital dentro de Scrum. Sus funciones son la de asegurar que se cumplen las buenas prácticas y valores descritos en el modelo Scrum.

El Scrum Máster tendría una figura similar a la de un *coach/mentor* que acompañará al equipo durante todo el desarrollo del proyecto y asegurará que se cumplan las buenas prácticas, actuando como un facilitador y solucionador de problemas.

Entre otras funciones, el Scrum Máster debe:

- ▶ Asesorar y formar a los diferentes miembros para trabajar de forma autoorganizada y con responsabilidad de equipo.
- ▶ Moderar las reuniones.
- ▶ Resolver impedimentos que en el *sprint* pueden entorpecer la ejecución de las tareas.
- ▶ Gestionar las dinámicas de grupo en el equipo.
- ▶ Encargarse de la configuración, diseño y mejora continua de las prácticas de Scrum en la organización.

Kanban

Kanban nace en Japón alrededor del año 1950 como una propuesta para manejar de forma eficiente el flujo de materiales en una línea de producción. Algunos de sus componentes principales son:

- ▶ La visualización del flujo de trabajo: dividiendo el trabajo en bloques y escribiendo cada elemento en una tarjeta para después ponerlo en la tabla o pizarra creada para ello.
- ▶ La limitación del trabajo en curso (*work in progress-WIP*) asignando límites concretos del número de elementos que pueden estar en progreso en cada estado del flujo trabajo.
- ▶ La medición del denominado tiempo de ciclo que es el tiempo medio para completar un elemento y también ayuda a optimizar el proceso para conseguir que este tiempo sea lo más pequeño y predecible posible.

El siguiente puede ser un ejemplo de tablero Kanban.

Backlog (4)	Análisis (3)		Construcción (3)		Revisión (4)		Terminado
	En curso	Hecho	En curso	Hecho	En curso	Hecho	
	<input type="checkbox"/>						
	<input type="checkbox"/>						
	<input type="checkbox"/>						
	<input type="checkbox"/>						

Figura 7. Tablero Kanban.

Fuente: <http://www.scrummanager.net/blog/2013/09/kanban-las-4-cosas/>

El control visual del proceso y la limitación del número de trabajos en curso contribuye a un entendimiento más completo del estado del proyecto y permite que las tareas avancen de forma eficiente.

Es muy común ver usar Kanban y Scrum simultáneamente en el mismo proyecto.

Existen multitud de herramientas para realizar Kanban y Scrum, pero por citar las más usadas puedes echar un vistazo a **Trello**, un gestor de tareas de tipo Kanban gratuito o **Pivotal Tracker** que usa la metodología Scrum y el concepto de historia y los puntos de esfuerzo que es capaz de calcular y actualizar de forma dinámica. Aunque hay otros como Active Collab, Taiga, Kanbanize, Atlassian Jira, iceScrum, Scrumblr, etc.

7.3 Ciclo de vida del proyecto de inteligencia artificial

Las metodologías y procedimientos explicados en los apartados anteriores sirven para cualquier tipo de proyecto de *software*, desde un entorno virtual hasta un *software* de gestión contable.

Sin embargo, la tipología de proyectos que tratamos en este máster tiene una serie de particularidades que es preciso reflejar. En primer lugar, porque **la inteligencia artificial como ciencia que es nos obliga a seguir el método científico**. El método científico ya lo hemos explicado en anteriores temas así que vamos a resumirlo, este método presenta un conjunto de pasos ordenados, que se emplea principalmente en el hallazgo de nuevos conocimientos en las ciencias. La figura 8 resume los pasos básicos del método científico:

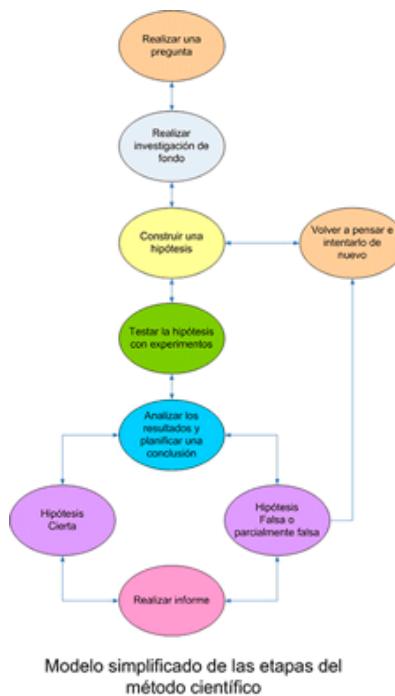


Figura 8. El método científico.

Fuente:

https://upload.wikimedia.org/wikipedia/commons/thumb/d/dd/M%C3%A9todo_cient%C3%ADfico.jpg/300px-M%C3%A9todo_cient%C3%ADfico.jpg?

Contextualizando dicho método a nuestro ámbito, y más concretamente en el terreno de la minería de datos, la metodología CRISP-DM, de Cross Industry Standard Process for Data Mining, ha tenido gran aceptación entre los distintos expertos. La figura 9 resumen los pasos esenciales:



Figura 9. Metodología CRISP-DM.

Fuente: <https://www.kdnuggets.com/2017/01/four-problems-crisp-dm-fix.html>

Es preciso mencionar que el proceso parte del entendimiento del negocio, de forma que podamos detectar necesidades y oportunidades susceptibles de ser resueltas con un enfoque innovador.

El ecosistema de la ciencia de datos viene a aportar un nuevo enfoque. En concreto, Rachel Schutt, en su libro *Doing Data Science* viene a proponer el siguiente esquema:

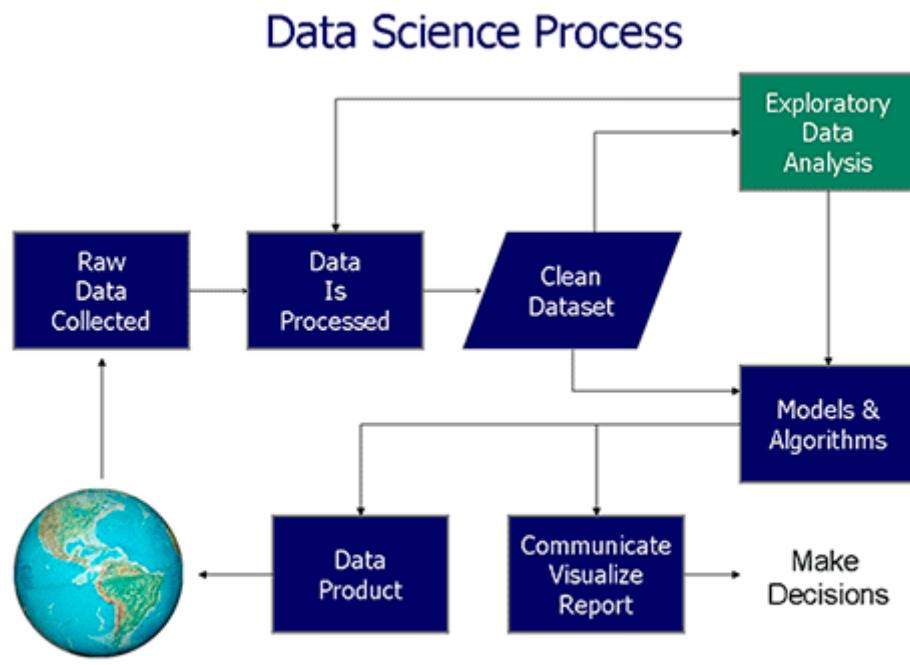


Figura 10. *Data science process*.

Fuente: <https://www.r-bloggers.com/a-data-science-solution-to-the-question-what-is-data-science/>

Dicho esquema detalla la parte de tratamiento de datos tan propia del ecosistema *big data* e incorpora nuevas relaciones entre las distintas fases. La generación de modelos que sean capaces de obtener patrones de los datos y realizar predicciones se convierte en parte central del esquema. Así como la visualización y comunicación de resultados adquiere especial relevancia frente a otros modelos más generalistas.

Una diferencia esencial cuando hablamos de modelos frente a un proyecto tradicional de *software* es que **una vez puesto el modelo a disposición de los usuarios puede ser necesario evaluar su comportamiento de forma periódica y ajustar los parámetros de este**. Los usuarios del modelo pueden cambiar sus hábitos y la propia solución puede inducir a comportamientos no previstos en un inicio.

7.4. Despliegue de soluciones de IA

Para que un especialista pueda realizar su trabajo necesita de una infraestructura informática con capacidad de cómputo y de almacenamiento de información. Dependiendo de las características del proyecto, la potencia de estos recursos puede variar.

En el entorno empresarial es habitual contar con dos infraestructuras diferenciadas: un entorno de análisis más adecuado para investigación y pruebas, y un entorno de producción donde reside el producto final para su interacción con los usuarios.

Este es el caso cuando, por ejemplo, un equipo desarrolla un nuevo modelo basado en inteligencia artificial para detección de ataques informáticos a una gran empresa. Los técnicos desarrollarían el modelo empleando el entorno de análisis. Una vez depurado, se instalaría en el entorno de producción analizando todo el tráfico de red a través de la compañía y detectando las amenazas correspondientes.

Evidentemente, cada proyecto tiene su tipología, en el caso de proyectos con robots, las propias unidades mecánicas o un prototipo podría emplearse hasta dar con la solución final. Una vez validada la solución pasaría a formar parte del producto final antes de salir al mercado o extender su uso.

En cualquier caso y aunque quizá sea más necesario en proyectos de aprendizaje automático en particular, conocer y comprender una serie de conceptos básicos es necesario para considerarse un experto en el área.

Vocabulario básico

Mencionaremos brevemente algunos conceptos que creemos necesario manejar:

Centro de proceso de datos (CPD)

Infraestructura informática donde se concentran los recursos necesarios para el procesamiento de la información de una organización.

Virtualización

La virtualización es una tecnología que permite crear múltiples entornos simulados o recursos dedicados desde un solo sistema de *hardware* físico. El *software* llamado «hipervisor» se conecta directamente con el *hardware* y permite dividir un sistema en entornos separados, diferentes y seguros, que se denominan «máquinas virtuales» (VM del inglés *virtual machine*). Estas VM dependen de la capacidad del hipervisor de separar los recursos de la máquina del *hardware* y distribuirlos adecuadamente. La máquina física original equipada con el hipervisor se denomina *host*.

Cloud computing (computación en la nube)

El paradigma que facilita acceder a programas y servicios alojados en un servidor conectado a una red informática y accesible desde otro ordenador con conexión a dicha red sin necesidad de instalar aplicaciones ejecutables en su disco duro y donde también se almacena la información generada por estas mismas aplicaciones o servicios. Posee varios modelos de servicio y modelos de despliegue.

Algunos términos **relacionados con la computación en la nube**:

- ▶ Modelos de servicio: establece el tipo de productos y servicios accesibles a través de la computación en la nube.
- ▶ Software as a Service (SaaS): modelo de servicio que ofrece acceso a aplicaciones y servicios que se ejecutan en la infraestructura *cloud*. A la aplicación se accede desde múltiples dispositivos. El usuario no tiene que gestionar redes, servidores, sistemas operativos, almacenamiento, versiones de *software*, etc. Un ejemplo de estos productos serían Google Drive, Dropbox o Gmail.
- ▶ Platform as a Service (PaaS): modelo de servicio que ofrece desarrollo y ejecución de aplicaciones propias o de terceros sobre las herramientas, bibliotecas y entornos de desarrollo disponibles en la nube. El usuario no tiene que gestionar las infraestructuras, solo las aplicaciones. Ejemplos de uso son Microsoft Azure, Google App Engine o Force.com.
- ▶ Infrastructure as a Service (IaaS): modelo de servicio que permite el uso y aprovisionamiento de procesamiento, almacenamiento, redes y otros recursos para ejecutar cualquier *software*, incluyendo sistemas operativos y aplicaciones. El usuario tiene control sobre el sistema operativo, configuración de redes, etc. Amazon Web Services (AWS), Cisco Metapod, Microsoft Azure o Google Compute Engine (GCE) son ejemplos destacados en este caso.
- ▶ Modelos de despliegue: establece y limita las características del acceso a los productos y servicios ofrecidos en la nube.
- ▶ Nube privada: acceso exclusivo para una organización, normalmente la dueña y gestor de la infraestructura.
- ▶ Nube comunitaria: acceso exclusivo para una comunidad de organizaciones alineadas por algún objetivo común (política, estándar, seguridad).
- ▶ Nube pública: acceso abierto para el público general, se encuentra en las instalaciones del proveedor de *cloud*.
- ▶ Nube híbrida: combina nubes privadas y públicas.

Por lo tanto, para desplegar un proyecto la principal idea que debemos tener en mente es que **tenemos que evitar tener duplicado el entorno de trabajo en dos entornos**. Una de desarrollo y otro de despliegue. Para ello podemos hacer uso de **DevOps** (acrónimo inglés de development [desarrollo] y operations [operaciones]) que es una práctica de ingeniería de *software* que tiene como objetivo unificar el desarrollo de *software* (Dev) y la operación del *software* (Ops). Utilizando este tipo de tecnologías explicadas anteriormente, podemos construir *software* directamente en el servidor usando un modelo **X As a Service**. Esa es la aproximación que hacen la mayoría de las grandes empresas que manejan actualmente los proyectos de IA y *machine learning* como ya vimos anteriormente. Y esto es así en gran medida gracias a las facilidades para el despliegue que estos entornos poseen y a las enormes capacidades de computación que ofrecen. Los **principales modelos** que podemos utilizar son:

- ▶ Software as a Service (SaaS): modelo de servicio que ofrece acceso a aplicaciones y servicios que se ejecutan en la infraestructura *cloud*. El usuario no gestiona nada a nivel de operaciones. Ejemplos: Algunos productos comerciales como Dropbox, Gmail, Google Maps, etc.
- ▶ Platform as a Service (PaaS): modelo de servicio que ofrece desarrollo y ejecución de aplicaciones propias o de terceros sobre las herramientas, bibliotecas y entornos de desarrollo disponibles en la nube. El usuario no tiene que gestionar las infraestructuras, solo las aplicaciones. Ejemplos: Microsoft Azure o Google App Engine.
- ▶ Infrastructure as a Service (IaaS): modelo de servicio que permite el uso y aprovisionamiento de procesamiento, almacenamiento, redes y otros recursos para ejecutar cualquier *software*, incluyendo sistemas operativos y aplicaciones. El usuario tiene control sobre el sistema operativo, configuración de redes, etc. Ejemplos: Web Services (AWS), Cisco Metapod, Google Compute Engine (GCE)

7.5. Perfiles de participantes en proyectos de inteligencia artificial

Uno de los mayores retos para el director de proyectos en cualquier ámbito es estimar las tres principales líneas base del proyecto:

- ▶ Alcance: detallando las funcionalidades a alcanzar y los servicios a ofrecer.
- ▶ Tiempo: estimar el marco temporal que guiará el proyecto.
- ▶ Coste: estimar los costes que ocasionará el proyecto. Implica estimar recursos teniendo en cuenta todos los objetivos del trabajo.

Dado que esta tarea no es en absoluto trivial, **las metodologías ágiles se están imponiendo como un estándar dado que permiten la descomposición del proyecto en bloques realistas, iterativos e incrementales**. No obstante, procede sacar a relucir ciertos costes ocultos que suelen ser frecuentes en este tipo de proyectos.



Figura 11. Enfoque tradicional versus enfoque agile.

Fuente: <http://www.pmoinformatica.com/2017/12/pmbok-6-metodologias-agiles.html>

Sculley saca a la luz gran parte de los costes asociados a una tipología especial de proyectos, los que están **basados en el desarrollo de modelos de aprendizaje automático** (Sculley, Phillips, Ebner, Chaundhary y Young, 2014). Los factores reflejados a continuación incrementan los costes económicos del proyecto y reducen el grado de innovación en el mismo:

- ▶ Complejidad estructural: a medida que la solución propuesta se vuelve más compleja la velocidad de desarrollo disminuye, cada nuevo componente requiere el análisis de dependencias adicionales, la detección de errores se convierte en un laberinto, se requiere personal más cualificado para el desarrollo, etc.
- ▶ Dependencias de datos: la creación de modelos requiere de datos. Las dependencias y relaciones entre datos pueden añadir inestabilidad en el modelo, dicha inestabilidad se ve reflejada por cambios notables en el comportamiento del modelo ante ligeras variaciones en los datos.
- ▶ Complejidad del código: no solo la proliferación de múltiples módulos interrelacionados entre sí incrementa la complejidad de la solución. Según crecen las líneas de código aumenta el esfuerzo necesario para validar y gestionar toda la codificación. El uso de librerías externas y código desarrollado por terceros puede insertar importantes capas de ineficiencias.
- ▶ Cambios en el entorno externo: la experiencia demuestra que el mundo externo es inestable por naturaleza, presentando cambios continuos de forma notable. En este sentido, es un **gran error entregar un modelo al cliente o usuario final y olvidarnos de él**. Es preciso revisar de forma más o menos constante, según el caso, los resultados que genera y comprobar que sigue alineado con los objetivos iniciales. Estas acciones generan un coste que debe ser considerado de antemano.

Precisamente debido a la dificultad para precisar de forma exacta y rigurosa los costes económicos y temporales del proyecto, un enfoque frecuentemente

adaptado consiste en **llegar a la mejor solución posible prefijando de antemano un plazo de tiempo y presupuesto inicial**. Acometiendo entregas iterativas y una gestión de proyectos ágil se intenta llegar a la mejor versión posible cumpliendo con las restricciones iniciales. Llegado el momento y en función de los resultados obtenidos se determina si procede la asignación de una nueva partida presupuestaria y nuevo límite temporal para la siguiente versión.

Gestión de equipos analíticos y de investigación

Los expertos en el área de inteligencia artificial no abundan en el mercado. El perfil más común acumula en su currículum una carrera técnica o científica además de un máster especializado y varios cursos de especialización.

Un factor intrínseco a estos profesionales es la renovación continua de conocimientos y capacidades. En un entorno tan innovador y cambiante es pura ilusión pensar que un conjunto de conocimientos estáticos adquiridos en un momento concreto servirá para toda la vida.

Evidentemente, existe un bloque común de conocimientos del que partirá el resto de las competencias. Sin embargo, la formación continua es una realidad ineludible en este tipo de perfiles.

Gestionar estos equipos suele implicar **habilidades para liderar un equipo de alto rendimiento**. Este tipo de equipos suele demandar autonomía, *feedback* preciso y explicación clara y rigurosa de lo que se espera de ellos. Las metodologías ágiles facilitan estos elementos.

Detallar de forma rigurosa técnicas efectivas de gestión de equipos escapa del ámbito de la asignatura por demandar competencias no contempladas en el programa. Sin embargo, sí que es necesario conocer las **características de algunos perfiles** que suelen estar involucrados en este tipo de proyectos:

- ▶ Analista de negocio: **perfil con formación variada y conocimiento concreto del negocio.** No suele ser un experto técnico, pero sí comprende las necesidades de los clientes y usuarios y conoce las tendencias del mercado. Pero los perfiles más científicos suele ser un elemento clave en el que apoyarse para entender las características que debe tener el producto final en un entorno empresarial.
- ▶ Jefe de proyecto: persona con formación específica en el área y en metodologías ágiles de gestión de proyectos que se responsabiliza de la gestión de equipo, gestiona las expectativas de los interesados y controla la calidad del producto final.
- ▶ Arquitecto de datos: perfil responsable de proporcionar una infraestructura, normalmente basada en entornos *cloud*, que almacena toda la información y proporciona capacidad de cómputo de forma escalable y tolerante a fallos. Es el encargado de diseñar los recursos técnicos adecuados para el desarrollo del proyecto.
- ▶ Ingeniero de datos: sobre la infraestructura diseñada principalmente por el arquitecto de datos, desarrolla *software* eficiente de diversa complejidad que procesa datos cumpliendo con los objetivos del proyecto.
- ▶ Científico de datos / experto en inteligencia artificial / experto en aprendizaje automático: profesional con altas capacidades analíticas y conocimiento experto que extrae valor de la información disponible, genera modelos analíticos y diseña algoritmos en base a los objetivos del proyecto.

La lista anterior está centrada en perfiles analíticos y más relacionados con el aspecto más puramente técnico y volcado en el tratamiento de la información. Por supuesto, no se trata de una lista cerrada ya que de forma frecuente perfiles menos técnicos y más centrados en cuestiones de diseño y experiencia de usuario, por ejemplo, están teniendo un papel muy relevante en este tipo de proyectos. Estos expertos tienen la misión de generar una experiencia de usuario gratificante y ejemplar, factor que suele ser una parte muy importante del éxito de un proyecto.

7.6. Referencias bibliográficas

Sculley, D., Phillips, T., Ebner, D., Chaudhary, V., y Young, M. (2014). Machine learning: The high-interest credit card of technical debt. *Google Reasearch*. Recuperado de

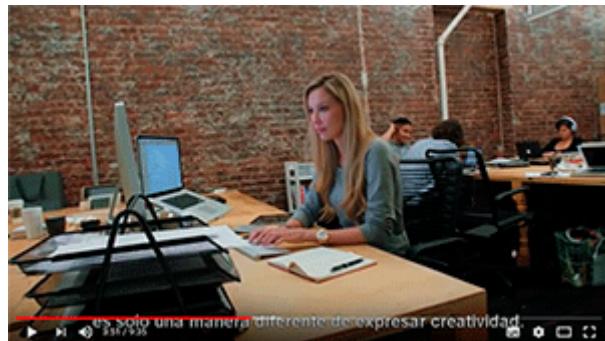
<https://static.googleusercontent.com/media/research.google.com/es//pubs/archive/43146.pdf>

A fondo

Code.org: la importancia de la programación

Quenerapú, I. (9 de marzo de 2013). *Code.org* [Vídeo]. Youtube.

<https://www.youtube.com/watch?v=Qhn48RekQeg>



En este vídeo se explica la importancia de la programación informática y por qué debería convertirse en un conocimiento transversal en la escuela.

WIP: why limiting work in progress makes sense (Kanban)

Lowe, D. (7 de octubre de 2013). *WIP: why limiting work in progress makes sense (Kanban)* [Vídeo]. YouTube.

<https://www.youtube.com/watch?v=W92wG-HW8gg>



Explicación del concepto *work in progress* (WIP) de la metodología Kanban y sus implicaciones para la gestión del proyecto.

1. Indica cuál de las siguientes frases no refleja una característica de un proyecto:

 - A. Es un esfuerzo temporal.
 - B. Incluye la fase de mantenimiento del producto.
 - C. Pretende crear un único producto o servicio.
 - D. Debe tener en cuenta a los interesados del proyecto.

2. El ciclo de vida del *software*:

 - A. Abarca la evolución desde el primer prototipo del producto hasta su retirada.
 - B. Abarca la evolución desde la concepción del producto hasta su retirada.
 - C. Abarca la evolución desde la concepción del producto hasta su puesta en producción.
 - D. Abarca la evolución desde que se consiguen los fondos necesarios hasta que no se dispone de presupuesto para continuar con el servicio.

3. El ciclo de vida en cascada:

 - A. Permite gestionar de forma eficaz todo tipo de proyectos.
 - B. Es ideal para proyectos de IA.
 - C. El diseño se hace de forma iterativa.
 - D. No permite una detección temprana de errores y fallos en la toma de requisitos.

4. El ciclo de vida iterativo e incremental:

- A. Permite entregar «algo» de valor al cliente o usuario final de forma iterativa.
- B. No permite entregar «algo» de valor al cliente o usuario final de forma iterativa.
- C. Incluye una única fase de análisis y en la primera iteración.
- D. Incluye una única fase de diseño y en la primera iteración.

5. Una metodología:

- A. Se limita a especificar cómo se realizará el diseño del producto.
- B. No debe contemplar elementos de gestión del equipo.
- C. Define un conjunto de procedimientos que permiten producir y mantener un producto *software*.
- D. Estándar es la PMP.

6. La gestión ágil de proyectos:

- A. Permite ir más rápido ya que no obliga a documentar.
- B. Es especialmente útil donde los requisitos son cambiantes y no están definidos con precisión.
- C. Se basa en poner en una pizarra las tareas pendientes.
- D. Sirve para todo tipo de proyectos en todos los ámbitos.

7. La limitación del trabajo en curso o *work in progress* (WIP):

- A. Es un concepto difundido por el PMI.
- B. Aplica a todo tipo de proyectos y metodologías.
- C. Es un concepto abstracto y poco práctico.
- D. En la metodología Kanban limita el número de elementos que pueden estar en progreso en cada estado del flujo de trabajo.

- 8.** El *data science process*, ¿en qué se diferencia fundamentalmente de los modelos ágiles?
- A. Que es más corto y con menos fases que los proyectos normales.
 - B. Que no necesita mantenimiento.
 - C. En que el modelo una vez puesto en marcha probablemente necesite ser reevaluado.
 - D. Es más lineal debido al poco riesgo en los requisitos de estos proyectos.
- 9.** Por *cloud computing* o computación en la nube entendemos (selecciona la opción más completa):
- A. Navegar por internet.
 - B. Navegar por internet y acceder al correo.
 - C. Acceder de forma remota a programas y servicios alojados en un servidor conectado a una red informática.
 - D. Un servidor FTP alojando en mi equipo personal.
- 10.** ¿Cuáles de las siguientes opciones son costes asociados a un proyecto de aprendizaje automático?
- A. La complejidad estructural.
 - B. La dependencia de datos.
 - C. La complejidad del código.
 - D. Todos los anteriores.

Investigación en Inteligencia Artificial

Investigación en agentes inteligentes y sistemas expertos

Índice

Esquema	3
Ideas clave	4
8.1. Introducción y objetivos	4
8.2. Introducción a los agentes inteligentes	4
8.3. Comportamiento y entorno de los agentes inteligentes	7
8.4. Estructura de los agentes inteligentes	11
8.5. Proyectos de investigación en agentes inteligentes	18
8.6. ¿Qué son los sistemas expertos?	19
8.7. Proyectos de investigación en sistemas expertos	23
8.8. Referencias bibliográficas	24
A fondo	25
Test	26

Esquema



8.1. Introducción y objetivos

En este tema se explica a alto nivel el **concepto y la estructura general de los agentes inteligentes y los sistemas expertos**. Estos conceptos forman parte de los fundamentos básicos de la inteligencia artificial y, de una forma u otra, deben considerarse a la hora de diseñar una solución basada en esta disciplina.

De forma detallada, los objetivos que persigue esta unidad son:

- ▶ Ser capaz de definir las peculiaridades de un agente inteligente.
- ▶ Identificar las aplicaciones de los agentes inteligentes.
- ▶ Conocer los fundamentos de diseño y estructura de los agentes inteligentes.
- ▶ Conocer los conceptos clave de los sistemas expertos.
- ▶ Identificar qué proyectos son más adecuados para los sistemas expertos.

8.2. Introducción a los agentes inteligentes

En temas anteriores se ha discutido el significado de inteligencia y comportamiento racional. Todos estos contenidos previos deben tenerse en mente a la hora abordar el presente tema.

Un agente es un ente que recibe información del entorno y actúa en consecuencia. Se espera de él que actúe de forma inteligente.

Un ejemplo de actuación inteligente es la que está orientada a la **maximización de un objetivo** previamente establecido usando toda la información disponible en ese

momento. Por ejemplo, si el objetivo es la supervivencia, una acción inteligente sería escapar u ocultarnos de un depredador.

De forma abstracta, un agente **es una función** que convierte las percepciones en acciones.

$$f: P^* \rightarrow A$$

P^* : indica un conjunto de percepciones finito.

A: indica la acción que llevamos a cabo en función de dichas percepciones.

Los agentes son muy útiles para modelar entornos complejos donde las acciones que lleven a cabo unos agentes puedan afectar a movimientos posteriores de otros entes. Se recibe información de un entorno y las propias acciones que los agentes llevan a cabo insertan nueva información en dicho entorno.

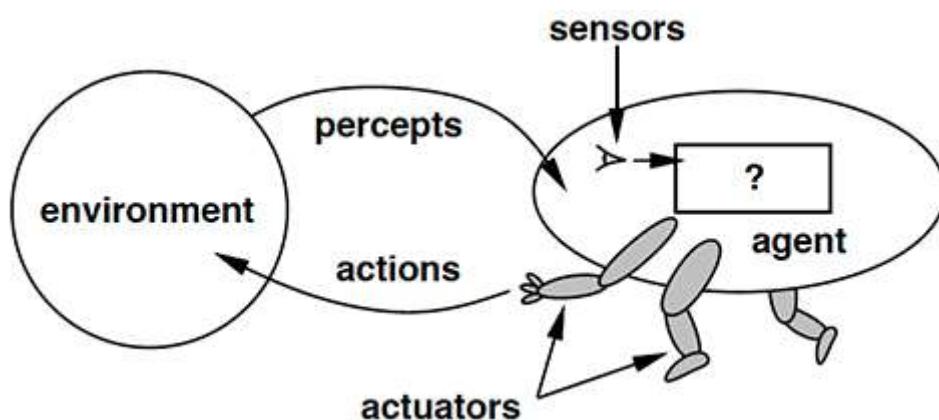


Figura 1. Un agente y su entorno.

Fuente: Russell, 2004.

Ilustraremos estos conceptos mediante un sencillo ejemplo práctico. Supongamos que hemos creado un robot aspirador. Hemos hecho un gran trabajo y, por tanto, nuestro robot puede considerarse un buen ejemplo de agente inteligente. ¿Cómo actuaría este robot aspirador? Resumiendo, la función asociada al agente (que estaría ejecutándose de forma continua mientras el robot estuviese encendido) podría ser como la que sigue (Russell, 2004):

```
function Reflex-Vacuum-Agent([location,status]) returns an action
if status = Dirty then return Suck
else if location = Right then return Right
else if location = Left then return Left
```

Explicado de forma sencilla, el robot dispone de un sensor que, dentro de un radio de acción, le indica que hay algún tipo de residuo o suciedad por recoger. Si así fuese, el mismo sensor le proporciona la localización concreta donde actuar. Si está a su derecha, el robot toma la decisión de moverse a la derecha. Una vez posicionado en el punto concreto, activa el aspirador. Tras ejecutar estas acciones, el entorno de partida ha cambiado, el residuo o resto de basura ha desaparecido y, por tanto, la información asociada a dicho entorno no es la misma que al inicio.

Como podemos esperar, sacar al mercado un robot aspirador implica crear un producto más completo del aquí reflejado. No obstante, la dinámica del procedimiento es común a todos los agentes inteligentes, todos tienen en cuenta la información que proporciona el entorno para, en función de una configuración previa, llevar a cabo una serie de acciones que permite maximizar el objetivo inicial planteado.

8.3. Comportamiento y entorno de los agentes inteligentes

Para diseñar un agente inteligente **debemos especificar de antemano el entorno en el que se va a ejecutar la tarea**. Esto implica definir cómo se va a evaluar si la tarea se está realizando correctamente, qué información se recoge del entorno, cuáles son las posibles acciones y qué tipo de sensores vamos a emplear. Todo ello constituye lo que se llama el **entorno de trabajo**, para cuya denominación se utiliza el acrónimo REAS (Rendimiento, Entorno, Actuadores/Acciones, Sensores).

Por ejemplo, supongamos que queremos diseñar ahora un coche autónomo. Podríamos trabajar con la siguiente descripción de elementos:

- ▶ Indicadores de rendimiento: número de kilómetros recorridos, número de infracciones de tráfico cometidas, número de accidentes, valoración media (o mediana) de la comodidad del viaje por parte de los ocupantes del vehículo, número de situaciones de riesgo detectadas, coste del sistema, etc.
- ▶ Entorno: calles de la ciudad, señales, otros vehículos, peatones...
- ▶ Acciones posibles: arrancar, parar, frenar, acelerar, girar, cambiar marcha...
- ▶ Sensores: velocímetros, frenos, GPS, cámaras de vídeo, etc.

A la hora de diseñar el agente se debe tener en especial consideración las **características del entorno**. Por ejemplo, no es lo mismo un entorno totalmente observable y una lógica de secuencias simple como puede ser la planteada para aspirar una habitación, que un entorno más estocástico e incluso con fenómenos no directamente observables como pueden ser los mercados financieros. En este último caso y en el ejemplo concreto del diseño de un agente inteligente para llevar a cabo operaciones bursátiles automatizadas, es probable que no se disponga del detalle de todas las acciones llevadas a cabo por todos los agentes. Podríamos llegar

a percibir las consecuencias de alguna acción concreta sin tener totalmente claro a quién asociar la acción raíz.

Propiedades de los entornos de trabajo

Los agentes inteligentes se emplean en entornos muy variados y diversos. Aun así, es posible identificar un número de dimensiones concretas en las que categorizar estos entornos. Estas dimensiones influyen de manera crítica en el diseño del agente. Las dimensiones que podemos considerar son las siguientes:

Totalmente observable vs. parcialmente observable

Diremos que el entorno de trabajo es totalmente observable si los sensores del agente le proporcionan información sobre todos los aspectos relevantes para la toma de decisiones. En otro caso, el entorno de trabajo es parcialmente observable. Un entorno puede ser parcialmente observable debido al ruido y a la existencia de sensores poco exactos o porque los sensores no reciben información de parte del sistema. Por ejemplo, en un sistema de inversión bursátil automatizado hay que tomar decisiones sin tener visión de qué están plateando o ejecutando otros actores en ese mismo momento.

Determinista vs. estocástico

Un entorno de trabajo es determinista cuando el siguiente estado del entorno está totalmente determinado por el estado actual y la acción ejecutada por el agente. En otro caso, hablamos de un entorno estocástico. Un entorno estocástico añade incertidumbre a la toma de decisiones. Por ejemplo, y en el caso de uso de un coche autónomo, el entorno es claramente estocástico, ya que en cualquier momento puede aparecer una avería, pinchazo o peatón inoportuno que altere el rumbo de los acontecimientos.

Secuencial vs. episódico

En un entorno secuencial las acciones realizadas en el corto plazo pueden afectar a largo plazo. Un ejemplo claro es el motor inteligente de un juego de ajedrez. Un mal movimiento puede dar al traste con una partida, aunque el jaque mate no llegue de manera inmediata. Por el contrario, en un entorno de trabajo episódico, las acciones a realizar en el siguiente episodio no dependen de las acciones que se realizaron en episodios previos. Por ejemplo, un robot ubicado en una cadena de montaje puede tener el objetivo de detectar piezas defectuosas y sacarlas del ciclo de fabricación. En este caso no se tiene en cuenta las piezas descartadas con anterioridad ni tampoco lo que se haga con esas piezas en el futuro. Los entornos episódicos son más simples puesto que no requieren el estudio de ciertas dependencias entre las acciones que se llevan a cabo.

Estático vs. dinámico

En un entorno de trabajo dinámico, el entorno puede cambiar mientras el agente está analizando la situación para decidir qué acción tomar. En otro caso, el entorno es estático. En el caso del coche autónomo, el entorno es dinámico puesto que todos los vehículos y peatones se están moviendo mientras el robot decide qué es lo próximo que debe hacer.

Discreto vs. continuo

Un entorno de trabajo discreto tiene un número finito de estados distintos, aunque este número sea muy elevado. Esta situación se da en juegos como el ajedrez, que poseen un elevadísimo número de combinaciones, pero aun así finito. Por el contrario, el recurrente caso del coche autónomo sería un ejemplo de entorno de trabajo continuo, puesto que la velocidad y ubicación de los coches y peatones tienen un rango de valores continuos a los que hay que unir parámetros varios como ángulos de giro, inclinación de la carretera, etc.

Agente individual vs. multiagente

Un entorno de trabajo donde un único agente trata de resolver un determinado problema sin interactuar con otros entes inteligentes sería un ejemplo de entorno de trabajo con agente individual. Por el contrario, entornos de trabajo como el ajedrez, donde dos agentes compiten entre sí o como el caso de uso del coche autónomo, donde varios agentes inteligentes tienen que convivir, serían ejemplos de entorno de trabajo multiagente.

Encontraremos el caso más complejo ante un entorno de trabajo que sea parcialmente observable, estocástico, secuencial, dinámico, continuo y multiagente. Realmente, este caso extremo suele ser uno de los más frecuentes.

La tabla siguiente muestra la clasificación de algunos ejemplos de uso según la tipología del entorno de trabajo (Russell, 2004). Todos los casos se refieren a un agente inteligente realizando la labor referenciada, ya puede ser jugar a un determinado juego, conducir un taxi (robotizado) o enseñar inglés.

Entorno de trabajo	Observable	Determinista/estocástico	Secuencial/episódico	Estático/dinámico	Discreto/continuo	Individual/multiagente
Crucigrama Ajedrez	Totalmente Totalmente	Determinista Determinista	Secuencial Secuencial	Estático Semi	Discreto Discreto	Individual Multi
Póker Backgammon	Parcialmente Totalmente	Determinista Estocástico	Secuencial Secuencial	Estático Estático	Discreto Discreto	Multi Multi
Taxi autónomo	Parcialmente	Estocástico	Secuencial	Dinámico	Continuo	Multi
Diagnóstico médico	Parcialmente	Estocástico	Secuencial	Dinámico	Continuo	Individual
Análisis de imagen	Totalmente	Determinista	Episódico	Semi	Continuo	Individual
Robot clasificador	Parcialmente	Estocástico	Episódico	Dinámico	Continuo	Individual
Tutor interactivo de inglés	Parcialmente	Estocástico	Secuencial	Dinámico	Discreto	Multi

Tabla 1. Ejemplos de entorno de trabajo y sus características. Fuente: Russell, 2004, p. 50.

8.4. Estructura de los agentes inteligentes

Un agente se compone de **arquitectura más software**. Por arquitectura entendemos el conjunto de sensores físicos, unidades de procesamiento y todo tipo de dispositivo que permite al agente comunicarse con el entorno y desencadenar acciones. El *software* recoge las instrucciones que guían el comportamiento del agente.

Podemos generalizar los distintos tipos de agentes inteligentes en cuatro grandes categorías de complejidad creciente. Esto significa que agentes categorizados en capas de mayor complejidad podrían ser capaces de realizar tipologías de tareas asociadas a agentes más simples, pero no al revés.

En primer lugar, encontramos a los **agentes reactivos simples**. Estos agentes reaccionan ante un estímulo o información de entorno externo y actúan en función de un conjunto de reglas o instrucciones previamente estipuladas. Toda la información previa o histórica es descartada. Por ejemplo, supongamos que los sensores de un robot aspirador informan de que justo en la ubicación actual hay unas migas de pan. El robot pasaría a succionar la suciedad sin plantearse nada más.

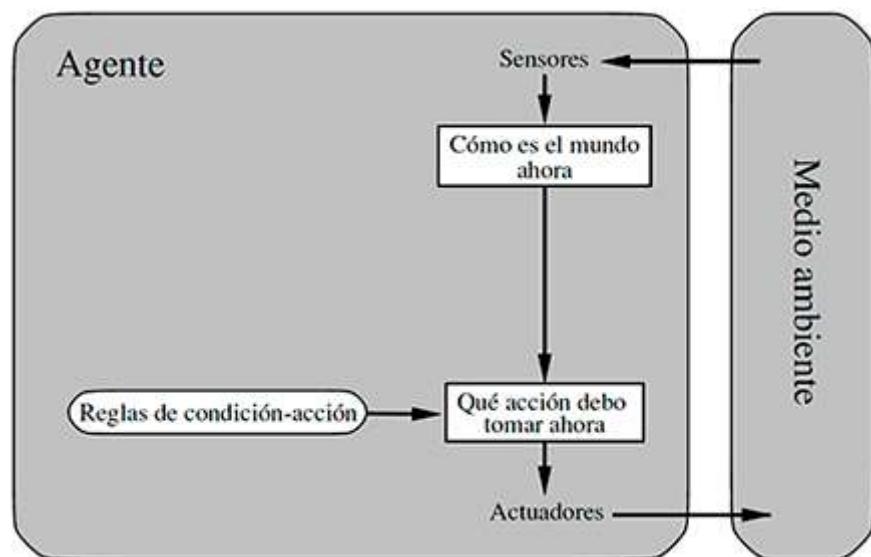


Figura2. Agente reactivo simple.

Fuente: Russel, 2004, p. 54.

Estos agentes se encuentran bastante limitados, puesto que solo se puede tomar la decisión correcta si el entorno es totalmente observable.

Una evolución más completa de estos agentes son los **agentes reactivos basados en modelos**. Estos agentes incorporan el histórico de información a la toma de decisiones. La historia previa se refleja en un determinado estado interno del agente, por lo que ya no solo se considera la información proveniente del entorno, sino que además debo tener en cuenta el estado en el que se encuentra el agente. Por ejemplo, imaginemos un videojuego de acción donde en un momento dado están disparando a nuestro personaje. El nivel de salud del personaje podría bajar o, por el contrario, podría mantenerse estático si nos encontramos en un estado «protegido» porque previamente hemos capturado algún tipo de objeto que ha creado un escudo protector. El estado del juego sería capaz de representar este amplio abanico de situaciones y entender cómo funciona su mundo particular. Y para entender su mundo necesita representar dicho conocimiento de manera formal, ya sea mediante un sistema de reglas, transacciones de estados, circuitos *booleanos* o ecuaciones matemáticas. Esa representación es lo que conocemos como modelo. Un agente que utilice este tipo de modelos es un agente basado en modelos.

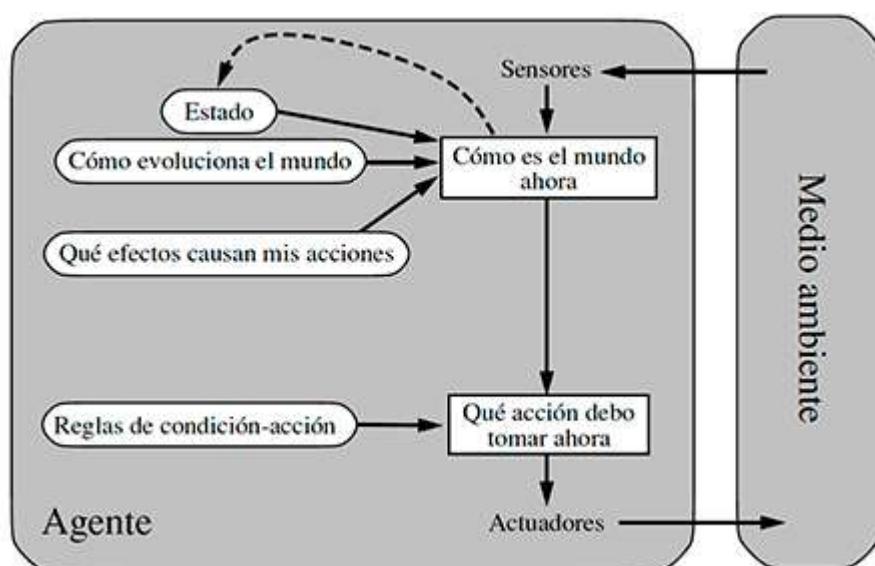


Figura 3. Agente reactivo basado en modelos.

Fuente: Russell, 2004, p. 56.

Los **agentes inteligentes como los anteriores hacen que sea posible determinar de antemano un conjunto de reglas de condición-acción**. Por ejemplo, si los sensores del vehículo autónomo detectan que el coche de delante frena, se debe iniciar la frenada. Si los sensores del robot aspirador detectan suciedad a la derecha, el robot debe desplazarse a la derecha.

En muchas ocasiones no se dispone del conocimiento necesario para poder construir este conjunto de reglas, por lo que se establece un objetivo a conseguir. Un objetivo o meta podría ser, por ejemplo, llegar a un destino concreto. Hablaremos en este caso de **agentes basados en objetivos**.

Agentes basados en objetivos y función de utilidad

La figura 4 muestra la arquitectura general de un agente basado en objetivos.

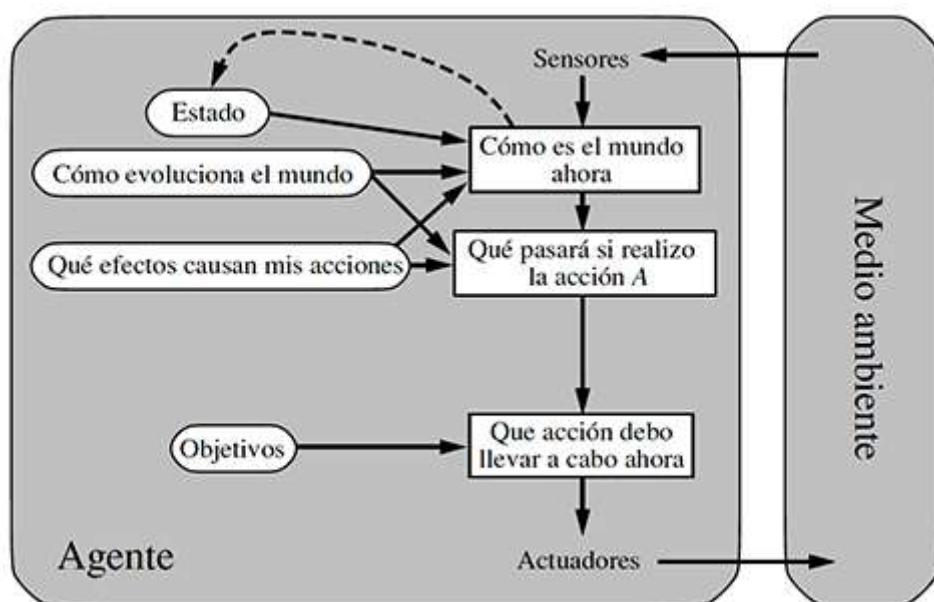


Figura 4. Agente basado en objetivos.

Fuente: Russell, 2004, p. 57.

Este tipo de agentes son más flexibles, puesto que el objetivo que les guía obliga a representar explícitamente el conocimiento que soporta su decisión. Pero también son más complejos (dependiendo del objetivo planteado). Por ejemplo, el objetivo

puede llegar con el coche a una determinada ubicación. Este tipo de tareas obliga a realizar una búsqueda de las distintas soluciones posibles y planificar un itinerario.

Los **agentes basados en objetivos** no permiten satisfacer nuestras necesidades en un gran número de ocasiones. Podemos pedir al navegador del GPS de nuestro vehículo que nos lleve a una ciudad concreta, pero nos sentiremos tremadamente defraudados y enfadados si nos damos cuenta de que nos está haciendo dar un rodeo innecesario o nos lleva por caminos poco transitables. Nos gustaría considerar no solo la meta a conseguir sino también en qué condiciones se debe alcanzar dicha meta para que consideremos plenamente satisfactoria la solución planteada por el algoritmo. Factores como llegar en el mínimo tiempo posible, recoger la mínima distancia posible, evitar peajes, etc., nos aporta valor sobre la utilidad de la solución.

Los **agentes basados en utilidad** incorporan una función de utilidad al modelo de tal forma que es posible medir la satisfacción de los usuarios con la solución. Esta función de utilidad suele contemplarse como una función matemática a maximizar o minimizar. Por ejemplo, y como se ha comentado antes, se puede minimizar la distancia recorrida, minimizar el gasto en combustible, minimizar el tiempo invertido. Debemos notar que, matemáticamente hablando, un problema de maximización es similar a un problema de minimización, puesto que el $\max\{f(x)\}$ es igual al $\min\{-f(x)\}$.

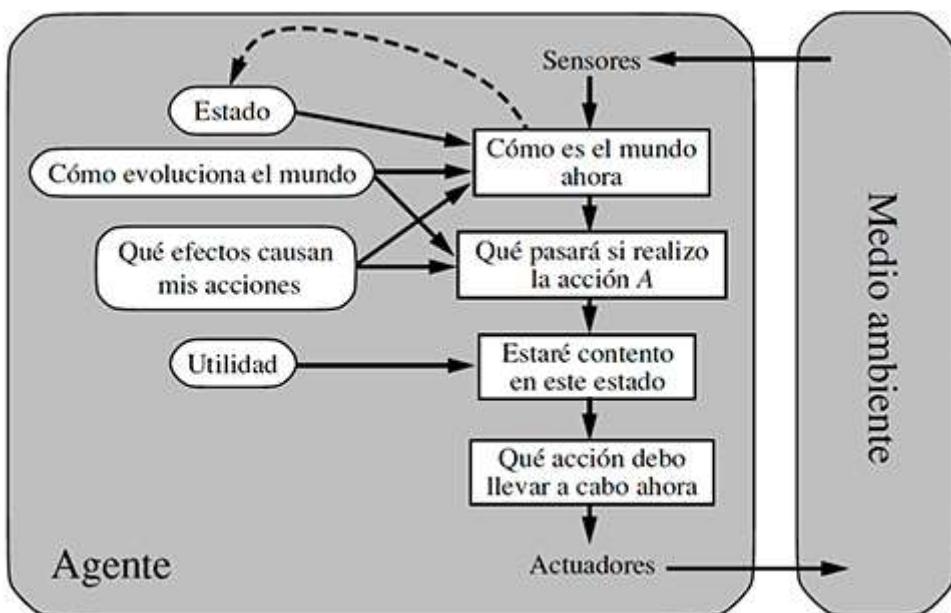


Figura 5. Agente basado en función de utilidad.

Fuente: Russell, 2004, p. 59.

Se considera que un agente que posea una función de utilidad explícita puede tomar decisiones racionales con la ayuda de un algoritmo de propósito general que no dependa de la función específica de utilidad a maximizar o minimizar.

Con un **algoritmo de propósito general**, el tiempo de ejecución depende solamente del volumen o tamaño de los datos de entrada. Por el contrario, en un **algoritmo de propósito específico** el tiempo de ejecución depende de las propiedades asociadas a los parámetros de entrada y no tiene por qué estar directamente relacionado con el tamaño de los datos de entrada.

Agentes que aprenden

Los agentes explicados hasta ahora reciben «su conocimiento» del entorno a través de sus programadores. Son estos quienes introducen las reglas correspondientes, los objetivos a conseguir o la función de utilidad. Un enfoque diferente es enseñar a las máquinas a aprender por sí mismas las características de la tarea a abordar. De esta forma, la propia máquina podría ser capaz de descubrir nuevos enfoques y

mecanismos eficientes para llevar a cabo la tarea asignada. Este tipo de agentes da pie a una nueva categoría, los **agentes que aprenden**.

Un agente que aprende se puede dividir en cuatro componentes conceptuales tal y como muestra la figura 6 (Russell 2004).

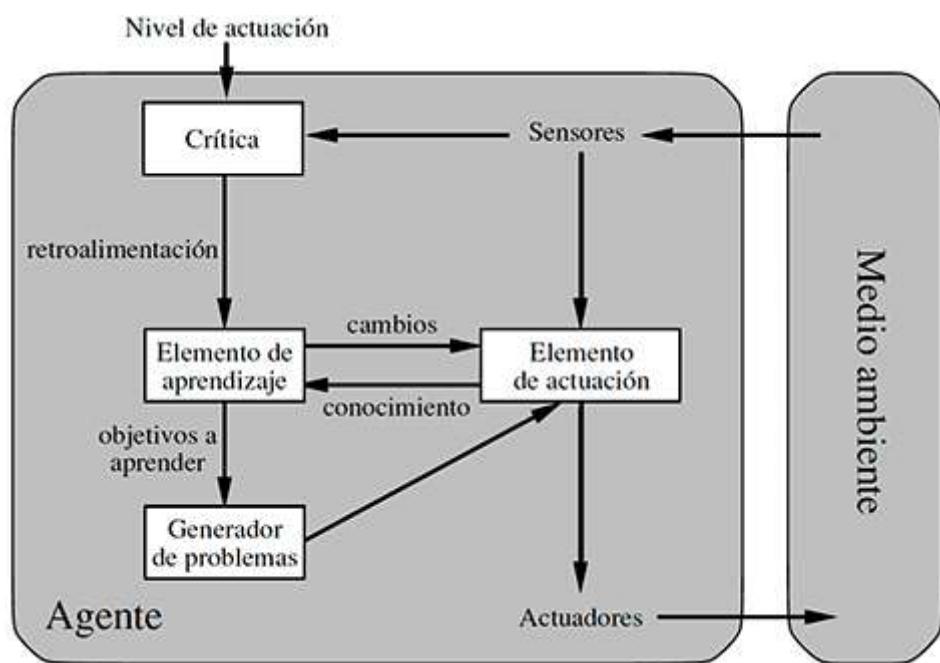


Figura 6. Estructura de un agente que aprende.

Fuente: Russell, 2004, p. 60.

La actuación del agente genera críticas, entendidas estas como evaluaciones sobre la actividad del agente inteligente. Los propios sensores del agente se convierten en receptores de las críticas. Debemos entender estas críticas como una retroalimentación muy valiosa para evolucionar y mejorar el funcionamiento del agente.

La retroalimentación recibida alimenta el elemento de aprendizaje, que es el motor encargado de desarrollar mejoras. Los cambios realizados son compartidos con el elemento de actuación. Este último componente es el responsable de seleccionar las acciones a ejecutar considerando los estímulos recibidos a través de los sensores.

Por último, el **generador de problemas** tiene la misión de sugerir acciones exploratorias (similares a un experimento) que, aunque puedan ser consideradas como no óptimas a corto plazo, son útiles para llegar a soluciones mejores que las actuales a largo plazo.

La arquitectura reflejada en la última imagen debe entenderse como una arquitectura de alto nivel donde solo se refleja la composición e interacción entre los elementos principales del esquema. De hecho, como **elemento de actuación** debe entenderse una representación simplificada de lo que anteriormente se había considerado como el agente completo ya que recibe estímulos y determina las acciones a realizar. Una posible implementación de este tipo de agentes es usando **aprendizaje por refuerzo**.

Para acabar de entender el último tipo de agente explicado podemos acudir al manual *Inteligencia Artificial: un enfoque moderno* (Russell, 2004). Los autores nos plantean el caso de un **taxi automatizado**, sin conductor. Podemos suponer que el usuario entra en el taxi e introduce la ubicación de destino. Aquí, el elemento de actuación agruparía todos los conocimientos y procedimientos que tiene el taxi para seleccionar las acciones a ejecutar en base a los requerimientos del conductor. La parte crítica, gracias a la observación del entorno, proporciona al elemento de aprendizaje la información necesaria para aprender y evolucionar su comportamiento. Un ejemplo puede ser un cambio de carril que ha provocado que un conductor proporcione un toque de atención con el claxon. El análisis de dicha situación puede servir para que el vehículo ajuste los parámetros de actuación ampliando el límite de separación con los vehículos circundantes antes de volver a realizar una operación de estas características. Este tipo de reacción no tiene por qué ser inmediata. Según el caso, se puede requerir un «volumen de críticas» previo determinado, antes de considerar un cambio de comportamiento.

El ciclo de las acciones está presente de forma constante en la filosofía de estos agentes. Mis acciones provocan reacciones en el mundo con el que interactúo y dichas reacciones provocan que ajuste mi comportamiento. Por supuesto, siempre teniendo en cuenta los principios básicos de diseño del agente, el objetivo a cubrir o función de utilidad.

Un ejemplo real de este esquema puede ser el motor de inteligencia artificial **AlphaGo** desarrollado por DeepMind (Google). En mayo de 2017, AlphaGo derrotaba a Ke Jie, campeón (humano) del mundo de *Go* por la mínima. Por primera vez en la historia, una máquina superaba a un campeón mundial en este milenario juego. AlphaGo sorprendió a su rival con movimientos imaginativos y poco usuales.

Las versiones iniciales de este tipo de juegos se basaban en el análisis histórico de millones de partidas previamente cargadas. AlphaGo basa su fortaleza en su capacidad de autoaprendizaje, siendo capaz de jugar millones de partidas contra sí mismo y aprender de todas las situaciones que se van produciendo durante el juego. De esta forma, no solo es capaz de descubrir por sí mismo diferentes trucos que los mejores profesionales usan de forma cotidiana en sus partidas, sino que es capaz de evolucionar las jugadas ideando acciones originales capaces de sorprender al rival.

8.5. Proyectos de investigación en agentes inteligentes

Los proyectos de investigación en agentes son un campo muy de moda y que tiene un enorme futuro por delante debido a que son aplicables a multitud de sistemas hardware y *software*.

Además, son fácilmente englobables en *topics* de investigación punteros como el *machine learning* ya que un agente puede estar implementado con modelos de sistemas expertos o aprendizaje automático. Y, de hecho, normalmente en los últimos años es lo que suele suceder. Sus principales campos de actuación más de moda son los siguientes:

- ▶ *Trading* automático en bolsa.
- ▶ Agentes de negociación automáticos (llegan a acuerdos sin transacciones, sin intervención de humanos).

- ▶ Internet de las cosas. Todo tipo de agentes que se conectan a internet y realizan operaciones de forma automática por ti.
- ▶ Drones autónomos.
- ▶ Vehículos autónomos.
- ▶ Robótica e industria 4.0.
- ▶ Videojuegos: comportamiento de NPC, jugadores de lucha entrenados por ti que juegan en red en tu nombre (*killer Instinct, Smash Bros. Ultimate...*).

En cuanto a publicaciones, las principales revistas sobre este tema son las siguientes:

- ▶ *IEEE Transactions on Automatic Control.*
- ▶ *Automática.*
- ▶ *IEEE Transactions on Systems, Man, and Cybernetics.*
- ▶ *Autonomous Agent and Multy Agents.*
- ▶ *System and Control lectures.*
- ▶ *International Journal of Control.*

8.6 ¿Qué son los sistemas expertos?

Un sistema experto es un **sistema que emplea conocimiento humano capturado por un analista especializado en la toma de requisitos de sistemas expertos**, que después se almacena en una computadora y que, mediante una serie de reglas y modelos, permite al sistema resolver problemas similares a los que puede resolver los expertos consultados. Algunos de estos sistemas imitan el proceso de razonamiento que los expertos utilizan para resolver problemas específicos, de esta forma estos razonamientos pueden ser validados por los propios expertos para comprobar que sean correctos. Normalmente **utilizan un sistema de razonamiento o bien inductivo o deductivo**. Aunque cada vez hay más sistemas que utilizan modelos basados en *machine learning* donde esta característica en ocasiones se ve

más comprometida, debido a que muchos de estos algoritmos son de caja negra (opacos, no se conoce exactamente cómo han llegado a la solución que generan).

Existen multitud de tipos de sistemas expertos pero los principales los podemos agrupar en tres tipos.

Basados en reglas

Se captura información de los expertos y se modeliza en forma de reglas. Estas reglas tienen una estructura:

SI X Entonces Y

Donde X es el antecedente de la regla e Y el consecuente.

El antecedente para cumplirse debe tener en la base de hechos (la memoria del sistema experto) una instancia de un objeto que sea compatible con esta. Ser compatible significa que cumpla con la descripción planteada en el antecedente.

Por ejemplo:

```
Si p: Paciente p.fiebre > 38 Entonces  
p.enfermo = true;
```

Esta regla se activará para cada uno de los elementos p de tipo paciente que tengan el atributo fiebre mayor que 38 y los marcará como enfermos.

De esta forma encadenando reglas se va generando nuevo conocimiento

```
Si p: paciente, p.enfermo && not p.ingresado && camaslibres > 0 entonces:  
camaslibres--, p.ingresado=true
```

A esta forma de razonar se le denomina **encadenamiento hacia adelante**. Y se basa en el proceso deductivo o en la aplicación de la regla *modus ponens*.

Este tipo de sistemas expertos suelen implementarse con sistemas de reglas de producción. Algunas herramientas para esto son Clips, Jess, Lisa o Prolog.

Basados en casos

Se le suele denominar *case base reasoning* y es un **sistema de razonamiento que se basa en la premisa de encontrar el caso más similar de una base de casos y ejecutar la acción que se realizó para ese caso en un momento anterior**. Estos sistemas han dado tradicionalmente muy buenos resultados debido a que son capaces de plantear soluciones a problemas en base a los criterios del experto. Pero tienen algunos problemas en ciertos entornos. En los métodos basados en razonamiento, el razonamiento realizado para llegar a la conclusión puede ser trazado y mostrado a un experto que valide el mismo. Esto da mucha confianza en los resultados que produce el sistema. Sin embargo, en los sistemas basados en casos, necesitamos encontrar un caso que sea lo más parecido posible. Esto tiene como problema principal que la función de similitud de casos a veces no funciona correctamente y no identifica bien el caso más similar, lo que puede conllevar elegir un caso que realmente no era tan parecido como pensábamos. Aquí razonar por qué un caso es más similar que otro de la base de casos no es tan claro como en el razonamiento deductivo, y puede haber problemas de trazabilidad de la toma de decisiones del sistema experto. Aun así, no llega a ser un algoritmo de caja negra ni mucho menos y es preferible cuando necesitamos una cierta explicabilidad a modelos más opacos como las redes de neuronas.

Se suele utilizar KNN (*k-nearest neighbors*) o modelos derivados de este para la selección del conjunto de casos candidatos de la base de datos a tratar. Donde K representa el número de casos que vamos a recuperar para su consideración.

Basados en redes bayesianas

Las redes bayesianas **representan un conjunto de variables aleatorias y sus dependencias condicionales a través de un grafo acíclico dirigido** (DAG por sus siglas en inglés). Por ejemplo, una red bayesiana puede representar las relaciones probabilísticas entre enfermedades y síntomas. Dados los síntomas, la red puede ser usada para computar la probabilidad de la presencia de varias enfermedades. Esto aporta algo que lo hace muy interesante y es el concepto de probabilidad. Al dar una probabilidad, el experto puede finalmente valorar entre diferentes opciones ordenadas según la más probable por el sistema experto y da también una idea del posible error que el sistema pueda tener. Cosa que es más complicada con los otros dos modelos.

Aunque hay versiones de sistemas expertos basados en reglas que utilizan lógica difusa. Este tipo de lógica sí que usa probabilidades y también puede aportar una idea de cuál es el error que puede tener el razonamiento obtenido.

Para ejecutar redes bayesianas existen diferentes sistemas *software*, por ejemplo, Winbugs, Jags u OpenBUGS.

8.7 Proyectos de investigación en sistemas expertos

Los proyectos de investigación en sistemas expertos se centran principalmente en las **siguientes ramas**:

- ▶ Medicina.
- ▶ Internet de las cosas.
- ▶ Cultivos.
- ▶ Ganadería.
- ▶ Economía.
- ▶ Visión por computador.
- ▶ Apoyo a las decisiones.

Estos proyectos se caracterizan por ser **entornos de aplicabilidad** donde el sistema normalmente apoya la decisión última de un humano y son sistemas que se basan principalmente en profesionales cualificados de los que extraen el conocimiento. Así que puede ser aplicado a cualquier entorno donde el conocimiento se pueda modelar a base de reglas, sistemas de inferencias o probabilidades. Actualmente los sistemas basados en **lógica difusa** son los que más activos están a nivel académico.

Las principales publicaciones de este medio son:

- ▶ *Expert Systems With Applications.*
- ▶ *International Journal of Information Management.*
- ▶ *IEE Access.*
- ▶ *Expert System.*
- ▶ *Decision Support System.*

8.8. Referencias bibliográficas

Russell, S. (2004). *Inteligencia artificial: un enfoque moderno*. México: Pearson Educación.

A fondo

Mastering the game of Go with deep neural networks and tree search

Silver, D. et al. (2016). Mastering the game of Go with deep neural networks and tree search. *Nature*, 529(7587), 484-489. doi: 10.1038/nature16961. Recuperado de <https://storage.googleapis.com/deepmind-media/alphago/AlphaGoNaturePaper.pdf>

Artículo científico publicado en la prestigiosa revista *Nature* donde se exponen los fundamentos de AlphaGo.

Expert Systems

Andy. (17 de agosto de 2018). What are expert system? *moral-robots.com*. Recuperado de <https://moral-robots.com/ai-society/what-are-expert-systems/>

Artículo interesante donde da una idea global de qué es un sistema experto.

Test

- 1.** ¿Cuáles de las siguientes frases son ciertas respecto a los agentes?

 - A. Solo pueden recibir información de forma directa de otros agentes.
 - B. Reciben información del entorno con el que se relacionan.
 - C. Sus acciones siempre afectan al entorno.
 - D. Su arquitectura se puede clasificar en diez tipologías distintas.

- 2.** ¿Qué significa el acrónimo REAS?

 - A. Rendimiento, Entorno, Avisador, Sensación.
 - B. Reutilización, Entorno, Acción, Sensor.
 - C. Reutilización, Entorno, Acciones, Sensores.
 - D. Rendimiento, Entorno, Actuadores, Sensores.

- 3.** ¿Cuáles de los siguientes pueden ser atributos de un entorno de trabajo?

 - A. Estocástico.
 - B. Parcialmente observable.
 - C. Episódico.
 - D. Todas las anteriores.

- 4.** ¿Cuáles de los siguientes pueden ser atributos de un entorno de trabajo?

 - A. Discreto.
 - B. Discrecional.
 - C. Multiepisódico.
 - D. Todas las anteriores.

- 5.** El entorno de trabajo en el que un agente inteligente debe desenvolverse para jugar al póker es un ejemplo de entorno de trabajo:
- A. Parcialmente observable, secuencial y continuo.
 - B. Totalmente observable, secuencial y continuo.
 - C. Parcialmente observable, estático y discreto.
 - D. Parcialmente observable, dinámico y continuo.
- 6.** Ejemplos válidos de distintos tipos de agentes inteligentes según su estructura son:
- A. Reactivo simple, basado en modelos, basado en alcance...
 - B. Reactivo simple, reactivo simple basado en modelos...
 - C. Basado en objetivos, basado en función dinámica.
 - D. Agente que aprende, agente que juega.
- 7.** Los agentes reactivos basados en modelos:
- A. Tienen en cuenta el histórico de información para la toma de decisiones.
 - B. Solo tienen en cuenta un conjunto de reglas de condición-acción.
 - C. Consideran una función de utilidad.
 - D. Un agente de estas características ha derrotado al campeón mundial de *Go*.
- 8.** Los sistemas expertos son útiles para...
- A. Obtener conocimiento más allá del aportado por los expertos.
 - B. Suplantar al experto en la toma de decisiones.
 - C. Modelizar el comportamiento de expertos humanos para apoyar sus decisiones.
 - D. Todas las anteriores.

9. El encadenamiento hacia delante:

- A. Es un método deductivo que se aplica en los sistemas expertos para obtener nuevo conocimiento a partir de una base de hechos y unas reglas de inferencia.
- B. Es el mecanismo por el cual funcionan los sistemas de producción.
- C. Permiten trazar el razonamiento del sistema experto.
- D. Todas las anteriores.

10. El razonamiento basado en casos:

- A. Es más fácil de trazar la respuesta del sistema que los algoritmos de encadenamiento hacia delante.
- B. Son un algoritmo de caja negra.
- C. Tienen peores resultados que los sistemas basados en redes bayesianas.
- D. Son más complejos de trazar que los sistemas de encadenamiento hacia delante.

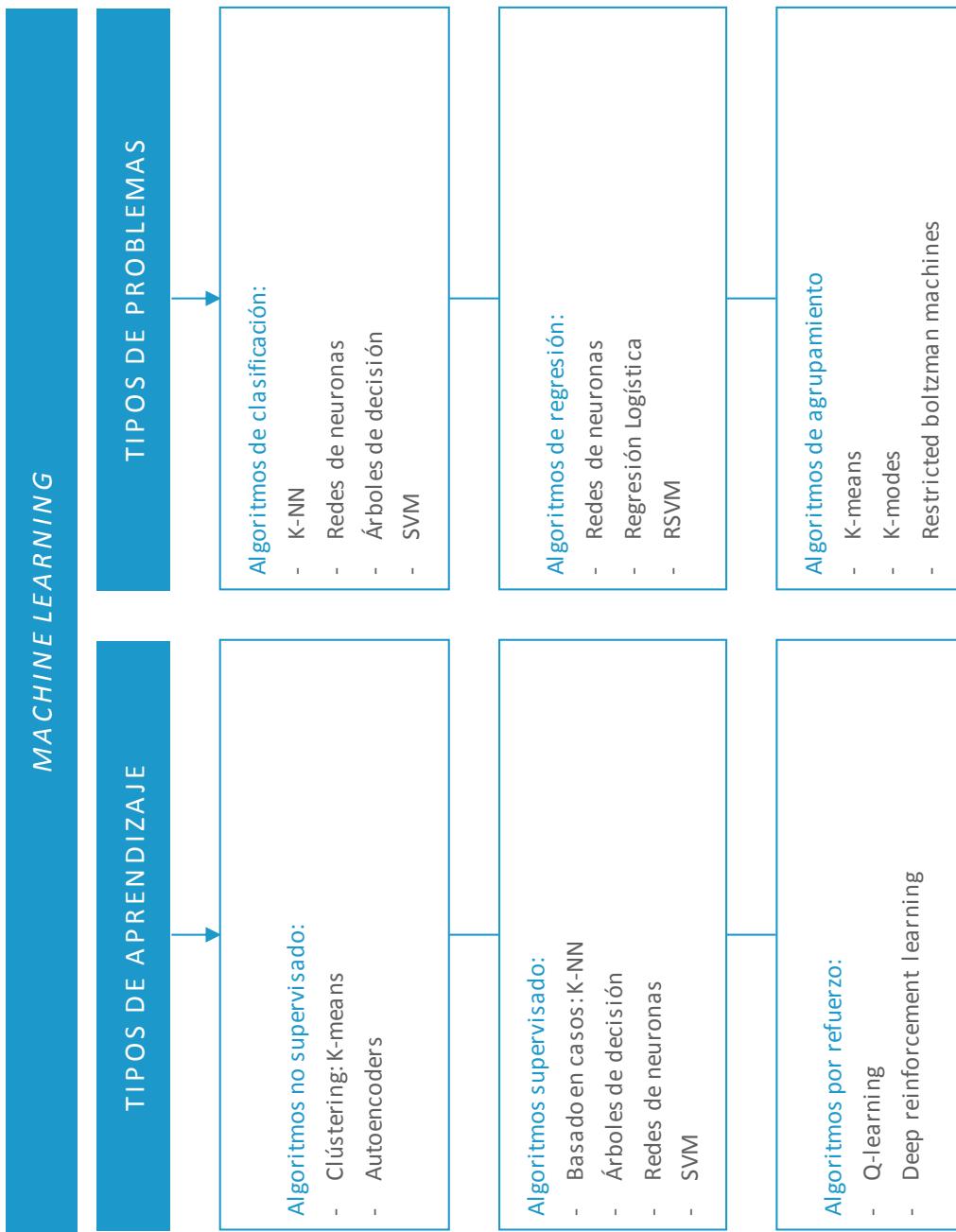
Investigación en Inteligencia Artificial

Investigación en aprendizaje automático

Índice

Esquema	3
Ideas clave	4
9.1. Introducción y objetivos	4
9.2. ¿Cómo aprenden las máquinas?	4
9.3. Tipología de un proyecto de aprendizaje automático	8
9.4. Técnicas de aprendizaje automático	14
9.5. Proyectos de investigación sobre aprendizaje automático	32
9.6. Referencias bibliográficas	33
A fondo	34
Test	36

Esquema



9.1. Introducción y objetivos

En este tema se explica a alto nivel el **concepto de aprendizaje automático o machine learning (ML)**. Estos conceptos son fundamentales hoy en día en la inteligencia artificial y es uno de los campos más activos en la misma.

De forma detallada, los objetivos que persigue esta unidad son:

- ▶ Ser capaz de entender cómo aprende una máquina.
- ▶ Conocer los tipos de algoritmos y problemas que existen y que pueden resolverse con *machine learning*.
- ▶ Identificar las principales técnicas y algoritmos de *machine learning* disponibles.
- ▶ Conocer los proyectos que se pueden crear usando *machine learning*.

9.2. ¿Cómo aprenden las máquinas?

Machine learning es uno de los temas más candentes de la inteligencia artificial actual. Pero como veremos durante este tema, es un inmenso campo de multitud de técnicas que darían cada una de ellas para una asignatura completa para poder tratarlas con profundidad. Lo que pretendemos en este tema es **dar las bases de cómo aprende una máquina y qué diversas implementaciones se han realizado en esta importante área de la inteligencia artificial**.

No está muy claro cuál es el mecanismo interno del aprendizaje del ser humano (o de otros animales). Por lo que **vamos a centrarnos en la funcionalidad**, más que en la estructura. Hay, por tanto, muchas definiciones de aprendizaje, pero vamos a resumirlo

en la **capacidad que tienen los seres vivos de mejorar sus aptitudes**. Es decir, la capacidad que tienen los seres vivos de realizar tareas que no sabían hacer y por otro lado de mejorar aquellas que ya sabían hacer. Para esto tiene que haber algún tipo de memoria donde se almacene este nuevo conocimiento, pero no es suficiente con una memoria. Se necesita una abstracción que permita enfrentarse a problemas nuevos que no han sido estudiados previamente.

Una de las principales cualidades que permite el aprendizaje es la **flexibilidad**. Un sistema que aprende se puede adaptar a entornos cambiantes. Frente a soluciones programadas que necesitan para cambiar la modificación del propio programa ya que son soluciones **específicas**. De esta forma el aprendizaje permite a un sistema adaptarse a los cambios en el entorno. Es por ello por lo que está especialmente recomendado en entornos dinámicos.

Pero ¿cómo aprende una máquina? El proceso de aprendizaje consiste **en seleccionar de entre un conjunto de hipótesis posibles para realizar una tarea o resolver un problema, aquellas que realizan la tarea o resuelven el problema de la mejor forma posible**. Así que hay ciertos conceptos que son claves. Por un lado, el sistema debe sacar conclusiones o crear un modelo que sirva para ser usado de forma general. Normalmente estas conclusiones se extraen de analizar casos particulares. Por lo cual, se necesita extraer cuál es el patrón o las características más relevantes en las que se basa el problema o la tarea. Esto no es más que una muestra de lo que se conoce como **abstracción**. Los humanos deben abstraerse de los pequeños detalles para saber resolver una tarea, ya que para resolver una tarea no se puede atender a todos los pequeños detalles a la vez. Pues básicamente eso es lo que en esencia hace un algoritmo de *machine learning*. Crea una descripción del problema de forma abstracta y resumida del mismo, que permite adaptar dicho modelo a múltiples problemas, que siendo diferentes entre ellos comparten unas características comunes que son las que lo identifican.

Por lo tanto, se necesita simplificar el problema y en este proceso de simplificación se introducen **sesgos**.

Los sesgos (o *bias* en inglés) son un **conjunto de datos establecidos *a priori* que ayuda a reducir la incertidumbre que caracteriza a la selección de la hipótesis correcta en el aprendizaje**. Por ejemplo, cuando usamos ejemplos para aprender, los algoritmos de *machine learning* introducen sesgos o generalizaciones que hacen perderse los detalles de los ejemplos introducidos. Pero esto es fundamental si queremos que el modelo pueda ser aplicado a otros ejemplos no usados en el entrenamiento. Porque, si no se introdujesen estos sesgos en el aprendizaje, entonces los algoritmos de aprendizaje automático podrían resolver muy bien los ejemplos que se han usado para el aprendizaje, pero no serían capaces de resolver otros ejemplos diferentes. Es decir, se habrían especializado, pero no tendrían capacidad de generalización.

Así que **la introducción de sesgos es esencial para que un algoritmo aprenda**. Y esto es algo que hacemos constantemente nosotros como humanos. La mayoría de las ecuaciones que se plantean en física tienen sesgos, ¿cuáles son esos sesgos? Por ejemplo, las ecuaciones de carga eléctrica asumen que los cuerpos están en el vacío. El movimiento uniformemente acelerado asume que lo haces en el vacío (sin fricción) y que la tierra no es curva (ni el espacio es curvo por la fuerza de la gravedad). La famosa ecuación de la relatividad de Einstein ($e=mc^2$), en realidad solo se aplica en los casos donde haya partículas con masa. Es decir, no se puede aplicar a la luz (que no tiene masa, pero sí energía. Por lo que la fórmula general es más compleja, pero esta nos sirve como buena aproximación para la mayoría de los casos). Las ecuaciones de física que estudias en el instituto y en la universidad tienen sesgos. Lo mismo que las reglas de ortografía que tienen excepciones. Como vemos no son reglas totalmente generales, pero se crean para ayudarnos en el proceso de abstracción.

Así que podemos decir que el mecanismo por el cual una máquina o un programa aprende es **gracias a que es capaz de construir un modelo simplificado de un problema en base a ejemplos parciales de ese problema**, intentando extraer las características generales que lo describen. A esto se le denomina sesgo y lo contrario, no introducir sesgos se le denomina **varianza (variance)** y es algo que normalmente no queremos. También se les denomina sobreajuste a los datos de entrenamiento. Este problema hace que el sistema pierda la capacidad de abstracción y, por tanto,

procesará mal los ejemplos no introducidos como entrenamiento de la red. Paradójicamente, el sesgo introduce un error en el modelo como se puede ver en la figura 1. Pero dicho error nos permite predecir el comportamiento de una nueva variable mucho mejor que el modelo que no proporciona dicho error. A esto tambien se le denomina en algunos entornos **sobre adaptación (overfitting)** del modelo a los datos de entrenamiento.

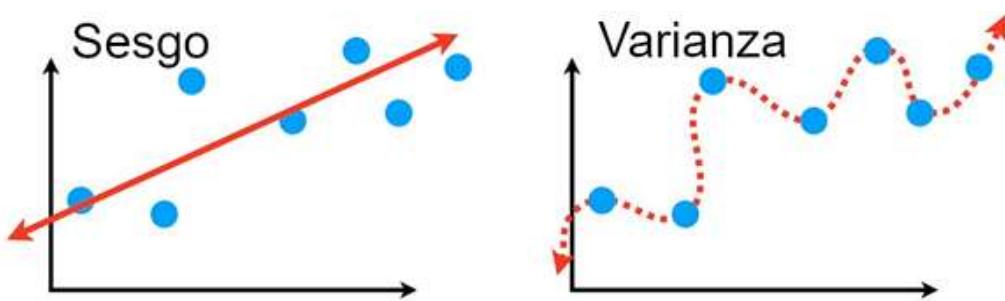


Figura 1. Definición gráfica de sesgo (*bias*) y varianza (*variance*).

Fuente: <https://skillsfuturetv.com/wp-content/uploads/2020/05/maxresdefault-360.jpg>

Ahora que tenemos claro que se necesita generalizar y, por tanto, introducir errores para conseguir aprender. Tambien debemos tener en cuenta que esos errores no deben ser inducidos por una mala selección de los casos de ejemplo. Seleccionar los casos de entrenamientos ayuda a que el sistema no aprenda sesgos innecesarios o que no son recomendables. Esto ya lo veremos con más detalles, pero por adelantar el concepto aquí vamos a ver un caso de ejemplo de un sesgo aprendido, podemos suponer que si no seleccionamos correctamente un patrón de rostros lo suficientemente diverso a nivel étnico, un algoritmo de *machine learning* puede aprender a reconocer solo caras de occidentales pero no de asiáticos o africanos, ya que utilizará probablemente los rasgos distintivos comunes de las caras occidentales como un patrón común que le maximiza la distinción entre lo que es una cara y lo que no.

Como podemos ver en este caso, el sesgo utilizado para aprender es un **sesgo perjudicial**. Así que hay que tener mucho cuidado cuando se presenten los ejemplos para no introducir sesgos artificiales o que induzcan a un comportamiento del modelo no deseado.

Una vez visto cómo aprenden las máquinas a nivel teórico, queda por describir cómo se realiza a nivel práctico. ¿Cómo llegamos a este aprendizaje? Hay diferentes caminos. Podemos clasificar las diferentes técnicas de aprendizaje en distintos tipos en base a si el aprendizaje es guiado por el diseñador del problema o no.

9.3. Tipología de un proyecto de aprendizaje automático

Podemos clasificar los algoritmos de aprendizaje en tres grandes tipos en función de la intervención que tenga el diseñador del algoritmo en el aprendizaje.

Aprendizaje supervisado

Este tipo de algoritmo **requiere que los ejemplos que se introduzcan para aprender tengan tambien la solución al problema**. De esta forma, el aprendizaje supervisado intenta deducir una función que permita obtener a partir de los datos de entrada la solución deseada. Ambos valores se proporcionan al algoritmo de forma que este puede analizar cómo de lejos está su modelo actual de conseguir representar correctamente las salidas deseadas. Es decir, puede usar el error de entrenamiento para tomar decisiones en el aprendizaje. En algunos de estos modelos este error de entrenamiento es clave para mejorar el algoritmo como en el caso de las redes de neuronas.

Como ya hemos comentado en la introducción, **no hay aprendizaje si no hay capacidad de generalización**. Es decir, si no somos capaces de aplicar el modelo a ejemplos que no hemos visto antes y obtener un resultado correcto o aproximado. Por lo tanto, necesitamos que nuestra función prediga el valor de un objeto una vez entrenado, sin conocer cuál es el resultado correcto del mismo. Es decir, **necesitamos que haga una**

generalización. Ejemplos de algoritmos de aprendizaje supervisado son las redes de neuronas, *support vector machines*, árboles de decisión, etc.

Para entrenar redes de aprendizaje supervisado, el diseñador debe seleccionar un conjunto de ejemplos de entrenamiento y un conjunto de ejemplos de validación. Esto es así para poder comprobar de forma empírica cuál es la capacidad de generalización de la red.

Así pues, se deberá crear una base de ejemplos con los atributos que describen dicho ejemplo y la clase a la que pertenece. De ellos, seleccionaremos un subconjunto de entrenamiento y otro de test. Normalmente se suele usar un 75 % de entrenamiento y un 25 % de validación o test. Pero no es una regla escrita y puede ser diferente en función del problema a tratar.

Una técnica de pruebas muy utilizada es el *cross validation* o la **validación cruzada**. Sirve para evaluar los resultados de un modelo y garantizar que son independientes de la partición entre datos de entrenamiento y prueba. Este método consiste en **calcular la media aritmética obtenida de las medidas de evaluación sobre diferentes particiones**. Se utiliza en entornos donde el objetivo principal es la predicción y se quiere estimar la precisión de un modelo que se llevará a cabo a la práctica. Por ejemplo, se puede dividir los datos en bloques en cuatro y coger tres de ellos para entrenar y uno para validar. Repitiendo el problema cuatro veces cambiando el bloque de validación en cada vez y calculando la media aritmética de las cuatro validaciones, obtendremos una mejor aproximación al error de aprendizaje y minimizaremos la introducción involuntaria de posibles sesgos a la hora de realizar la partición.

El aprendizaje no supervisado

El aprendizaje no supervisado **es aquel que se lleva a cabo sobre ejemplos que no informan del resultado esperado del mismo**. Por lo tanto, podemos decir que no hay nadie que le está indicando al sistema cómo se resuelve el ejemplo propuesto. Esto debe descubrirlo el propio algoritmo. Por lo tanto, los algoritmos de aprendizaje no

supervisado tratan las entradas como variables aleatorias y crean modelos de densidad para los diferentes conjuntos de datos. Ejemplos de algoritmos de aprendizaje no supervisado son los mapas autoorganizados, el análisis multivariante, el clústering, los estimadores de máxima verosimilitud o las redes de neuronas de base radial. Algunos modelos de *deep learning* como los modelos de extracción de características también son no supervisados. Aunque las redes de neuronas son algoritmos supervisados, existen algunos trucos que detallaremos más adelante para conseguirlo.

Para realizar un entrenamiento de un algoritmo de aprendizaje no supervisado, no debemos construir pares atributos-valor esperado, ya que no sabemos el valor esperado (o no lo necesita saber el algoritmo en cuestión). Esto hace más complejo e impreciso calcular el error de estos modelos. Pero debemos medir si nuestro algoritmo está funcionando bien de todas formas, ¿cómo lo hacemos? Debemos introducir medidas artificiales. Por ejemplo, en un algoritmo de agrupamiento podemos estimar la varianza total del sistema. Pero en estos casos, visualizar las agrupaciones realizadas ayuda mucho a comprender si el agrupamiento se ha realizado correctamente.

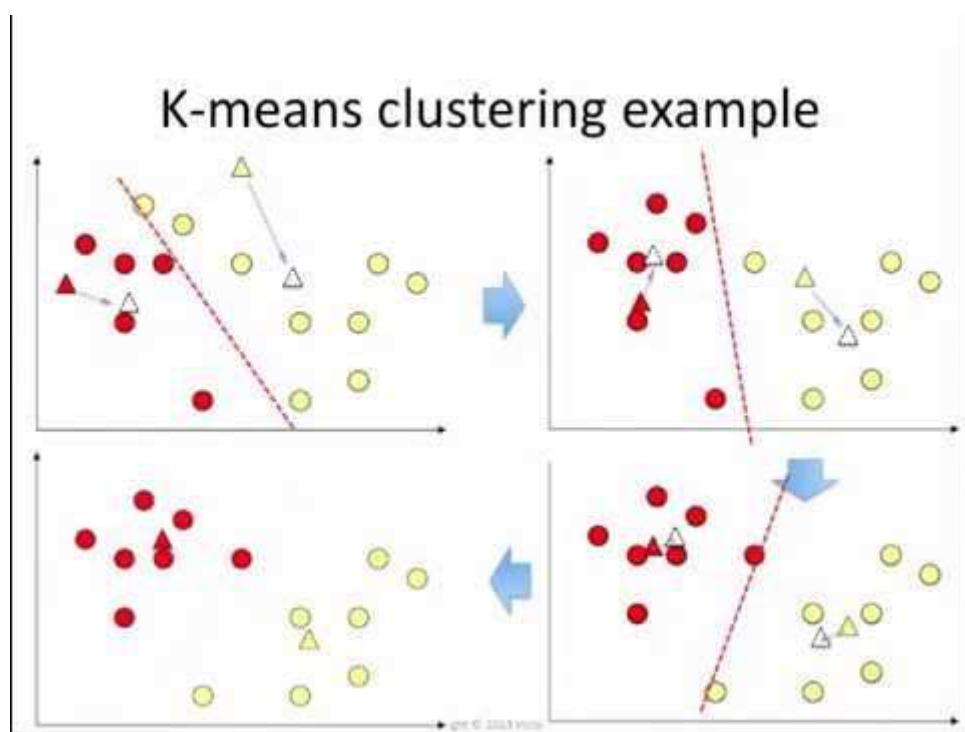


Figura 2 Ejemplo de funcionamiento de k-means, un algoritmo no supervisado y cómo visualizar si el agrupamiento realizado es correcto. Fuente: <http://www.cs.us.es/~fsancho/?e=77>

En el caso de que tengamos las soluciones de las clases de algunos ejemplos, podemos comprobar si ha clasificado correctamente los ejemplos *a posteriori*. Aunque como decimos, esta información solo nos servirá para estimar el error cometido, pero el algoritmo no toma ventaja de esta información para mejorar los resultados.

Aprendizaje por refuerzo

El algoritmo aprende observando el mundo que le rodea. Su información de entrada es el *feedback* o retroalimentación que obtiene del mundo exterior como respuesta a sus acciones. Por lo tanto, el **sistema aprende a base de ensayo-error**. Aunque no de forma aleatoria. Se podría considerar como un aprendizaje no supervisado por nadie, pero hay un pequeño matiz y es que los algoritmos no supervisados ni siquiera tienen el *feedback* de si lo están haciendo bien o mal. Aquí el entorno les está dando una retroalimentación de si lo que hacen está siendo efectivo o no. Por lo tanto, no podemos clasificarlos ni como aprendizaje supervisado ni como no supervisado.

El aprendizaje por refuerzo es el más general entre las tres categorías ya que en vez de que un instructor indique al agente qué hacer, **el agente inteligente debe aprender cómo se comporta el entorno mediante recompensas (refuerzos) o castigos de forma autónoma**, derivados del éxito o del fracaso de sus decisiones. El objetivo principal es aprender la función de valor que ayude al agente inteligente que lo utiliza a maximizar la señal de recompensa recibida y así optimizar sus políticas para comprender el comportamiento del entorno y de esta forma ser capaz de tomar buenas decisiones para conseguir sus objetivos. Por lo tanto, la versatilidad del sistema es máxima y un mismo sistema podría aprender a realizar cosas completamente diferentes en función del entorno donde se mueva. Ejemplos de este tipo de técnicas son Q-learning, deep reinforcement learning, simulaciones de Montecarlo, etc.

¿Y cómo se puede evaluar un algoritmo de aprendizaje por refuerzo? Pues hay que ejecutarlo en un escenario diferente al aprendido y tener una medida de calidad de este para saber si el algoritmo funciona y ha conseguido generalizar un modelo. En parte la propia función le dice al algoritmo si está haciendo las cosas bien o no sirve

como medida de la calidad de este. Pero hay que probar el sistema en otros entornos para ver si ha sido capaz de aprender una política válida para cualquier entorno similar. Y ahí necesitaremos una medida de rendimiento. Imaginemos que tenemos un sistema de aprendizaje por refuerzo que ponemos a entrenar para que aprenda a jugar a un juego. El sistema debe aprender a jugar al juego a base de prueba y error y en cada decisión que tome de forma acertada, el sistema debe premiar esa política. Pero luego lo tenemos que poner a jugar al juego contra otros jugadores diferentes y tendremos que evaluar si consigue ganar a esos otros jugadores y cómo de rápido o de eficiente ha ganado (cuántas piezas ha perdido, cuántos puntos ha conseguido, en cuánto tiempo ha ganado...). De esta forma validaremos el sistema.

Tipos de problemas

Vistos los diferentes tipos de algoritmos de aprendizaje automático, nos queda por describir los diferentes tipos de problemas que estos pueden abordar. Principalmente **podemos agrupar los problemas en tres tipos:** problemas de clasificación, de regresión y de agrupación.

Es importante detectar cuándo un problema es de clasificación, de regresión o de agrupamiento, porque las técnicas empleadas y los métodos para obtener los datos no son iguales. Tampoco es igual la forma con la que evaluamos los errores que cometen los algoritmos en ambos modelos.

Problema de clasificación

La clasificación es el problema de identificar a cuál, de un conjunto de categorías (subpoblaciones), pertenece una nueva observación, sobre la base de un conjunto de datos de entrenamiento que contienen observaciones (o instancias) cuya categoría es conocida.

Para que haya un problema de clasificación se deben conocer las clases *a priori* y además estas deben ser un conjunto finito.

Algunos algoritmos que resuelven problemas de clasificación son: árboles de decisión, support vector machine, redes de neuronas (discretizando la salida de la red), K-NN, redes bayesianas, random forest, etc.

Problema de regresión

Los problemas de regresión **son problemas en los que la salida que se espera es una salida continua**. Un número de casos infinito o al menos indefinido y, por tanto, lo que queremos normalmente es usar una estimación de esta salida lo más precisa posible. Los algoritmos que intentan realizar regresión intentan estimar la función de mapeo (f) de las variables de entrada (x) a las variables de salida numéricas o continuas (y). Ahora, la variable de salida podría ser un valor real, que puede ser un valor entero o de coma flotante.

Algunos algoritmos que resuelven problemas de regresión son: regresión logística, redes de neuronas, *regression support vector machines*, etc.

Problema de agrupamiento

Un problema es de agrupamiento **si no se conoce *a priori* el número de clases ni el tipo de estas y lo que se pretende es precisamente conocer qué clases existen**. Los algoritmos que se aplican al agrupamiento pretenden encontrar cuáles son los grupos que existen en una población y cuáles son los criterios o las características que separan unas clases de otras.

Algunos algoritmos que resuelven problemas de agrupación son: k-means, k-modes, mean-shift, binary split, deep belief networks, restricted boltzmann machines, etc.

9.4. Técnicas de aprendizaje automático

A continuación, vamos a citar las principales técnicas y algoritmos de aprendizaje automático de forma resumida, identificado en qué se basan, cuáles son sus ventajas y cuál son sus inconvenientes.

K-NN (K-nearest neighbor o k vecinos más cercanos)

Este algoritmo de aprendizaje supervisado es tambien conocido como **aprendizaje perezoso**. El algoritmo no intenta extraer una característica de los datos haciendo un proceso de entrenamiento exhaustivo con ellos. **Simplemente los almacena para recuperarlos en el futuro**. Cuando se presenta un nuevo caso, este es comparado con la base de casos y se extrae aquel más similar (o los K más similares) para comprobar cuál era la clase (o la acción) que se llevó a cabo para el caso concreto. Asumiendo que, si el caso es similar, la acción tomada anteriormente será una acción que también será válida para el nuevo caso.

Para ello hay que definir una función de similitud entre dos ejemplos. La calidad de esta media de similitud es la que más ayudará a que el sistema se comporte correctamente. Existen multitud de medidas de similitud, por ejemplo, la distancia euclídea, la distancia de Manhattan, la distancia de edición, la distancia de Minkowski, la distancia de Mahalanobis, etc. Cada una de ellas funciona mejor en unos dominios que en otros. Tambien se pueden ponderar los pesos de los atributos para ajustar más la similitud, dando más importancia a ciertos rasgos que a otros.

Ventajas:

- ▶ No paramétrico (salvo que usemos distancias ponderadas). No hace suposiciones explícitas sobre la forma funcional de los datos, evitando los peligros de la distribución subyacente de los datos.

- ▶ Algoritmo simple tanto de explicar como de interpretar.
- ▶ Alta precisión (relativa). Es bastante alta, aunque no superior a otros modelos más sofisticados.
- ▶ El proceso de entrenamiento es inmediato.

Las desventajas de este algoritmo son:

- ▶ Es muy sensible a los atributos irrelevantes. Hacer una buena selección de atributos relevantes es fundamental para este algoritmo.
- ▶ Es sensible al ruido, ya que, si un ejemplo es un mal ejemplo de entrenamiento y es el seleccionado como el más similar, daremos una solución errónea. Esto se puede mitigar haciendo que K sea grande, es decir que el número de elementos cercanos sea mayor para que el peso del ruido sea menor.
- ▶ La ejecución de este es lenta si hay muchos datos de entrenamiento, ya que tiene que procesar todos los datos. Existen métodos para optimizarlo usando partición espacial pero aún así es más costoso que otros algoritmos.
- ▶ Es caro en memoria en tiempo de ejecución. Si la base de casos es muy grande ocupa mucha memoria.

Árboles de decisión

Es un modelo de predicción supervisado utilizado en diversos ámbitos que van desde la inteligencia artificial hasta la economía. Estos árboles **son muy útiles para visualizar las diversas opciones de las que se dispone para resolver un problema o modelar un comportamiento y cuál es la secuencia de pasos necesaria para llegar a dicha decisión**, ya que se pueden convertir en reglas, que son mucho más legibles para el que investiga el funcionamiento del modelo. Además, se les pueden aplicar métodos de inducción, como por ejemplo la inducción hacia atrás, gracias a los cuales, mediante sencillos razonamientos, se puede descubrir la lógica que el sistema ha tomado para llegar a seleccionar una acción.

Un árbol de decisión está **compuesto de dos tipos de nodos**: un nodo condicional y una clase. Los nodos condicionales son los nodos internos del árbol y la clase, los nodos hoja o terminales.

Cada **nodo condicional** es una pregunta que se hace a las variables del entorno y solo tiene dos caminos, que la condición sea cierta o falsa. En función de si es una u otra, se tomará un camino y se llegará a un nodo hoja u otro. Cuando se llega a un nodo hoja, este determina la clase a la que pertenece el ejemplo suministrado.

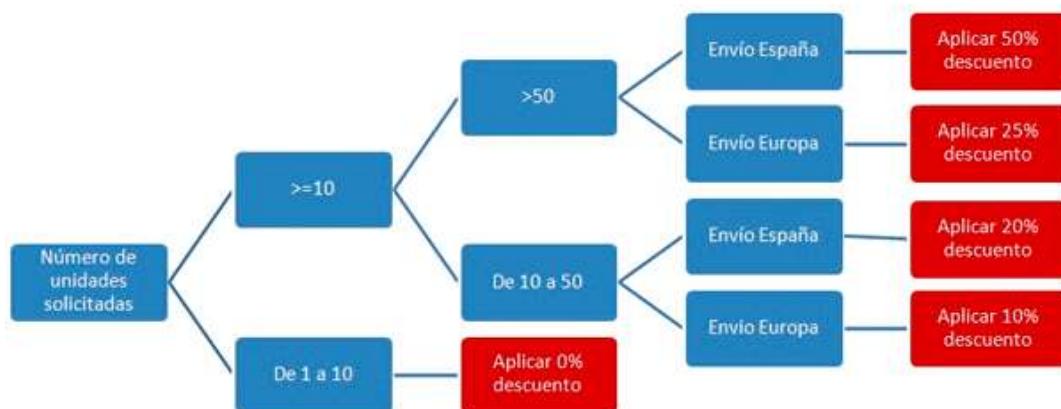


Figura 3. Ejemplo de árbol de decisión.

Fuente: https://es.wikipedia.org/wiki/%C3%81rbol_de_decision#/media/Archivo:Arbol_decision.jpg

Algunas de las implementaciones más famosas de estos algoritmos son el ID3, J48 o C4.5.

Este algoritmo **utiliza la entropía** (medida de incertidumbre o de desorden) para ayudar a decidir qué atributo debe ser el siguiente en ser evaluado en el árbol. También nos puede ayudar a descubrir qué atributos son los más relevantes, ya que el algoritmo los colocará más cerca de la raíz. En concreto se pretende maximizar la ganancia de información, donde la entropía se utiliza como una medida del orden de los datos. Es decir, el atributo seleccionado es aquel que deja la información más

ordenada o, dicho de otra forma, mejor clasificada. La entropía se calcula con la ecuación.

$$Entropía(s) = \sum_{n=1}^c -p_i \log_2 p_i$$

Y la ganancia con la ecuación:

$$Ganancia(S, A) = Entropía(S) - \sum_{v \in Valores(A)} \frac{|A|}{s} Entropía(Sv)$$

Donde $Valores(A)$ es el conjunto de todos los valores posibles para el atributo A , y Sv , es el subconjunto de S para el cual el atributo A tiene el valor v .

En general, los atributos que separan mejor las clases tienden a reducir más la entropía y, por tal motivo, debe ser seleccionado primero.

Por lo tanto, este algoritmo divide el espacio de soluciones mediante hiperplanos, fijando una de las variables a un valor frontera.

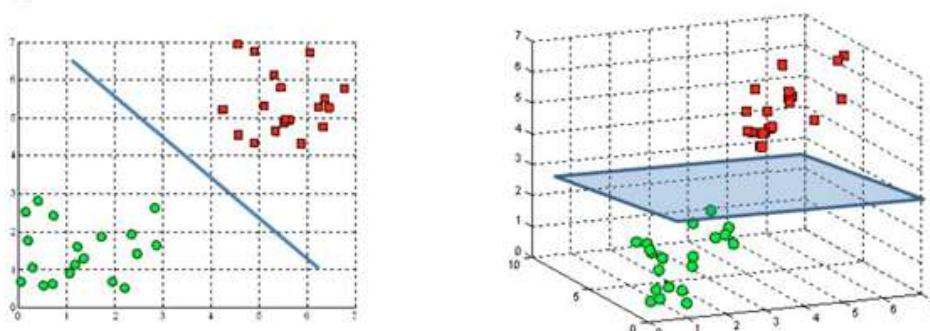


Figura 4. Ejemplo de un hiperplano en 2 y 3 dimensiones que separa dos conjuntos de clases.

Ventajas:

- ▶ El entrenamiento es muy rápido.
- ▶ Es fácil de interpretar los resultados por un humano, es un algoritmo de caja blanca.
- ▶ Para algunos problemas consigue una buena precisión.
- ▶ Se pueden convertir fácilmente reglas.
- ▶ No requiere una preparación de los datos demasiado exigente.
- ▶ Puede trabajar con variables cualitativas y cuantitativas.

Desventajas:

- ▶ Es muy dependiente al ruido de la entrada.
- ▶ Los árboles de decisión tienden al sobreentrenamiento.
- ▶ No se puede garantizar que el árbol generado sea el óptimo.
- ▶ Hay conceptos que no son fácilmente aprendibles por los árboles de decisión, ya que las particiones del espacio de soluciones que puede hacer son aquellas que son representables mediante una sucesión de hiperplanos. Si no hay una aproximación lineal al problema, puede que den un modelo poco efectivo.
- ▶ Se recomienda balancear el conjunto de datos antes de entrenar.

Existen versiones mas modernas de árboles de decisión como es el **random forest**. Este ejecuta diferentes árboles de decisión y se realiza un proceso de votación para elegir cuál de los árboles ha generado una mejor predicción. Por ejemplo, nos podemos quedar con la predicción mayoritaria, con la media, etc. Estos métodos obtienen mejores resultados en general que los árboles de decisión convencionales, aunque dificultan la comprensión del modelo generado.

Redes de neuronas

Las redes de neuronas son una analogía del funcionamiento del cerebro humano, que **permiten a las computadoras adoptar ciertas capacidades que se pueden definir como inteligentes**, y que resuelven un importante subconjunto de problemas que otras técnicas computacionales o matemáticas no resuelven o lo hacen de forma más ineficiente, tanto en tiempo como en los resultados obtenidos. Debido a su semejanza con el cerebro, posee muchas capacidades similares a la mente humana. Por ejemplo, son capaces de aprender de la experiencia, de generalizar de casos anteriores a nuevos casos, de abstraer características esenciales a partir de entradas que representan información irrelevante, etc. Esto hace que ofrezcan numerosas ventajas y que este tipo de tecnología se esté aplicando en múltiples áreas, como aprendizaje adaptativo, autoorganización, tolerancia a fallos, etc.

Las redes de neuronas **se basan en los trabajos realizados por Warren McCulloch y Walter Pittis** en 1943, que han servido de base para la aparición de multitud de arquitecturas obtenidas durante los últimos setenta años. Las redes de neuronas se caracterizan por contar con un número de neuronas organizadas de diversas formas, que están conectadas unas a otras a través de enlaces ponderados por unos pesos, que suelen ser recalculables por la propia red, de forma más o menos autónoma, para proporcionar una de las capacidades más importantes de la mayoría de las redes de neuronas, que es su capacidad de aprendizaje. Al igual que el cerebro humano, las neuronas y las redes de neuronas artificiales sustentan su capacidad de cálculo no en el núcleo de las neuronas sino principalmente en las **interconexiones que se realizan entre ellas**.

Normalmente se suelen diferenciar **tres tipos de neuronas en una red**, dependiendo de su función:

- ▶ Neuronas de entrada. Son las encargadas de recibir los datos del exterior de la red e iniciar la propagación de dichos datos por toda la estructura. Las conexiones de entrada de estas neuronas (sus dendritas) no realizan ningún cálculo, solo el hecho de captar la información para la red.
- ▶ Neuronas ocultas. Son aquellas neuronas que permiten enlazar las neuronas de entrada y las neuronas de salida, aunque también pueden estar conectadas a otras neuronas ocultas; y en sus interconexiones es donde precisamente se realiza el proceso de cálculo, tanto al enviar la información a todas las neuronas con las que está interconectada, como al recibir la información.
- ▶ Neuronas de salida. Son las que muestran al exterior los resultados obtenidos después de que la entrada haya recorrido toda la red. Las conexiones de salida de las neuronas de salida y no realizan ningún cálculo.

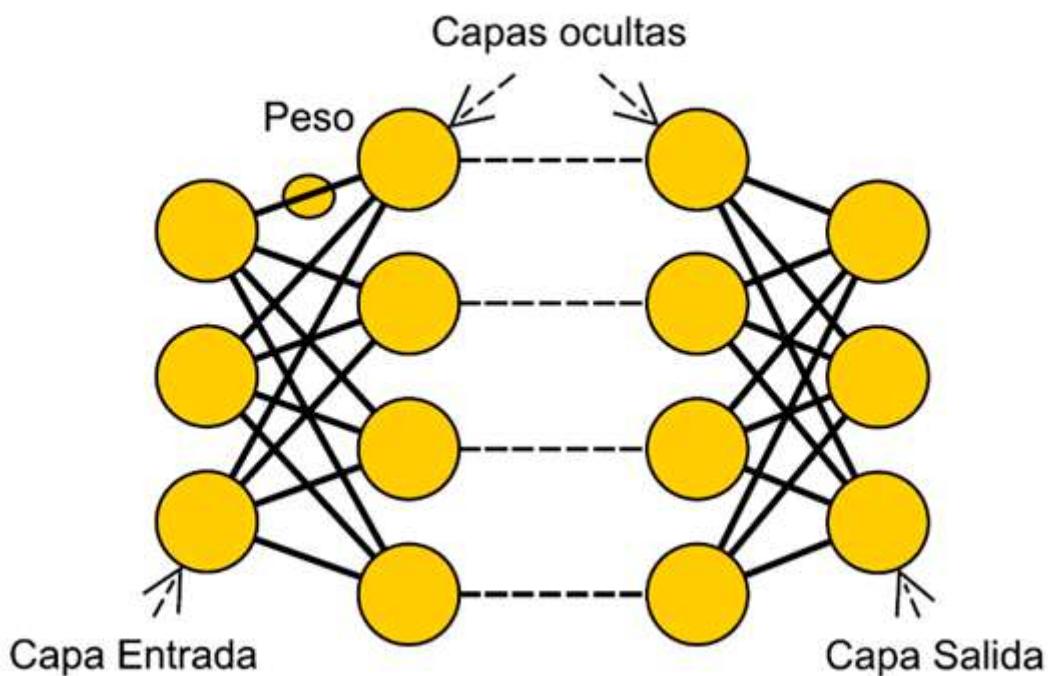


Figura 5. Esquema de una red de neuronas.

Existen **multitud de tipos de redes** de neuronas las principales son las siguientes:

- ▶ Perceptrón multicapa. Es el modelo más simple que consta de una capa de entrada, n capas ocultas y una capa de salida. Es una aproximador universal.
- ▶ Redes de neuronas recurrentes: algunas de las neuronas de salida son tambien neuronas de entrada del siguiente ejemplo a procesar. Este tipo de neuronas viene muy bien para predecir sistemas que tengan alguna relación temporal entre los diferentes ejemplos. Por ejemplo, para procesar series temporales o datos secuenciales.
- ▶ Las redes de base radial: calculan la salida de la función en función de la distancia a un punto denominado centro. Al igual que con los perceptrones multicapa, sirven como aproximadores universales y solo tienen una capa oculta. Mientras que las neuronas ocultas poseen carácter local, las neuronas de salida realizan una combinación lineal de las activaciones de las neuronas ocultas. Este tipo de redes construyen aproximaciones que son combinaciones lineales de múltiples funciones locales no lineales. Entre sus aplicaciones se encuentran análisis de series temporales, procesamiento de imágenes, etc.
- ▶ Redes estocásticas. Son redes recurrentes que utilizan redes bayesianas.
- ▶ Red de Hopfield. Las redes de Hopfield se usan como sistemas de memoria asociativa con unidades binarias. Están diseñadas para converger a un mínimo local, pero la convergencia a uno de los patrones almacenados no está garantizada.

Normalmente, se utiliza para aprender el algoritmo de **retropropagación del error cometido por la red**. No entraremos en detalles, pero la idea es que la derivada del error producido entre la salida generada y la esperada va modificando los pesos de la red en una especie de descenso de gradiente. Esto se hace indispensable para que la red aprenda y conozca el dato correcto del ejemplo que procesa. Como otros algoritmos, si se entrena demasiado tiende a la especialización y a perder capacidad de generalización. Es importante detener el entrenamiento cuando se detecte que este comienza a ofrecer un error de validación creciente. Es decir, que en este tipo

de redes es importante entrenar y evaluar con datos no entrenados para saber si la red se sobreadapta a los ejemplos de entrenamiento o no.

El **concepto de *deep learning*** viene precisamente de crear modelos de redes de neuronas con un gran número de capas ocultas. Estos modelos de redes de neuronas ahora son posibles debido a las nuevas técnicas de aprendizaje que se utilizan y al incremento de la potencia de cálculo disponible. Pero en líneas generales siguen las mismas directrices que los modelos simples. Una descripción más detallada de este tipo de redes se sale del objetivo de esta asignatura, pero vamos a enumerar los modelos más conocidos.

Autoencoders

Son **redes neuronales con el objetivo de generar nuevos datos**, primero comprimiendo la entrada en un espacio de variables latentes y luego reconstruyendo la salida en base a la información adquirida. La parte comprimida es una abstracción que sirve para extraer las características importantes de los datos.

Este tipo de red **consta de dos partes**:

- ▶ Encoder: la parte de la red que comprime la entrada en un espacio de variables latentes.
- ▶ Decoder: la parte que trata de reconstruir la entrada en base a la información recolectada previamente. Se representa mediante la función de decodificación $r = g(h)$.

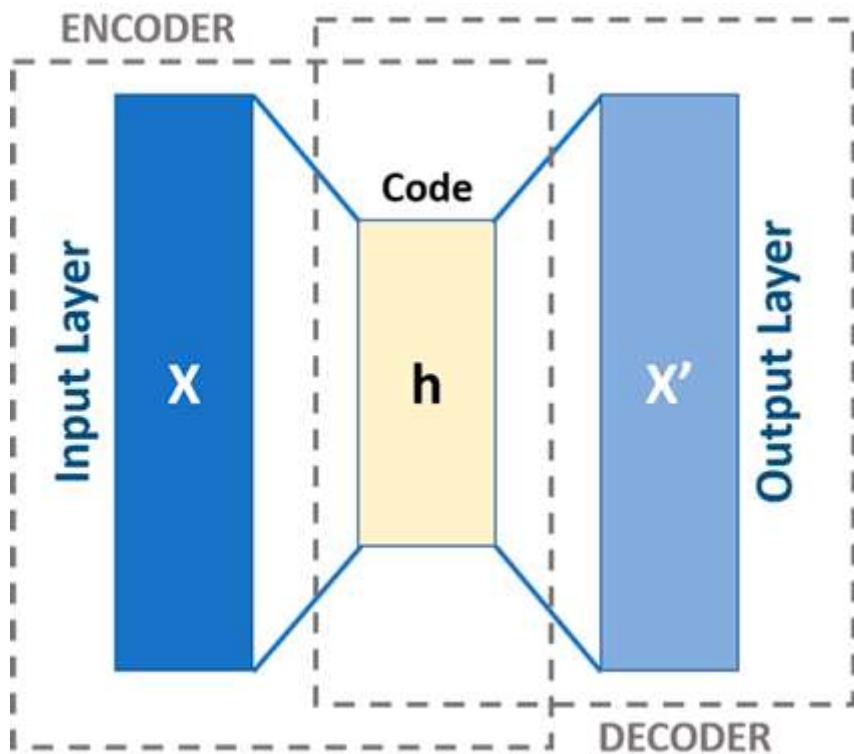


Figura 6. Esquema de un autoencoder.

Redes convolucionales

Las redes neuronales convolucionales son **un tipo de redes neuronales donde las neuronas corresponden a campos receptivos de una manera muy similar a las neuronas en la corteza visual**. Este tipo de red es una variación de un perceptrón multicapa, sin embargo, debido a que su aplicación es realizada en matrices bidimensionales, son muy efectivas para tareas de visión artificial, como en la clasificación y segmentación de imágenes, entre otras aplicaciones. Consisten en múltiples capas de filtros convolucionales de una o más dimensiones. Después de cada capa, por lo general, se añade una función para realizar un mapeo causal no lineal. Como cualquier red empleada para clasificación, al principio estas redes tienen una fase de extracción de características, compuesta de neuronas convolucionales, luego hay una reducción por muestreo y al final tendremos neuronas de perceptrón más sencillas para realizar la clasificación final sobre las características extraídas.

Neural Turing machines

Una máquina de Turing neural es una **red neural extendida con una memoria de trabajo que le proporciona unas altas capacidades de aprendizaje**. La arquitectura de una máquina de Turing neural (NTM) contiene dos componentes básicos: un controlador de red neural y un banco de memoria. Como la mayoría de las redes neuronales, el controlador interactúa con el mundo exterior a través de vectores de entrada y salida. A diferencia de una red estándar, también interactúa con una matriz de memoria utilizando operaciones selectivas de lectura y escritura. Por analogía con la máquina de Turing, las salidas de la red que parametrizan estas operaciones se llaman cabezas, como la cabeza lectora de la máquina de Turing.

Redes recurrentes profundas

Son versiones profundas de las redes recurrentes. Tienen conexiones de retroalimentación. No solo pueden procesar puntos de datos individuales (como imágenes), sino también secuencias enteras de datos (como voz o vídeo). Una de las redes de este tipo más conocidas es la red LSTM (Long short-term memory). Esta red es aplicable a tareas como el reconocimiento de escritura, traducción automática, el reconocimiento de voz y la detección de anomalías en el tráfico de la red o sistemas de detección de intrusos.

Ventajas:

- ▶ De muy alta precisión.
- ▶ Sirven para regresión y para clasificación.
- ▶ Las nuevas técnicas las hacen muy escalables.
- ▶ Muy tolerantes al ruido.

Inconvenientes:

- ▶ Los modelos profundos son muy caros de calcular, se necesita muchísima potencia de cómputo para entrenar.
- ▶ Tienden a la sobreadaptación. Hay que vigilar con cuidado el entrenamiento para que no pierdan capacidad de abstracción.
- ▶ Son algoritmos de caja negra muy difíciles de analizar por los diseñadores.
- ▶ Dependen de ciertos pesos o parámetros iniciales y de como se inicialicen los valores.

Máquinas de vectores de soporte (SVM)

Son un **conjunto de algoritmos de aprendizaje supervisado desarrollados por Vladimir Vapnik** y su equipo en los laboratorios AT&T.

Estos métodos están propiamente relacionados con problemas de clasificación y regresión. Dado un conjunto de ejemplos de entrenamiento podemos etiquetar las clases y entrenar una SVM para construir un modelo que prediga la clase de una nueva muestra.

Los **vectores de soporte** son los puntos que definen el margen máximo de separación del hiperplano que separa las clases. Se llaman vectores porque estos puntos tienen tantos elementos como dimensiones tenga nuestro espacio de entrada. Es decir, estos puntos multidimensionales se representan con vector de n dimensiones.

Hay veces en las que no hay forma de encontrar un hiperplano que permita separar dos clases. En estos casos, decimos que las clases no son linealmente separables. Para resolver este problema **se puede usar un kernel**. El truco del kernel consiste en inventar una dimensión nueva en la que podamos encontrar un hiperplano para separar las clases.

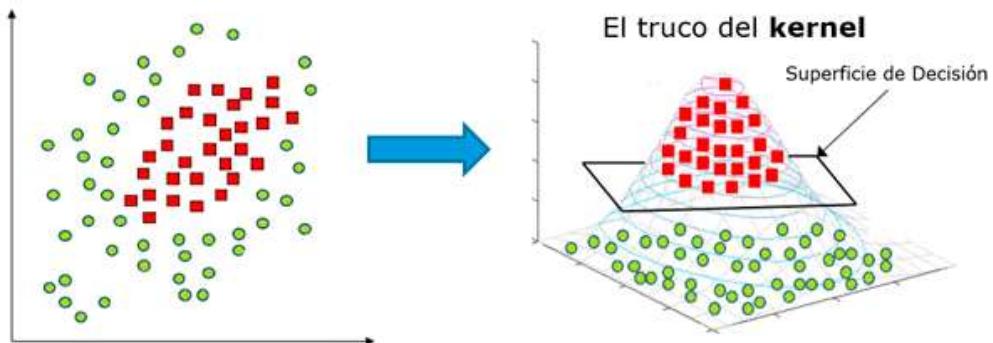


Figura 7. Kernel en SVM.

Fuente: <https://www.iartificial.net/maquinas-de-vectores-de-soporte-svm/>

Algunos casos de éxito de las máquinas de vectores de soporte son:

- ▶ Reconocimiento óptico de caracteres.
- ▶ Detección de caras para que las cámaras digitales enfoquen correctamente.
- ▶ Filtros de *spam* para correo electrónico.
- ▶ Reconocimiento de imágenes a bordo de satélites (saber qué partes de una imagen tienen nubes, tierra, agua, hielo, etc.).

Ventajas:

- ▶ Eficaz en espacios de grandes dimensiones.
- ▶ Todavía eficaz en casos donde el número de dimensiones es mayor que el número de muestras.
- ▶ Utiliza un subconjunto de puntos de entrenamiento en la función de decisión (llamada vectores de soporte), por lo que también es eficiente en memoria.
- ▶ Versátil: se pueden especificar diferentes funciones del núcleo para la función de decisión. Se proporcionan kernels comunes, pero también es posible especificar kernels personalizados.

Las **desventajas** son:

- ▶ Si el número de características es mucho mayor que el número de muestras evita el exceso de ajuste al elegir las funciones del kernel y el término de regularización es crucial.
- ▶ Los SVM no proporcionan directamente estimaciones de probabilidad, estas se calculan utilizando una validación cruzada.

Aprendizaje por refuerzo

El aprendizaje por refuerzo es una **serie de técnicas inspiradas en la psicología conductista**, cuya ocupación es determinar qué acciones debe escoger un agente *software* en un entorno dado con el fin de maximizar alguna noción de recompensa o premio acumulado. Esta recompensa o premio se da cuando el agente realiza la acción correcta en un momento dado.

El agente que aprende **sigue un proceso de decisión de Marcov**:

- ▶ El agente percibe un conjunto finito (S) de estados distintos en su entorno, y dispone de un conjunto finito (A) de acciones para interactuar con él.
- ▶ El tiempo avanza de forma discreta, y en cada instante de tiempo (t) el agente percibe un estado concreto.
- ▶ El entorno responde a la acción del agente por medio de una recompensa. Se formaliza esta recompensa/castigo por medio de un número.
- ▶ Tanto la recompensa como el estado siguiente obtenido no tienen por qué ser conocidos *a priori* por el agente, y dependen únicamente del estado actual y de la acción tomada.

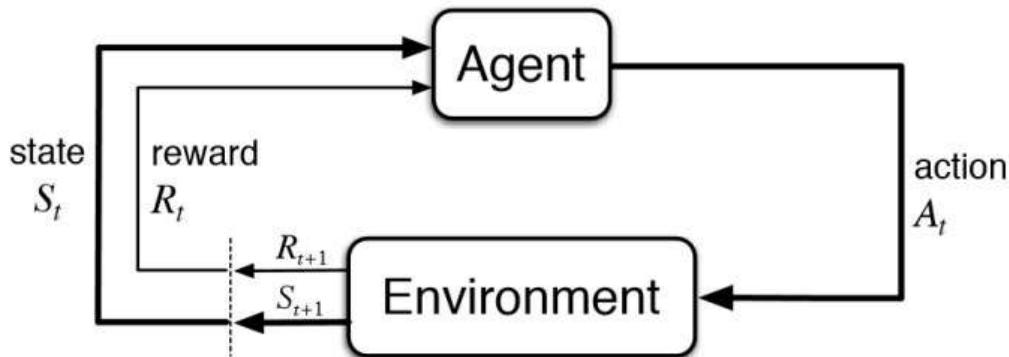


Figura 8. Esquema de aprendizaje por refuerzo.

Fuente: <https://wiki.pathmind.com/deep-reinforcement-learning>

El objetivo del aprendizaje por refuerzo es extraer qué acciones deben ser elegidas en los diferentes estados para maximizar la recompensa. Buscamos, por tanto, **que el agente aprenda una política**. Esta política no es más que un criterio a la hora de seleccionar una acción o la otra.

Existen diferentes implementaciones de aprendizaje por refuerzo, pero nos vamos a centrar en **cómo aprende la política el algoritmo Q-learning**.

El algoritmo, por tanto, tiene una función que calcula la calidad de una combinación estado-acción:

$$Q : S \times A \rightarrow \mathbb{R}.$$

Antes de que comience el aprendizaje se inicializa a un valor arbitrario constante. Después, en cada tiempo el agente selecciona una acción (a) observa una recompensa (r) introduce un estado nuevo $s(t+1)$ que depende del estado anterior y de la acción seleccionada, actualizando Q. El núcleo del algoritmo es una actualización del valor de la iteración simple, haciendo la media ponderada del valor antiguo y la información nueva:

$$Q^{new}(s_t, a_t) \leftarrow \underbrace{Q(s_t, a_t)}_{\text{old value}} + \underbrace{\alpha}_{\text{learning rate}} \cdot \underbrace{\left(\underbrace{r_t}_{\text{reward}} + \underbrace{\gamma}_{\text{discount factor}} \cdot \underbrace{\max_a Q(s_{t+1}, a)}_{\text{estimate of optimal future value}} \right)}_{\text{learned value}}$$

Estos valores se van almacenando en lo que se denomina tabla Q que almacena los valores que se van produciendo. La probabilidad de seleccionar un camino bueno es mayor que uno malo. De esta forma el agente cada vez tiende a hacer las acciones que más recompensa le han generado. Aunque se permite la exploración de nuevas políticas dejando siempre una probabilidad de seleccionar otra acción.



The diagram illustrates the evolution of a Q-table. On the left, under the heading 'Initialized', is a 5x6 grid representing the Q-table. The columns are labeled 'Actions' (South (S), North (N), East (E), West (W), Pickup (P), Dropoff (D)) and the rows are labeled 'States' (0, 1, 2, 3, 4). All values in the grid are zero. An arrow labeled 'Training' points to the right, leading to the second part of the diagram. On the right, under the heading 'Training', is another 5x6 grid representing the Q-table. The columns are the same as the first. The rows are labeled 'States' (0, 1, 2, 3, 4). The values are now non-zero and reflect the results of training: State 0 has values [0, 0, 0, 0, 0, 0]; State 1 has values [0, 0, 0, 0, 0, 0]; State 2 has values [-2.3010805, -1.97092096, -2.30357004, -2.20599839, -10.3607744, -8.5583017]; State 3 has values [0, 0, 0, 0, 0, 0]; State 4 has values [0, 0, 0, 0, 0, 0].

Figura 9. Tabla Q inicializada y actualizada con el entrenamiento.

Existe una versión más avanzada de Q-learning que se denomina **deep reinforcement learning**, que es el algoritmo usado por AlphaGo y AlphaStar para aprender a jugar a *Go* y *Starcraft*. También es el utilizado por OpenAI para aprender a jugar juegos arcade.

El aprendizaje por refuerzo profundo combina redes neuronales artificiales con una arquitectura de aprendizaje por refuerzo que permite a los agentes aprender las mejores acciones posibles en un entorno virtual para alcanzar sus objetivos. Es

decir, une la aproximación a la función y la optimización del objetivo, mapeando los pares de estado y acción a las recompensas esperadas. Una red neuronal puede utilizarse para aproximar una función de valor o una función de política. Es decir, las redes neuronales pueden aprender a mapear estados a valores o pares estado-acción a valores Q. En lugar de utilizar una tabla de búsqueda para almacenar, indexar y actualizar todos los posibles estados y sus valores (lo cual es imposible con problemas muy grandes) podemos entrenar una red neuronal en muestras del espacio de estado o acción para aprender a predecir cuan valiosos son estos en relación con nuestro objetivo en el aprendizaje de refuerzo.

Clústering

Los algoritmos de clústering o agrupamiento **son algoritmos de aprendizaje no supervisado que pretenden obtener el conjunto de clases que conforman una determinada colección de valores de entrenamiento.** Suelen realizar una clasificación a partir de un representante, denominado centroide. Este representante es el elemento que mejor divide cada clúster o grupo y los demás se asocian a él por proximidad. Este tipo de algoritmos normalmente son dependientes del número de clases que estimemos que existen. Aunque existen métodos que son capaces de inferir el número de clases que hay.

Para evaluarlos es muy importante la visualización de los mismos porque no hay un método analítico para saber si la agrupación es del todo correcta si asumimos que los datos no están etiquetados. Podemos, para entrenar el sistema, tener datos etiquetados y con ellos validar el clúster que ha realizado el algoritmo. Pero estos valores no son tenidos en cuenta por el algoritmo. La medida más utilizada para medir la similitud entre los casos es la matriz de correlación entre los $n \times n$ casos. Sin embargo, también existen muchos algoritmos que se basan en la maximización de una propiedad estadística llamada verosimilitud. Generalmente, los vectores de un mismo grupo (o clústeres) comparten propiedades comunes. El conocimiento de los grupos puede permitir una descripción sintética de un conjunto de datos multidimensional complejo. De ahí su uso en minería de datos. Esta descripción

sintética se consigue sustituyendo la descripción de todos los elementos de un grupo por la de un representante característico del mismo.

Uno de los algoritmos más conocidos es **k-means**.

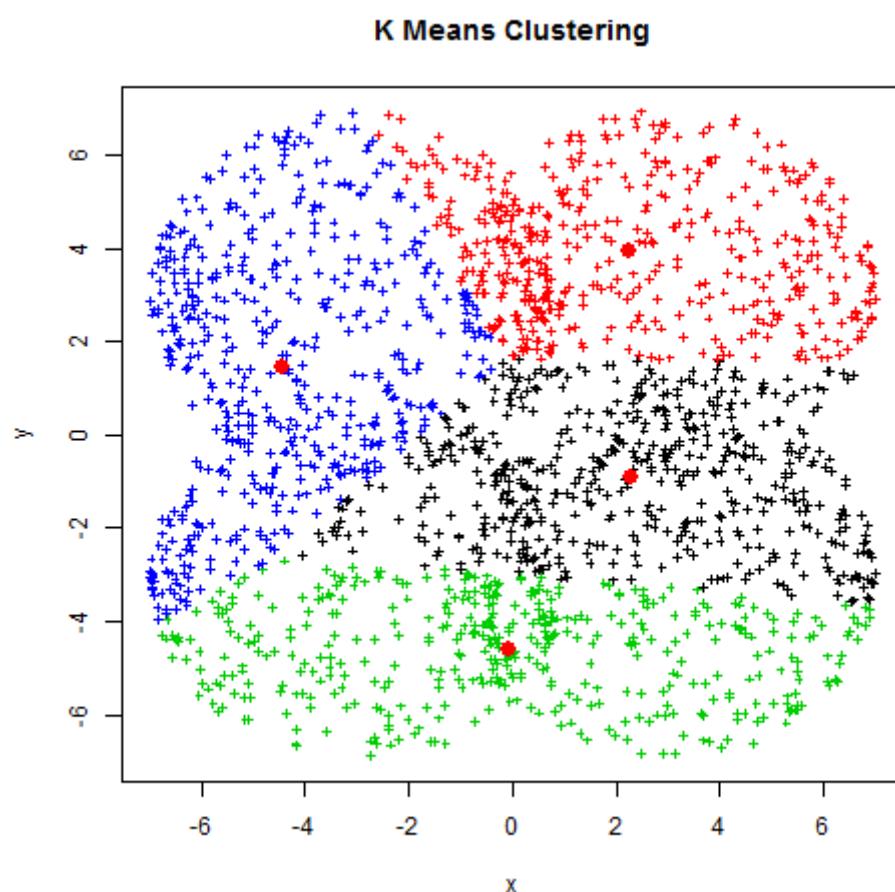


Figura 10. Ejemplo de cómo k-means agrupa un conjunto de datos.

K-means necesita como dato de entrada el número de grupos en los que vamos a segmentar la población. A partir de este número k de clúster, el algoritmo coloca primero k puntos aleatorios (centroídes). Luego asigna a cualquiera de esos puntos todas las muestras con las distancias más pequeñas.

A continuación, el punto se desplaza a la media de las muestras más cercanas. Esto generará una nueva asignación de muestras, ya que algunas muestras están ahora más cerca de otro centroide. Este proceso se repite de forma iterativa y los grupos se van ajustando hasta que la asignación no cambia más moviendo los puntos.

9.5. Proyectos de investigación sobre aprendizaje automático

Los proyectos de investigación en *machine learning* son actualmente los más prolíficos en inteligencia artificial. Pueden ser aplicados a prácticamente todo, pero por hacer un resumen rápido podemos centrarnos en:

- ▶ Todo tipo de clasificación automática.
- ▶ Reconocimiento de imágenes.
- ▶ Programación de agentes autónomos y robótica.
- ▶ Regresión.
- ▶ Sistemas de recomendación.
- ▶ Procesado de imágenes.
- ▶ Generación de modelos de predicción de todo tipo.
- ▶ Procesado del lenguaje natural.
- ▶ Sistemas cognitivos.
- ▶ Experiencia de usuario.
- ▶ Lead scoring: algoritmo que tiene en cuenta una serie de parámetros de los diferentes clientes potenciales que llegan para ordenarlos por prioridad y que los agentes comerciales trabajen de una manera más eficaz.
- ▶ Producción inteligente. Con estos proyectos se logra identificar nuevos modelos, tendencias y estacionalidades que permiten a las empresas adelantarse a futuras ventas y cubrir las necesidades del stock.

Los proyectos de *machine learning* suelen requerir de grandes cantidades de computación, por lo que es muy recomendable acudir a soluciones de terceros. Además, como hemos visto, son muy dependientes de la calidad de los datos y de poder acceder a ellos. Se estima que la mayor parte del coste en este tipo de proyectos es la adquisición, procesado y refinamiento de estos datos. Hay que tener

en cuenta estas cuestiones a la hora de embarcarse en un proyecto de aprendizaje automático.

Las principales publicaciones de este medio son:

- ▶ *IEEE Transactions on Pattern Analysis and Machine Intelligence.*
- ▶ *arXiv: Computer Vision and Pattern Recognition.*
- ▶ *Journal of Machine Learning Research.*
- ▶ *arXiv: Learning.*
- ▶ *Expert System with Application.*
- ▶ *International Journal of Computer Vision.*

9.6. Referencias bibliográficas

Marsland, S. (2015). *Machine learning: an algorithmic perspective*. CRC press.

A fondo

How AI is changing the video game industry: augmentation and synthetic media

AI Business. (26 de febrero de 2020). How AI is changing the video game industry: augmentation and synthetic media. *aibusiness.com*. Recuperado de https://aibusiness.com/author.asp?section_id=789&doc_id=761220

Artículo donde se habla de cómo la IA revolucionará la industria del videojuego.

4 Social Media Data Mining Techniques to Help Grow Your Online Business

Lim, H. (15 de marzo de 2020). 4 Social Media Data Mining Techniques to Help Grow Your Online Business. *hackernoon.com*. Recuperado de <https://hackernoon.com/4-social-media-data-mining-techniques-to-help-grow-your-online-business-o6ch32q4>

Artículo que describe diferentes técnicas de minería de datos útiles para ayudar al crecimiento del negocio *online*.

Deep Reinforcement Learning

Silver, D. (17 de junio de 2016). Deep Reinforcement Learning [Mensaje en un blog]. DeepMind. Recuperado de <https://deepmind.com/blog/article/deep-reinforcement-learning>

Artículo donde *Deep Mind* explica cómo funciona el *deep reinforcement learning* y los experimentos que lo demuestran.

Neural Turing Machines

Graves, A., Wayne, G. y Danihelka, I. (1 de junio de 2014). Neural Turing Machines. *deepmind.com*. Recuperado de <https://deepmind.com/research/publications/neural-turing-machines>

Artículo donde *Deep Mind* donde explica cómo funciona el neural Turing machine.

Test

- 1.** La capacidad de generalización de un algoritmo de *machine learning* viene determinado por...

 - A. Las máquinas no son capaces de aprender, solo repiten patrones previamente almacenados. Es solo una ilusión de que están aprendiendo.
 - B. El sesgo que introduce en el aprendizaje.
 - C. Su capacidad para adaptarse a los datos de entrenamiento.
 - D. A su similitud con la forma que tiene el cerebro de aprender.
- 2.** Los algoritmos de *machine learning* pueden ser de los siguientes tipos:

 - A. Supervisado, individual y multivariante.
 - B. No supervisado, supervisado y clústering.
 - C. No supervisado, supervisado y por refuerzo.
 - D. Por refuerzo, clústering y supervisado.
- 3.** El aprendizaje supervisado se caracteriza por:

 - A. Usar la solución de cada ejemplo como parte del aprendizaje.
 - B. No conocer la solución de los ejemplos para realizar el aprendizaje.
 - C. Usar o supervisado, supervisado y por refuerzo.
 - D. Por refuerzo, clústering y supervisado.

- 4.** El aprendizaje por refuerzo se caracteriza por:
- A. No necesitar a un instructor que guie el entrenamiento.
 - B. Usar el *feedback* del entorno como guía para saber si la acción realizada es la correcta.
 - C. Selecciona de forma aleatoria la siguiente acción en base al *feedback* acumulado.
 - D. Todas las anteriores son correctas.
- 5.** Un problema de clasificación es un problema donde...
- A. La variable dependiente es discreta y finita.
 - B. Es continua e infinita.
 - C. Es discreta e infinita.
 - D. Ninguna de las anteriores.
- 6.** ¿Cuál de estas afirmaciones es cierta sobre K-NN?
- A. Es un algoritmo de aprendizaje no supervisado que busca almacenar estados antiguos y su solución para recuperarlas posteriormente.
 - B. Es un algoritmo de aprendizaje supervisado que busca almacenar estados antiguos y su solución para recuperarlas posteriormente.
 - C. Es un algoritmo de aprendizaje no supervisado que agrupa a los ejemplos por proximidad formando clúster.
 - D. Ninguna de las anteriores.
- 7.** ¿Cuál de estas afirmaciones es cierta sobre los árboles de decisión?
- A. Es un algoritmo no supervisado que permite crear reglas.
 - B. Segmenta el problema con aproximaciones no lineales.
 - C. Es un algoritmo de caja negra.
 - D. Ninguna de las anteriores.

8. ¿Cuál de estas afirmaciones es cierta sobre las redes de neuronas?

- A. Es un algoritmo de aprendizaje supervisado muy resistente al ruido.
- B. Es un algoritmo de caja negra.
- C. Sirve para clasificación y para regresión.
- D. Todas las anteriores son correctas.

9. ¿Cuál de estas afirmaciones es cierta sobre el aprendizaje por refuerzo ?

- A. Utiliza la función de refuerzo que le proporciona el entorno para actualizar la política.
- B. Solo se almacena el último *feedback* recibido.
- C. El aprendizaje por refuerzo profundo sustituye la función de recompensa por una red de neuronas.
- D. Todas las anteriores son correctas.

10. ¿Cuál de estas afirmaciones es cierta sobre el k-means?

- A. K-means agrupa en base a la distancia de un centroide que se le proporciona como ejemplo.
- B. K-means agrupa en base a la distancia de un centroide que se va calculando en cada iteración como la media de las muestras clasificadas.
- C. K-means agrupa en base a la distancia de un centroide que se va calculando en cada iteración como el elemento más alejado de los elementos del grupo.
- D. Todas las anteriores son falsas.

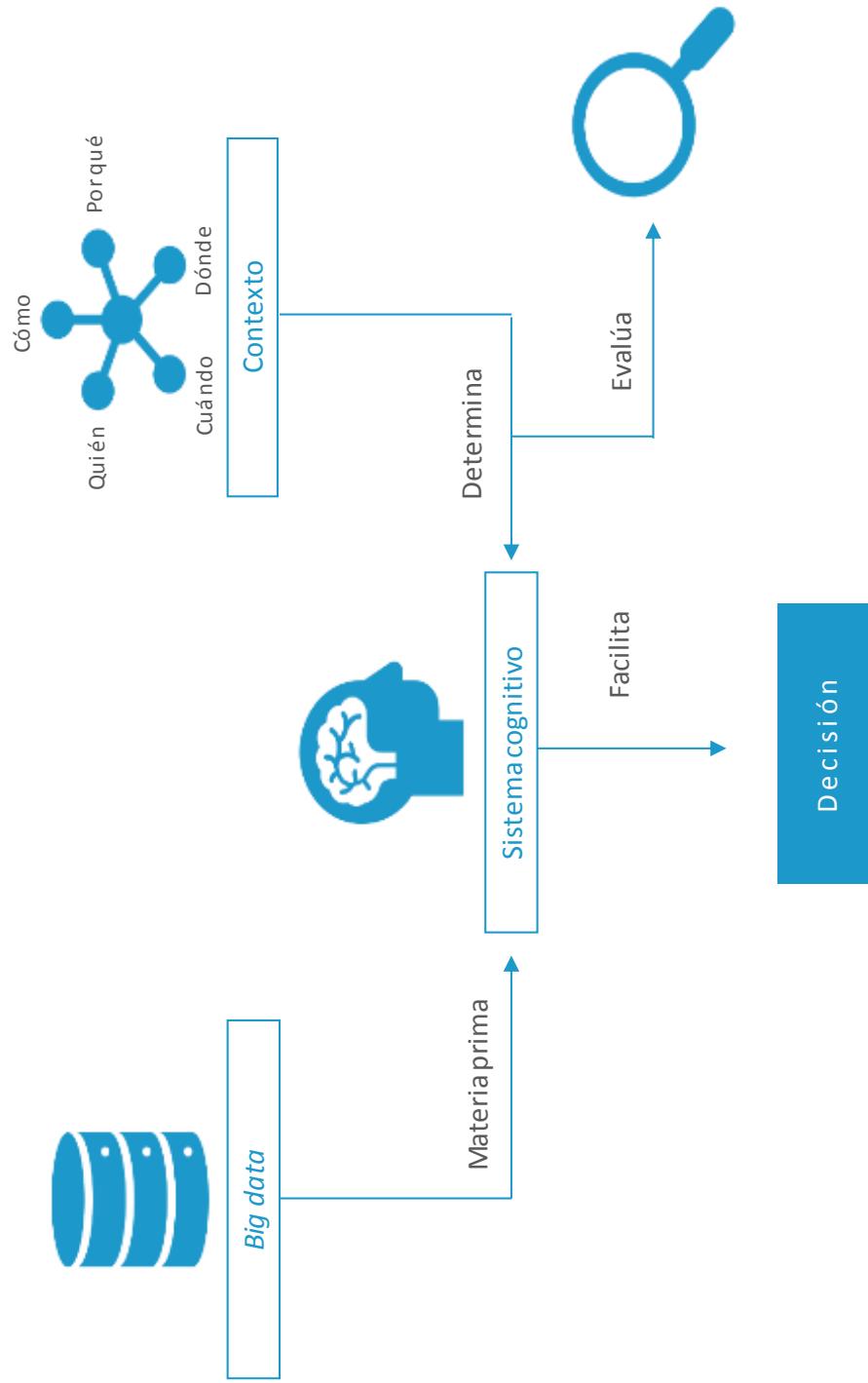
Investigación en Inteligencia Artificial

Investigación en sistemas cognitivos

Índice

Esquema	3
Ideas clave	4
10.1. Introducción y objetivos	4
10.2. Introducción a la computación cognitiva	4
10.3. Elementos de un sistema cognitivo	7
10.4. <i>Big data</i> y computación cognitiva	11
10.5. Percepción computacional	14
10.6. Procesamiento de lenguaje natural	17
10.7. Proyectos de investigación sobre sistemas cognitivos	20
10.8. Referencias bibliográficas	21
A Fondo	22
Test	23

Esquema



10.1. Introducción y objetivos

Los sistemas cognitivos están en auge en los últimos tiempos. Aquí se proporciona una **descripción conceptual de la computación cognitiva y un listado de sus aplicaciones más inmediatas**. Los objetivos planteados son los siguientes:

- ▶ Reconocer las peculiaridades de un sistema cognitivo.
- ▶ Conocer las utilidades de las aplicaciones cognitivas en el ámbito empresarial.
- ▶ Ser capaz de identificar limitaciones de los sistemas cognitivos.

10.2. Introducción a la computación cognitiva

En los últimos años algunos errores asociados a las aplicaciones de inteligencia artificial han aparecido en los medios especializados. Conocido fue el caso de *chatbot* de Microsoft, que en el mismo día realizó comentarios racistas, antisemitas e incluso misóginos.

Accede a la noticia a través del siguiente enlace:

<http://www.elmundo.es/tecnologia/2016/03/28/56f95c2146163fdd268b45d2.html>

Pero hay más, como por ejemplo el caso del *chatbot* de Facebook que «inventó» un nuevo idioma, el sistema de detección facial del iPhone X engañado por una máscara, un coche de conducción autónoma involucrado en un accidente en Las Vegas, etc.

Accede al artículo a través del siguiente enlace:

<https://medium.com/@Synced/2017-in-review-10-ai-failures-4a88da1bdf01>

Estos errores son la prueba de que, aunque son muy importantes los logros conseguidos por la disciplina, todavía queda mucho camino por recorrer.

Muchos de estos fallos se deben a que los creadores de las aplicaciones no han sido capaces de proporcionar a la máquina el contexto adecuado a su caso de uso.

Por ejemplo, en el caso de Face ID, la aplicación de reconocimiento facial del móvil iPhone X, los desarrolladores han entrenado a la aplicación para poder identificar rostros y, por consiguiente, personas. Pero una persona está unida a un tronco, tiene una cabeza, respira... Realmente, ¿entiende la máquina que está identificando a una persona o, por el contrario, se limita a analizar unas similitudes entre dos representaciones de una realidad? ¿Entiende la máquina de forma plena el contexto de la actividad que está realizando? La respuesta a esta pregunta no es trivial, porque primero tendremos que acordar qué significa «entiende». No obstante, parece obvio que la máquina maneja un contexto mucho más reducido que el que contempla un humano, puesto que este último nunca hubiese proporcionado acceso al teléfono a alguien con una máscara. El humano (medianamente avisado) reconocería que se trata de una máscara y pediría a su portador que se identificase correctamente.

La computación cognitiva se ocupa de la búsqueda de soluciones eficientes que permitan a los humanos colaborar con las máquinas de forma plena y eficaz, para generar nuevos conocimientos y experiencias.

Para ello, una de las principales necesidades es **proporcionar a la máquina contexto sobre el entorno y el objetivo a conseguir**. Por este motivo, las soluciones cognitivas deben hacer uso de un volumen de información muy amplio y diverso, surge aquí la relación entre la computación cognitiva y el paradigma *big data*.

La gran disponibilidad de datos, la reducción del coste de almacenamiento y las mejoras tecnológicas que permiten procesar grandes volúmenes de datos a altas velocidades han impulsado este tipo de soluciones.

Un sistema cognitivo se compone fundamentalmente de **tres principios fundamentales**:

- ▶ Aprendizaje: el sistema debe ser capaz de aprender en base a un conjunto de observaciones y realizar predicciones sobre un dominio concreto.
- ▶ Modelado: el aprendizaje toma como base la representación de un modelo y un conjunto de reglas de inferencia.
- ▶ Generación de hipótesis: un sistema cognitivo debe asumir que no existe una única respuesta válida. Es decir, un sistema cognitivo es un sistema probabilístico capaz de emitir varias respuestas dando una probabilidad a cada una de ellas.

Estos tres principios fundamentales **se basan todos en el almacenamiento, catalogación y uso eficiente de datos** que guían la toma de decisiones. Los sistemas cognitivos se encuentran todavía en una fase de desarrollo incipiente. El desarrollo de este tipo de productos involucra aspectos provenientes de otras disciplinas técnicas como el aprendizaje automático, el procesamiento de grandes volúmenes de datos, el internet de las cosas, procesamiento del lenguaje natural, razonamiento probabilístico, visualización, etc.

Los sistemas cognitivos están orientados a entornos probabilísticos, no determinísticos.

Un ejemplo típico de este escenario es el entorno clínico. Los síntomas del paciente invitan a pensar en una enfermedad concreta con una determinada probabilidad de acierto. El trabajo del facultativo en muchas ocasiones consiste en ir explorando el amplio abanico de opciones hasta localizar la raíz del problema de forma concreta. En general, se puede decir que este tipo de sistemas está enfocado para contextos que deben trabajar en un entorno de incertidumbre teniendo a su disposición una importante variedad de datos de distinta complejidad.

10.3. Elementos de un sistema cognitivo

Un sistema cognitivo consta de diferentes elementos que incluyen desde la capa de *hardware* al *software* y modelos implementados. Aunque cada solución concreta particulariza su propio diseño, es posible caracterizar una serie de elementos comunes. La figura 1 muestra los componentes principales de un sistema cognitivo:

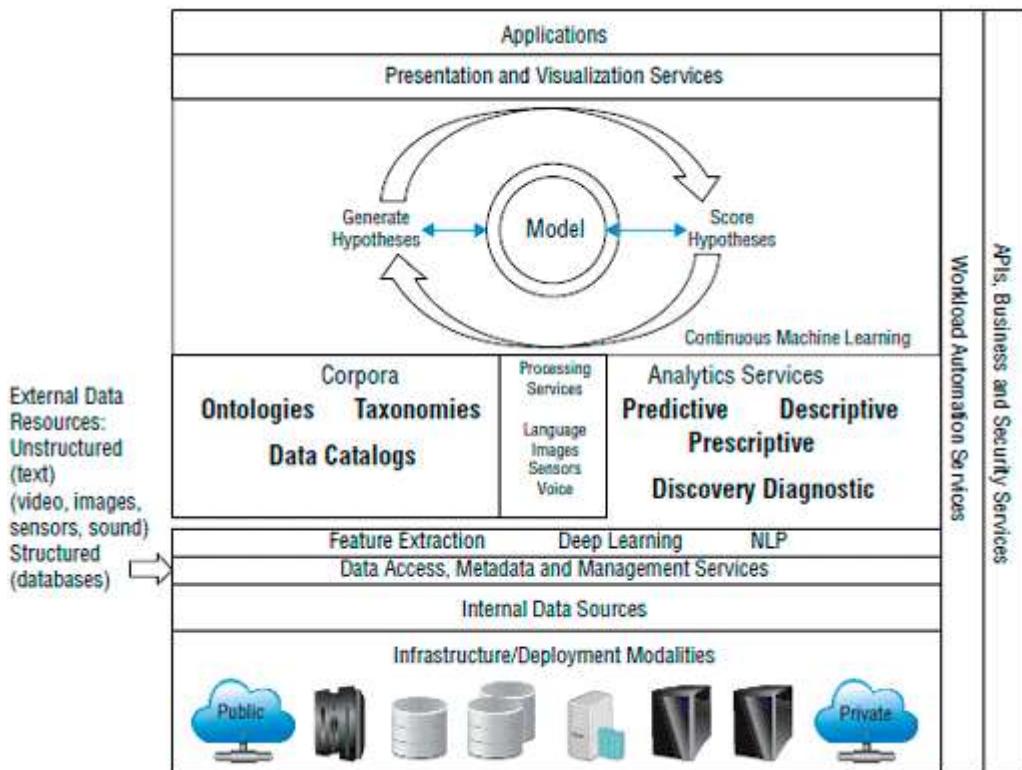


Figura 1. Elementos de un sistema cognitivo.

Fuente: Hurwitz, Kaufman y Bowles, 2015.

Un **sistema cognitivo se alimenta de datos**. Estos datos pueden ser **estructurados** (por ejemplo, los procedentes de bases de datos relacionales) o **no estructurados** (fotografías, vídeos, sonido, texto...).

Los datos constituirán la base para la toma de decisiones. **Gracias a ellos el sistema puede interpretar el entorno y crear los modelos adecuados**. Los datos precisan de una infraestructura de almacenamiento. Esta infraestructura admite muchas modalidades, normalmente basadas en productos en la nube.

De cara a planificar los recursos necesarios, debemos comprender que un sistema de estas características no solo tiene que almacenar los datos que recibe de entrada, la propia actividad del sistema y los modelos desarrollados crean nuevos datos que también alimentan al sistema. Además, el producto también puede contar con fuentes de datos internas y permanentes como, por ejemplo, las referidas a restricciones de contexto que se quieren mantener de forma constante.

El acceso a los datos se gestiona a través de una capa cuya labor es precisamente intermediar con la capa de almacenamiento. **El módulo de acceso a datos prepara los datos para su consumo y es el encargado de verificar que la calidad de los datos es adecuada**, monitorizando constantemente el proceso. Sobre esta capa se mantiene un corpus, con el cometido de codificar computacionalmente el conocimiento existente, una capa de servicios de procesamiento y una capa de servicios analíticos.

Un corpus de conocimiento estructura y codifica todo el conocimiento asociado a un ámbito concreto. Para su construcción es habitual recurrir al uso de ontologías. Estas ontologías permiten definir los conceptos, establecer sus propiedades y fijar relaciones entre ellos.

De forma continuada un proceso de aprendizaje automático alimenta la toma de decisiones del sistema. El proceso parte de una hipótesis como explicación verificable ante un hecho observado. Dicha hipótesis es verificada mediante datos para finalmente llegar a una conclusión que culmina en una acción o decisión. El proceso de aprendizaje conlleva un ajuste continuo del modelo o de los modelos.

Por último, es preciso crear varios componentes que facilitan la interacción máquina-humano e, incluso, máquina-máquina. La capa de visualización ayuda a interpretar los resultados y acciones sugeridas por el sistema, mientras que los servicios API (por sus siglas en inglés: Application Program Interface) facilitan el acceso automatizado a las funcionalidades que ofrece el sistema.

Diseño de sistemas cognitivos

En su sistema de computación cognitiva el modelo hace referencia tanto al corpus como a los algoritmos que permiten ejecutar la toma de decisiones. El modelo desarrollado por los programadores iniciales está cambiando continuamente. El **aprendizaje continuo es la esencia de un sistema cognitivo**. Dicho aprendizaje solo se culmina si el sistema es capaz de actualizar constantemente el modelo que guía su actividad. Asegurar esta actualización continua del modelo es el principal requisito a la hora de diseñar un sistema cognitivo.

Todo sistema cognitivo se debe diseñar de forma que:

1. Permita la ingesta en el sistema de toda la información necesaria.
2. Organice los datos de forma que se permita el acceso y análisis de forma eficiente.
3. Permita actualizar el corpus y los modelos iniciales.
4. Asegure el cumplimiento de la normativa vigente y los estándares de seguridad adecuados.
5. Permita emplear las técnicas adecuadas para la extracción de conocimiento y toma de decisiones.

Ontologías

Las ontologías **permiten la representación formal de los conceptos de un dominio y las relaciones entre ellos**. De esta forma no solo se modela el conocimiento existente, sino que, además, permite la estandarización del dominio creando un mapa conceptual que es compartido y comúnmente usado. Las ontologías facilitan el almacenamiento ordenado de información, la búsqueda de recursos y la comunicación entre distintos dispositivos. Gracias a las ontologías podemos también formalizar muchas de las suposiciones de partida entre los elementos, en base a relaciones entre conceptos, atributos de los elementos, etc.

Las ontologías se suelen representar gráficamente en forma de grafo para facilitar su comprensión.

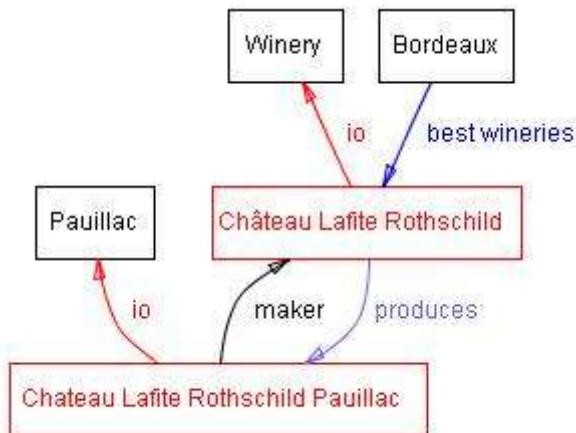


Figura 2. Ejemplo de parte de una ontología asociada al dominio del vino.

Fuente: https://protege.stanford.edu/publications/ontology_development/image002.jpg

Otro ejemplo algo más elaborado podría ser el que representa la figura 3.

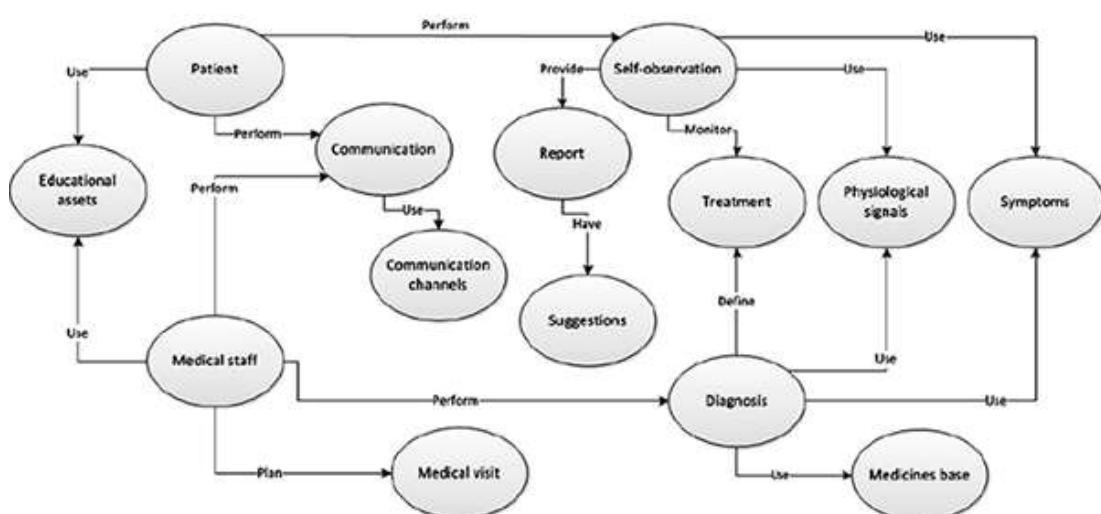


Figura 3. Ejemplo de ontología clínica para una aplicación de telemedicina.

Fuente: https://www.researchgate.net/figure/E-health-application-ontology-Medical-applications-which-are-currently-available-on-the_269700233

La herramienta Prótegé (<https://protege.stanford.edu/>), gestionada ahora por la Universidad de Stanford en colaboración con la Universidad de Manchester, es una de las herramientas más útiles para la creación de ontologías.

10.4. *Big data* y computación cognitiva

Se ha insistido mucho en este tema en la necesidad de disponer de grandes volúmenes de datos como materia prima para generar aplicaciones realmente eficaces. En otro caso, los sistemas cognitivos podrían encontrar serias dificultades para mostrar resultados consistentes, fiables y relevantes. Además, y dado que se precisa tanto de datos estructurados como no estructurados, es necesario emplear otras tecnologías distintas de las bases de datos relacionales, puesto que estas últimas están diseñadas específicamente para interactuar con datos relacionales.

Big data no hace referencia a una tecnología concreta. *Big data* hace referencia a un paradigma centrado en capturar información de fuentes de datos diversas, que se generan a una velocidad considerable y con un gran volumen. Y esta actividad viene inevitablemente unida a la obtención de valor para el negocio.

Acumular datos sin más no tiene ningún sentido. Es la obtención de información útil y conocimiento que apoye la toma de decisiones la razón de ser de este paradigma.

Las tecnologías que apoyan este nuevo proceso son varias y diversas. Intentando hacer una generalización, podríamos detectar algunas grandes categorías:

- ▶ Bases de datos NoSQL (por sus siglas en inglés Structured Query Language).
- ▶ Sistemas de archivos distribuidos.
- ▶ Sistemas de cómputo en paralelo.

Bases de datos NoSQL

NoSQL hace referencia a Non-only SQL. Estas bases de datos **se caracterizan por la facilidad de desarrollo, el desempeño escalable, la alta disponibilidad y la resiliencia.**

A cambio de permitir mayores cotas de flexibilidad y versatilidad, estas herramientas han debido renunciar en mayor o menor medida a las propiedades ACID (Atomicity, Consistency, Isolation and Durability) de los sistemas de bases de datos relacionales tradicionales.

ACID, como acabamos de ver, es el acrónimo para referirse a las propiedades de atomicidad, consistencia, aislamiento y durabilidad:

- ▶ La atomicidad implica que una transacción se ejecuta completamente o no se ejecuta en absoluto.
- ▶ Consistencia quiere decir que una vez se ha ejecutado una transacción, los datos deben acoplarse al esquema de la base de datos.
- ▶ El aislamiento requiere que las transacciones simultáneas se ejecuten por separado.
- ▶ La durabilidad es la capacidad de recuperarse de un error inesperado del sistema o de un corte de energía y volver al último estado conocido.

Mantener estas propiedades ayuda a construir bases de datos más robustas, menos propensas a fallos y más formales. Sin embargo, para conseguir estas propiedades, los sistemas gestores de bases de datos incurren en penalización de rendimiento que no son sostenibles cuando la base de datos escala a grandes tamaños. Por ese motivo, los sistemas NoSQL buscan crear bases de datos sin estas restricciones para ser mucho más rápidos y poder procesar mayor número de datos. A cambio, estos sistemas son más vulnerables a fallos.

Existen diversos tipos de bases de datos NoSQL, incluyendo bases de datos documentales y bases de datos orientadas a grafos. Un ejemplo es MongoDB o Cassandra.

Sistemas de archivos distribuidos

Un sistema de archivos distribuidos **proporciona un sistema de almacenamiento permanente apoyándose en una infraestructura en red**. De esta forma se puede aprovechar el potencial de una red de computadoras y escalar las capacidades de almacenamiento en un sistema multiusuario.

Los sistemas de archivos distribuidos son esenciales en el ecosistema *big data*. La **replicación de la información es una de sus características esenciales**. De esta forma se facilita el acceso concurrente de los usuarios a la información. Adicionalmente, los sistemas *big data* de archivos distribuidos cuentan con una configuración diferente a los sistemas de archivos tradicionales.

Una de las diferencias más relevantes es el **tamaño de bloque de disco**. El tamaño de bloque de disco es mucho más grande en los sistemas *big data* que en sistemas de archivos tradicionales. De esta forma y con una sola búsqueda y acceso se accede a mucha más información de golpe. A cambio, los sistemas *big data* son una muy mala opción si una de las funcionalidades principales del sistema debe ser la actualización y modificación continua de la información.

Sistemas de cómputo en paralelo

Gran parte de los procesamientos de aprendizaje automático implican la realización de múltiples cálculos repetitivos o susceptibles de ser descompuestos en tareas similares que se realizan una y otra vez. Los sistemas de cómputo en paralelo aprovechan la capacidad de cálculo de redes de computadoras conectadas entre sí

proporcionando la escalabilidad de cómputo necesaria para procesar grandes volúmenes de información de forma eficiente.

10.5. Percepción computacional

Los **sistemas cognitivos intentan interactuar con los humanos usando su mismo lenguaje**. Como hemos comentado anteriormente, la idea es que los sistemas cognitivos puedan procesar el lenguaje natural para que puedan entender a los usuarios que los usan, por ejemplo, mediante órdenes de voz. También necesitan ensegún qué entornos percibir lo que le rodea. La percepción computacional es una serie de algoritmos que permiten a un *software* reconocer imágenes, sonidos etc. De forma que puede procesar la voz humana o puede procesar el rostro de una persona para reconocerle.

Podemos definir la **percepción** como la capacidad de los organismos para obtener información sobre su ambiente a partir de los efectos que los estímulos producen sobre los sistemas sensoriales. La **percepción computacional** es la capacidad que tiene las máquinas de imitar la percepción de los organismos.

Tradicionalmente, hacer que una computadora procesara imágenes, por ejemplo, en tiempo real, era algo bastante difícil de conseguir. Sin embargo, con las mejoras de las técnicas y el aumento de la potencia de cálculo disponible, **actualmente un computador puede reconocer, clasificar, etiquetar y analizar imágenes de forma bastante solvente**. Aunque sigue siendo una de las tareas pendientes de mejora, las técnicas de *machine learning* utilizadas para la visión por computador, por ejemplo, nos permiten hacer cosas que eran impensables hace tiempo.

Las redes más usadas para el procesado de imágenes son las **redes convolucionales y los autoencoders**. Estas se van aplicando por capas y estas capas se concatenan unas con las otras. Es muy típico que en las primeras capas la red aprenda unas

simples características y sobre ellas se va construyendo otras capas que realizan otras áreas, por ejemplo, la diferenciación de objetos.

Los **autoencoders** ayudan a la extracción de características de una imagen, lo que hace que se elimine el paso previo de clasificación que se solía dar antiguamente.

Las **redes convolucionales** se utilizan también para procesar las imágenes. Normalmente se aplican en las primeras capas para extraer características, reducir la dimensión de la imagen, etc.

Finalmente, se suelen utilizar **perceptrones multicapa simples** para la etapa de clasificación. Un ejemplo de esta arquitectura por capas se puede ver en la figura 4.

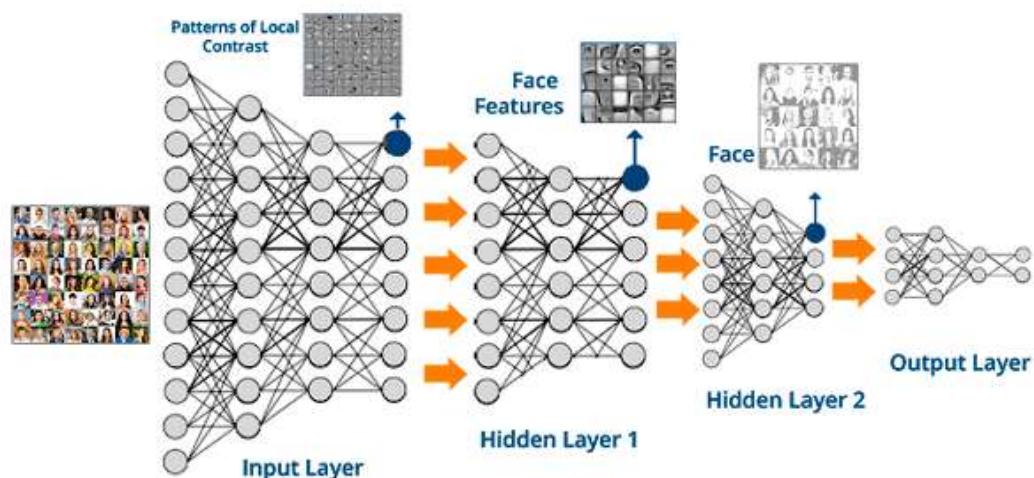


Figura 4. Esquema de una red que procesa imágenes con diferentes capas.

Fuente: <https://www.ellaberintodefalken.com/2019/10/vision-artificial-redes-convolucionales-CNN.html>

Con estos modelos se han conseguido importantes avances en esta área. Aquí tenemos un ejemplo de una aplicación para crear tiendas autónomas sin dependientes:

<https://computerhoy.com/noticias/tecnologia/startup-francesa-tiendas-autonomas-inteligencia-vision-artificial-712177>

La **visión artificial** tiene diferentes aplicaciones, podemos resumirlas como:

- ▶ Reconocimiento de objetos: decir qué objetos hay en una imagen, etiquetar las caras en una fotografía, etc.
- ▶ Reconstrucción de objetos 3D a partir de objetos 2D. Poder construir a partir de una serie de imágenes en 2D un objeto en 3D. Se le suele denominar **fotometría**.
- ▶ Análisis de vídeo: reconocer en tiempo real intrusos, la posición de un objeto, la distancia. A veces se apoyan en sensores de infrarrojo que ayudan a estos cálculos.
- ▶ Aplicaciones industriales. Por ejemplo, se utiliza visión artificial para el control de calidad de ciertos productos, para su clasificación, el guiado de robots, la medición de objetos, etc.

Para conseguir esos resultados se necesita normalmente varias etapas. Normalmente hay que procesar de alguna forma la señal visual para ayudar a los algoritmos de *machine learning* a trabajar con las imágenes. Así que pueden procesarse la imagen para eliminar ruido, aplicar la transformada de Fourier para analizarla, detectar los bordes de los objetos, etc.

La **iluminación** es un aspecto clave en la visión artificial. Muchas veces un objeto no se reconoce por cambios en la iluminación de la escena. Así que, si utilizamos la visión artificial en un entorno controlado, es mejor ayudar al sistema de visión con una iluminación apropiada.

Ventajas de la visión artificial

Algunas **ventajas de la visión artificial sobre la humana** son las siguientes: la visión humana es mejor para la interpretación cualitativa y para interpretar escenas sin estructura, mientras que en la visión artificial destaca que es mejor en tomar medidas cuantitativas sobre escenas con cierta estructura, precisión y repetibilidad. Por ejemplo, en una línea de producción, un sistema de visión artificial puede

inspeccionar cientos, o incluso miles, de piezas por minuto. También puede procesar detalles más pequeños si la cámara tiene mucha resolución o aumento.

Procesamiento y generación de sonido

Además de la visión como percepción computacional debemos tener en cuenta también el **procesamiento del sonido**. Y no solo el procesamiento sino la generación de sonido. Trabajos relacionados con esta área son los siguientes:

- ▶ Síntesis de voz.
- ▶ Reconocimiento del habla.
- ▶ Traducción automática.
- ▶ Generación de música.
- ▶ Reconocimiento de objetos mediante el audio.

10.6. Procesamiento de lenguaje natural

Se ha insistido mucho en lo que llevamos de tema en que los sistemas cognitivos introducen nuevos elementos en la relación humano-máquina. Efectivamente, muchos de estos sistemas contemplan en su funcionalidad una interacción continua entre las personas y las máquinas.

El principal medio de expresión de los humanos es el lenguaje, ya sea oral o escrito. Por tanto, las **técnicas de procesamiento del lenguaje natural** forman un papel esencial aquí.

Para comprender el mensaje emitido, las técnicas de procesamiento del lenguaje natural deben considerar el momento o fecha en la que se emite el mensaje, identificar al emisor, detectar los pronombres y reconocer a quién se están refiriendo, quién habla de quién, etc.

Es preciso catalogar correctamente los diferentes conceptos (personas, lugares, cosas) y la relación que guardan entre ellos. El orden de las palabras, la propia sintaxis, el uso de sinónimos y antónimos, frase hechas..., producen variaciones y particularizaciones del significado que es preciso aprehender.

La ambigüedad y el uso de palabras como los conectores contribuyen a añadir una capa adicional de complejidad al proceso. El análisis probabilístico de las palabras que aparecen (y el orden en el que lo hacen) en los textos empleados durante el proceso de aprendizaje puede añadir algo de claridad a la tarea.

Las **cadenas de Márkov** son ampliamente utilizadas para el análisis probabilístico de cadenas de texto. En la teoría de la probabilidad se conoce como cadena de Márkov o modelo de Márkov a un **tipo especial de proceso estocástico discreto** en el que la probabilidad de que ocurra un evento depende solamente del evento inmediatamente anterior.

Esta característica de falta de memoria recibe el nombre de propiedad de Markov:

https://es.wikipedia.org/wiki/Cadena_de_M%C3%A1rkov

El lenguaje hablado o escrito constituye un ejemplo de dato no estructurado. Interpretar correctamente mensajes lingüísticos es una tarea nada trivial debido a la importancia del contexto. El significado de una expresión concreta puede variar en función de la situación y la entonación. Por ejemplo, imaginemos el siguiente diálogo:

—Buenas tardes, Sr. Martín. Le llamamos del servicio de atención al cliente para hacerle una pregunta rápida que nos ayuda a mejorar: ¿está satisfecho con nuestros servicios?
—Sí, mucho.

¿Qué ha interpretado la operadora ante la respuesta del cliente? Leyendo el texto sin disponer de más información, parece simple deducir que la respuesta del cliente es positiva. Por tanto, un punto extra para la compañía y

su impecable servicio. No obstante, un mensaje oral viene acompañado de una entonación concreta. El tono del mensaje forma parte del mensaje al igual que los gestos del emisor. El tono del mensaje puede transmitir sarcasmo, ironía, alegría y un sinfín de connotaciones que complementan el mensaje.

Las técnicas de *deep learning* han permitido evolucionar las técnicas de procesamiento del lenguaje gracias a su potencia para representar el proceso de aprendizaje y el propio conocimiento. En este ámbito, una de las mayores innovaciones ha sido **considerar el vecindario de una palabra como fuente muy relevante para determinar el significado concreto de esa palabra**. De esta forma, gracias al *deep learning* se ha conseguido crear redes que son capaces de categorizar una palabra por proximidad, usando redes de reducción de dimensionalidad con algoritmos no supervisados. Estos algoritmos se conocen con el nombre de **embedding** y permiten comprimir una red de palabras representada como vectores equidistantes de un espacio multidimensional K (por ejemplo, usando one-hot encoding) a un espacio multidimensional N menor que K que representa la distancia semántica entre diferentes palabras. Una de las principales herramientas para conseguir esto es **Word2vec**. Word2Vec es una red preentrenada que agrupa palabras de forma compacta.

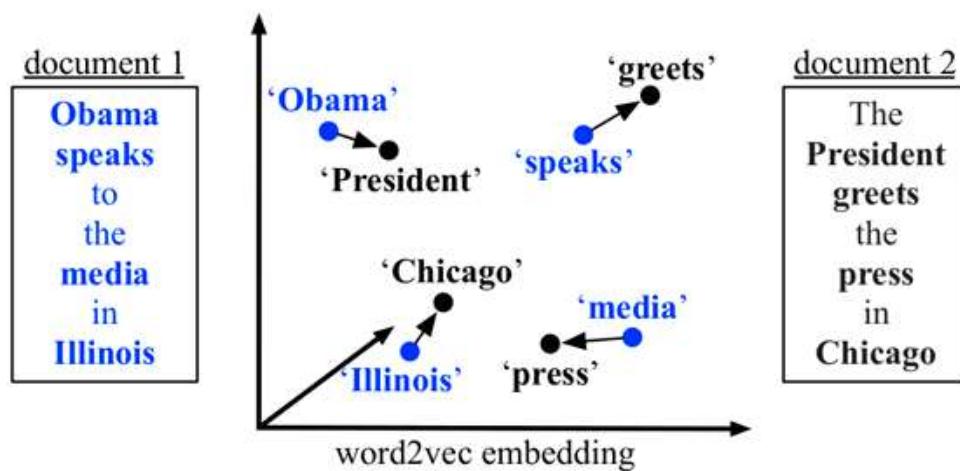


Figura 5. Ejemplo de uso de Word2vec con ciertas palabras.

Fuente: <https://www.springboard.com/blog/introduction-word-embeddings/>

Usando este tipo de herramientas se consigue codificar palabras que, usando redes de neuronas profundas, son capaces de realizar multitud de tareas.

De forma concreta, se han mostrado **aplicaciones muy relevantes** en campos como:

- ▶ Clasificación de texto y documentos: incluyendo clasificar mensajes según la tonalidad positiva o negativa mostrada, filtrado de correo electrónico no deseado, clasificación automática de noticias, sentencias, informes, según su categoría, etc.
- ▶ Modelado del lenguaje: con el objetivo de sugerir correcciones gramaticales, sugerencias de palabras, etc.
- ▶ Reconocimiento del habla: posibilitando, por ejemplo, generar de forma automática los subtítulos de un vídeo.
- ▶ Descripción automática: posibilitando describir una imagen, vídeo... de forma automática.
- ▶ Traducción automática.
- ▶ Resumen automático de documentos: lo que permitiría, por ejemplo, resumir de forma automática una sentencia judicial.
- ▶ Respuesta automática: faculta a la máquina para interactuar con el humano proporcionando de forma directa la pregunta a una pregunta formulada.
- ▶ Generación automática de texto (GPT 2/3).

10.7. Proyectos de investigación sobre sistemas cognitivos

A nivel empresarial, el objetivo principal de la computación cognitiva es apoyar el proceso de toma de decisiones presentando el conocimiento necesario en el momento oportuno de la forma adecuada. Gracias a estas aplicaciones, las

compañías pueden diferenciar sus productos y servicios y obtener ventaja sobre sus competidores.

10.8. Referencias bibliográficas

Hurwitz, J., Kaufman, M. y Bowles, A. (2015). *Cognitive Computing and Big Data Analytics*. Estados Unidos: Wiley.

The rise of cognitive computing

Dickinson, M. (9 de noviembre de 2016). The rise of cognitive computing. *BBC Future*. Recuperado de <http://www.bbc.com/storyworks/future/an-intelligent-future/the-rise-of-cognitive-computing>

Reportaje de la *BBC* sobre la computación cognitiva, aplicaciones y futuro.

Watson and the Jeopardy! Challenge

IBM Research (6 de noviembre de 2013). *Watson and the Jeopardy! Challenge* [Vídeo]. YouTube. Recuperado de <https://youtu.be/P18EdAKuC1U>



Este vídeo nos permite conocer un poco más a fondo la historia de Watson y su participación en el programa *Jeopardy*.

1. En lo que se refiere a la relación entre *big data* y computación cognitiva:

 - A. Las soluciones cognitivas ayudan a que las tecnologías *big data* sean más eficientes.
 - B. Los sistemas cognitivos se alimentan de información *big data* para tomar las mejores decisiones.
 - C. Ambos usan las mismas tecnologías.
 - D. La computación cognitiva abarata los costes de las tecnologías *big data*.
2. Los tres principios fundamentales que componen un sistema cognitivo son:

 - A. Aprendizaje, modelado y análisis estadístico.
 - B. Inferencia, modelado y módulo de procesamiento de lenguaje natural.
 - C. Aprendizaje, modelado y módulo de procesamiento de lenguaje natural.
 - D. Aprendizaje, modelado y generación de hipótesis.
3. Son elementos de un sistema cognitivo:

 - A. Una infraestructura de almacenamiento.
 - B. Una capa de servicios analíticos.
 - C. Una capa de servicios de visualización.
 - D. Todos los anteriores.
4. Las ontologías sirven para:

 - A. Representar formalmente el conocimiento asociado a un dominio con las relaciones entre conceptos.
 - B. Analizar grafos.
 - C. Acceder a datos *big data*.
 - D. Ninguno de los anteriores.

5. Para diseñar un sistema cognitivo se debe tener en cuenta:

- A. Organizar los datos de forma eficiente.
- B. Permitir la ingesta de todos los datos necesarios.
- C. Asegurar el cumplimiento de la normativa vigente.
- D. Todas las anteriores.

6. ¿Qué es el embedding?

- A. Usar tecnologías *deep learning* para clasificar palabras en grupos gramaticales.
- B. El uso de *deep learning* para agrupar texto como vectores de distancia de forma compacta.
- C. Emplear *deep learning* para generar texto de forma automática.
- D. Ninguna de las anteriores.

7. Las bases de datos NoSQL:

- A. Son adecuadas para datos estructurados.
- B. Son las más lentas de todas.
- C. Son adecuadas para datos no estructurados.
- D. Prohíben el uso de SQL o variantes parecidas.

8. Los sistemas de archivos distribuidos para el *big data*:

- A. No replican los datos, pues es ineficiente.
- B. Replican la información para agilizar el acceso.
- C. Tienen un tamaño de bloque similar a los sistemas operativos básicos.
- D. Todas la anteriores son ciertas.

9. El sistema de IBM Watson:

- A. Es un ejemplo muy primitivo de lo que se puede conseguir con la computación cognitiva.
- B. Es una innovación total que ya está preparado para usarse a gran escala independientemente del dominio en multitud de entornos, aunque todavía hay un campo grande de mejora e investigación.
- C. Ha resultado un fracaso total.
- D. No tiene aplicaciones comerciales.

10. Los sistemas de percepción computacional han mejorado mucho en los últimos años principalmente debido a...

- A. Al descubrimiento de nuevos algoritmos de filtrado de imagen.
- B. A la mejora de los modelos basados en *deep learning* y el aumento de la capacidad computacional.
- C. A un aumento de la capacidad de las cámaras y sensores disponibles.
- D. Ninguna de las anteriores.

Investigación en Inteligencia Artificial

Investigación en computación bioinspirada

Índice

Esquema	3
Ideas clave	4
11.1. Introducción y objetivos	4
11.2. Introducción a la computación bioinspirada	5
11.3. Algoritmos bioinspirados	9
11.4. Sistemas emergentes	27
11.5. Proyectos de investigación sobre computación bioinspirada	33
11.6. Referencias bibliográficas	34
A fondo	35
Test	37

Esquema

COMPUTACIÓN BIOINSPIRADA			
Algoritmos evolutivos	Sistemas emergentes bioinspirados	Machine learning	Optimización multiobjetivo
Algoritmos genéticos Se usan para optimización de datos discretos	Colonias de hormigas Optimización en entornos dinámicos con computación en enjambre	Redes de neuronas Tanto clasificación como regresión o extracción de características	Optimización por frente de Pareto - NGA II - Optimización por enjambre de partículas - Optimización multiobjetivo
Estrategias evolutivas Se usan para la optimización de datos continuos	Optimización por enjambre de partículas Optimización en entornos dinámicos con computación en enjambre		
Programación genética Se usa para generar programas de forma automática			
Sistema inmunitario artificial Se usa para optimizar en entornos dinámicos			

11.1. Introducción y objetivos

En este tema se explica a alto **nivel las diferentes técnicas y algoritmos de inteligencia artificial que han surgido con inspiración en la biología**. Así pues, en este tema presentaremos un conjunto de técnicas y algoritmos o aproximaciones que se han inspirado de alguna forma en sistemas presentes en la naturaleza. Este campo es inmenso y solo queremos dar unas pinceladas sobre el tema, para que tengas un concepto general sobre este tipo de tecnologías.

Tradicionalmente, los humanos hemos intentado imitar el comportamiento de ciertos individuos o procesos naturales para el avance tecnológico. No siempre se han adaptado los sistemas biológicos tal cual existen en la naturaleza (por ejemplo, el hombre no vuela como las aves), pero su estudio siempre ha sido un campo de inspiración para el avance científico y técnico.

No hay que confundir la **computación biológica con la biología computacional**. La computación biológica estudia cómo podemos utilizar elementos de naturaleza biológica para obtener mejores resultados a problemas en la computación. Sin embargo, la biología computacional es el uso de técnicas de computación en el estudio de la biología molecular, procesado de ADN, etc. Las ideas detrás de ambos enfoques son similares pero la finalidad no tiene nada que ver.

De forma detallada, los **objetivos** que persigue esta unidad son:

- ▶ Ser capaz de entender cómo funcionan estos sistemas bioinspirados.
- ▶ Conocer los tipos de algoritmos y problemas que existen y que pueden resolverse con ellos.
- ▶ Identificar las principales técnicas y algoritmos, en especial, los relacionados con la computación evolutiva.
- ▶ Conocer los proyectos que se pueden crear o resolver usando computación bioinspirada.

11.2. Introducción a la computación bioinspirada

Como hemos definido anteriormente, la computación bioinspirada o computación biológica es un **área de la inteligencia artificial que se apoya en soluciones inspiradas en la biología para resolver problemas computacionales**. La variedad de algoritmos y de técnicas es enorme y, además, muchas de ellas están siendo la punta de lanza de la IA (inteligencia artificial) actualmente, obteniendo unos resultados asombrosos. Por la inmensidad del área a abarcar, vamos a centrarnos en los **algoritmos de computación evolutiva** y después mencionaremos otros enfoques con menor profundidad.

Para comprender la computación evolutiva tenemos que acudir a las bases de la teoría de la evolución **propuesta por Charles Darwin** (1809-1882) y su **teoría de la selección natural**.

Esta teoría surge después de visitar las islas Galápagos en una expedición geológica. En ellas Darwin observó algunas anécdotas biológicas muy interesantes que le hizo interesarse por la evolución de las especies, ya que encontró similitudes con especies de América pero a la vez diferencias. Darwin en realidad no era biólogo sino geólogo, y no fue hasta este momento cuando comenzó a interesarse más a fondo en temas de biología y de especies.

Por ejemplo, se percató de que, en cada isla, un grupo de especies diferentes de pájaros, muy similares en color, aspecto y tamaño, tenían modificaciones en su pico adaptadas a las necesidades de obtención de alimento, que eran diferentes en cada isla. Por ejemplo, unos tenían el pico más largo, o más curvo o de diferente tamaño.

Tardo varios años en recopilar toda la información que necesitaba para formular su teoría. En ella, describe que las especies evolucionan unas de otras, ramificándose a partir de padres comunes. Algunas de estas especies terminan en extinción y otras volviéndose a ramificar en nuevas especies, lo que le confiere una forma de árbol. También relacionó el desarrollo embrionario de un individuo con el desarrollo evolutivo de sus estirpes (esto lo extrajo de los fósiles que recolectó en su vida como geólogo), y que las islas u otras barreras geológicas hacían que las especies se adaptaran al entorno de esas zonas.

Darwin sabía que las especies evolucionaban y que el medio en el que se desarrollaban influía, pero no sabía el mecanismo de cómo evolucionaban estas especies. Esto se descubrió paralelamente por **Mendel** (1957-1968) que describió cómo funcionaba la mutación haciendo unos experimentos con plantas de guisantes.

De esta forma, años después con el descubrimiento del ADN, se conformó la teoría completa que sustenta lo que hoy conocemos como la **teoría de la evolución**.

La teoría de la evolución dice que la presión ambiental (reproductiva, por obtener alimentación, competencia con otros animales) es el vehículo del cambio en las especies y el mecanismo del cambio es el cruce genético y la mutación. Es decir, el cruce entre dos especies provoca que parte del ADN de una de ellas se mezcle con parte del ADN de la otra. De forma que el nuevo individuo no tiene exactamente el mismo ADN que los progenitores, sino una mezcla de ambos. Además de esto, otro mecanismo para el cambio es la mutación, que consiste en pequeños cambios en los genes que se producen o no, de forma aleatoria en cada réplica del ADN. De esta forma, un nuevo individuo puede tener un ADN diferente al de sus progenitores. Algunos de estos cambios producen individuos más aptos para el medio en el que se desenvuelven. Mientras que otros cambios hacen a estos individuos menos aptos para

su supervivencia. La presión selectiva hace que no todos los individuos acaben transmitiendo sus genes a sus descendientes, bien porque no consiguen aparearse (no son el macho dominante con ese derecho en una manada, por ejemplo) o bien porque mueren antes de poder hacerlo. De esta forma, la probabilidad de que un cambio genético que perjudique la supervivencia de un individuo en un entorno concreto se transmita a los descendientes es menor que la probabilidad de que lo haga uno beneficioso, ya que este tendrá más probabilidades de reproducirse. Y el tiempo hace el resto. De forma que, en sucesivas generaciones, los individuos con modificaciones que le permiten adaptarse mejor al medio han proliferado y han transmitido estas modificaciones a la mayoría de la población.

Si miramos esto desde un punto de vista computacional, lo que hace la selección natural es un **método de optimización**. La selección natural optimiza a los individuos para resolver una tarea concreta. Que en el caso de la biología es sobrevivir y reproducirse, con todo lo que ello implica. Si una especie no se adapta a los cambios desaparece, lo que da lugar a las extinciones, ya sea porque no se adapten a un cambio repentino del medio, ya sea porque una versión mejorada de ellos mismos los sustituya en su nicho ecológico.

Ahora extrapolemos esto a un problema de computación. ¿Podemos generar una solución a un problema de computación y hacerla evolucionar para que encuentre la forma de resolver ese problema de la forma más óptima posible?

De esta pregunta surge la **computación evolutiva**.

Computación evolutiva

La computación evolutiva es una **rama de la inteligencia artificial que busca resolver problemas de optimización, inspirándose en los mecanismos de evolución biológica**. Existen diferentes métodos y algoritmos que se pueden englobar dentro de este campo como los algoritmos genéticos, las estrategias evolutivas, la programación genética etc.

Ya los desgranaremos en el resto del tema. Pero por ahora vamos a asentar las bases de los conceptos en los que se sustentan todas ellas.

Podemos definir **gen como la unidad mínima de transferencia genética**. Es decir, la unidad mínima que es capaz de sintetizar una macromolécula con función celular. Estos genes tienen memoria y esto permite la herencia entre descendientes.

El **genotipo** es el conjunto de todos los genes de un individuo.

El **fenotipo** es la expresión del genotipo en un individuo concreto, dentro de un ambiente concreto. Es decir, la implementación de ese genotipo.

Ejemplo: hay un gen que codifica el color de los ojos de un individuo. Su genotipo sería la secuencia de bases del gen en concreto y su fenotipo el color que genera al final.

Con estas definiciones, lo que buscan los algoritmos evolutivos es crear una analogía con la evolución natural. Así pues, habrá que representar una cadena de ADN con información mínima útil para realizar una tarea (gen). Estos genes se reproducirán (o no) cruzándose con otros genes y ocasionalmente se cambiará la información de alguno de estos genes mediante una mutación. Este proceso se repetirá hasta encontrar un individuo lo suficientemente bueno como para que resuelva la tarea que debe realizar.

¿Y cómo sabemos si resuelve la tarea? Pues ejecutando dicha tarea y «puntuando» lo bien que lo ha hecho. Esta puntuación es una valoración de lo adaptado que está el individuo al medio.

Aplicaciones

Las aplicaciones de este tipo de algoritmos evolutivos son muchas y están enmarcadas dentro de los algoritmos de optimización. Son capaces de encontrar una forma eficiente de realizar una tarea. Vistos así puede el lector pensar que no son muy

diferentes a un algoritmo de búsqueda, que busca la mejor solución entre las posibles soluciones y es cierto. La mayoría de estos algoritmos pueden ser considerados algoritmos de búsqueda, pero tiene ciertas peculiaridades que los hacen especialmente efectivos en determinados entornos. Y es que son bastante buenos en encontrar soluciones subóptimas en problemas que tienen una alta complejidad y donde algoritmos como A* no son viables. También son bastante robustos a mínimos locales en las búsquedas debido a su componente aleatorio. Por otro lado, su complejidad computacional viene determinada principalmente por lo **costoso que sea evaluar la calidad de la solución**. Es decir, la calidad que cada individuo tiene para resolver el problema. Si es muy costoso hacerlo, estos algoritmos suelen ser demasiado costosos ya que normalmente necesitan poblaciones grandes de individuos (o soluciones), y habrá que ejecutar la función de evaluación para cada individuo y para cada generación.

Más en concreto, se han utilizado en diferentes campos como:

- ▶ La robótica.
- ▶ La optimización de procesos.
- ▶ Fabricación industrial.
- ▶ Economía.
- ▶ Aprendizaje automático en por ejemplo la optimización de algoritmos de *machine learning* como redes de neuronales.

A continuación, veremos algunas de estos métodos con más detalle.

11.3. Algoritmos bioinspirados

A continuación, explicaremos las principales técnicas englobadas en los denominados algoritmos bioinspirados. Haremos más hincapié en los más comunes y utilizados,

basándonos en las explicaciones de la selección natural que hemos realizado en el anterior apartado.

Algoritmos genéticos

Los algoritmos genéticos son unos **algoritmos evolutivos que se basan en los principios explicados de la selección natural**. Estos algoritmos pretenden resolver problemas de optimización y búsqueda representando la solución en forma de genes. Estos se cruzan y mutan en sucesivos ciclos para conseguir individuos que son capaces de resolver el problema cada vez mejor.

Así pues, debemos obtener la mejor solución dentro del espacio de soluciones que resuelva un problema planteado, atendiendo a un criterio de evaluación del rendimiento de la solución obtenida. Esto se puede aplicar por tanto a:

- ▶ Búsqueda en un espacio de estados.
- ▶ Clasificación: cuál es el procedimiento que mejor clasifica.
- ▶ Predicción: encontrar el procedimiento que tenga menos fallos.
- ▶ Control: procedimiento que maximice una tarea de control.

En los algoritmos genéticos es muy importante **la codificación**. Es la principal tarea del diseñador del algoritmo. Los algoritmos genéticos deben codificar las diferentes soluciones como un vector o matriz de genes. Estos vectores de genes se los denomina individuos y codifican una solución particular dentro del espacio de soluciones.

Una **población** es el conjunto de soluciones (individuos) que disponemos en un momento dado. Inicialmente esta población se puede generar de forma aleatoria o con algo más de conocimiento del dominio. A cada uno de estos individuos se le aplica una serie de **operadores genéticos** que transforman una población en otra.

¿Y cuáles son esos operadores genéticos?

- ▶ Selección. Se selecciona a los mejores candidatos para la reproducción. Se genera una población intermedia del mismo tamaño que la original normalmente. Hay diferentes técnicas a la hora de seleccionar cuáles son los individuos candidatos para reproducirse.
- ▶ Cruce. Se seleccionan de dos en dos los individuos de esa población intermedia y se cruzan de alguna forma. Hay diferentes técnicas de cruce como cruce simple por la mitad, cruce uniforme donde cada gen se cruza con su homónimo en la pareja, cruzar por múltiples puntos, etc.
- ▶ Mutación. Se modifica de forma aleatoria mediante una probabilidad de mutación los genes de los individuos recién cruzados.
- ▶ Inversión. Se invierten algunos de los genes (en caso de que la codificación sea binaria), también se puede coger un grupo de ellos y darles la vuelta.

Para finalizar tenemos una función que se denomina **función de fitness** que evalúa las soluciones (los individuos) para determinar cuáles son los más aptos, es decir, los que mejor resuelven el problema. Esta función de fitness debe resumir el comportamiento del individuo con un número para poder comparar unos con otros.

Los algoritmos genéticos son muy eficientes a la hora de encontrar soluciones más o menos optimas a un problema. Y son propensos a una convergencia prematura en un mínimo local. Sobre todo, si solo usamos cruce. Esto es debido a que **ciertos rasgos genéticos se pierden** (a esto se le denomina perdida de alelos) en los cruces con lo que se pierde variabilidad genética. La mutación aquí es un factor importante para mantener esa variabilidad genética. Después de unas cuantas generaciones, todos los individuos tienen un valor de adecuación o fitness similar, por lo tanto, deja de haber **presión selectiva** y, por tanto, se deja de evolucionar (todos tiene la misma probabilidad de reproducirse). Para evitarlo, **surgen diferentes técnicas** que favorecen que aumente o disminuya la presión selectiva para dar oportunidades a que rasgos óptimos en individuos no óptimos puedan formar parte de soluciones mejores y no se pierdan por el camino. Para ellos hay diferentes tipos de selección.

Tipos de selección

La selección por **ruleta** es similar a tirar un dado donde los individuos con mejor fitness tienen más probabilidades de acertar. Acertar en este caso es reproducirse.

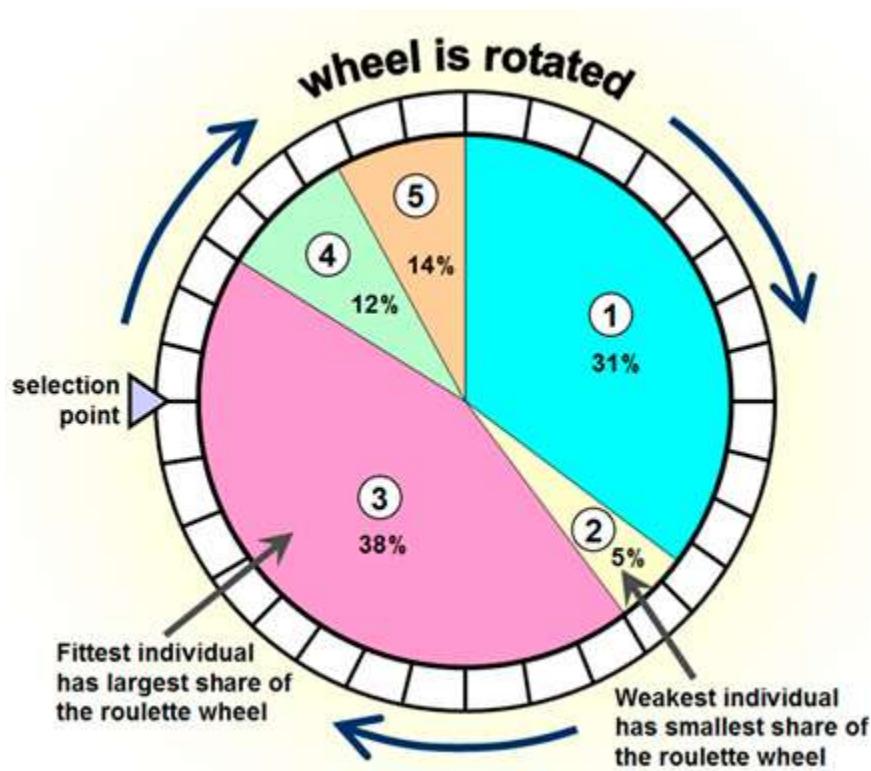


Figura 1. Selección por ruleta.

Fuente: <https://stackoverflow.com/questions/23183862/genetic-programming-difference-between-roulette-ranking-and-tournament-selection>

En la selección **jerárquica** se ordena a la población de mayor a menor según su fitness y se selecciona el individuo en proporción al rango que ocupa calculando una probabilidad.

En la selección por **torneos** se eligen K elementos de la población y se selecciona aquel que tenga mejor fitness. Normalmente este tipo de selección relaja la presión selectiva lo que hace que la diversidad genética sea mayor.

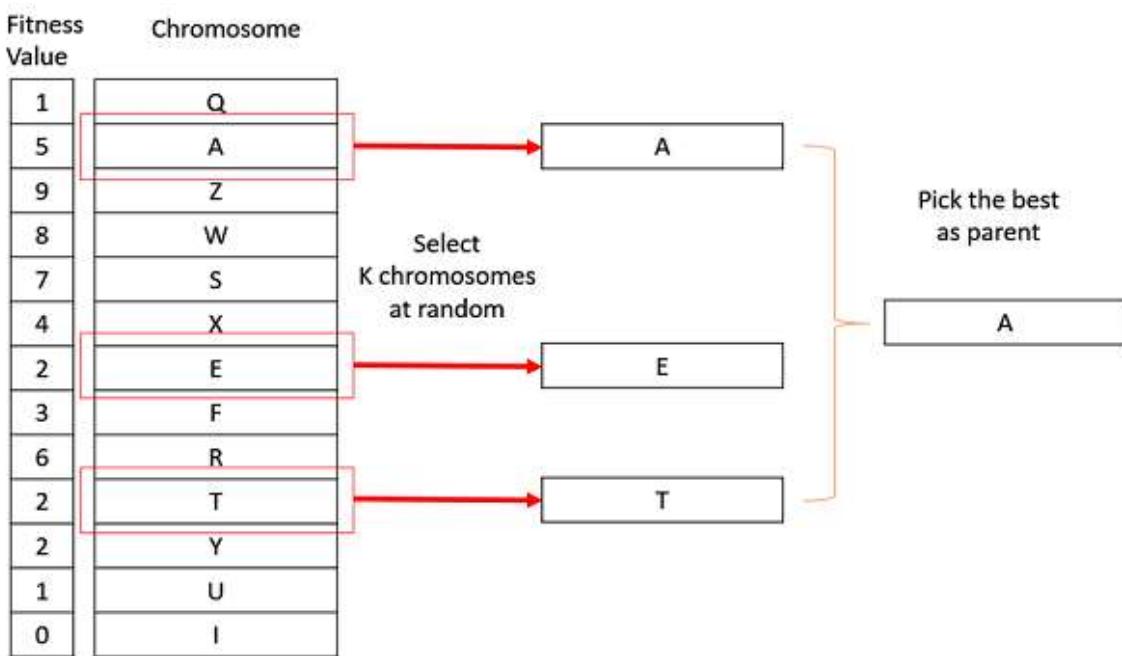


Figura 2. Selección por torneos.

Fuente: <https://jeongchul.tistory.com/573>

La mutación es muy importante en los algoritmos genéticos. Sin ella, estos algoritmos convergerían muy rápidamente a un mínimo local. Además, la mutación permite evitar la pérdida de información genética ya que puede recuperar alelos perdidos mediante cruces. También permite aleatorizar las búsquedas y así poder encontrar nuevos mínimos locales. Normalmente en las primeras fases se potencia más la exploración (mutación) y después la explotación (cruce), aunque para salir de un mínimo se puede dar algún arreón de mutación en algún momento concreto. En cualquier caso, la probabilidad de mutación puede ser variable bajo algún criterio que fije el diseñador del algoritmo.

Y, por último, la función de **fitness** que es la que nos determina cómo de bueno es un individuo y la que guía la búsqueda. Normalmente se busca minimizar la función, aunque también se puede maximizar en función de cómo lo consideres.

Ejemplo

Vamos a ver un ejemplo de cómo codificar el problema del viajero usando algoritmos genéticos y realizaremos una iteración del mismo.

Imaginemos que queremos codificar cuatro ciudades, podemos utilizar 2 bits.

Codificación
00
01
10
11

La codificación de un individuo puede ser 2 bits y un individuo codificarse así:

1 => 3 => 2 => 4 = 00100111

Generamos una población aleatoria:

10101010
11011001
00010101
01010101

Después valoramos con la función de fitness (la distancia recorrida):

10101010 => 3
11011001 => 2
00010101 => 4
01010101 => 2

Si usamos ruleta decimos que la probabilidad de ser elegido será más alta para el individuo 2 y 4. Supongamos que se eligen ambos para el cruce. Se cruzan con cruce simple.

11011001 x 01010101 => 11010101

Se muta con una probabilidad 11010101 => 11010111

Se repite hasta tener cuatro nuevos individuos.

Se repite el proceso hasta que no se mejore más.

Esta codificación tiene un problema y es que permite visitar la misma ciudad dos veces. Estos individuos serían erróneos. ¿Qué sucede con ellos? Se les pueden penalizar fuertemente con el fitness o impedir que se generen. Pero la solución más eficiente es codificar de otra forma el problema.

Podemos codificar el problema de forma que ordenamos las ciudades de menor a mayor según un criterio arbitrario. Cuando codificamos un número, en binario, codificamos la posición de la ciudad en la lista. De forma que 0101 no significa que vayamos de la ciudad 2 a la 2 sino de la 2 a la 3 (la numero 2 de la lista una vez que hemos quitado la segunda). De esta forma podemos codificar el sistema de forma más eficiente (con menos bits) y evitar individuos erróneos. Piensa que cuanto más grandes sean los individuos, el espacio de búsqueda será mayor.

Otra de las cuestiones que debemos tener en cuenta es que, en el cruce, normalmente va implícito una relación entre las diferentes posiciones de los genes. Si la codificación no mantiene esa relación de posición, el cruce guía peor la búsqueda y esta se vuelve más aleatoria.

El algoritmo se muestra en la siguiente figura:

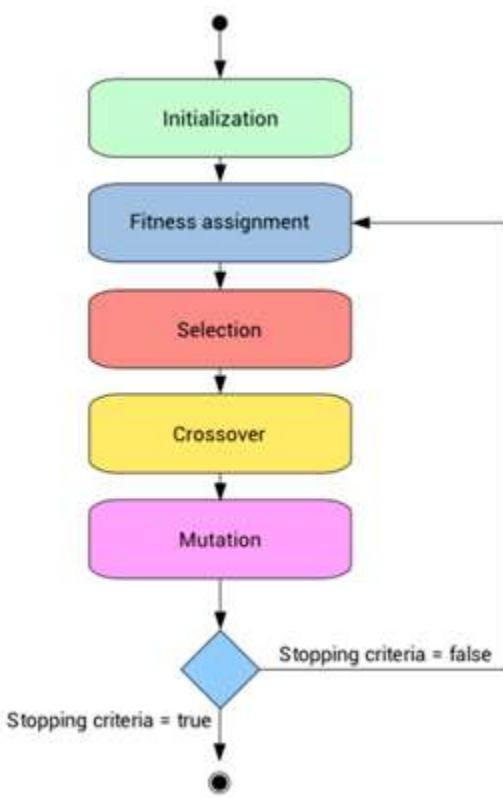


Figura 3. Diagrama de un algoritmo genético.

Estrategias evolutivas

Las estrategias evolutivas son similares a los algoritmos genéticos. La principal diferencia es que **opera con una codificación de valores continuos**. Esto implica que la mutación debe cambiar ya que no existe el concepto de mutación discreta, es decir, no podemos elegir una de las diferentes alternativas del valor del gen, ya que ahora hay infinitos valores. La mutación va a seguir una distribución normal de probabilidad. Esto hace que la heurística de búsqueda se establezca precisamente en el operador de mutación y no en el cruce como en los algoritmos genéticos.

Se codifican los individuos con un vector de codificación x_i y un vector de varianzas σ_i que indica lo alejado que se encuentra de la solución del problema. Pero no sabemos cuál es la solución óptima, por lo tanto, este vector de varianzas en realidad son

predicciones de la distancia al óptimo. Este vector también se evoluciona, por lo tanto, es de esperar que esta predicción mejore con el tiempo.

Los diferentes algoritmos de estrategias evolutivas siguen la notación $(\mu / \rho, \alpha)$ donde:

μ Tamaño de la población.

ρ número de padres que se seleccionan para recombinarse.

α número de individuos en la descendencia.

Lo más simple, el modelo (1+1) - EE

Este es el **algoritmo más básico y es un método de escalada puro**. Un individuo se denota de la siguiente forma $P = x, \sigma$ donde x y σ son el vector de codificación y el de varianza.

En este algoritmo se genera una población intermedia mutando al individuo y después se evalúa. Si es mejor se reemplaza y si no se mantiene el actual. Como solo hay un individuo no hay cruce.

La mutación del vector de soluciones se realiza usando una distribución normal de media 0 y varianza σ .

$$x'_i = x_i + \sigma'_i \cdot N(0,1)$$

Esto tiene sentido debido a que la varianza al principio será muy grande (la distancia a la solución) y los saltos que se produzcan serán más grandes. A medida que nos vayamos acercándonos a la solución, los saltos son más pequeños porque estamos más cerca (exploración vs. explotación).

Para actualizar la varianza se usa la regla de 1/5. Dicha regla lleva una estadística del número de veces que la población ha cambiado y se ha encontrado una mejor solución. Para ello establecemos un valor del número de generaciones que vamos a tener en

cuenta y cuántas veces se ha cambiado en esas generaciones. Esto se denota como γ_s^t donde t es el número de veces que se ha realizado reemplazo y s el número de generaciones anteriores que tenemos en cuenta. Si la ratio entre ellas varía de 1/5 se incrementa o se decrementa el valor de la varianza siguiendo la siguiente regla:

$$\begin{aligned}\sigma' = \sigma \cdot const_d &\leftrightarrow \gamma_s^t < 1/5 \\ \sigma' = \sigma \cdot const_i &\leftrightarrow \gamma_s^t > 1/5\end{aligned}$$

Donde $const_d$ y $const_i$ son dos valores. El primero menor que 1 y el segundo mayor que 1.

De esta forma vamos evolucionando el algoritmo hasta que no cambie sustancialmente en una iteración o hasta que encuentre la solución (el error sea admisible).

Este es el ejemplo más simple, pero se pueden extender a modelos con múltiples individuos.

Modelos con múltiples individuos

Estos modelos comparten gran parte de las ideas del ejemplo simple, pero incorporan el operador de cruce. El cruce de dos individuos se realiza haciendo la media aritmética de cada uno de ellos. También hay que cruzar el vector de varianzas, pero este se debe hacer calculando la raíz cuadrada de la suma de las varianzas de los dos individuos.

$$x' = \frac{1}{2} \cdot (x_1 + x_2)$$

$$\sigma' = \sqrt{\sigma_1 + \sigma_2}$$

El ámbito de aplicación de estos algoritmos es el mismo que los algoritmos genéticos, y se utilizan en vez de estos cuando el sistema a optimizar maneja valores continuos. Así pues, cualquier aplicación realizada con algoritmos genéticos con números reales puede aplicarse a las estrategias evolutivas. Se han demostrado especialmente útiles

por ejemplo en el entrenamiento de redes de neuronas, modificando sus pesos mediante estrategias evolutivas.

Programación genética

La programación genética es un algoritmo evolutivo derivado de los algoritmos genéticos propuestos en 1989 por John Koza.

La idea de la programación genética es **evolucionar programas completos**. Para poder hacer esto, debemos crear un lenguaje de programación más sencillo que nos permita encontrar una solución a un problema concreto que queremos obtener. De esta forma a la programación genética debemos darle los operadores y funciones necesarios que estimemos que va a necesitar el algoritmo de la solución. Por ejemplo, si vamos a resolver una ecuación, pues introduciremos operadores, variables, constantes, etc.

La idea del algoritmo consiste en generar una población de programas que intentan resolver el problema que se plantea. En la figura se muestra un ejemplo de un individuo.

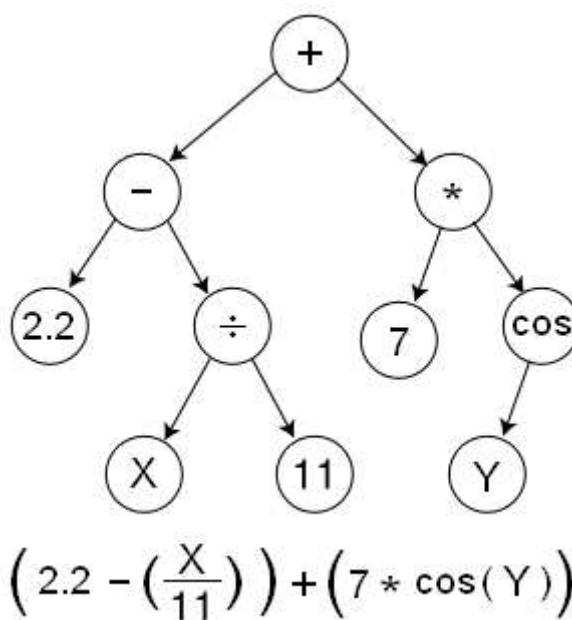


Figura 4. Ejemplo de un individuo que calcula una función usando programación genética.

Fuente: https://es.wikipedia.org/wiki/Programaci%C3%B3n_gen%C3%A9tica

Si queremos hacer un programa que calcule una ecuación, el algoritmo funcionaría de la siguiente forma:

- ▶ Se genera un conjunto de ecuaciones aleatorias en forma de árbol (notación polaca o polaca inversa, por ejemplo).
- ▶ Cada ecuación del conjunto de ejemplos contendrá un valor para las variables que se necesitan.
- ▶ Se sustituyen dichos valores en la ecuación representada por el individuo a evaluar, y así se obtiene un valor de la ecuación.
- ▶ Se compara el valor con el esperado y si es el mismo, el individuo resuelve la ecuación y si no es así, la diferencia se utilizará para el cálculo del fitness del individuo.

Se aplican las mismas ideas que en los algoritmos genéticos. Por ejemplo, el **cruce** consistirá en coger un subárbol de cada individuo e intercambiarlos.

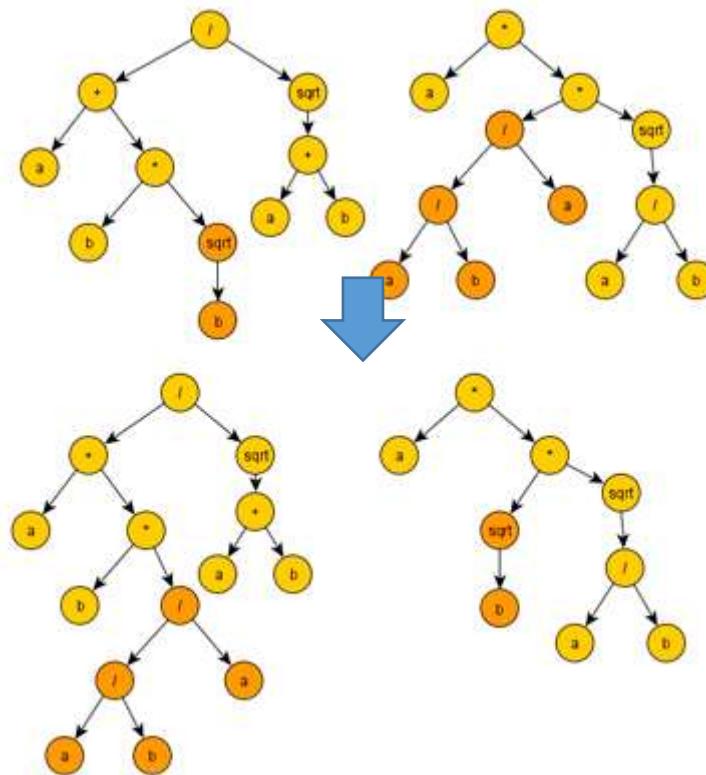


Figura 5. Cruce en programación genética.

La **mutación** consiste en cambiar un operador por otro o un valor o variable por otro. También se puede mutar un subárbol generándolo aleatoriamente.

Redes de neuronas

Las redes de neuronas son otros de los sistemas más exitosos con inspiración biológica que se utilizan actualmente. Ya hablamos de ellas anteriormente, por lo que no nos extenderemos demasiado aquí. Simplemente recordaremos su inspiración biológica, basada en la organización del cerebro animal, donde **la potencia de cálculo principal se encuentra en las reacciones químicas que se producen en las interconexiones de las células o neuronas**. Así pues, las redes de neuronas artificiales llevan el cálculo a las interconexiones de las neuronas y no tanto a estas, que simplemente procesan mínimamente la señal recibida con funciones simples como la sigmoidal.

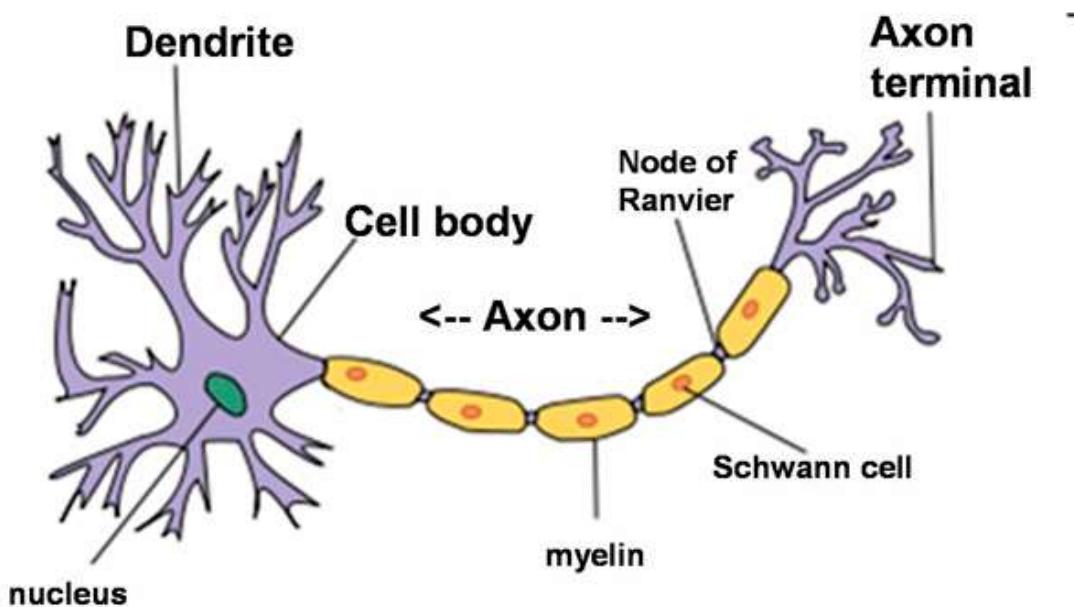


Figura 6. Esquema de una neurona.

Su uso, como sabemos con el éxito del *deep learning* está muy extendido actualmente y puede utilizarse en multitud de dominios, debido a que se le considera un aproximador universal de funciones.

Sistema inmunitario artificial

Los sistemas inmunitarios artificiales (AIS) son una clase de máquinas de aprendizaje basado en reglas computacionalmente inteligentes, inspiradas en los principios y procesos del sistema inmunitario de los seres vivos. Esos algoritmos son modelados a partir de las características de aprendizaje y memorización del sistema inmunitario y están enfocados en la resolución de problemas, de forma similar a los algoritmos genéticos.

Las técnicas comunes están inspiradas en teorías inmunológicas que explican el funcionamiento y el comportamiento del sistema inmunitario adquirido. Existen diferentes técnicas, pero nos centramos en la más común, el **clonal selection algorithm o algoritmo de selección clonal**.

Los algoritmos de selección clonal son una clase de algoritmos inspirados en la teoría de selección clonal de la inmunidad adquirida, que explica cómo los linfocitos B y T mejoran su respuesta ante antígenos con el tiempo mediante la maduración de la afinidad. Son interesantes porque pretenden obtener las características del sistema inmunitario que son:

- ▶ Tolerante a fallos.
- ▶ Robusto.
- ▶ Dinámico.
- ▶ Descentralizado.
- ▶ Adaptativo.
- ▶ Autoprotegido.
- ▶ Diverso.

¿Cómo funciona el sistema inmune?

Cuando un animal es expuesto a un antígeno, algunas de sus células derivadas de la médula ósea (linfocitos B) responden produciendo anticuerpos. Cada célula secreta solo un tipo de anticuerpo, que es específico para el antígeno detectado. Al unirse a estos anticuerpos (receptores), y con una segunda señal de células accesorias, como las células T, el antígeno estimula al linfocito B para que prolifere (se divida) y madure en células secretoras de anticuerpos terminales. A diferencia de los linfocitos B, estas células no se pueden clonar. A estas células terminales se la denomina **células de plasma**. Los linfocitos T son los reguladores de esta respuesta de los linfocitos B. En realidad, son los que detectan a los antígenos cuando estos han infectado alguna célula o han sido procesados por macrófagos mediante un patrón de afinidad. Cada linfocito T responde a un antígeno concreto.

La **selección clonal** hace una analogía de las soluciones candidatas a anticuerpos buscando afinidad. La **afinidad se mide como la semejanza con un patrón antígeno** mediante la evaluación de una función, similar a la función de fitness. De forma que el antígeno en realidad es la función de afinidad. Los anticuerpos con mejor afinidad se someten a una clonación proporcional a dicho valor de afinidad y a la mutación de dichos clones. Esta mutación es mayor cuanto menor es la afinidad. Una vez realizada la clonación, estos compiten con la población de anticuerpos existente por ser miembro de la próxima generación.

El algoritmo sería el siguiente:

1. Se genera un conjunto de soluciones P candidatas aleatorias.
2. Se selecciona los n mejores individuos en base a una función de afinidad con el patrón antígeno presentado (el problema a resolver).
3. Se clonian los n individuos un número de veces por cada individuo seleccionado. Este número es proporcional a la afinidad de cada individuo. Se puede utilizar la siguiente ecuación donde β es un parámetro ajustable.

$$N_i = \beta \cdot \frac{N}{\text{affinity}(i)}$$

4. Se mutan en proporción a la afinidad. Si hay menor afinidad, la mutación es más agresiva. Se puede utilizar la siguiente ecuación para calcular la tasa de mutación donde ρ es un parámetro ajustable.

$$\text{mutation rate}(\alpha) = \exp(-\rho \cdot \frac{\text{affinityMin}}{\text{affinityMax}})$$

5. El número de clones se puede realizar siguiendo una distribución normal de media 0 y desviación típica α .
6. Se evalúan de nuevo cada anticuerpo de los generados por clonación.
7. De entre las nuevas soluciones generadas se seleccionan los n mejores y se remplazan en la población original.
8. Los d anticuerpos con menor afinidad son sustituidos por otros anticuerpos generados aleatoriamente.
9. El proceso continúa hasta alcanzar el criterio de parada.

Algunas aplicaciones sobre de estos algoritmos son:

- ▶ Detección de intrusos.
- ▶ Extracción de características.
- ▶ Problemas de optimización.

Optimización multiobjetivo

En un problema de optimización **se tratará de encontrar una solución que represente el valor óptimo para una función objetivo**. Pero en ciencias e ingeniería se dan, en bastantes ocasiones, problemas que requieren la optimización simultánea de más de un objetivo. A esto se le denomina **optimización multiobjetivo**.

Estos problemas se resuelven principalmente con la **optimización de frente de Pareto**. El frente de Pareto es un concepto proveniente de las ciencias económicas que consiste en **realizar una asignación de recursos a los diferentes objetivos de forma que no se pueda mejorar uno sin perjudicar otro**. A esto se le denomina frente de Pareto óptimo o eficiente. Existen multitud de algoritmos que buscan resolver este problema. Uno de los más comunes es NSGA II.

NSGA II (Non dominated sorting genetic algorithm)

Se deben definir unas funciones objetivas que se pretenden maximizar. Normalmente ambas funciones son contrarias, es decir, si mejoramos una empeoramos otra. Por ejemplo, una función de beneficio y una función de coste.

Se denomina **dominancia de una solución sobre otra** si cumple las siguientes condiciones:

- ▶ La primera solución es mejor que la segunda en todos los objetivos.
- ▶ La primera solución es estrictamente mejor que la segunda en al menos uno de los objetivos.

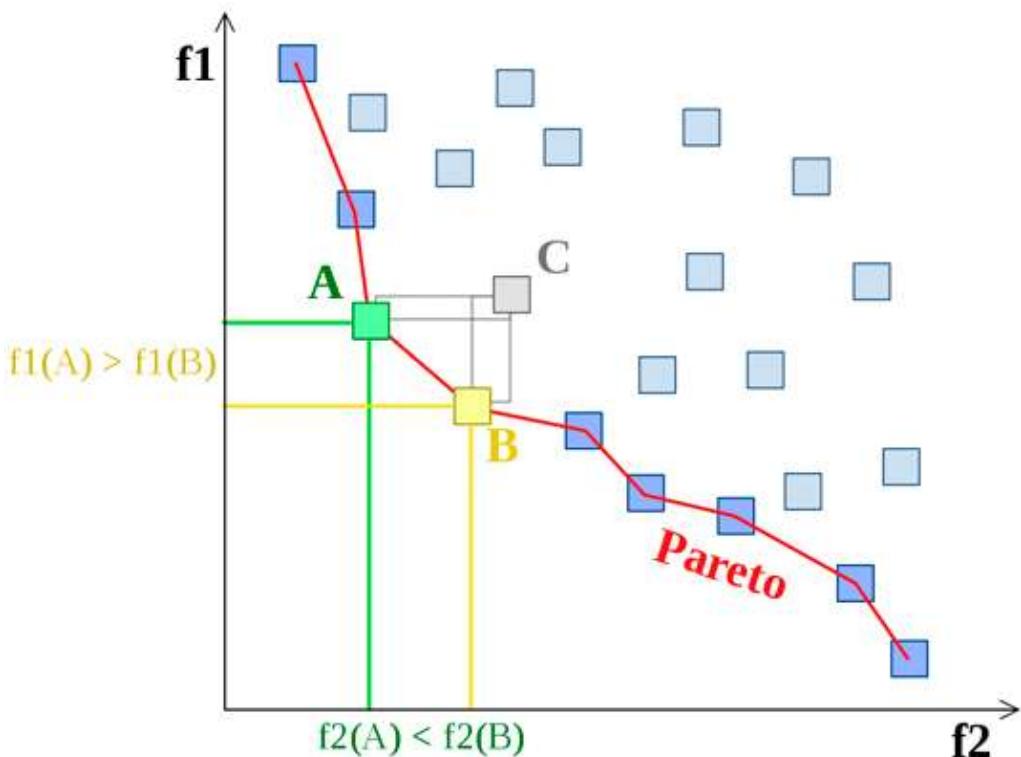


Figura 7. Frente de Pareto. La opción C no está en el frente de Pareto ya que A y B lo dominan. Estos a su vez no son dominados por ningún otro, consecuentemente están en la frontera.

Fuente: https://es.wikipedia.org/wiki/Eficiencia_de_Pareto#/media/Archivo:Front_pareto.svg

Para la optimización de un problema con múltiples objetivos no existe una única solución, sino un conjunto de soluciones no dominadas, en la cual no existen soluciones mejores en todos los objetivos.

NSGA II funciona muy similar a un algoritmo genético normal. Tiene las mismas etapas de selección cruce y mutación. La diferencia es que se debe calcular el frente de Pareto de la solución frente a las diferentes funciones de coste y beneficio. Y los criterios de selección se realizan en función de aquellos individuos que generan un

frente de Pareto más eficiente sin generar soluciones no dominadas. La nueva población es generada a partir de las configuraciones de los frentes no dominados. Esta nueva población empieza a ser construida con el mejor frente no dominado (F_1), continúa con las soluciones del segundo frente (F_2), tercero (F_3) y así sucesivamente. Como la población es de tamaño $2N$, y solamente existen N configuraciones que conforman la población descendiente, no todas las configuraciones podrán ser acomodados en la nueva población. Aquellos frentes que no pueden ser acomodados desaparecen.

Este proceso se repite hasta que se obtiene un individuo que satisface los criterios planteados como criterio de parada.

11.4. Sistemas emergentes

Los sistemas emergentes son **sistemas complejos compuestos normalmente por elementos con un comportamiento muy simple, pero que realizan en conjunto un comportamiento poco esperado**. Algunas técnicas empleadas en computación se han basado en este tipo de sistemas para conseguir solucionar problemas complejos. Se caracterizan por resolver problemas, al menos en apariencia, espontáneamente; es decir, sin recurrir a una inteligencia de tipo centralizado o jerarquizado (descendente), sino de forma ascendente, desde la base, a partir de masas de elementos relativamente no inteligentes.

Existen muchas aproximaciones de este tipo para modelizar comportamientos y simularlos computacionalmente como por ejemplo para **simular el comportamiento de manadas (*flock*) o de multitudes se pueden utilizar sistemas de enjambres (*swarm*)** en los que cada individuo se modela con simples reglas que afectan solo a su entorno local. A parte de estas simulaciones que están inspiradas en cómo se comportan los enjambres y manadas de aves, mamíferos o insectos y cómo evitan, por ejemplo, chocarse, existe un grupo de algoritmos que utilizan

estos conceptos más allá de la modelización de sistemas sino para resolver problemas de optimización. A continuación, veremos algunos de ellos.

Colonias de hormigas

Los algoritmos basados en colonias de hormigas son algoritmos bioinspirados, que se aprovechan de una característica interesante de las colonias de hormigas. Y es que son capaces de encontrar el camino más corto entre dos puntos de forma automática.



Figura 8. Las hormigas son capaces de encontrar el camino más corto.

Fuente: <http://ornewspaper.blogspot.com/2011/11/la-investigacion-de-operaciones-en-la.html>

El sistema funciona a través del concepto de feromonas. Las hormigas artificiales son unas unidades o pequeños agentes que se mueven de forma aleatoria por el espacio de soluciones. Al moverse van dejando una feromona a su paso.

La sustancia emitida puede ser detectada por otras hormigas:

- ▶ La hormiga elige el camino proporcional a la feromona que esté en dicho camino.
- ▶ La feromona del camino se disipa con el tiempo, y si no se vuelve a generar desaparece.
- ▶ Los caminos poco frecuentados casi no tienen posibilidad de ser utilizados, a menos que se encuentre comida y se repita el proceso.

Este comportamiento elemental explica como las hormigas son capaces de seguir caminos cortos hacia lugares relativamente alejados. También son capaces de adaptarse a cambios producidos en el entorno, por ejemplo, a encontrar un nuevo camino si hay un obstáculo.

Lo interesante de las colonias de hormigas es que encontrar el camino más corto entre dos puntos es una **propiedad emergente** de la propia interacción con el entorno.

Llevado esto al terreno computacional, una hormiga no es más que un pequeño agente que se mueve de forma semialeatoria. Tiene mayor probabilidad de moverse hacia una posición donde antes ha pasado una hormiga que haya dejado su feromona depositada, pero no lo garantiza. Esta aleatoriedad le da al algoritmo cierta capacidad de exploración. También hace que pueda adaptarse las soluciones si el entorno ha cambiado. Las hormigas, por tanto, solo utilizan información local para tomar las decisiones y la memoria colectiva que las comunica es la feromona.

Para que el algoritmo funcione debemos **discretizar el espacio de búsqueda**. La emisión de feromona de nuestra hormiga virtual puede ser global en todo el camino o no. Las hormigas tienen también una cierta memoria. Dicha memoria impide que la hormiga vuelva sobre sus pasos.

La feromona que van dejando las hormigas se va disipando con las sucesivas iteraciones. Esto implica que se van olvidando los caminos por donde no pasan las hormigas (se van olvidando soluciones antiguas subóptimas).

Cada hormiga tiene una simple tabla de decisiones que tiene una entrada por cada nodo al que pueda transitar dentro del espacio de estados discreto. Esta entrada es inversamente proporcional al coste de alcanzar el nodo y directamente proporcional a la feromona del tramo. Se puede usar la siguiente ecuación donde α y β son parámetros de ajuste.

$$\text{probabilidad(tramo } i \text{ a } j) a_{ij} = \frac{\text{feromon}a_{ij}^{\alpha} \cdot \frac{1}{\text{coste}_{ij}}^{\beta}}{\sum I \in \text{Nodos}_i \Rightarrow \text{feromon}a_{ii}^{\alpha} \cdot \frac{1}{\text{coste}_{ii}}^{\beta}}$$

Es decir, la probabilidad de elegir un nodo es el producto entre la feromona del nodo y la inversa del coste de dicho nodo, ajustado por dos parámetros α y β entre el acumulado de todas las feromonas y los costes de todos los posibles nodos adyacentes al nodo actual donde está la hormiga.

El rastro de la feromona que deja la hormiga en cada tramo se puede calcular con la siguiente ecuación:

$$\Delta\tau_{ij} = \begin{cases} \frac{1}{\text{Longitud}(C)} & \text{si } i, j \in C \\ 0 & \text{en caso contrario} \end{cases}$$

La disipación de la feromona con el tiempo se calcula de la siguiente forma:

$$\tau_{ij} = (1 - \rho) \cdot \tau_{ij} + \Delta\tau_{ij}$$

Donde ρ es un parámetro entre 0 y 1 que se le conoce como coeficiente de evaporación.

Podamos añadir **elitismo** si establecemos que la hormiga que ha hecho el camino más corto agrega más feromona.

Aplicaciones de las colonias de hormigas

Los algoritmos de optimización de colonias de hormigas son aplicados en muchos algoritmos de **optimización combinatorio**. Se han adaptado a problemas dinámicos en variables reales, problemas estocásticos, programación paralela y multiobjetivo. Estos algoritmos tienen una ventaja sobre algoritmos genéticos cuando el grafo puede cambiar su estructura de manera dinámica. Los algoritmos de colonias de hormigas pueden seguir ejecutándose de forma permanente y adaptarse a la nueva estructura del grafo. Esto es de mucho interés en los campos de enrutamiento de redes y en los sistemas de transportes urbanos. También se han utilizado para generar arte.

Optimización de enjambres de partículas

Es una **metaheurística que evoca el comportamiento de las partículas en la naturaleza**. En un principio fueron concebidos para elaborar modelos de conducta social, como por ejemplo los movimientos descritos por los seres vivos en una bandada de aves o en un banco de peces. Sin embargo, se vio que estos algoritmos eran adecuados para resolver problemas de optimización.

Estos algoritmos permiten optimizar un problema a partir de una población de soluciones candidatas, que denominamos partículas. Estas partículas se mueven dentro del espacio de búsqueda según una serie de reglas matemáticas sencillas que tienen en cuenta la posición y la velocidad de la partícula. El movimiento de cada partícula depende de su mejor posición obtenida, así como de la mejor posición global hallada en todo el espacio de búsqueda. A medida que se descubren nuevas y mejores posiciones, estas pasan a orientar los movimientos de las partículas. El proceso se repite con el objetivo, no garantizado, de hallar en algún momento una solución lo suficientemente satisfactoria.

Así pues, cada partícula recuerda su mejor posición y se desplaza con cierta aleatoriedad por el espacio, pero siempre en torno a su mejor posición. Con el tiempo las partículas convergen en los mínimos locales.

Como tienen cierta aleatoriedad, pueden salir de los mínimos locales. Se puede jugar con el grado de aleatoriedad en el cambio de la dirección de la partícula.

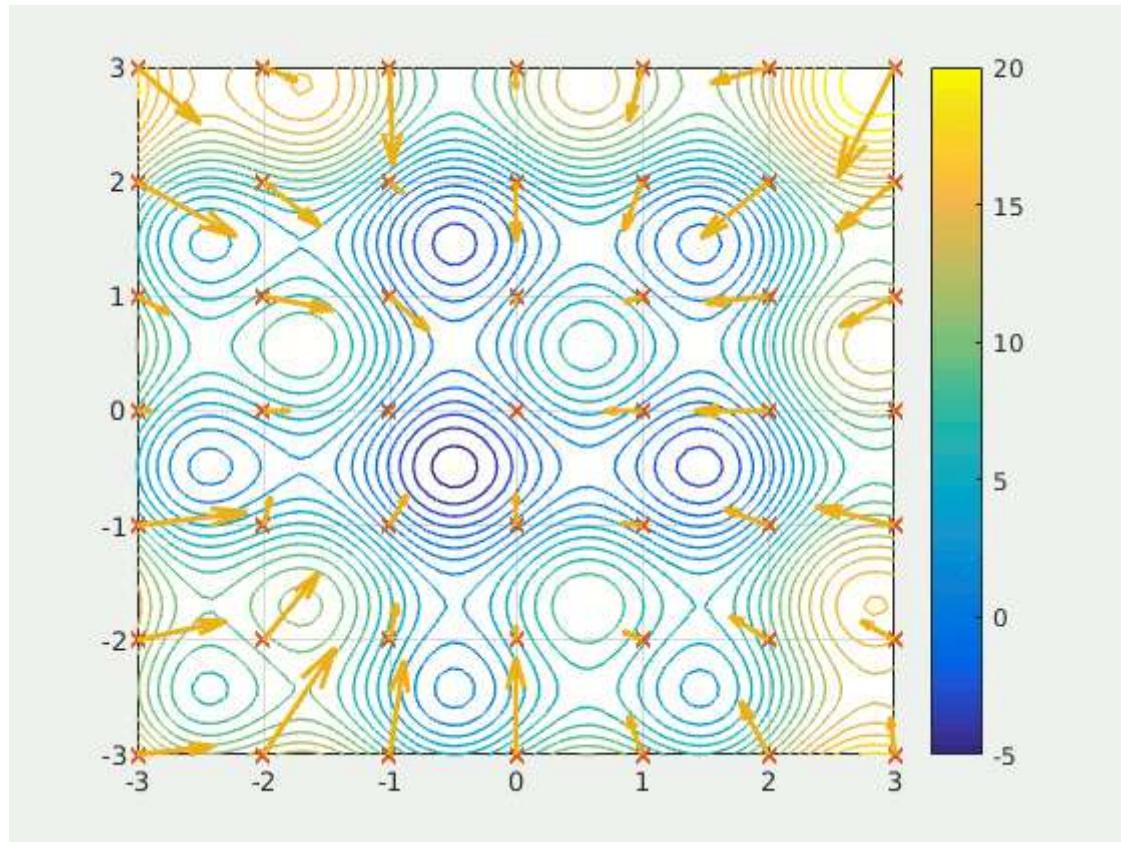


Figura 9. Ejemplo de optimización por enjambre de partículas.

Fuente: https://en.wikipedia.org/wiki/Particle_swarm_optimization#/media/File:ParticleSwarmArrowsAnimation.gif

11.5. Proyectos de investigación sobre computación bioinspirada

Los proyectos de investigación en computación bioinspirada son uno de los más activos dentro de la investigación actual. Son áreas candentes en esta investigación el *deep learning*, los algoritmos genéticos, colonias de hormigas, sistemas de optimización de partículas y enjambres para drones, robótica, etc., Los problemas donde se pueden aplicar estos sistemas de algoritmos son casi infinitos, pero vamos a concretar los más importantes describiendo algunos de ellos:

- ▶ Todo tipo de clasificación automática.
- ▶ Reconocimiento de imágenes.
- ▶ Generación artística artificial.
- ▶ Simulación de vida artificial.
- ▶ Programación de agentes automáticos.
- ▶ Optimización.
- ▶ Sistemas dinámicos.
- ▶ Sistemas de optimización multiobjetivo.

Los proyectos de computación evolutiva y optimización suelen requerir de grandes cantidades de computación, por lo que es muy recomendable acudir a soluciones de terceros. Están en una situación similar a los algoritmos de *machine learning*. Aquí la dependencia de los datos es menor que en *machine learning*, ya que la mayoría de estos algoritmos se emplean en modelos de optimización más que en la de ciencia de datos. Aunque se han visto soluciones hibridas de *machine learning* y metaheurísticas bioinspiradas en multitud de proyectos.

Las principales publicaciones de este medio son:

- ▶ *Computers & Structures.*
- ▶ *Journal of Global Optimization.*
- ▶ *IEEE Transactions on Evolutionary Computation.*
- ▶ *Expert System with Application.*
- ▶ *Evolutionary Computation.*

11.6. Referencias bibliográficas

Davis, L. (1991). *Handbook of genetic algorithms*. New York: Van Nostrand Reinhold Company.

Gestal, M., Rabuñal, J. R., Dorado, J. Pazos, A. y Rivero, D. (2010). *Introducción a los algoritmos genéticos y la programación genética*. A Coruña: Universidade da Coruña.

A fondo

Algoritmos genéticos y computación evolutiva

Marczyk, A. (2004). Algoritmos genéticos y computación evolutiva. *the-geek.org*. Recuperado de <https://the-geek.org/docs/algen/algen.html>

Libro web donde puede expandir los conocimientos de algunos de los algoritmos que exponemos en este tema.

¿Qué son los algoritmos genéticos?

Brunvelop. (2 de abril de 2020). *Esta I.A. Aprende a caminar con algoritmos genéticos* [Vídeo]. YouTube.

https://www.youtube.com/watch?time_continue=27&v=06iwEZokIGk&feature=emb_title



Vídeo explicativo de cómo funciona un algoritmo genético.

Algoritmo Inmune de Selección Clonal para el problema de Job Shop Scheduling

Gómez, W. A. (agosto de 2013). Algoritmo Inmunse de Selección Clonar para el problema de Job Shop Scheduling. *Conference IV Congreso Internacional de Computación e Informática del Norte de Chile*. Recuperado de

https://www.researchgate.net/publication/258110429_Algoritmo_Inmune_de_Selección_Clonal_para_el_problema_de_Job_Shop_Scheduling#:~:text=El%20algoritmo%20de%20selección%20clonal%20%28CLONALG%29%20%5B16%5D%20se,la%20cu%20al%20implica%20la%20selección%20de%20anticuerpos

Artículo donde se ve una aplicación de la selección clonal de sistemas inmunes en un problema de planificación de tareas.

Drawing with Ants: Generative Art with Ant Colony Optimization Algorithms

LOBSTERS. (18 de enero de 2020). Drawing with Ants: Generative Art with Ant Colony Optimization Algorithms. *colabug.com*. Recuperado de

<https://www.colabug.com/2020/0118/6871092/>

Artículo donde se explica cómo generar arte usando colonias de hormigas.

1. ¿Qué es la computación biológica?

- A. Es el área de conocimiento que estudia cómo podemos utilizar elementos de naturaleza biológica para obtener mejores resultados en problemas de computación.
- B. Es el uso de técnicas de computación en el estudio de la biología.
- C. Es el estudio de la biología haciendo una analogía con la computación.
- D. Ninguna de las anteriores.

2. En la teoría de la selección natural, la presión selectiva sirve para...

- A. Que se detenga la evolución.
- B. Que la evolución se produzca, ya que es el mecanismo por el cual las células transmiten la herencia genética.
- C. Guiar a la evolución para adaptar los individuos al entorno.
- D. Favorece la mutación de los individuos.

3. ¿Qué es la computación evolutiva?

- A. Es un conjunto de técnicas que permite a las máquinas aprender de forma supervisada y usar la solución de cada ejemplo como parte del aprendizaje.
- B. Es la rama de la IA que busca resolver problemas de optimización, inspirándose en mecanismos evolutivos.
- C. Es la rama de la IA que busca resolver problemas de clasificación, inspirándose en mecanismos evolutivos.
- D. Ninguna de las anteriores.

- 4.** En los algoritmos genéticos, la mutación sirve para...
- A. Aportar variabilidad genética.
 - B. Buscar nuevas soluciones y salir de mínimos locales.
 - C. Mantener y/o recuperar los alelos perdidos.
 - D. Todas las anteriores son correctas.
- 5.** Si hay más presión selectiva en un algoritmo genético, ¿qué sucede?
- A. Que disminuye la diversidad genética.
 - B. Que aumenta la diversidad genética.
 - C. Que la solución que se obtenga será más óptima.
 - D. Ninguna de las anteriores.
- 6.** Indica el motivo por el cual un algoritmo genético se puede estancar prematuramente:
- A. Que apenas haya presión selectiva.
 - B. Que apenas haya diversidad genética.
 - C. Que esté estancado en un mínimo local y el mínimo global esté muy lejos, siendo la tasa de mutación no lo suficientemente grande.
 - D. Todas las anteriores.
- 7.** ¿Cómo configurarías un algoritmo genético para maximizar los resultados?
- A. Al principio pondría tasas de mutación baja para explotar y con el tiempo pondría tasas de mutación alta para explorar.
 - B. No pondría mutación.
 - C. Al principio pondría tasas de mutación alta para explorar y con el tiempo pondría tasas de mutación baja para explotar.
 - D. Ninguna de las anteriores.

8. Las estrategias evolutivas...

- A. Son similares a los algoritmos genéticos pero los datos que procesan son discretos.
- B. Son similares a los algoritmos genéticos pero los datos que procesan son continuos.
- C. Sirve para clasificación y para regresión.
- D. Todas las anteriores son correctas.

9. En cuanto a la optimización multiobjetivo...

- A. Se calcula el frente de Pareto que determina la región de decisión óptima que hace que no se mejore ni empeore ninguna de las funciones objetivo para cada uno de los posibles valores del dominio.
- B. Pretende optimizar dos algoritmos simultáneamente.
- C. Pretende aprender un único valor que optimice los múltiples objetivos buscados.
- D. Todas las anteriores son correctas.

10. ¿Qué es un sistema emergente?

- A. Son sesgos no esperados en los comportamientos aprendidos por los algoritmos de *machine learning*.
- B. Sistemas complejos compuestos por elementos muy simples de cuyo comportamiento no se puede extraer el comportamiento final del conjunto.
- C. Un sistema que es capaz de extraer soluciones que otros sistemas no permite extraer.
- D. Todas las anteriores son falsas.

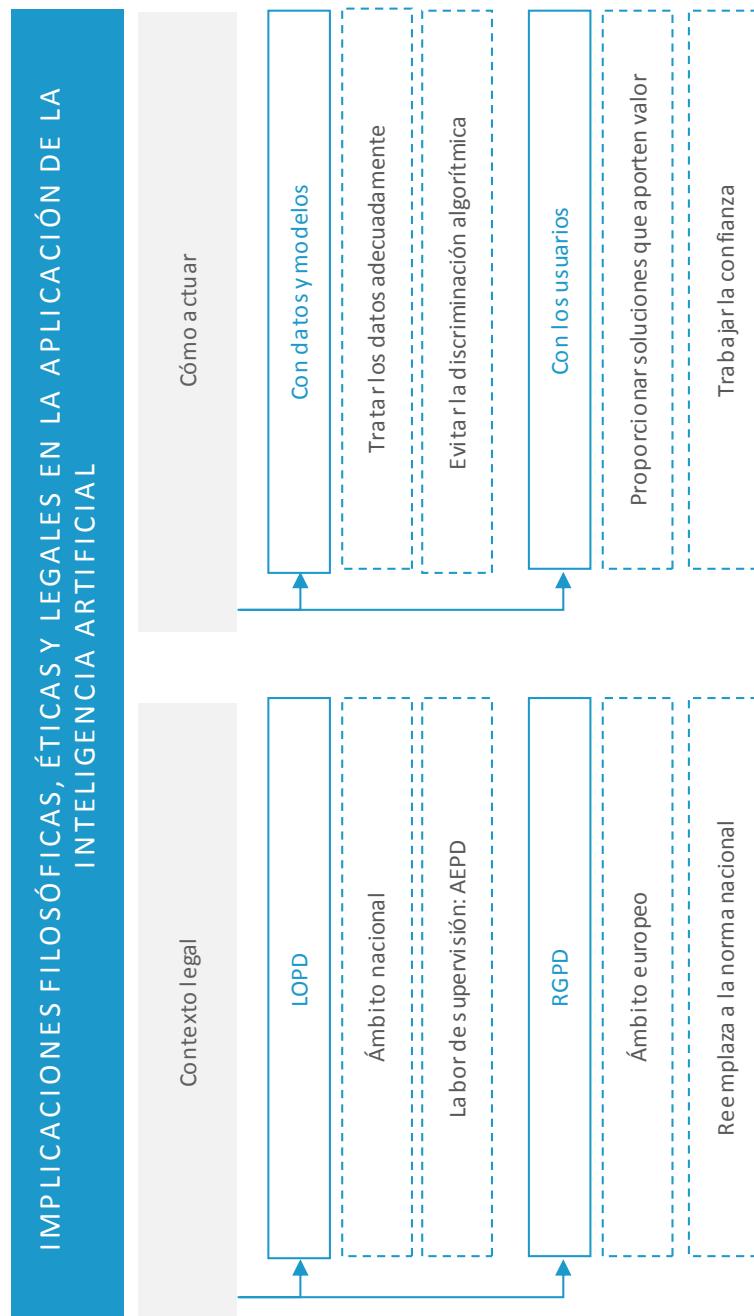
Investigación en Inteligencia Artificial

Implicaciones filosóficas, éticas y legales en la aplicación de la inteligencia artificial

Índice

Esquema	3
Ideas clave	4
12.1. Introducción y objetivos	4
12.2. Contexto legal aplicable a proyectos de inteligencia artificial	4
12.3. Sesgos, legalidad y responsabilidad	14
12.4. Seguridad y tolerancia ante ataques	20
12.5. Explicabilidad de algoritmos	23
12.6. Referencias bibliográficas	26
A Fondo	27
Test	29

Esquema



12.1. Introducción y objetivos

La aplicación de la inteligencia artificial está unida no solo a retos técnicos o científicos sino también a **retos éticos y legales** propios de una disciplina que está llamada a cambiar notablemente la forma en la que interactuamos con los demás, los métodos de trabajo y toda la actividad económica. En esta unidad nos planteamos los objetivos siguientes:

- ▶ Identificar retos éticos y filosóficos en el empleo de la IA (inteligencia artificial).
- ▶ Conocer las peculiaridades legales del trabajo basado en datos.
- ▶ Identificar nuevas tendencias regulatorias que implican cambios a la hora de trabajar con datos.

12.2. Contexto legal aplicable a proyectos de inteligencia artificial

Los proyectos de inteligencia artificial **están basados en datos**. Necesitamos datos para entrenar y para validar los modelos. También necesitamos almacenar y analizar los datos que genera la interacción con el modelo, solo así seremos capaces de determinar si la puesta en práctica de la solución cumple con los objetivos previstos. La evaluación constante de los modelos es lo que permite su mejora y evolución.

Por tanto, los proyectos de inteligencia artificial se ven claramente influenciados por la legislación vigente en materia de protección de datos.

En España, la Agencia Española de Protección de Datos (AEPD), como autoridad de control independiente, tiene encargada la misión de velar por el cumplimiento de la normativa de protección de la información personal.

La agencia es un ente de derecho público, con personalidad jurídica propia y plena capacidad pública y privada, que actúa con plena independencia de las Administraciones públicas en el ejercicio de sus funciones. Se relaciona con el Gobierno a través del Ministerio de Justicia. La Ley Orgánica de Protección de Datos de Carácter Personal (LOPD) 15/1999, de 13 de diciembre, regula gran parte de las directrices que se deben tener en cuenta a la hora de manipular información personal.



Figura 1. Página web de la AEPD.

Fuente: <https://www.agpd.es/portalwebAGPD/index-ides-idphp.php>

Bajo el paraguas de la Unión Europea surge el Reglamento General de Protección de Datos (RGPD), con la aspiración de unificar los regímenes de todos los Estados miembros sobre la materia. Aunque entró en vigor el día 25 de mayo de 2016, su cumplimiento solo será obligatorio transcurridos dos años desde esta fecha.

A continuación, se desglosarán **algunas de las características más destacadas de estas normativas**, así como sus implicaciones para la tipología de los proyectos que nos ocupa.

Protección de datos de carácter personal

Se considera un dato de carácter personal **cualquier información concerniente a personas físicas identificadas o identificables**. Por tanto, la legislación no solo contempla como datos de carácter personal aquellos casos en los que es posible identificar únicamente al individuo en base a un atributo exclusivo como el documento nacional de identidad. También contempla la posibilidad de que dicha identificación pueda producirse en el futuro vía cruce de datos de distintas tipologías.

Por ejemplo, supongamos que tenemos almacenada en un fichero o base de datos información sobre los clientes de una determinada compañía. Supongamos ahora que contamos con campos poco relevantes y genéricos. En esta situación, la compañía decide añadir la edad de cada usuario a la información existente. Si los datos anteriores realmente son irrelevantes (siempre referido a cuestiones de identificación personal), la edad por sí sola no podría servir para identificar a un individuo, puesto que, ¿cuántas personas hay de 20 años?, ¿y de 35? Las opciones son muy diversas. En estas condiciones, la compañía decide incorporar a la base de datos el género. Ahora las opciones se reducen, evidentemente, hay más personas que tienen 35 años a personas que tienen 35 años y son mujeres (por ejemplo). No obstante, parece poco creíble pensar que solo existe una única persona que tenga 35 años y sea mujer. Animados por lo útil que es insertar este tipo de atributos en el estudio, el responsable del proyecto decide ahora incorporar el código postal del usuario. En esta situación, y si la compañía dispone de especialistas en seguridad y normativa de protección de datos, se levantará una señal de alarma. Con la edad, género y código postal sí que existe una mínima posibilidad de que se pueda identificar de forma única a una persona (especialmente en grupos de edades menos frecuentes). En diversos entornos de la materia, a la combinación de los atributos edad, género y código postal se la conoce con el nombre de triada.

Antes de continuar, queremos dejar patente que no se prohíbe el almacenamiento de información personal. Si así fuese, ninguna empresa o institución pública podría funcionar. La **misión de la ley es establecer distintos niveles obligatorios de seguridad y control** en función de la tipología de la información. Incluso dentro de

la información que puede ser considerada como información personal existen distintos niveles de criticidad. No es lo mismo almacenar edad, género y código postal, que almacenar el número de la cuenta bancaria o, más crítico todavía, el número completo de la tarjeta de crédito con su fecha de caducidad y código de seguridad.

Las distintas iniciativas regulatorias que van surgiendo se encaminan a posibilitar a los usuarios finales o clientes un mayor control sobre los datos que ceden a las compañías y el uso que estas hacen de estos datos. En su artículo 4, la LOPD dice:

- «1. Los datos de carácter personal solo se podrán recoger para su tratamiento, así como someterlos a dicho tratamiento, cuando sean adecuados, pertinentes y no excesivos en relación con el ámbito y las finalidades determinadas, explícitas y legítimas para las que se hayan obtenido.
- »2. Los datos de carácter personal objeto de tratamiento no podrán usarse para finalidades incompatibles con aquellas para las que los datos hubieran sido recogidos. No se considerará incompatible el tratamiento posterior de estos con fines históricos, estadísticos o científicos.»

Es por tanto preciso que **el propietario de los datos autorice el almacenamiento y tratamiento de su información personal**. Así mismo, los responsables del repositorio de datos deben garantizar a los usuarios los métodos adecuados para la rectificación, modificación, acceso y cancelación de la información suministrada.

Confianza

Dentro de ciertos sectores sociales existe una preocupación relevante por el uso que determinadas compañías están haciendo con sus datos, y quizás esta preocupación esté justificada si nos basamos en ciertos titulares periodísticos que se han publicado en los últimos años. En este tipo de cuestiones entraremos más en detalle en el próximo apartado. Ahora intentaremos explicar cómo se puede establecer una relación de confianza entre el usuario y la organización para que este ceda sus datos.

Supongamos que un paciente estándar de 50 años está en la revisión anual con su médico. El hospital está interesado en crear un banco de datos y compartirlos con personal externo. Dado que es obligatorio otorgar permiso explícito para ceder datos de carácter personal, el facultativo nos pone delante un formulario donde simplemente se indica el texto: «autorizo la cesión de mis datos ...».

Sin disponer de más información, ¿qué haría el lector? Cambiemos la situación yendo a un caso algo más extremo.

Supongamos que ese mismo paciente ha sido diagnosticado recientemente de una enfermedad que tiene una incidencia muy baja, pero, desgraciadamente, sus efectos en el enfermo son importantes. Textualmente, lo que nos comenta ahora el médico es:

«Nos gustaría que nos cediese su datos personales y clínicos para incorporarlos a nuestro banco de datos. Lo que pretendemos con este proyecto es compartir información con especialistas de todo el país e investigadores muy destacados en el área. Dado que su enfermedad es extremadamente rara, consideramos que sería muy útil compartir sus datos con la comunidad científica. Seguro que así podremos recibir recomendaciones que nos ayudaría a personalizar el tratamiento a sus características. Además, no compartiremos sus datos personales más íntimos, toda la información estará anonimizada. Ante cualquier cuestión, seremos nosotros quienes nos comunicaremos con usted como venimos haciendo hasta ahora».

¿Qué haría el lector en esta hipotética situación?

Es probable que las respuestas ante ambos casos de uso sean muy distintas. Y sería normal, puesto que se trata de dos casos muy diferentes. En la primera situación, el paciente no aprecia el retorno de valor que le produce la cesión de sus datos. Además, al recibir poca información sobre el proyecto es difícil generar una relación de confianza. En el segundo caso, el beneficio que el paciente percibe como consecuencia de la cesión de datos es evidente y relevante. La comunidad podría aportar un tratamiento que retarde la enfermedad o incluso la cure. Pero para que esto sea una realidad es preciso que especialistas comparten información sobre qué

tratamientos han funcionado, cuáles no, cuáles han sido los efectos secundarios, qué peculiaridades tenían los pacientes... Además, el facultativo ha trabajado la confianza al solicitar la cesión de los datos, ha explicado la utilidad de la cesión de los datos y ha establecido unas garantías que invitan al paciente a estar tranquilo sobre quién podrá contactarle. ¿Cuántas veces hemos recibido llamadas publicitarias ofreciendo productos sin ningún tipo de interés para nosotros y sin saber cómo habían accedido a nuestros datos?

Lo que se ha pretendido explicar en los párrafos anteriores se puede resumir en los siguientes puntos:

- ▶ Los algoritmos de inteligencia artificial (por ejemplo, los algoritmos de aprendizaje automático) emplean datos como materia prima.
- ▶ Muchas veces estos datos incluyen información personal.
- ▶ Para manipular información personal (aunque esté anonimizada) el usuario debe haber emitido su autorización.
- ▶ Para que el usuario ceda sus datos personales es preciso trabajar un entorno de confianza. Este entorno de confianza se basa en los siguientes puntos:
 - El usuario tiene claro a quién (a nivel de institución) y para qué va a acceder los datos.
 - El usuario tiene claro qué se va a hacer con sus datos.
 - El usuario percibe un retorno de valor evidente que le invita a permitir la cesión de datos.
 - El usuario es informado de que sus datos van a ser tratados de forma adecuada y segura.

Establecer mecanismos que cultiven e implementen esta relación de confianza entre el usuario o cliente y la empresa o la institución es uno de los grandes retos a los que se enfrenta la industria hoy en día.

Anonimización de datos personales

El tratamiento, análisis y explotación de grandes volúmenes de datos puede producir infinitud de beneficios a la sociedad, pero es preciso compatibilizar dicho almacenamiento y manipulación con el respeto a la protección de datos personales.

El mecanismo principal que permite alinear la generación de beneficios tangibles con el respeto a la privacidad es **la anonimización de la información**. Según la RAE, anonimizar es «expresar un dato relativo a entidades o personas, eliminando la referencia a su identidad». Por otro lado, y según la AEPD, «un proceso de anonimización es aquel que elimina o reduce al mínimo los riesgos de identificación de los datos anonimizados manteniendo la veracidad de los resultados tras el tratamiento de los mismos».

Para garantizar su efectividad, **este proceso de anonimización debe ser irreversible**. Es decir, no debe existir posibilidad de recuperar el dato adicional empleando solo los datos anonimizados. Es necesario precisar que, debido a los avances tecnológicos y algorítmicos, no es posible garantizar de forma absoluta y completa la anonimización de la información. El objetivo realizable es proporcionar las mayores garantías posibles de cara a asegurar el respeto a la privacidad de las personas.

Según la AEPD, el proceso de anonimización debe atender a los siguientes principios:

- ▶ Principio proactivo. La privacidad se debe garantizar de forma proactiva y no de forma reactiva y una vez que se haya producido alguna fuga de información.
- ▶ Principio de veracidad por defecto. Se debe considerar la granularidad o grado de detalle final que deben tener los datos anonimizados. Esto lleva a que, en ocasiones, se exija la eliminación de ciertos datos para garantizar la anonimización del conjunto.

- ▶ Principio de privacidad objetiva. Siempre existirá un error residual de riesgo de reidentificación que deberá ser aceptable en función de la información anonimizada, conocido por el usuario y asumido por el responsable del fichero.
- ▶ Principio de plena funcionalidad. El proceso de anonimización debe garantizar la utilidad de los datos anonimizados en base a los objetivos inicialmente establecidos.
- ▶ Principio de privacidad en el ciclo de vida de la información. El respeto a la privacidad de los usuarios debe garantizarse durante todo el proceso de anonimización. Por ejemplo, realizando el proceso de anonimización en los sistemas preparados y autorizados a almacenar la información sin anonimizar.
- ▶ Principio de información y formación. Todo el personal con acceso a datos anonimizados o no deben estar correctamente formados e informados acerca de sus obligaciones.

Técnicamente hablando, existen varios **procedimientos para asegurar la anonimización de la información**. Algunos de los más destacados son:

- ▶ Desnaturalizar: consistente en transformar la naturaleza del dato. Por ejemplo, en lugar de representar la edad (edad = 42), podemos indicar el rango de edad al que pertenece en base a alguna división previamente establecida (edad = 5 donde 5 hace referencia al intervalo [40, 50]).
- ▶ Cifrar: consiste en hacer ilegible un mensaje concreto en base a la aplicación de un algoritmo que precisa de un conjunto de claves. En este caso, el descifrado de la información es posible siempre que se disponga del algoritmo de las claves necesarias.
- ▶ *Tokenizar*: se reemplaza el valor a anonimizar por un valor distinto (*token*) que no suele respetar la naturaleza del dato. Por ejemplo, se *tokeniza* el DNI 04345566D cambiándolo por el valor YID884S3VVQW4ZZY1. Se puede observar cómo no se respeta el formato estándar de un DNI. Para que el proceso mantenga la coherencia, al mismo valor le debe siempre corresponder el mismo *token*. La reversibilidad es posible siempre que se disponga del *token* correspondiente a cada valor.

- ▶ Funciones *hash*: es un método parecido a la *tokenización*. En este caso se aplica una función matemática al valor a anonimizar. Dicho valor reemplaza al valor original. En este caso, y por la propia naturaleza del proceso, se garantiza la irreversibilidad del proceso. No obstante, esta técnica podría, ocasionalmente, generar el mismo valor *hash* para distintos valores de entrada.
- ▶ Disociar: eliminar parte de la información para evitar la identificación personal. Por ejemplo, pongamos que disponemos de la siguiente información sobre un paciente: fecha de la consulta, hora de la consulta, código postal, edad, sexo y síntomas. Con esa información encima de la mesa existiría un riesgo de identificar al paciente, ya que solo la fecha y hora de la consulta proporcionan bastante información. La disociación consistiría en eliminar ciertos campos quedándonos solo (por ejemplo), con código postal, sexo y síntomas. De esta forma se reduce el riesgo de identificación.

Reglamento General de Protección de Datos (RGPD)

Aunque diseñada en el año 2016, el RGPD de la Unión Europea pasó a ser de obligado cumplimiento en el año 2018. La aplicación de esta normativa supone un gran reto para muchas compañías en las que el dato y la analítica sobre el mismo constituye un eje esencial de su actividad. El incumplimiento de la normativa implica cuantiosas multas para los infractores.

El RGPD **exige medidas adicionales que garanticen la transparencia en el tratamiento de datos personales**. Se pretende así empoderar al ciudadano para que tome las decisiones más adecuadas. Las organizaciones deben informar al usuario sobre el tipo de perfilado o modelización que realizan en base a su dato personal facilitándole la denegación del permiso para realizar dicho tratamiento si así es solicitado.

Especialmente en el entorno financiero y asegurador, una de las medidas que más impacto causará es el **derecho a explicación**. Esta medida obliga, por ejemplo, a las entidades financieras a explicar las razones por las que un crédito ha sido

denegado. De esta forma, el usuario puede actuar en consecuencia buscando alternativas para obtener una mejor clasificación en el futuro.

Además, los algoritmos implementados deben asegurar que se garantiza la no discriminación a la hora de tomar decisiones basadas en datos. No podrán tomarse decisiones basadas en criterios como la raza, la edad, el sexo, la religión, etc.

La Unión Europea ha habilitado el portal <https://www.eugdpr.org/> donde se desglosa todo tipo de información sobre esta regulación.



Figura 2. Portal de la RGPD.

Fuente: <https://www.eugdpr.org/>

En esta unidad se propondrá al alumno la realización de un trabajo que versará, precisamente, sobre esta normativa.

12.3. Sesgos, legalidad y responsabilidad

Lo que hace unos años era una cuestión ética, evitar la discriminación a la hora de emplear datos y algoritmos en la toma de decisiones, ahora se ha convertido en ley. En un entorno tecnológico y científico con una evolución tan rápida e impredecible, el entorno regulatorio no siempre es capaz de llegar a tiempo para proteger los derechos de los ciudadanos, por lo que el especialista, científico o técnico debe mantener altos estándares éticos en el día a día de su trabajo.

Hay una serie de puntos para tener en cuenta en este sentido:

1. Tener siempre presente la normativa de protección de datos.
2. Asegurarse que los algoritmos que empleamos no conllevan la toma de decisiones implicando la discriminación de algún colectivo por edad, sexo, raza, religión o cualquier otro aspecto.
3. Comprobar que los datos empleados no contienen sesgos que puedan llevar a tomar decisiones equivocadas.
4. Interpretar los resultados de los modelos científicamente, evitando interpretaciones interesadas y no ajustadas a la realidad.
5. Emplear los métodos de trabajo adecuados que garanticen la fiabilidad de los resultados.

Conceptos como inteligencia artificial y aprendizaje automático forman parte del vocabulario diario de comités de empresa y gobiernos de todo el mundo. Las decisiones que se toman en base a estos modelos afectan a millones de personas. Por tanto, es responsabilidad de los expertos asegurar que el objetivo final es aportar beneficio a la sociedad en general o los clientes de la compañía en particular, respetando los derechos fundamentales de la ciudadanía.

Sesgos: el motivo por el que los algoritmos aprenden y su principal punto débil

Como ya vimos anteriormente, **los sesgos son imposibles de eliminar ya que es el mecanismo por el que los algoritmos aprenden**. Crear sesgos y hacer suposiciones globales despreciando los detalles concretos es la base del aprendizaje y, por tanto, las excepciones siempre van a estar ahí. Lo importante en este caso es minimizarlos y que estos sesgos no aprendidos no sean sesgos inducidos por el entrenamiento debido a una mala elección de los datos de entrenamiento.

Es por tanto muy importante tener en cuenta este tipo de cuestiones en entornos sensibles y con sesgos que pueden ocurrir en discriminaciones de cualquier tipo. Si no se puede discriminar a un individuo por razones ideológicas, de sexo, étnicos, etc., los algoritmos no pueden hacerlo tampoco.

¿Cómo evitar este hecho?

- ▶ Seleccionar cuidadosamente los datos de entrenamiento.
- ▶ Validar los algoritmos no solo con los datos provenientes del primer mundo o de nuestra área de influencia sino de otras partes del mundo con rasgos, culturas o éticas diferentes.
- ▶ Mantener una vigilancia continua de las decisiones que estos toman, para intervenir lo antes posible si se detectan estos sesgos.
- ▶ Tener evaluadores humanos que confirmen las decisiones tomadas por los algoritmos o al menos que los usuarios que se vean afectados por ellos puedan acudir para que se revise su caso particular.

Este va a ser uno de los retos más importante de aquí a unos años cuando los algoritmos de IA sean los que vayan controlando cada vez más y más procesos en los que la vida de los ciudadanos se vea afectada.

En cuanto a la **responsabilidad de los errores de estos algoritmos**, pues es bastante complicado establecerla y es algo en lo que debemos esforzarnos como sociedad. Obviamente, la responsabilidad siempre debe ser de la organización que use estos algoritmos que deben probarlos correctamente. Pero nada está exento de errores al 100 %. Así que hay que establecer cuál es el margen de error admisible para cada tarea que se delegue a la IA. Al fin y al cabo, los humanos también nos equivocamos. La diferencia aquí es que la cadena de responsabilidad entre los humanos es trazable y queda más o menos clara. Cuando entra dentro un algoritmo de IA ya no está tan clara, ¿es de los diseñadores del algoritmo?, ¿es de quien seleccionó los datos de entrenamiento? ¿Es de quien no revisó correctamente dicha decisión? Se entra en un terreno desde nuestro punto de vista pantanoso que tendremos que ir valorando con el tiempo y con leyes nuevas que probablemente lleguen tarde para algunos casos. Este es un reto importante a tener en cuenta en el futuro.

Implicaciones de la inteligencia artificial

La explosión de la inteligencia artificial ha marcado o marcará el inicio de una nueva revolución de dimensiones equivalentes a la Revolución Industrial. Como en toda gran transformación, y a pesar de las nuevas posibilidades que deslumbran, surgen también dudas y temores. Por ejemplo:

- ▶ ¿Perderán las personas su trabajo siendo reemplazadas por máquinas?
- ▶ ¿En qué trabajarán los humanos? ¿Cuánto tiempo?
- ▶ ¿Cómo se transformará nuestro tiempo de ocio?
- ▶ ¿Qué sucederá con nuestro derecho a la privacidad?
- ▶ ¿Cómo interactuaremos con los sistemas basados en inteligencia artificial?
¿Quién liderará la relación?
- ▶ ¿Podrán las máquinas sentir? ¿Cómo actuaremos entonces?
- ▶ ¿Dominarán los robots a la raza humana?

A la mayoría de estas preguntas no es posible responderla en estos momentos. Sin embargo, el debate sobre las implicaciones laborales de la robótica y la inteligencia artificial ya ha comenzado.

En primer lugar, la gran mayoría de flujos de trabajo serán automatizados y no requerirán intervención humana. Esto permitirá mejorar la eficiencia de los procesos y, quizás, reducir los costes.

Los algoritmos se convertirán en los principales agentes en ciertos escenarios. Por ejemplo, muchas de las operaciones bursátiles que se llevan a cabo hoy en día son ejecutadas de forma automática mediante algoritmos que evalúan la evolución del mercado e incluso el contexto informativo analizando la información que circula sobre el contexto económico y social.

Los robots no solo han transformado las fábricas del siglo XXI, también están introduciéndose en el terreno sanitario e incluso militar.

El reto que tenemos por delante es **convertir los riesgos en oportunidades reales**.

Igual que sucedió con la Revolución Industrial, muchos trabajos desaparecerán porque ya no serán necesarios al poder realizarse de forma automática. Sin embargo, surgirán otros muchos nuevos trabajos y, por supuesto, los métodos de trabajo cambiarán radicalmente. La **educación adquiere una relevancia especial** para dirigir la transformación global de habilidades y conocimientos que permita la inmersión en el nuevo ecosistema. Algunos autores sugieren gravar mediante nuevos impuestos ciertos productos basados en inteligencia artificial, como los robots industriales para que el Estado reciba, al menos temporalmente, fuentes adicionales de financiación que permitan ayudar a aquellas personas que se queden sin trabajo debido a la automatización de sus posiciones y, además, pueda impulsar un nuevo sistema educativo que promueva la creatividad y genere el ecosistema humano que este nuevo paradigma necesita. A esto se le ha denominado que los **robots paguen impuestos**. Otros autores comentan que se debe crear una renta

básica que permita la subsistencia mínima individual. En cualquier caso, es un reto muy importante que debemos abordar en los próximos años.

La inteligencia artificial redefinirá la economía del futuro impulsando la productividad y generando una oleada de productos personalizados basados en las necesidades del cliente. Este impacto se extenderá por todos los ámbitos de actividad.

Regulando la inteligencia artificial

En los últimos años, destacados y muy relevantes personajes públicos han planteado la necesidad de regular de forma más exigente el ámbito de la inteligencia artificial de cara a evitar un mal uso de esta.

Entre estos actores destaca especialmente la organización Future of Life (<https://futureoflife.org/>). Esta fundación propone medidas para mitigar los riesgos de un mal uso no solo de la inteligencia artificial, sino también de la biotecnología o de las armas nucleares. Además, contemplan acciones para mitigar el cambio climático.

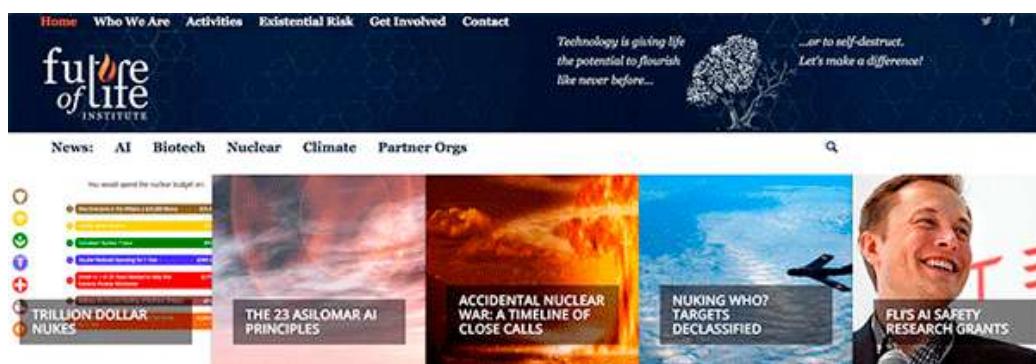


Figura 3. Página web de Future of Life.

Fuente: <https://futureoflife.org/>

La lista de personalidades detrás de este proyecto es absolutamente espectacular. Entre sus fundadores podemos encontrar destacados científicos procedentes del MIT, DeepMind, Universidad de Santa Cruz o Boston, además de a Jaan Tallinn,

cofundador de Skype. La labor de la institución está apoyada públicamente por figuras como Alan Alda, Erik Brynjolfsson (director del MIT Center for Digital Business), Morgan Freeman, el conocidísimo Stephen Hawking, Elon Musk (fundador de Tesla), Stuart Russell (autor del libro de referencia en esta asignatura) y un largo etcétera.

Una de las grandes preocupaciones de este grupo es **la aplicación de la inteligencia artificial con fines militares**. Atributos como la empatía, la justicia, la responsabilidad, la compasión..., son de momento atributos puramente humanos que las máquinas no tienen ocasión de contemplar.

Future of Life también se plantea romper con muchos de los mitos ligados a la inteligencia artificial:



Figura 4. Mitos asociados a la inteligencia artificial según Future of Life.

Fuente: <https://futureoflife.org/background/benefits-risks-of-artificial-intelligence/>

La imagen anterior podría resumirse en los siguientes puntos:

- ▶ No es preciso definir qué es lo que seremos capaces de conseguir y cuándo.
- ▶ Muchos y grandes expertos están muy preocupados sobre el uso que se podría hacer de la inteligencia artificial.
- ▶ En ocasiones la inteligencia artificial puede producir soluciones no alienadas con los objetivos adecuados.
- ▶ No es solo cuestión de preocuparse por los robots, los algoritmos autónomos son también una cuestión que hay que vigilar.
- ▶ Ciertas soluciones de inteligencia artificial podrían condicionar y controlar el comportamiento humano.
- ▶ Las máquinas pueden plantearse un objetivo de forma predefinida.
- ▶ Es mejor prevenir que curar. Actuemos antes de que sea tarde estableciendo las bases que permitan el desarrollo de una inteligencia artificial segura y alineada con las necesidades humanas.

Hay que destacar que este grupo no está en contra de la inteligencia artificial. De hecho, mucho de sus colaboradores forman parte de la élite investigadora de la materia. Lo que se pide no es abandonar el desarrollo de la inteligencia artificial, sino **establecer unas bases regulatorias adecuadas** que permitan asegurar que su desarrollo se produce de forma segura.

12.4. Seguridad y tolerancia ante ataques

Con el uso cada vez mayor de la inteligencia artificial y de la compartición de datos, las posibilidades de ataques en este tipo de sistemas están aumentando considerablemente. El uso intensivo de datos en los sistemas de IA hace que estos mejoren y se puedan aplicar a múltiples entornos que antes no eran posibles. Pero nos surge una duda, ¿cómo queda la seguridad y la privacidad de mis datos al adoptarla?

Ahora mismo muchas compañías están utilizando sistemas y modelos que van aprendiendo sobre la marcha y estos modelos no solo son usados por una compañía. Muchas veces, distintas compañías hacen uso de los mismos modelos debido a que la mayoría de estos modelos se enfocan al uso del *software* como servicio. De forma que, de una u otra forma, tus datos forman parte de una red que usa más gente. Por esta razón, ¿están seguros esos datos?

Por otro lado, con la creciente digitalización cada vez habrá más datos conectados a internet y, por tanto, más puertas abiertas que pueden ser aprovechadas por los ciberatacantes. Por eso, es muy importante que estos datos estén bien protegidos; sobre todo, si son datos sensibles.

También comienzan a verse ataques a los propios algoritmos de IA. Como todo sistema informático, estos sistemas pueden tener agujeros de seguridad. Pero también podemos ir más allá. Los sistemas de IA y de *machine learning* como sabemos tienen sesgos y los atacantes pueden utilizar esos sesgos a su favor. Por ejemplo, falseando los datos de entrada, aprovechando de alguna vulnerabilidad del algoritmo para que este falle en su beneficio. Por ejemplo, para conseguir que una IA descarte o acepte un préstamo en un banco.

Los sistemas de reconocimiento de texto y de lenguaje natural, por ejemplo, han avanzado enormemente en los últimos años, pero los programas de IA que analizan el texto pueden ser engañados por frases cuidadosamente elaboradas. Una frase que parece sencilla para un humano puede engañar a un algoritmo de IA. Esto puede implicar, por ejemplo, que un sistema que filtre noticias falsas sea engañado colando una noticia falsa sutilmente camuflada o que rechace o clasifique correctamente un posible candidato a un empleo. A este tipo de ejemplos que se le conoce con el nombre de *adversarial examples*.

Podemos definir, por tanto, un **adversarial example** como un caso con pequeñas perturbaciones intencionales de características que hacen que un modelo de aprendizaje de una máquina haga una predicción falsa.

Algunos ejemplos de este tipo de adversarial examples pueden ser:

- ▶ Un auto que se conduce a sí mismo choca con otro coche porque ignora una señal modificada para que el algoritmo lo identificara erróneamente.
- ▶ Armas diseñadas para engañar los sistemas de escaneo de un aeropuerto.
- ▶ Engañamos a un sistema recomendador para que recomiende nuestros productos cuando los usuarios buscan el de la competencia.

En la siguiente figura podemos ver un ejemplo de cómo se ha conseguido engañar a AlexNet, un conocido clasificador de imágenes. En las imágenes de la izquierda el algoritmo los reconoce, pero al aplicar el ruido de la imagen central, el algoritmo clasifica todas las imágenes como avestruces.

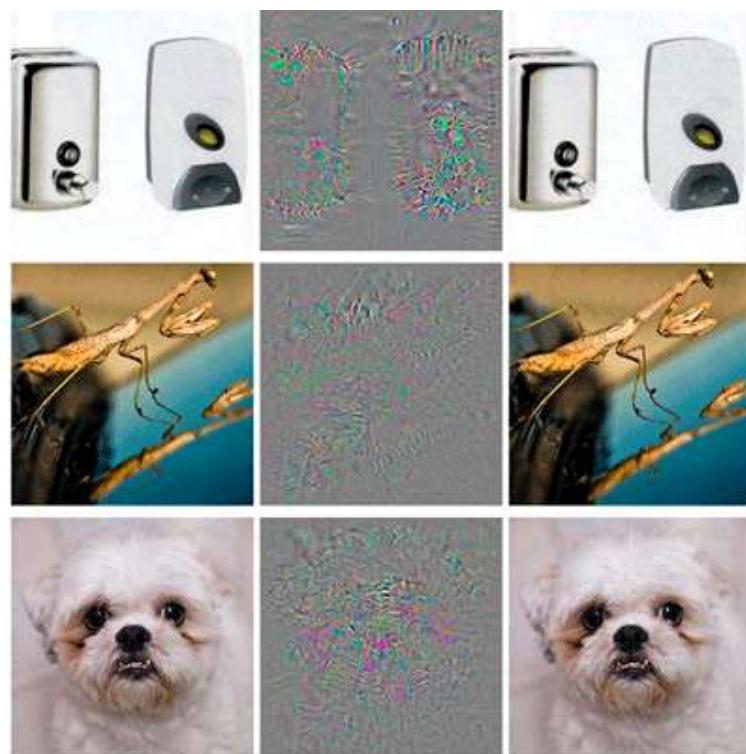


Figura 5. Algunas imágenes adversarias que engañan a Alexnet.

Fuente: <https://christophm.github.io/interpretable-ml-book/adversarial.html>

¿Cómo podemos defendernos de estos ataques? Es complicado, pero debemos ser conscientes de ello para intentar remediarlo. Podemos mitigar el riesgo con las **siguientes medidas**:

1. Conocer a tu adversario. Conocer cuáles son las motivaciones que un atacante tiene para hacerlo.
2. Ser proactivo, esto es, intentar engañar constantemente a los sistemas con tus propios ejemplos de adversario para intentar detectar los posibles errores.
3. Protegerte a ti mismo con reentrenamientos activos con adversarios o utilizar varios clasificadores y decidir en mayoría.

12.5. Explicabilidad de algoritmos

Como hemos visto en el apartado anterior, es posible engañar a los algoritmos de *machine learning* entregando ejemplos preparados para que estos se equivoquen. Si podemos explicar el modelo que se genera en el proceso de aprendizaje es más fácil evitar o predecir estos errores. Pero esto no siempre es posible, debido a que no todos los algoritmos son explicables o fácilmente explicables.

Los **modelos que generan los algoritmos de *machine learning* podemos clasificarlos en función de su explicabilidad**: en modelos de caja negra y modelos de caja blanca.

Los **modelos de caja negra** son aquellos en los que conseguir entender el modelo para poder analizarlo es muy complejo o casi imposible. Potencialmente todos pueden ser de caja negra si el número de parámetros de configuración es muy alto. Pero, en general, son propensos a ser algoritmos que generan modelos de caja negra las redes de neuronas, los random forest, el razonamiento basado en casos con medidas de distancia compleja, etc.



Figura 6. Modelo de caja negra.

Los **modelos de caja blanca**, por el contrario, son más sencillos de explicar y, por tanto, de analizar. Los árboles de decisión simples son algoritmos que generan modelos de cajas blancas. También las redes bayesianas o cualquier sistema que utilice lógica de predicados, con encadenamiento hacia delante o hacia atrás.

La **explicabilidad** nos da una serie de ventajas que debemos tener en cuenta como, por ejemplo:

- ▶ Confidabilidad. **Es muy importante poder confiar en las decisiones de un algoritmo**, sobre todo si este está a cargo de algo importante. Por ejemplo, está conduciendo un vehículo por ti, tomando decisiones de compra en bolsa o manejando una central nuclear. Pero también para otros temas menores es importante saber qué decisión está tomando el algoritmo debido a que este puede suceder en, como hemos visto durante este tema, sesgos, discriminaciones etc. Saber cómo llega a esa decisión puede ayudarnos a prevenir estos malos comportamientos de los algoritmos.
- ▶ Adquirir nuevo conocimiento. **Los algoritmos a veces son capaces de resolver problemas o descubrir nuevas soluciones a problemas que antes no se conocían**. Pero estos problemas muchas veces no pueden ser analizados correctamente debido a que no sabemos cómo el algoritmo ha llegado a esa

conclusión. Por lo tanto, perdemos los detalles de ese nuevo conocimiento adquirido.

- ▶ Detección de fallos. **Si el modelo tiene fallos y conocemos el modelo, podremos predecirlos, mitigarlos o reentrenarlo.** Hasta ahora, solo podemos saber si un algoritmo de caja negra tiene fallos probándolo exhaustivamente. Pero siempre puede haber casos que no se han contemplado en los que el algoritmo falle. Durante el tema hemos visto varios de ellos.

Pero eso no significa que dejemos de usar algoritmos de caja negra. Hay ciertos entornos donde no hay problema por usar algoritmos de caja negra debido a que la necesidad de verificación del modelo no es crítica. Pensemos en algoritmos que controlan la calidad de imagen en un videojuego (DLSS), el algoritmo de YouTube o el recomendador de Netflix, o en algoritmos que ayuden a los científicos de la NASA a encontrar nuevos exoplanetas. Si hay algún error en estos algoritmos que clasifica un exoplaneta como planeta que no lo es, el error no es de vital importancia. Con el tiempo se descubrirá que no lo es, pero no tiene un coste de error alto y es mayor el beneficio de poder haber encontrado cientos de exoplanetas que sin la IA no hubieran podido ser encontrados.

En resumen, hay que ser conscientes en qué dominios que el modelo no sea explicable no es un problema y que los beneficios de estos modelos compensen la falta de explicabilidad, y en qué dominios es recomendable usar algoritmos explicables, aunque obtengan peor rendimiento.

La aproximación, normalmente, a este tipo de dominios sensible es **tener soluciones híbridas entre algoritmos de caja negra y algoritmos de caja blanca**. Es decir, que parte del razonamiento esté inducido por algoritmos de caja blanca para que el cuerpo principal de la solución sea explicable y se use los algoritmos de caja negra en aquellas tareas donde la explicabilidad sea menos crítica. O también se busca intentar explicar los algoritmos de caja negra con otros métodos que ayuden a entenderlos.

12.6. Referencias bibliográficas

Ley Orgánica 15/1999, de 13 de diciembre, de Protección de Datos de Carácter Personal. *Boletín Oficial del Estado*, 298, de 14 de diciembre de 1999. Recuperado de <https://www.boe.es/buscar/doc.php?id=BOE-A-1999-23750>

Turing, A. M. (1950). Computing machinery and intelligence. *Mind*, 59(236), 433-460.

Orientaciones y garantías en los procedimientos de anonimización de datos personales

Agencia Española de Protección de Datos. (2016). Orientaciones y garantías en los procedimientos de anonimización de datos personales. *aepd.es*. Recuperado de: <https://www.aepd.es/sites/default/files/2019-09/guia-orientaciones-procedimientos-anonimizacion.pdf>

En el siguiente enlace podemos consultar las orientaciones y garantías en los procedimientos de anonimización de datos personales en la Agencia Española de Protección de Datos.

Código de buenas prácticas en protección de datos para proyectos Big Data

Sáiz, A. (coord.). (2017). Código de buenas prácticas en protección de datos para proyectos Big Data. *aepd.es*. Recuperado de <https://www.aepd.es/sites/default/files/2019-09/guia-codigo-de-buenas-practicas-proyectos-de-big-data.pdf>

En el siguiente enlace podemos consultar el código de buenas prácticas en protección de datos para proyectos *big data*.

Adversarial Examples

Molnar, C. (2020.). Adversarial Examples. En *Interpretable Machine Learning. A Guide for Making Black Box Models Explainable* (capítulo 6). Recuperado de <https://christophm.github.io/interpretable-ml-book/adversarial.html>

Este autor nos explica lo que son los adversarial examples.

- 1. La Agencia Española de Protección de Datos:**
 - A. Se limita a aconsejar al gobierno sobre cómo debería legislarse en materia de protección de datos.
 - B. Tiene encargada la misión de velar por el cumplimiento de la normativa de protección de la información personal.
 - C. Dependiendo del Ministerio de Interior vela porque los datos de las Administraciones públicas se gestionen correctamente.
 - D. Depende directamente de la Unión Europea.
- 2. El Reglamento General de Protección de Datos:**
 - A. Es de obligado cumplimiento desde el año 2016.
 - B. Se convierte en normativa de obligado cumplimiento en el año 2019.
 - C. Se convierte en normativa de obligado cumplimiento en el año 2018.
 - D. Se convierte en normativa de obligado cumplimiento en el año 2020.
- 3. En un contexto de anonimización y privacidad de la información, la triada hace referencia a:**
 - A. Seguridad, consistencia y persistencia: elementos básicos de un repositorio que cumpla con la normativa oficial.
 - B. La combinación de los atributos edad, género y código postal, y que, ocasionalmente, podrían permitir la identificación única de una persona.
 - C. Las tres instituciones encargadas de velar por la privacidad de los usuarios, la AEPD, el Ministerio de Justicia y el Ministerio de Interior.
 - D. Ninguna de las anteriores.

- 4.** A la hora de gestionar la relación con los usuarios o clientes:
- A. Es preciso generar un entorno de confianza.
 - B. Los clientes deben percibir que reciben un beneficio por conceder permiso para trabajar con sus datos.
 - C. Se debe informar a los clientes de cuál es el objetivo que persigue el tratamiento de sus datos personales.
 - D. Todas las anteriores.
- 5.** La anonimización de los datos personales:
- A. Implica eliminar o reducir al mínimo los riesgos de identificación de los datos anonimizados.
 - B. Implica mantener la veracidad de los resultados tras el tratamiento de los datos.
 - C. Implica eliminar la referencia a la entidad.
 - D. Todas las anteriores.
- 6.** Según la AEPD, el proceso de anonimización debe atender a los siguientes principios:
- A. Principio proactivo, principio de veracidad, principio de privacidad objetiva, principio de plena funcionalidad y principio de privacidad.
 - B. Principio proactivo, principio de veracidad, principio de privacidad objetiva, principio de plena funcionalidad, principio de privacidad y principio de información y formación.
 - C. Principio activo, principio de veracidad, principio de privacidad objetiva, principio de plena funcionalidad, principio de privacidad y principio de información y formación.
 - D. Principio reactivo, principio de veracidad, principio de privacidad objetiva, principio de plena funcionalidad, principio de privacidad y principio de información y formación.

- 7.** Una diferencia fundamental entre *tokenizar* y aplicar una función *hash* es:
- A. No existe esa diferencia, son la misma cosa.
 - B. El proceso de *tokenización* es irreversible, lo contrario que si aplicamos una función *hash*.
 - C. La anonimización mediante función *hash* es irreversible y la *tokenización* no.
 - D. La *tokenización* tarda mucho más tiempo que aplicar una función *hash*.
- 8.** ¿Qué podemos decir de las implicaciones de la inteligencia artificial?
- A. Es posible determinar de antemano casi la totalidad de estas implicaciones, tanto para lo bueno como para lo malo.
 - B. El panorama es muy esperanzador puesto que solo se perciben beneficios.
 - C. Los problemas que se perciben a corto, medio y largo plazo superan a los beneficios.
 - D. La educación será esencial para conseguir que la sociedad se adapte a los nuevos puestos laborables.
- 9.** ¿Qué son los adversal examples?
- A. Ejemplos atípicos en los datos de entrenamiento.
 - B. Ejemplos manipulados sutilmente que hacen que los algoritmos de IA fallen.
 - C. Ejemplos que no son capaces de ser reconocidos por una IA pero que también engañarían a un humano.
 - D. Son los datos de entrañamiento que se utilizan en las Generative Adversarial Networks.

10. Un algoritmo de caja negra es aquel que...

- A. Permite analizar fácilmente cómo han llegado a obtener una conclusión.
- B. Es aquel en el que el análisis de cómo ha llegado a obtener una conclusión es muy complejo.
- C. Son algoritmos que han aprendido sesgos y que pueden hacerlos tomar decisiones poco éticas.
- D. No se usan ya que se prefiera usar algoritmos de caja blanca.