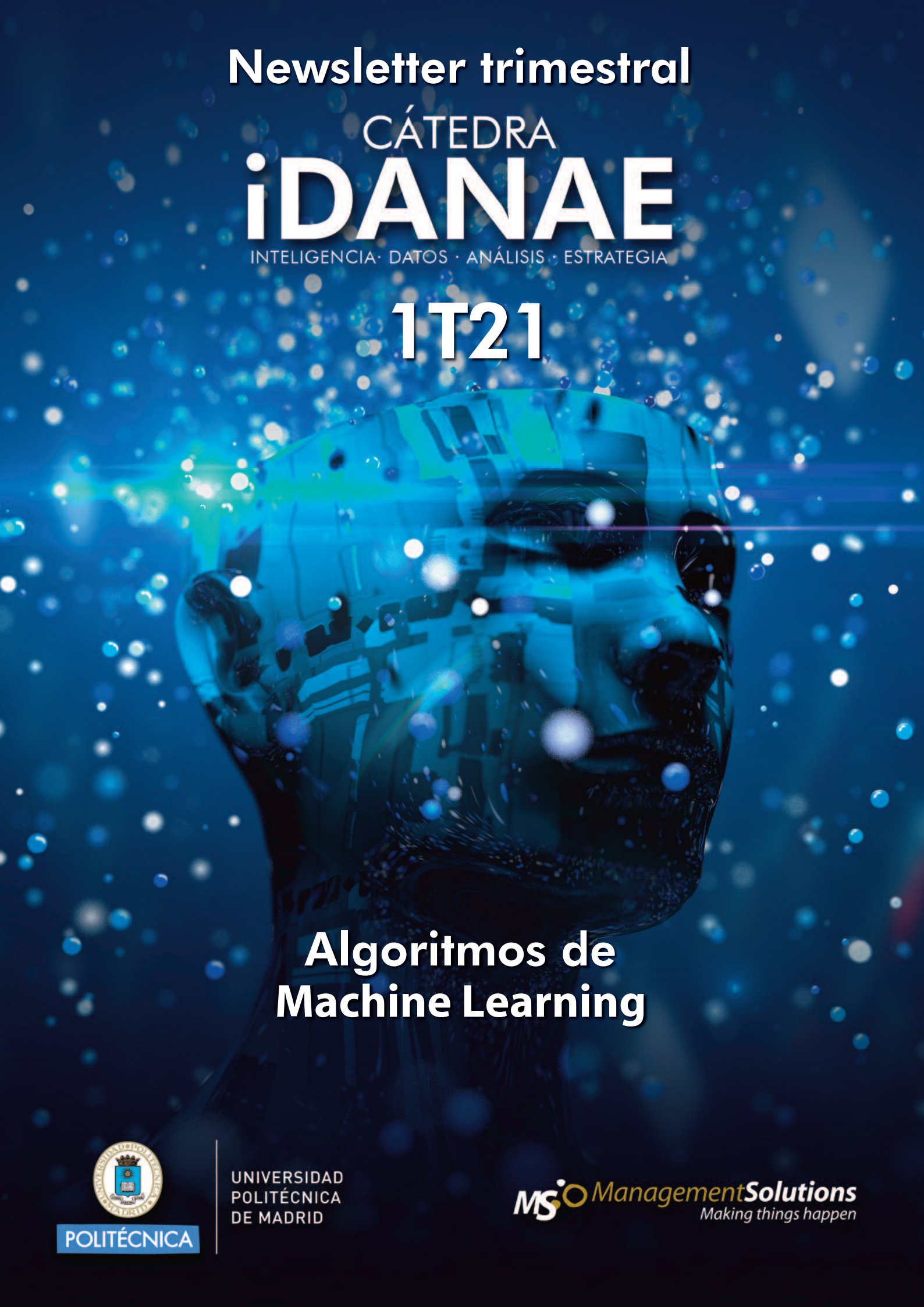
Newsletter trimestral

CÁTEDRA

iDANAE

INTELIGENCIA · DATOS · ANÁLISIS · ESTRATEGIA

1T21



Algoritmos de Machine Learning

UNIVERSIDAD POLITÉCNICA DE MADRID

POLITÉCNICA

ManagementSolutions
Making things happen

Introducción

En los últimos años, el auge de la Inteligencia Artificial, el Aprendizaje automático o *Machine Learning*, y el Aprendizaje profundo o *Deep Learning*, se ha incorporado a todas las industrias y sectores.

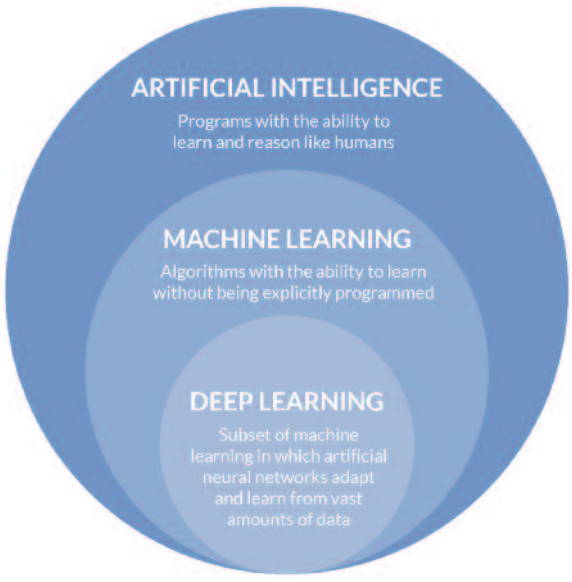
Estos tres términos suelen usarse indistintamente como sinónimos. Sin embargo, conceptualmente se puede considerar que la Inteligencia Artificial es el campo más amplio en el que se pueden enmarcar las otras disciplinas (véase Figura 1).

Más concretamente, la Inteligencia Artificial es un área de la computación que estudia cómo lograr que las máquinas realicen tareas propias del ser humano, es decir, intenta simular el comportamiento inteligente[1]. Por otra parte, el *Machine Learning*[2] se puede considerar un subcampo de la inteligencia artificial que permite a los ordenadores aprender de los datos sin estar programados explícitamente para ello, pudiéndose ver como una posible metodología de cara a dotar a un sistema inteligente de la capacidad de realizar una determinada tarea. Así, el *Machine Learning* puede definirse como el conjunto de métodos que pueden detectar patrones automáticamente en un conjunto de datos y usarlos para predecir datos futuros, o para llevar a cabo otro tipo de decisiones en entornos de incertidumbre[3].

Por último, el *Deep Learning*[4] también permite a los ordenadores aprender de los datos, pero se basa en el uso de complejas redes neuronales, de varias capas, donde el nivel de abstracción aumenta gradualmente mediante transformaciones no lineales de los datos. Dentro de este marco, la presente newsletter se centra en la disciplina del *Machine Learning*.

El auge de este área conlleva múltiples debates en torno a los principales desafíos que presenta, como pueden ser la capacidad para modelar la causalidad[5], la aproximación empresarial en el uso de este campo desde un punto de vista ético[6], organizativo[7], etc., así como otros elementos técnicos como la calidad de los datos, su insuficiencia o no representatividad, la capacidad computacional necesaria, entre otros. Adicionalmente a dichos elementos, y como fundamento de la disciplina, surge también la necesidad de entender los algoritmos de aprendizaje en los que se basa este campo de conocimiento. Por ello, en esta newsletter se pretende dar una visión general sobre los algoritmos de *Machine Learning* más utilizados, con el objetivo de aportar una base de conocimientos para el correcto uso de los diferentes algoritmos.

Figura 1. Relación entre los conceptos de inteligencia artificial, Machine Learning y Deep Learning[8]



[1] Luger, G.F., 2005.
[2] En Samuel, A., 1959. Arthur Samuel define el término de *Machine Learning* como "Programming computers to learn from experience should eventually eliminate the need for much of this detailed programming effort".
[3] Murphy, K.P., 2012.
[4] A pesar de que el origen de este paradigma se remonta a 1943, esta línea de investigación se paralizó debido a la no disponibilidad de la capacidad computacional necesaria. De este modo, el incremento de la capacidad computacional, así como el aumento del volumen y tipología de los datos, ha sido el detonante del apogeo del *Deep Learning* en los últimos años. Gerón, A., 2019.
[5] iDanae, 2020b.
[6] iDanae, 2019.
[7] iDanae, 2020a.
[8] Srivastav, S., 2020.

Tipos de algoritmos

Dentro de los algoritmos de *Machine Learning*, existen diferentes categorizaciones de acuerdo a distintos criterios: grado de supervisión durante el entrenamiento, posibilidad de llevar a cabo un aprendizaje incremental, forma de generalización, etc. Cabe señalar que estas clasificaciones no son necesariamente estáticas, ya que en algunos casos se han incorporado nuevas ramas como consecuencia de la aparición de nuevas técnicas cuya naturaleza no permitía clasificarlas con ninguna de las categorías existentes[9].

En primer lugar es necesario realizar una distinción, basada en la funcionalidad, entre algoritmos descriptivos y predictivos, siendo una categorización extendida dentro del campo de la minería de datos[10]. Un algoritmo descriptivo tiene como objetivo describir la población mediante la exploración de las propiedades y características de los datos, mientras que un algoritmo predictivo tiene como objetivo realizar predicciones sobre eventos futuros desconocidos a partir de datos históricos. A su vez, cada categoría se puede desglosar en varias subcategorías, lo que permite una clasificación más detallada de los algoritmos. En el caso de los algoritmos descriptivos se pueden encontrar las subcategorías (1) clustering, (2) *summarization*, (3) reglas de asociación y (4) *sequence discovery;* mientras que, en el caso de los algoritmos predictivos, se encontrarían (1) clasificación y (2) estimación de valor.

Sin embargo, hoy en día, es más común utilizar una clasificación basada en la tipología de datos disponibles, es decir, de acuerdo a si se dispone de datos etiquetados o no etiquetados. Se pueden distinguir cuatro categorías[11]:

Aprendizaje supervisado: los algoritmos de aprendizaje supervisado intentan modelar las relaciones existentes entre los datos de entrada y la variable objetivo, de forma que se puedan predecir los valores de la variable objetivo para datos nuevos en función de las relaciones aprendidas a partir de datos históricos etiquetados. Para inferir estas relaciones, los conjuntos de entrenamiento utilizados deben incluir los valores de la variable objetivo, llamados comúnmente "etiquetas".

Hay dos tipos principales de aprendizaje supervisado:

- *Clasificación:* su objetivo es predecir la pertenencia a una determinada clase, por ejemplo, determinar si un correo es spam o no. Por tanto, la variable objetivo es discreta (generalmente, se trata de un conjunto finito de clases).

[9]Un ejemplo sería la inclusión del aprendizaje por refuerzo como un nuevo paradigma como complemento a la clasificación en aprendizaje supervisado y no supervisado.
[10]Agyapong, K.B., Hayfron-Acquah, J.B. and Asante, M., 2016.
[11]Géron, A., 2019.

- *Estimación de valor:* su objetivo es predecir un valor numérico, por ejemplo, el precio de venta de una casa. Por tanto, la variable objetivo suele ser continua (aunque la predicción final puede ser un conjunto discreto de valores representados por un indicador, como, por ejemplo, la media de un grupo).

Aprendizaje no supervisado: los algoritmos de aprendizaje no supervisado permiten inferir patrones o relaciones existentes en conjuntos de datos sin etiquetar. Al contrario que en el aprendizaje supervisado, no se predice un valor objetivo (no se dispone de etiquetas), sino que se exploran los datos y se realizan inferencias que permitan descubrir estructuras ocultas en los datos.

Se pueden destacar los siguientes tipos de algoritmos de aprendizaje no supervisado (lista no exhaustiva):

- *Clustering:* su objetivo es encontrar grupos o clusters de individuos o variables similares existentes en los datos, por ejemplo, grupos homogéneos de clientes para llevar a cabo una segmentación comercial.

- *Aprendizaje de reglas de asociación:* su objetivo es descubrir relaciones entre las variables, extrayendo reglas y patrones a partir de los datos; por ejemplo, para determinar patrones de consumo.

- *Otros algoritmos:* existen múltiples algoritmos que no se pueden encuadrar en las categorías anteriores; algunos de ellos se utilizan con frecuencia en otros procesos, como sería el caso de los algoritmos de reducción de dimensionalidad y detección de anomalías, utilizados habitualmente en el preprocesamiento de los datos. Debido a su utilización en fases distintas al entrenamiento del modelo propiamente dicho, estos algoritmos no se tratarán en esta newsletter.

Aprendizaje semisupervisado: se trata de un enfoque híbrido entre el aprendizaje supervisado y no supervisado. En este caso, los datos de entrenamiento son conjuntos parcialmente etiquetados, donde unos pocos ejemplos están etiquetados y una gran cantidad de ejemplos no lo están. El objetivo de estos algoritmos es hacer un uso efectivo de todos los datos disponibles, no solo de los etiquetados. Estos modelos se emplean cuando el etiquetado de los datos puede llevar mucho tiempo, ser demasiado costoso, o no está disponible para parte de la muestra.

Aprendizaje por refuerzo: engloba aquellos problemas en los que un agente aprende a operar en un entorno mediante un proceso de retroalimentación (es decir, prueba y error). El objetivo de estos algoritmos es utilizar la información obtenida de la interacción con el entorno para llevar a cabo aquellas acciones que maximicen la recompensa o minimicen el riesgo; se trata de un aprendizaje iterativo mediante la exploración de las distintas posibilidades. No se dispone de un conjunto de datos de entrenamiento, sino de una meta que el agente ha de lograr, un conjunto de acciones que este puede realizar y retroalimentación sobre su desempeño.

Bajo esta categorización, en los siguientes apartados se ahondará en los principales algoritmos correspondientes a las categorías de aprendizaje supervisado y no supervisado, pues son los más comúnmente utilizados hoy en día.

Algoritmos supervisados y no supervisados

En este apartado, se proporciona una breve visión de los principales algoritmos de aprendizaje supervisado y no supervisado que pueden utilizarse de cara a abordar la resolución de un problema. La elección del modelo dependerá tanto del tipo de datos como del tipo de la tarea a realizar, siendo fundamental la validación del modelo.

Aprendizaje supervisado

Tal y como se expuso en el apartado anterior, los algoritmos de aprendizaje supervisado intentan modelar las relaciones existentes entre los datos de entrada y la variable objetivo. En función de si la variable objetivo es discreta o continua, se tratará de un problema de clasificación o estimación de valor, respectivamente.

Entrenamiento del modelo

En términos generales, un algoritmo de aprendizaje supervisado construye un modelo a partir de modelos patrón, examinando un conjunto de ejemplos e intentando encontrar los parámetros que minimicen o maximicen una función objetivo, comúnmente denominada función de pérdida. De este modo, los parámetros o pesos del modelo se irán ajustando iterativamente[12] con el objetivo de aproximarse al óptimo de dicha función.

Más detalladamente, la función de pérdida se utiliza como una medida del buen funcionamiento del modelo en términos de su poder predictivo. Hay varios factores involucrados en la elección de esta función[13], pero existe una diferenciación clara según el tipo de tarea que se vaya realizar, estimación de valor o problemas de clasificación.

Entre las funciones de pérdida de estimación de valor, una de las más utilizadas es el error cuadrático medio (MSE)[14], calculado como el promedio de la diferencia cuadrática entre las observaciones reales y las predicciones, representándose mediante la siguiente ecuación:

$$MSE = \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{n}$$

De este modo, esta función mide la magnitud promedio del error, independientemente de su dirección.

Por otra parte, una de las funciones de pérdida de clasificación más utilizadas es la entropía cruzada, que mide el rendimiento de aquellos modelos cuya salida es un valor de probabilidad entre 0 y 1. Matemáticamente se puede formular mediante la ecuación:

$$H = -\sum_{i=1}^{n}\sum_{c=1}^{C} y_{i,c}\log(\hat{y}_{i,c})$$

El valor de esta función aumentará a medida que la probabilidad predicha difiera de la etiqueta real, penalizando fuertemente las predicciones que sean seguras, pero erróneas.

Rendimiento del modelo

El rendimiento es un factor clave a la hora de elegir el modelo. Un buen modelo de clasificación no debería ajustarse exclusivamente a los datos con los que ha sido entrenado, sino que debería funcionar correctamente sobre nuevas instancias

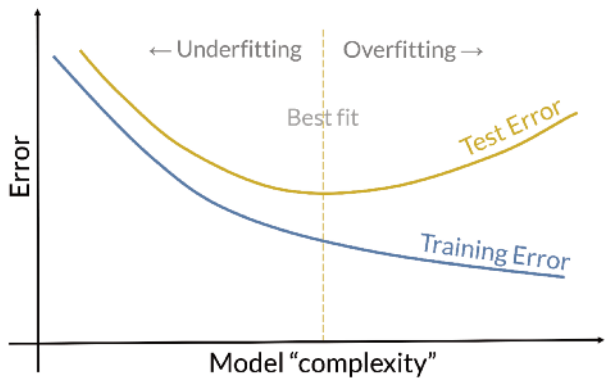
[12]Utilizando algoritmos de optimización como el descenso del gradiente.
[13]Tipo de algoritmo elegido, facilidad para calcular derivadas, porcentaje de valores atípicos en el conjunto de datos, etc.
[14]Otras funciones de pérdida de regresión son el error absoluto medio (MAE), el error absoluto medio suavizado o de Huber, el error de sesgo medio (MBE) o el error log-cosh.

nunca antes vistas[15]. Esta segunda parte hace referencia a la capacidad de generalización del modelo. Por tanto, se distingue entre error de entrenamiento (calculado sobre el conjunto de datos con el que fue entrenado el modelo) y error de validación o de test[16] (calculado sobre un conjunto de datos independiente, que no se usó en el entrenamiento).

Así, se dice que el modelo tiene un buen rendimiento si el error de validación se mantiene acotado, y en el mismo orden de magnitud que el error de entrenamiento. Por tanto, el error de validación es una buena forma de medir el rendimiento del modelo[17]. Para su estimación, debido a que los datos suelen ser limitados, se suele recurrir a utilizar el método *hold-out*, a partir del cual se divide el conjunto de datos en los subconjuntos de entrenamiento y test. Sin embargo, otra alternativa más robusta para estimar este error es la validación cruzada, método mediante el que se divide el conjunto de datos en k grupos, se utilizan k-1 grupos para entrenar y uno para testear el modelo[18,19].

Si el error de generalización fuera significativamente mayor que el error de entrenamiento, el modelo estaría sobreajustando los datos de entrenamiento[20], lo cual implica que el modelo no es capaz de generalizar bien sobre datos nuevos. Esta situación se denomina *overfitting*, y su probabilidad de ocurrencia aumenta a medida que aumenta la complejidad del modelo. Por ello, de acuerdo al principio de la navaja de Ockham[21], entre dos modelos con el mismo error de validación, se seleccionará el más simple. No obstante, si el error de entrenamiento es muy alto, el modelo no estaría siendo capaz de modelar los datos de entrenamiento, y, por consiguiente, tampoco de generalizar a datos nuevos, denominándose *underfitting*. En la Figura 2 se pueden observar estas casuísticas.

Figura 2. Overfitting vs Underfitting[24]



No obstante, a pesar de la utilización de una metodología común, la forma en la que se evaluarán las predicciones variará en función de si se trata de un problema de clasificación o estimación de valor. A continuación, se exponen las principales medidas utilizadas en cada categoría[22].

Clasificación

- *Matriz de confusión:* matriz n x n[23], donde las filas hacen referencia a las clases reales y las columnas a las clases predichas, mostrando de forma explícita el nº de instancias de cada clase predichas correctamente y el nº de instancias en el que una clase se ha confundido con otra.
- *Accuracy:* proporción de predicciones correctas realizadas por el modelo.
- *Precision:* fracción de predichos clasificados correctamente, es decir, cuántos de los predichos son verdaderamente de la categoría predicha.
- *Recall:* fracción de ejemplos clasificados correctamente del total de ejemplos de una determinada clase.
- *F1-Score:* media armónica de las medidas *Precision y Recall.*

Estimación de valor

- *Error cuadrático medio:* promedio de la diferencia cuadrática entre las observaciones reales y las predicciones.
- *Raíz del error cuadrático medio:* raíz cuadrada del promedio de las diferencias al cuadrado entre las observaciones reales y las predicciones.
- R^2: proporción de variabilidad explicada por el modelo.

[15]Extraídas de la misma distribución que se utilizó para crear el modelo.
[16]El cual pretende medir el rendimiento del modelo en un entorno real, denominado error de generalización.
[17]Un buen rendimiento en el conjunto de test será un indicador útil de un buen desempeño en los nuevos datos en general.
[18]La repetición de este proceso genera k estimaciones del error cuyo promedio se utiliza como estimación final.
[19]Otras estrategias de validación son Leave-One-Out CrossValidation, Stratified K-Fold Cross Validation o Repeated K-Fold Cross Validation
[20]Aprendiendo incluso el ruido contenido en los datos de entrenamiento.
[21]Cuanto menos complejo sea un modelo, más probable es que un buen resultado empírico no se deba solo a las peculiaridades de nuestra muestra.
[22]Véase Zheng, A., 2015. para una descripción más detallada.
[23]Siendo n el número de clases a predecir.
[24]Saxena, S., 2020.

Finalmente, es preciso recordar que la consecución de un buen rendimiento no depende únicamente del algoritmo elegido. Sin un preprocesamiento de los datos no se obtendrán unos resultados precisos, independientemente del algoritmo[25].

Principales algoritmos

Una vez abordado tanto el entrenamiento como la validación de un modelo, se exponen los principales algoritmos recogidos bajo la categoría de aprendizaje supervisado. No se realiza una diferenciación entre algoritmos de clasificación y de estimación de valores debido a que algunos se podrán utilizar para ambas tareas.

Regresión lineal

Es un enfoque lineal para modelar la relación entre una variable respuesta continua (variable dependiente) y una o más variables explicativas (variables independientes). El caso de una variable explicativa se denomina regresión lineal simple, representándose matemáticamente mediante la ecuación:

$$Y=a+bX$$

Donde X es la variable independiente e Y es la variable dependiente. La pendiente es b y el intercept es a, siendo estos los coeficientes del modelo que se ha de estimar, ajustándose habitualmente por el método de mínimos cuadrados. En la Figura 3, se puede visualizar un ejemplo de regresión lineal simple.

Este modelo suele ser un buen baseline para los problemas de estimación de valor. No obstante, es importante corroborar el cumplimiento de las hipótesis del modelo: (1) linealidad, (2) homocedasticidad, 3) independencia y (4) normalidad[26].

Regresión logística

Es un método estadístico que se utiliza en problemas de clasificación, prediciendo la probabilidad de ocurrencia de los diferentes resultados posibles. En concreto, este algoritmo busca un hiperplano que sea capaz de separar linealmente las clases (véase Figura 4).

A diferencia de la regresión lineal, la variable dependiente es categórica, es decir, toma un número limitado de valores. Cuando esta variable solo tiene dos valores posibles, se denomina regresión logística simple, prediciendo la probabilidad de ocurrencia de ese evento[27]. Matemáticamente, se formula mediante la ecuación:

$$y= \sigma(a+bX)$$

[25]En esta línea, surge la importancia de la ingeniería de características, la optimización de hiperparámetros y el uso de técnicas de regularización para mejorar el rendimiento del modelo.
[26]Gujarati, D., 1978.
[27]Una vez calculadas estas probabilidades, se establecerá un umbral (comúnmente 0.5), bajo el cual, si la probabilidad de ocurrencia es mayor a este valor, el evento ocurrirá.

Figura 3. Ejemplo de regresión lineal simple

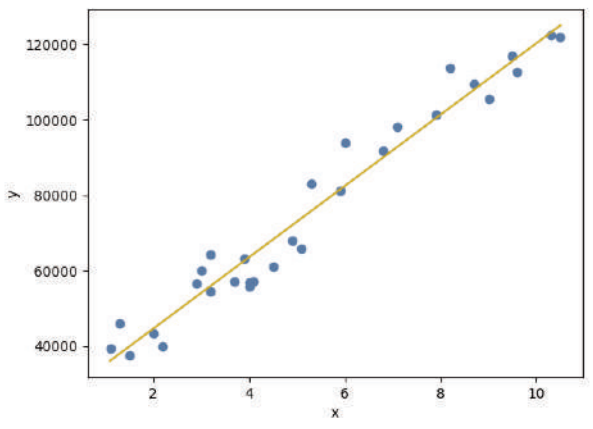
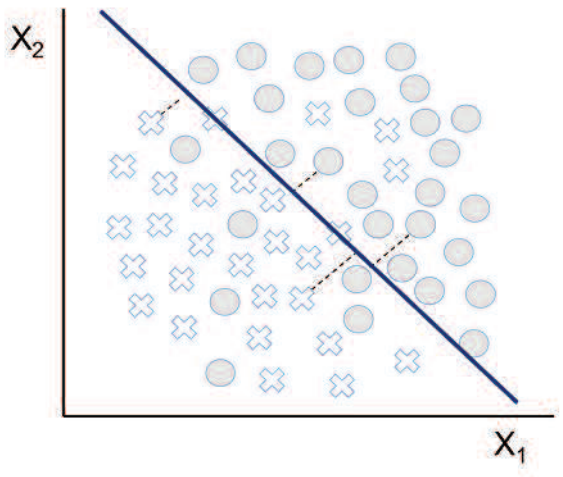


Figura 4. Regresión Logística



Donde σ es la función sigmoide[28], la cual se encarga de asignar probabilidades a los valores predichos (figura 5)

Como consecuencia de su simple formulación, este modelo se considera un buen baseline a emplear en problemas de clasificación.

K-vecinos más cercanos (k-nn)

Es un algoritmo de aprendizaje perezoso[29] no paramétrico que se puede usar para resolver tanto problemas de clasificación como de estimación de valor.

Más concretamente, dada una nueva instancia que se quiere clasificar, se calculan las distancias de este punto respecto a las instancias que conforman el conjunto de entrenamiento, se seleccionan las k-instancias más cercanas y se calcula la media de las etiquetas numéricas (en un problema de regresión) o se selecciona la clase más frecuente (en un problema de clasificación)[30]. En la Figura 6, se presenta un ejemplo de 3-nn, donde la nueva instancia pertenecerá a la clase B.

Su uso generalizado se debe a su buen rendimiento, sencillez y versatilidad; aunque también hay que tener en cuenta sus inconvenientes: (1) dificultad para determinar el k óptimo; (2) necesidad de almacenar el conjunto de entrenamiento; y (3) tiempo de ejecución proporcional al tamaño de los datos.

Árboles de decisión

Este algoritmo se basa en la estrategia "divide y vencerás"[31], utilizándose tanto para problemas de clasificación como de estimación de valor[32], siendo su objetivo predecir el valor de la variable objetivo a partir de reglas de decisión simples inferidas a partir de los datos. En otras palabras, este algoritmo subdivide el espacio de características en regiones, mapeando cada una de las regiones finales a un valor de la variable objetivo. De este modo, se trata de un modelo simple e interpretable, que proporciona una explicación de los resultados inferidos por el modelo. En la Figura 7, se puede visualizar un árbol de decisión utilizado en una tarea de clasificación binaria.

Naive Bayes

Es un algoritmo de clasificación que se basa en el teorema de Bayes[33]. Este clasificador supone que las variables son independientes y que cada una contribuye de forma equivalente al resultado. No obstante, estas hipótesis no se suelen satisfacer en problemas reales; el supuesto de independencia rara vez se cumple, pero, aun así, este algoritmo suele funcionar bien en la práctica[34].

[28] La función sigmoide se representa mediante la ecuación: $\sigma(z) = \frac{1}{1+e^{-z}}$ la cual toma un valor real y le asigna un valor entre 0 y 1.
[29] Se dice que un algoritmo es perezoso cuando no aprende una función discriminativa de los datos de entrenamiento, sino que memoriza el conjunto de entrenamiento, retrasando el proceso inductivo hasta la llegada de una instancia de prueba. Daelemans, W. and Van den Bosch, A., 2005.
[30] Este procedimiento se basa en la hipótesis de que instancias de la misma clase (o similares en el caso de problemas de regresión), tendrán un comportamiento similar, estando cerca unas de otras.
[31] Véase Brassard, G. and Bratley. P., 2006. para más información.
[32] También es habitual, denominar como árboles de predicción a los árboles utilizados en tareas de estimación de valor.
[33] El teorema de Bayes se usa en Estadística para calcular probabilidades condicionales, siendo su formulación:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}, \qquad P(A|B) = \frac{P(A \cap B)}{P(B)}$$

[34] No obstante, esta limitación hace que los valores de las probabilidades que se obtienen deban considerarse con cautela.
[35] Atul, 2020.

Figura 5. Función sigmoide

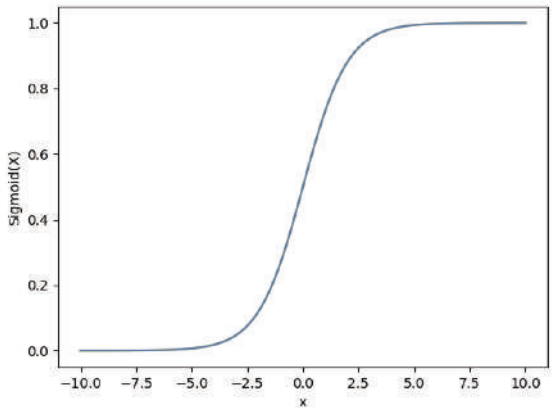
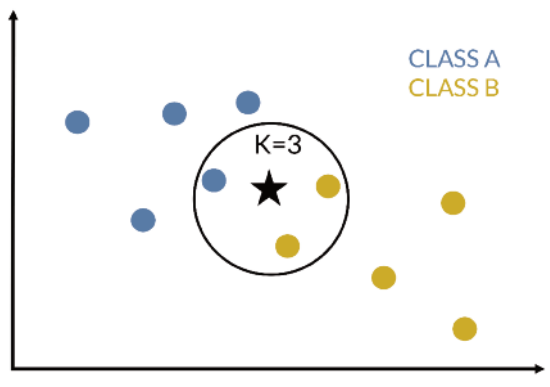


Figura 6. Ejemplo de 3-vecinos más cercanos[35]



Matemáticamente, mediante el teorema de Bayes, se estable la siguiente relación[37]:

$$P(y|X) = \frac{P(X|y)P(y)}{P(X)} \propto P(y)\prod_{i=1}^{n} P(x_i|y)$$

Donde y es la clase y $X=(x_1,x_2,...,x_n)$ el conjunto de características. De este modo, la clase predicha será aquella que presente la mayor probabilidad.

$$y = argmax_y P(y)\prod_{i=1}^{n} P(x_i|y)$$

Se trata de una técnica simple y robusta, que tolera de forma satisfactoria los valores missing. No obstante, si una variable categórica tiene una categoría que no se observa en el conjunto de entrenamiento, el modelo asignará una probabilidad de 0 y no se podrá realizar una predicción.

Support Vector Machine

Este popular algoritmo se puede utilizar tanto para tareas de clasificación como de estimación de valor, aunque su uso está más extendido en tareas de clasificación. Su objetivo radica en encontrar un hiperplano, en un espacio N-dimensional, que separe las clases de puntos, transformando para ello las variables explicativas mediante funciones kernel. Estas funciones se utilizan para mapear funciones separables no linealmente en funciones separables linealmente de dimensión superior.

Más concretamente, se debe encontrar el hiperplano en un espacio de dimensión superior al espacio en el que se encuentran los datos que maximice la distancia entre los puntos de datos de las clases (denominada margen). De este modo, se intenta tener un mayor margen de confianza para predicciones futuras. En la Figura 8, se puede visualizar un ejemplo en el que se representa el hiperplano que separa mejor los datos, y el margen.

De cara a emplear este algoritmo, hay que tener en cuenta que su entrenamiento puede prolongarse bastante en el tiempo si se dispone de un gran conjunto de datos.

Redes neuronales

El incremento de la capacidad computacional, así como el aumento del volumen y tipología de los datos, ha dado lugar a un amplio desarrollo en el ámbito de las redes neuronales[38], lo que ha propiciado un gran auge de la disciplina *Deep Learning*.

En este contexto, en el que el procesamiento de los datos no estructurados (texto, audio, imágenes y video) se ha convertido en una importante fuente de información, las áreas de *Natural Language Processing* y *Computer Vision* han avanzado rápidamente.

No obstante, debido al elevado número de tipos de redes neuronales existentes y a la complejidad de las mismas, este tipo de algoritmos no se aborda en esta newsletter.

[37]Las probabilidades $P(y)$ y $P(x_i|y)$ se calculan a partir de los datos de entrenamiento.
[38]No siendo todas las arquitecturas de redes neuronales necesariamente supervisadas, pudiendo ser también no supervisadas, p.ej. autoencoders.
[39]Mohan, A., 2019.
[40]VanderPlas, J., 2016.

Figura 7. Ejemplo de árbol de decisión[39]

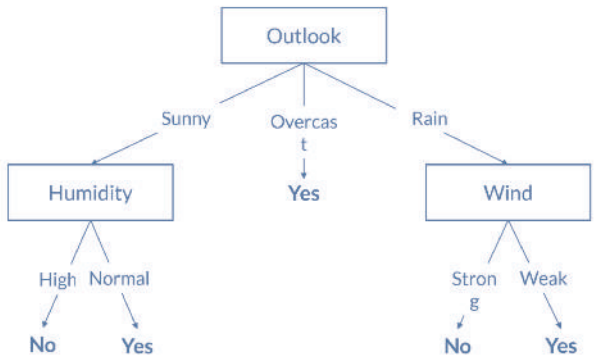
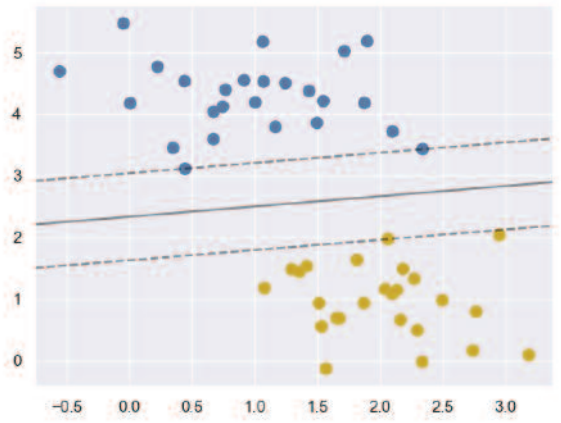


Figura 8. Support Vector Machine[40]



Técnicas para la mejora del aprendizaje. Métodos Ensemble

Los métodos ensemble se basan en la combinación de múltiples modelos en aras de obtener mejores resultados utilizando el conocimiento conjunto, pudiéndose utilizar tanto para tareas de clasificación como de estimación de valor. Hay dos técnicas principales:

- **Bagging:** los modelos se entrenan en paralelo y la combinación de las predicciones se utiliza como predicción final.
- **Boosting:** los modelos se entrenan secuencialmente de forma que cada modelo trata de corregir los errores del modelo anterior.

A continuación, se describen brevemente algunos de los métodos ensemble más utilizados[40]:

- **Random Forest:** este algoritmo de bagging crea varios árboles de decisión de forma paralela y los fusiona combinando sus predicciones para obtener predicciones más precisas y estables.
- **Gradient Boosting:** como el propio nombre indica, se trata de un algoritmo boosting. Este método combina secuencialmente una serie de árboles, de modo que cada árbol se centre en corregir el error del árbol anterior. Más concretamente, esto se realiza modelando el error de predicción del árbol de decisión anterior. En particular, una implementación específica de este algoritmo es el XGBoost, que trata de mejorar la velocidad de ejecución y el rendimiento del modelo.
- **AdaBoost:** es un algoritmo similar al anterior, diferenciándose en el modo en el que cada árbol corrige el error del árbol anterior. En este caso, en cada iteración se actualizan las ponderaciones de las observaciones en función de los errores cometidos, es decir, se asigna mayor peso a aquellas observaciones que el modelo no haya sido capaz de predecir correctamente.

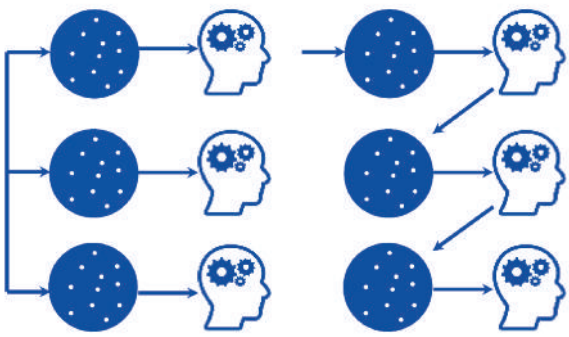
Aprendizaje no supervisado

Los algoritmos de aprendizaje no supervisado tienen como objetivo descubrir patrones ocultos en datos sin etiquetar. La ausencia de estas etiquetas supone la ausencia de un punto de referencia con el que evaluar la calidad del modelo, lo que aumenta la complejidad de la validación del modelo. En este apartado, se exponen algunos ejemplos enmarcados en las categorías descritas anteriormente.

Clustering

Estos algoritmos[41], cuyo objetivo es descubrir agrupaciones naturales en los datos, se basan en la hipótesis de que observaciones pertenecientes al mismo grupo deben tener características similares; y, al contrario, observaciones

Figura 9. Bagging - Figura 10. Boosting



[40]Otros algoritmos ensemble que pueden resultar de interés son LightGBM y CatBoost.
[41]Véase Aggarwal, C.C. and Reddy, C.K., 2013. para más información.

pertenecientes a distintos grupos deben tener características diferentes. De este modo, este tipo de algoritmos pretenden agrupar los datos de forma que las instancias de un grupo sean similares entre sí (minimizando la distancia intra-grupo) y diferentes respecto a otros grupos (maximizando la distancia inter-grupo).

Existe una gran variedad de algoritmos de clustering, y cada uno de ellos ofrece un enfoque diferente de cara a abordar esta tarea[42]. Las principales categorías en las que se agrupan estos algoritmos son:

- **Clustering partitivo:** las particiones se generan de acuerdo con una medida de similitud. Ej. k-means, pam.
- **Clustering jerárquico:** busca construir una jerarquía de clusters, denominada dendograma. Existen dos enfoques claramente diferenciados:
  - Aglomerativo (enfoque de abajo hacia arriba): cada observación constituye su propio cluster inicialmente, y se van fusionando de acuerdo a una condición de similitud. El algoritmo suele detenerse cuando se cumple un cierto criterio, con lo que genera varios clusters. Ej. agnes.
  - Divisivo (enfoque de arriba a abajo): todas las observaciones se incluyen inicialmente en un único cluster y se realizan divisiones de forma recursiva. El algoritmo suele detenerse cuando se cumple un cierto criterio, con lo que genera varios clusters. Ej. diana.
- **Clustering difuso:** la asignación de las instancias a cualquiera de los clusteres no es fija, pudiendo una instancia pertenecer a más de un cluster. Ej. fanny, daisy.
- **Clustering basado en densidades:** los clusters se crean según la densidad de los puntos de datos. Ej. DBSCAN, OPTICS, meanshift.
- **Otros:** basados en grafos (espectral), basados en modelos probabilísticos (modelos de mixturas), entre otros.

La elección de qué algoritmo se debe emplear es compleja y no inmediata, y es necesaria la experimentación sobre el conjunto de datos. Por tanto, no existe un mejor algoritmo, sino que la adecuación de uno u otro dependerá de la distribución inherente a los datos.

En cualquier caso, es importante cuantificar el rendimiento de la agrupación realizada. En general, se puede distinguir entre medidas extrínsecas, las cuales requieren el conocimiento de las etiquetas verdaderas[43], y medidas intrínsecas, que miden la bondad de los resultados sin tener en cuenta información externa. De acuerdo con la propia definición de aprendizaje no supervisado, bajo la cual no se disponen de datos etiquetados, en esta newsletter se pone el foco en la validación intrínseca, siendo habitual distinguir entre medidas de cohesión y de separación. Concretamente, las medidas de cohesión determinan el grado de proximidad de los puntos que conforman un cluster, mientras que las medidas de separación determinan el grado de separación de los clusters. Algunas de las medidas intrínsecas más utilizadas son:

- *Coeficiente de Silhouette:* calculado para cada instancia del conjunto de datos como

$$\frac{b-a}{\max(a,b)},$$

  donde a mide la cohesión como la distancia media de esa instancia al resto de instancias de ese cluster y b la separación como la distancia media de esa instancia al conjunto de instancias que conforman el cluster más cercano. El cálculo de esta medida a nivel de cluster se calculará como la media de este coeficiente para todas sus instancias.

  Este coeficiente está acotado al intervalo [-1, 1] donde un valor más alto será indicativo de clusters mejor definidos[44].
- *Índice Davies-Bouldin:* se calcula como la similitud promedio de cada cluster con su cluster más similar. Cuanto menor sea su valor, mejor será la partición, siendo el valor mínimo de este índice 0.
- *Índice de Dunn:* se define como la proporción entre la mínima distancia inter-grupo y la máxima distancia intra-grupo. Cuanto mayor sea su valor, mejor será la agrupación.

Finalmente, se exponen algunos de los algoritmos de clustering más utilizados:

- **K-means:** es probablemente el algoritmo de clustering más conocido y utilizado. Especificando un número de clusters a priori, el algoritmo permite asignar cada observación al

[42]Por ejemplo, algunos algoritmos requieren que se especifique el número de clusters, mientras otros requieren que se especifique la distancia máxima para que dos observaciones se consideren cercanas.
[43]Evaluando el grado de coincidencia respecto a éstas.
[44]Un valor de 0 hará referencia a clusters superpuestos y un valor cercano a -1 a clusters erróneos.

centroide más cercano[45]. En particular, se seleccionarán al azar k centroides, y mediante un proceso iterativo se asigna cada punto a su centroide más cercano, actualizando en cada paso el valor de los centroides[46].

- **DBSCAN (density-based spatial clustering of applications with noise):** es un algoritmo basado en densidades, es decir, asume la existencia de un cluster cuando identifica una región densa, lo que permite detectar como outliers aquellos puntos que no superan un umbral de densidad establecido. De esta manera, este algoritmo se basa en detectar puntos con densidad suficiente y construir clusters en torno a ellos, añadiendo puntos cercanos al mismo. Para ello, es necesario especificar la distancia máxima para que dos puntos se consideren vecinos y el número mínimo de puntos para conformar un cluster.

En la Figura 11, se puede visualizar una comparativa de los resultados obtenidos con estos dos métodos sobre distintos conjuntos de datos.

Así pues, se pueden percibir las siguientes ventajas del algoritmo DBSCAN respecto al algoritmo k-means: (1) Puede encontrar clusters con formas geométricas arbitrarias, (2) Tiene capacidad de detectar outliers, (3) No es necesario especificar el número de clusters y (4) Consistencia en sus resultados en diferentes ejecuciones. En contrapartida, el rendimiento del algoritmo k-means es ligeramente mejor gracias a su simplicidad.

Aprendizaje de reglas de asociación

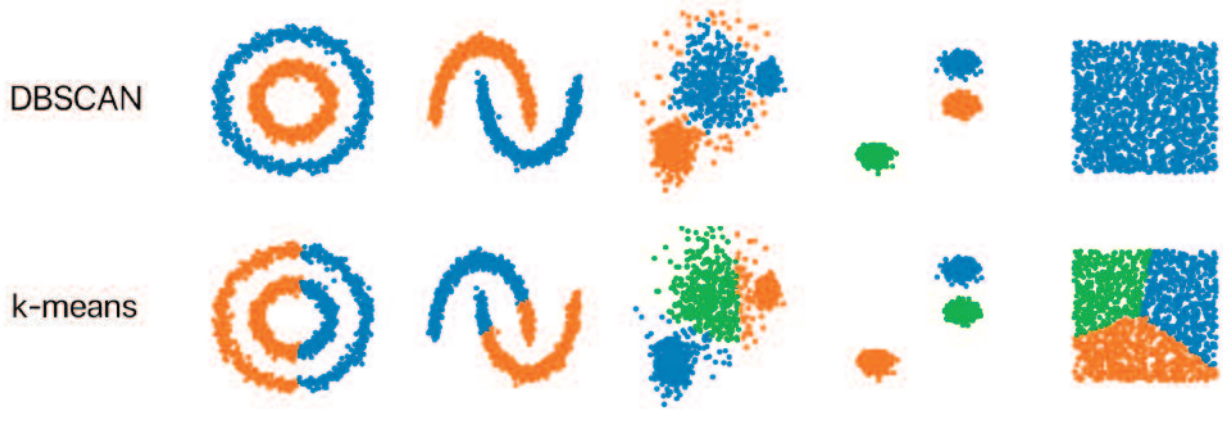
Recoge métodos basados en reglas que se utilizan para descubrir relaciones ocultas en los datos[47]. Más concretamente, una regla de asociación consiste en un antecedente (si) y un consecuente (entonces), reflejando esta implicación una relación de coocurrencia, es decir, el consecuente es algo que ocurre cuando ocurre un antecedente[48].

$$\{\text{Antecedente}\} \rightarrow \{\text{Consecuente}\}$$

Cabe mencionar que este tipo de extracción de reglas a partir de los datos tiene dos problemas: (1) descubrir reglas a partir de grandes conjuntos de datos puede resultar computacionalmente costoso y (2) algunos de los patrones extraídos pueden ocurrir por casualidad. Para ayudar a solventar estos problemas, existen una serie de métricas que

[45]Calculado como la media de los puntos que conforman dicho cluster.
[46]Hasta que se alcanza el criterio de convergencia establecido.
[47]El aprendizaje de reglas de asociación a veces se denomina "análisis de la cesta de la compra", ya que fue su primer área de aplicación.
[48]Este enfoque puede verse ampliamente extendido por la teoría de lógica borrosa, la cual no se aborda en esta newsletter.
[49]Chauhan, N. S., 2020.

Figura 11. Comparativa k-means y DBSCAN[49]



determinar la fuerza de la asociación: (1) *Support*: frecuencia con la que aparece un elemento o conjunto de elementos en el conjunto de datos, (2) *Confidence*: frecuencia con la que la regla es cierta y (3) *Lift*: aumento en la probabilidad de que ocurra el consecuente cuando el antecedente ya ha ocurrido.

- **Algoritmo a priori:** fue el primer algoritmo que se propuso para esta problemática. Conceptualmente, se basa en generar un conjunto de elementos frecuentes (elementos cuyo *support* es mayor que un umbral establecido) y en generar reglas de asociación a partir de este conjunto reducido, seleccionando aquellas con un alto nivel de *confidence*.

- **Algoritmo ECLAT (Equivalence Class Transformation):** este algoritmo utiliza una búsqueda en profundidad para encontrar el conjunto de elementos más frecuentes, siendo así un algoritmo más rápido que el algoritmo a priori.

Algoritmos más frecuentes

Una vez que se han expuesto los principales algoritmos de *Machine Learning*, resulta interesante visualizar cuáles son los algoritmos más utilizados en aplicaciones reales. Para ello, en la Figura 13 se presentan los resultados obtenidos según la encuesta realizada por KDnuggets en el periodo 2018/2019. Bajo este estudio, los tres métodos más utilizados son la regresión, los árboles de decisión y los algoritmos de clustering, tratándose de modelos simples e interpretables.

Respecto a la encuesta realizada en 2017[50] , se puede observar un aumento en el uso de técnicas de *Deep Learning* (redes adversarias generativas, redes neuronales recurrentes, redes convolucionales, …), acompañado de un descenso en el uso de algoritmos como el SVM y las reglas de asociación.

[50]Piatetsky, G., 2017.
[51]Great Learning Team, 2020.
[52]Mayo, M., 2019.

Figura 12. Métricas utilizadas para analizar una regla de asociación[51]

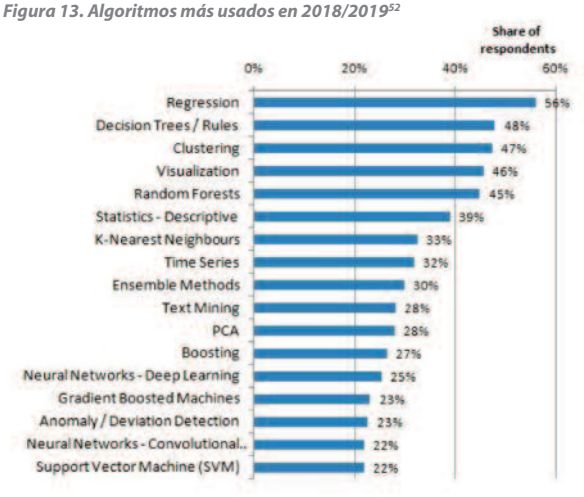
$$Rule: X \Rightarrow Y$$

$$Support = \frac{frq(X,Y)}{N}$$

$$Confidence = \frac{frq(X,Y)}{frq(X)}$$

$$Lift = \frac{Support}{Supp(X)xSupp(Y)}$$

Figura 13. Algoritmos más usados en 2018/2019[52]



¿Cómo llevar estos algoritmos a la práctica?

Tras exponer de forma breve los principales algoritmos de *Machine Learning*, se ha de abordar cómo se pueden utilizar desde un punto de vista práctico para resolver una determinada tarea.

Una pregunta frecuente radica en qué lenguaje utilizar para tareas de ML, dada la gran diversidad de de lenguajes de programación existentes[53]. No existe un mejor lenguaje de programación, aunque sí que hay algunos lenguajes que son más apropiados que otros en función del problema que se va abordar. A continuación, se exponen algunos de los más populares[54]:

- **Python:** se ha convertido en el lenguaje de programación preferido en el ámbito del *Machine Learning*[55] gracias a: (1) su amplio ecosistema de bibliotecas (scikit-learn, numpy, pandas, keras, tensorflow, …), (2) la legibilidad del código gracias a su simple sintaxis y (3) su naturaleza multiparadigma y flexible.

- **R:** es otra de las opciones más populares, siendo un lenguaje orientado a la estadística, que proporciona una amplia gama de paquetes. Concretamente, este lenguaje se centra en el modelado y el análisis estadístico. Entre sus principales inconvenientes, destaca su curva de aprendizaje y su velocidad.

- **Julia:** es un lenguaje de programación de alto nivel que fue diseñado para el análisis numérico de alto rendimiento y la ciencia computacional. Julia intenta suplir las carencias de Python y R en términos de rendimiento, manteniendo un desarrollo rápido. No obstante, no ha conseguido alcanzar la popularidad de Python o R.

Además, todo ello se ha generalizado al poder disponer de estos lenguajes tanto en entornos de laboratorio como en entornos productivos, bien on-premise, bien en Cloud. Estos entornos integran grandes avances recientes en la tecnología (como la incorporación de GPU con extensiones para *Deep Learning*, o de tensor processing units o TPU), lo que aumenta de forma drástica tanto la potencia de cálculo como el rendimiento energético de los sistemas de cómputo. Todo ello permite aprovechar la infraestructura tecnológica disponible, a un coste cada vez menor.

[53] Algunos de propósito general y otros orientados a un determinado fin.
[54] Otros lenguajes populares son Scala, Java y C++.
[55] Además, Python ocupa el primer lugar en la clasificación anual de lenguajes de programación populares del IEEE Spectrum.

Bibliografía

Aggarwal, C.C. and Reddy, C.K., 2013. Data Clustering: Algorithms and Applications. CRC Press.

Agyapong, K.B., Hayfron-Acquah, J.B. and Asante, M., 2016. An overview of data mining models (Descriptive and predictive). International Journal of Software & Hardware Research in Engineering, 4(5), pp.53-60.

Ameisen, E., 2018. Always start with a stupid model, no exceptions. Disponible en: https://blog.insightdatascience.com/always-start-with-a-stupid-model-no-exceptions-3a22314b9aaa

Atul, 2020. K-Nearest Neighbors Algorithm Using Python. Disponible en: https://www.edureka.co/blog/k-nearest-neighbors-algorithm/

Bishop, C.M., 2006. Pattern recognition and Machine Learning. Springer.

Bohnhoff, T., 2019. Machine Learning as a Service — The Top Cloud Platform and AI Vendors. Disponible en: https://medium.com/appanion/machine-learning-as-a-service-the-top-cloud-platform-and-ai-vendors-2df45d51374d

Boudreau, E., 2020. What Language Should You Learn For Data Science In 2021? Disponible en: https://towardsdatascience.com/what-language-should-you-learn-for-data-science-in-2021-fdeebb88d6e

Brassard, G. and Bratley, P., 2006. Fundamentos de Algoritmia. Prentice Hall.

Burkov, A., 2019. The Hundred-Page Machine Learning Book.

Chauhan, N. S., 2020. DBSCAN Clustering Algorithm in Machine Learning. Disponible en: https://www.kdnuggets.com/2020/04/dbscan-clustering-algorithm-machine-learning.html

Daelemans, W. and Van den Bosch, A., 2005. Memory-based language processing. Cambridge University Press.

Géron, A., 2019. Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow, 2nd Edition. O'Reilly Media, Inc.

Goldstein, M. and Uchida, S., 2016. A comparative evaluation of unsupervised anomaly detection algorithms for multivariate data. PloS one, 11(4), p.e0152173.

Great Learning Team, 2020. What is Apriori Algorithm? Apriori Algorithm Explained. Disponible en: https://www.mygreatlearning.com/blog/apriori-algorithm-explained/

Gujarati, D., 1978. Basic Econometrics. International Edition, Prentice-Hall International, Inc.

Hadidi, S., 2016. Anthony Goldbloom gives you the Secret to winning Kaggle competitions. Disponible en: https://www.kdnuggets.com/2016/01/anthony-goldbloom-secret-winning-kaggle-competitions.html

Harasymiv, V., 2015. Lessons from 2 Million Machine Learning Models on Kaggle. Disponible en: https://www.kdnuggets.com/2015/12/harasymiv-lessons-kaggle-machine-learning.html

iDanae, 2019. Ética e Inteligencia Artificial.

iDanae, 2020a. Data Democratization.

iDanae, 2020b. Una introducción a la causalidad y el aprendizaje automático.

Jain, N. and Srivastava, V., 2013. Data mining techniques: a survey paper. IJRET: International Journal of Research in Engineering and Technology, 2(11), pp.2319-1163.

Kinha, Y., 2020. An easy guide to choose the right Machine Learning algorithm. Disponible en: https://www.kdnuggets.com/2020/05/guide-choose-right-machine-learning-algorithm.html

Luger, G.F., 2005. Artificial intelligence: structures and strategies for complex problem solving. Pearson education.

Mayo, M., 2019. Top Data Science and Machine Learning Methods Used in 2018, 2019. Disponible en: https://www.kdnuggets.com/2019/04/top-data-science-machine-learning-methods-2018-2019.html

Mohan, A., 2019. Decision Tree Algorithm With Hands-On Example. Disponible en: https://medium.datadriveninvestor.com/decision-tree-algorithm-with-hands-on-example-e6c2afb40d38

Murphy, K.P., 2012. Machine Learning: a probabilistic perspective. MIT press.

Ng, A. and Soo, K., 2016. Principal Component Analysis Tutorial. Disponible en: https://algobeans.com/2016/06/15/principal-component-analysis-tutorial/

Patel, A.A., 2019. Hands-On Unsupervised Learning Using Python: How to Build Applied Machine Learning Solutions from Unlabeled Data. O'Reilly Media.

Piatetsky, G., 2017. Top Data Science and Machine Learning Methods Used in 2017. Disponible en: Top Data Science and Machine Learning Methods Used in 2017 - KDnuggets

Samuel, A., 1959. Some studies in Machine Learning using the game of checkers. IBM Journal.

Sancho, F., 2019. Redes Neuronales: una visión superficial. Disponible en: http://www.cs.us.es/~fsancho/?e=72

Sarbach, A., 2012. Bases biológicas del comportamiento. Disponible en: https://filosert.wordpress.com/temas/4-bases-biologicas-del-comportamiento/

Sarkar, P.K., 2019. Association rule learning: A brief overview. Disponible en: https://medium.com/data-science-vibes/association-rule-part-1-f37e3cc545a0

Saxena, S., 2020. Underfitting vs. Overfitting (vs. Best Fitting) in Machine Learning. Disponible en: https://www.analyticsvidhya.com/blog/2020/02/underfitting-overfitting-best-fitting-machine-learning/

Shin, T., 2020. All Machine Learning Algorithms You Should Know in 2021. Disponible en: https://towardsdatascience.com/all-machine-learning-algorithms-you-should-know-in-2021-2e357dd494c7

Singh, J., 2020. Major Machine Learning Algorithms. Disponible en: https://inblog.in/Major-Machine-Learning-Algorithms-K8qQfEf2ey

Springboard, 2020. Best language for Machine Learning: Which Programming Language to Learn. Disponible en: https://in.springboard.com/blog/best-language-for-machine-learning/

Srivastav, S., 2020. Artificial Intelligence, Machine Learning, and Deep Learning. What's the Real Difference? Disponible en: https://medium.com/swlh/artificial-intelligence-machine-learning-and-deep-learning-whats-the-real-difference-94fe7e528097

VanderPlas, J., 2016. Python data science handbook: Essential tools for working with data. O'Reilly Media, Inc.

Veen, F.V., 2016. The Neural Network Zoo. Disponible en: https://www.asimovinstitute.org/neural-network-zoo/

Witten, I.H., Frank, E., Pal, C.J. and Hall, M., 2016. Data Mining: Practical Machine Learning Tools and Techniques. Morgan Kaufmann.

Zheng, A., 2015. Evaluating Machine Learning models: a beginner's guide to key concepts and pitfalls.

Zhou, Z.H., 2012. Ensemble methods: foundations and algorithms. CRC press.

Autores

Ernestina Menasalvas (UPM)

Alejandro Rodríguez (UPM)

Manuel Ángel Guzmán (Management Solutions)

Segismundo Jiménez (Management Solutions)

Silvia Duque (Management Solutions)

UNIVERSIDAD POLITÉCNICA DE MADRID

La Universidad Politécnica de Madrid es una Entidad de Derecho Público de carácter multisectorial y pluridisciplinar, que desarrolla actividades de docencia, investigación y desarrollo científico y tecnológico.

www.upm.es

Management Solutions es una firma internacional de consultoría, centrada en el asesoramiento de negocio, finanzas, riesgos, organización, tecnología y procesos, que opera en más de 40 países y con un equipo de más de 2.500 profesionales que trabajan para más de 900 clientes en el mundo.

www.managementsolutions.com

Para más información visita

blogs.upm.es/catedra-idanae/