Contenido

1. Principios de los sistemas de aprendizaje automático.

El **aprendizaje automático (Machine Learning)** es una rama de la **Inteligencia Artificial (IA)**, que a su vez forma parte del amplio campo de la **informática**. Estas áreas no siempre presentan límites claramente definidos, ya que el aprendizaje automático comparte fundamentos con otras disciplinas, como la **estadística** o la **matemática aplicada**.

En términos sencillos, el aprendizaje automático permite crear **sistemas capaces de mejorar su rendimiento de manera autónoma** a medida que adquieren experiencia o procesan más datos. Una definición breve podría ser:

"Sistemas que optimizan la forma en que realizan una tarea conforme acumulan experiencia o datos."

Gracias al aprendizaje automático, los sistemas de IA se vuelven **adaptativos**, es decir, pueden aprender de los datos y ajustar su comportamiento sin necesidad de ser programados de forma explícita.

Dentro de este campo se encuentra el **aprendizaje profundo (Deep Learning)**, una subdisciplina del aprendizaje automático basada en redes neuronales artificiales con múltiples capas. Este tema se abordará con más detalle en unidades posteriores.

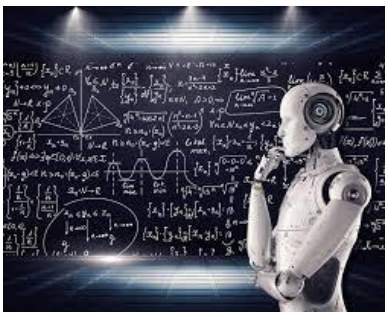
De la definición anterior surgen tres conceptos esenciales que constituyen la base del aprendizaje automático:

1. **Capacidad de aprender a partir de la experiencia.** Los sistemas de Machine Learning no siguen instrucciones fijas, sino que **aprenden de los datos**. La *experiencia* del sistema se representa mediante conjuntos de datos que contienen ejemplos del mundo real. Cuantos más datos recibe el sistema, más puede aprender y mejorar sus resultados.

2. **El algoritmo como mecanismo de aprendizaje.** Los datos se introducen en un **algoritmo matemático**, que analiza la información, detecta patrones y **construye un modelo**. Este modelo es la representación del conocimiento aprendido y será capaz de realizar predicciones o clasificaciones.

3. **Predicción y mejora continua.** Una vez entrenado, el modelo puede aplicarse a **nuevos datos** para generar predicciones. Si el modelo no ofrece resultados satisfactorios, puede **reentrenarse** con más datos o con un algoritmo diferente, buscando siempre mejorar su rendimiento.

En resumen, el aprendizaje automático combina tres elementos clave: **datos (experiencia), algoritmos (aprendizaje) y modelos (predicción).** El sistema aprende de datos históricos, construye un modelo basado en ellos y lo aplica para predecir o tomar decisiones sobre nuevos casos.

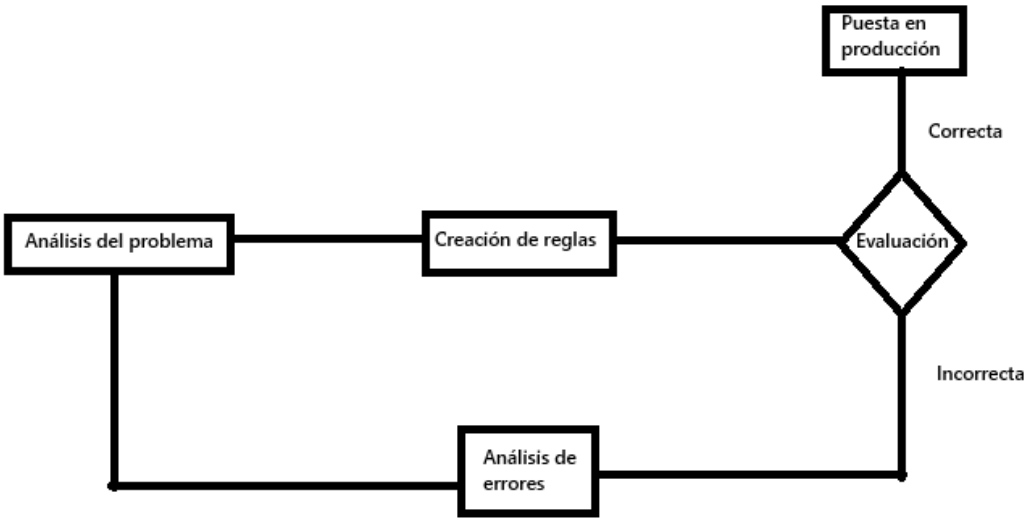


Vamos a ver un ejemplo de aplicación de técnicas de Machine Learning con un ejemplo realista que vimos anteriormente, el filtro capaz de identificar y bloquear correos electrónicos spam. Este ejemplo es uno de los primeros problemas que se resuelven aplicando técnicas de Machine Learning.

Hoy en día, prácticamente todos los servicios de correo electrónico incorporan un **filtro de spam** que utiliza **técnicas de aprendizaje automático** para identificar y bloquear mensajes no deseados. Solo en sistemas muy antiguos o desactualizados se emplean aún métodos basados exclusivamente en reglas programadas manualmente.

Enfoque tradicional

Si nos remontamos a unos años atrás y vemos cómo funcionaba esta tecnología antes, probablemente un filtro de spam funcionaba de la siguiente manera:



En el enfoque tradicional, un **analista** (generalmente especializado en seguridad informática) debía realizar un **análisis manual del problema**. Para ello, recopilaba ejemplos de correos electrónicos identificados previamente como *spam* y buscaba **patrones comunes** que los distinguieran de los mensajes legítimos. Por ejemplo, podía observar que muchos correos de spam:

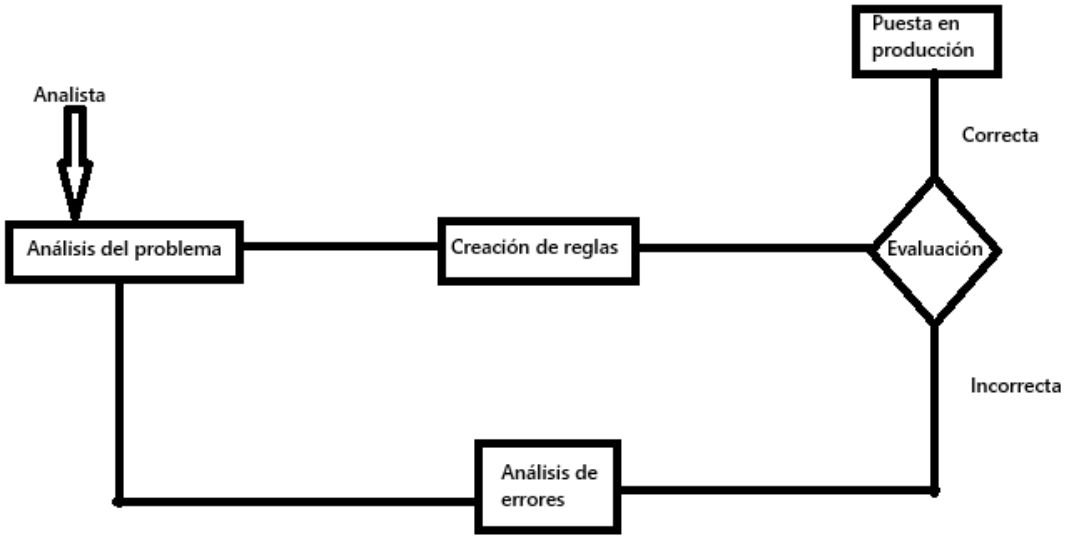
- No incluían el nombre real del destinatario.
- Contenían un exceso de etiquetas HTML.
- Presentaban errores ortográficos o palabras en varios idiomas.

A partir de estas observaciones, el analista elaboraba un **conjunto de reglas**, conocido como *motor de reglas*, que definía explícitamente las condiciones bajo las cuales un correo debía clasificarse como spam o como mensaje legítimo.

En este enfoque, el **motor de reglas** se compone de un conjunto de **condiciones programadas manualmente** que permiten **distinguir entre un correo electrónico legítimo y uno de spam**. Cuando llega un nuevo mensaje, el sistema analiza sus características y verifica si cumple alguna

de las reglas establecidas: por ejemplo, si incluye o no el nombre del destinatario, si contiene muchas etiquetas HTML o si utiliza palabras en otros idiomas.

Estas reglas y heurísticas permiten al sistema **decidir con cierto grado de confianza** si el mensaje debe clasificarse como correo legítimo o como spam. Sin embargo, todas ellas deben ser **definidas y actualizadas manualmente** por el analista.



En este caso, el **motor de reglas** se compone de una serie de **instrucciones diseñadas para diferenciar** entre correos electrónicos legítimos y mensajes de spam. Cuando llega un nuevo correo, el sistema analiza su contenido y **verifica si cumple determinadas condiciones**, como la presencia del nombre del destinatario, el uso excesivo de etiquetas HTML o la inclusión de palabras en otros idiomas. A partir de estas comprobaciones, el motor aplica un conjunto de **reglas y heurísticas** que le permiten determinar, con cierto grado de confianza, si el mensaje es legítimo o debe clasificarse como spam. Todas estas reglas deben ser **definidas, probadas y actualizadas manualmente** por el analista.

Una vez definidas todo este conjunto de reglas que parecen adecuadas para resolver el problema, el siguiente paso consiste en **evaluar el motor de reglas**.

Para ello, el analista pone a prueba el sistema utilizando un conjunto de **correos electrónicos nuevos**, distintos a los empleados durante la fase de análisis inicial, con el fin de comprobar su capacidad de clasificación. Por ejemplo, se le presentan mensajes legítimos y se verifica si los clasifica correctamente, y lo mismo con los mensajes de spam.

Si el sistema logra **clasificar los correos con precisión aceptable**, se considera listo para su **implantación en producción**, es decir, para operar en la bandeja de entrada de los usuarios y filtrar automáticamente los mensajes entrantes.

Sin embargo, si el rendimiento no es satisfactorio, se inicia una **fase de análisis de errores**. En ella, el analista revisa qué reglas no están funcionando correctamente, cuáles resultan irrelevantes y qué nuevas condiciones podrían añadirse. Este proceso suele requerir **reiterar el**

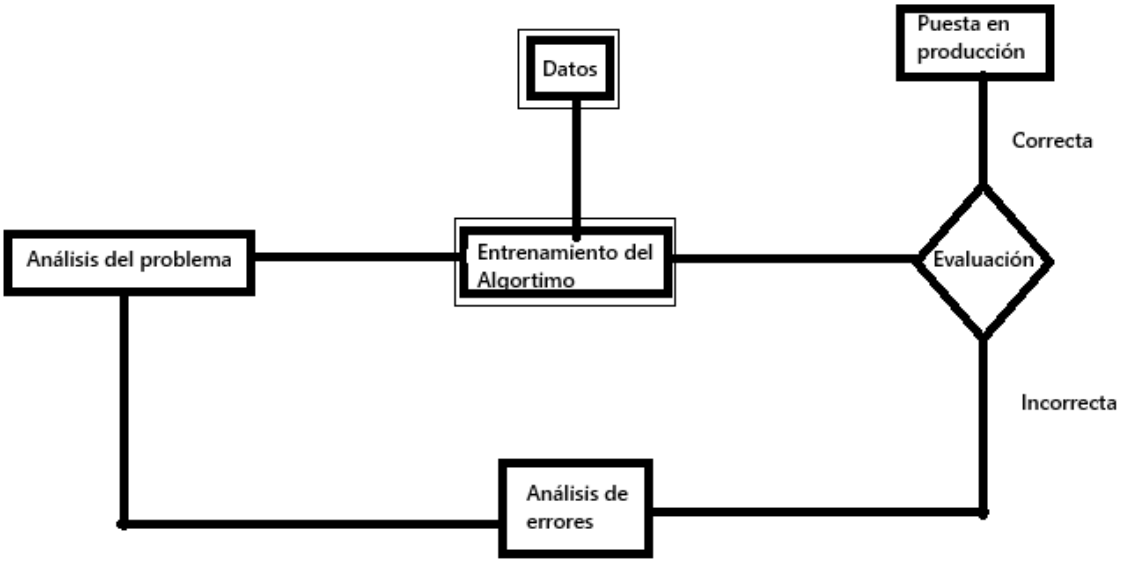
análisis del conjunto de datos, identificar nuevos patrones en los correos de spam y **ajustar o ampliar las reglas** para mejorar el comportamiento del sistema.

Como puede observarse, este **enfoque tradicional** depende en gran medida del **analista**, ya que es él quien diseña, implementa y ajusta manualmente el conjunto de reglas del sistema. Todo el funcionamiento del motor se apoya, por tanto, en su conocimiento y experiencia.

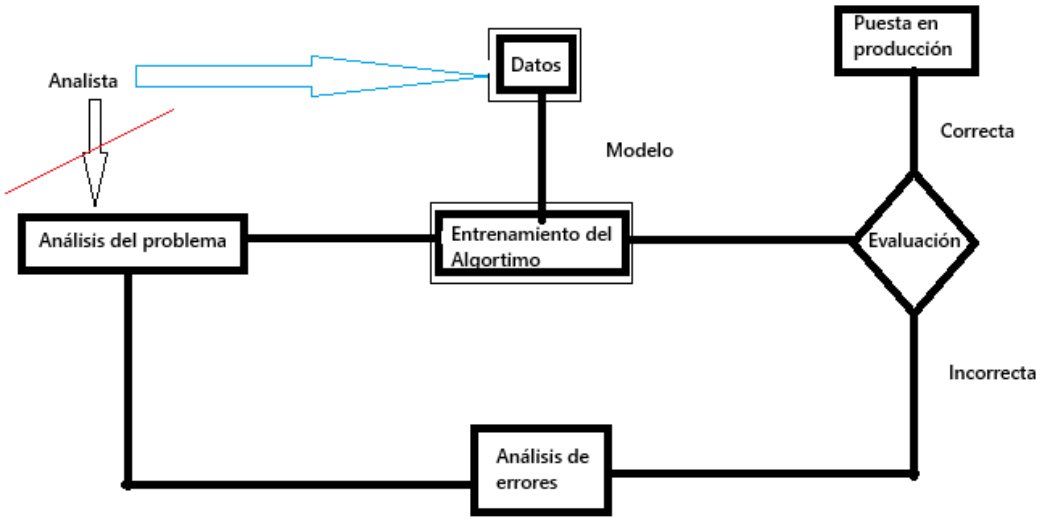
Además, este tipo de sistemas se basan en **reglas estáticas**, lo que implica que deben ser revisadas y actualizadas de forma periódica. Un motor de reglas puede resultar eficaz durante un tiempo, pero con el paso de los meses (a medida que cambian los patrones del spam) **acaba quedando obsoleto**. Mantenerlo operativo requiere un trabajo continuo de supervisión y ajuste por parte del analista, que debe volver a analizar los datos y redefinir las reglas cuando sea necesario.

Aprendizaje automático

Si trasladamos este mismo problema al ámbito del **aprendizaje automático (Machine Learning)**, el enfoque cambia radicalmente. En lugar de depender de reglas fijas creadas manualmente, el sistema **aprende directamente de los datos** y es capaz de adaptarse de forma autónoma a nuevos patrones, como veremos a continuación:



En el enfoque basado en **aprendizaje automático**, el papel del analista sigue siendo esencial, aunque cambia significativamente. El analista ya **no necesita definir manualmente las reglas** que diferencian un correo legítimo de uno de spam. En su lugar, su labor se centra en **preparar y seleccionar los datos adecuados**, es decir, recopilar ejemplos de correos electrónicos legítimos y de spam que representen correctamente el problema a resolver.



Una vez creado este **conjunto de datos etiquetado**, el analista elige el **algoritmo de aprendizaje automático** que considere más apropiado (por ejemplo, una regresión logística, un árbol de decisión o una red neuronal) para que el sistema aprenda automáticamente los patrones que distinguen ambos tipos de correos.

El **analista**, por tanto, se encarga principalmente de reunir los **datos** representativos del problema y de seleccionar el **algoritmo matemático** que considere más adecuado para resolverlo.

Una vez elegido el algoritmo, inicia la fase de entrenamiento, en la que proporciona los datos al sistema de aprendizaje automático. Durante este proceso, el algoritmo ajusta progresivamente sus parámetros internos en función de los ejemplos que recibe, hasta construir un modelo capaz de reconocer patrones.

En términos simples, el modelo "aprende" las características que diferencian un correo electrónico de spam de uno legítimo, sin necesidad de que el analista programe esas reglas de forma explícita.

Una vez completado el proceso de entrenamiento, **el algoritmo ha generado un modelo capaz de realizar predicciones**.

Para comprobar su eficacia, el modelo se somete a una **fase de evaluación**, en la que se le presentan **nuevos correos electrónicos** que no formaron parte del conjunto de datos utilizado durante el aprendizaje. El objetivo de esta fase es **verificar si el modelo generaliza correctamente**, es decir, si es capaz de clasificar con precisión los nuevos mensajes como spam o legítimos. Si los resultados son satisfactorios y el modelo demuestra un rendimiento adecuado, se considera listo para su **implementación en producción**, donde podrá analizar y clasificar correos electrónicos en tiempo real.

El analista **no necesita conocer las reglas internas** que utiliza el modelo para distinguir entre un correo legítimo y uno de spam, ya que estas se generan automáticamente durante el proceso de aprendizaje.

Lo realmente importante es que el modelo **realice predicciones precisas y fiables**. Una vez validado, el modelo puede **implantarse en producción**, donde comenzará a analizar los correos

entrantes de los usuarios y a **clasificarlos automáticamente** como legítimos o como spam en función de lo que ha aprendido.

En caso de que las predicciones del modelo **no sean correctas o suficientemente precisas**, se debe volver a una **fase de análisis de errores**. Sin embargo, este análisis es muy diferente al que se realizaba en el enfoque tradicional. Ya no se trata de revisar manualmente un conjunto de reglas para identificar cuáles funcionan y cuáles no, sino de **evaluar la calidad de los datos y del algoritmo utilizado**.

El analista debe comprobar si los **datos de entrenamiento representan adecuadamente el problema**, es decir, si incluyen ejemplos suficientemente variados y significativos de correos legítimos y de spam. También es importante analizar si el **algoritmo seleccionado** es el más apropiado para el tipo de tarea o si conviene probar con otro modelo que se ajuste mejor a la naturaleza de los datos.

En esta etapa, el foco principal está en **mejorar los datos y el modelo**, no en modificar reglas manuales, lo que hace que el sistema pueda evolucionar de forma más eficiente y automática.

La principal diferencia es que, en este caso, **el modelo asume el papel principal en la toma de decisiones**, ya que es él quien **realiza las predicciones** y determina si un correo electrónico es legítimo o spam. El analista ya no necesita intervenir directamente en la clasificación ni definir reglas específicas, sino que se convierte en el **responsable de supervisar, ajustar y mejorar** el modelo a partir de los resultados obtenidos.

2. ¿Cuándo utilizar Machine Learning?

El **aprendizaje automático (Machine Learning)** resulta especialmente útil en los siguientes casos:

- **Cuando las soluciones se basan en un conjunto extenso de reglas o heurísticas**, cuya gestión manual resulta compleja o ineficiente. *Ejemplo:* filtros de correo electrónico capaces de identificar mensajes de spam.
- **En problemas complejos donde un analista no puede definir fácilmente una solución a partir de la información disponible**, debido a la cantidad, variabilidad o ambigüedad de los datos. *Ejemplo:* clasificación automática de imágenes o reconocimiento de objetos.
- **En entornos que cambian con frecuencia o presentan comportamientos impredecibles**, donde las reglas fijas dejan de ser efectivas con el tiempo. *Ejemplo:* detección de amenazas en el tráfico de red.
- **Como apoyo a los métodos tradicionales de análisis**, especialmente cuando se dispone de grandes volúmenes de datos que resultan difíciles de interpretar manualmente. *Ejemplo:* análisis de registros de actividad o patrones de comportamiento de usuarios.

3. Clasificación de los sistemas de Machine Learning.

Existen diversos criterios para **clasificar los sistemas y técnicas de aprendizaje automático**, atendiendo, por ejemplo, a su rendimiento, complejidad o forma de aprendizaje. No obstante, el criterio más común se basa en **cómo el modelo se entrena y se ajusta a lo largo del tiempo** utilizando un conjunto de datos o experiencias previas.

Según este enfoque, los métodos de Machine Learning pueden agruparse en varias categorías (no necesariamente excluyentes), ya que un mismo algoritmo puede compartir características de más de un tipo.

Según la forma en que se entrenan:

- **Aprendizaje supervisado:** el modelo aprende a partir de datos etiquetados, es decir, ejemplos donde se conoce la respuesta correcta.
- **Aprendizaje no supervisado:** el modelo trabaja con datos sin etiquetar, buscando patrones o estructuras ocultas.
- **Aprendizaje semi-supervisado:** combina datos etiquetados y no etiquetados para mejorar la precisión del modelo.
- **Aprendizaje por refuerzo:** el sistema aprende mediante la interacción con un entorno, recibiendo recompensas o penalizaciones según sus decisiones.

Según la forma en que aprenden con el tiempo:

- **Aprendizaje online:** el modelo se actualiza de manera continua conforme recibe nuevos datos.
- **Aprendizaje batch (por lotes):** el modelo se entrena de una sola vez con un conjunto completo de datos y no se actualiza hasta que se entrena nuevamente.

Según la forma en que realizan las predicciones:

- **Aprendizaje basado en instancias:** las predicciones se realizan comparando nuevos datos con ejemplos almacenados en el conjunto de entrenamiento (por ejemplo, K-Nearest Neighbors).
- **Aprendizaje basado en modelos:** el sistema construye una representación general del problema a partir de los datos y utiliza esa abstracción para realizar predicciones (por ejemplo, regresión lineal o redes neuronales).

4. Técnicas y herramientas de aprendizaje automático.

Existen diversas **técnicas de aprendizaje automático** que permiten abordar distintos tipos de problemas según la naturaleza de los datos y los objetivos del modelo. Entre las más utilizadas se encuentran:

- **Regresión lineal y regresión logística:** empleadas para **predecir valores numéricos continuos** o **clasificar datos en categorías simples**, respectivamente.
- **Árboles de decisión y bosques aleatorios (Random Forest):** útiles en tareas de **clasificación y toma de decisiones**, especialmente cuando intervienen múltiples variables.

- **K-Means:** técnica de **aprendizaje no supervisado** que permite **agrupar datos en clústeres o grupos** según su similitud.
- **Redes neuronales artificiales:** modelos inspirados en el funcionamiento del cerebro humano, ampliamente utilizados en **reconocimiento de imágenes, procesamiento de voz y texto**, entre otros campos.

Para implementar estas técnicas, existen diversas **herramientas y bibliotecas** que facilitan su aplicación práctica:

- **Scikit-learn:** biblioteca de Python ampliamente utilizada para el aprendizaje automático clásico; ideal para principiantes y proyectos educativos.
- **Weka y Orange:** entornos gráficos interactivos que permiten **experimentar con diferentes algoritmos** sin necesidad de programar.
- **TensorFlow y Keras:** bibliotecas de alto rendimiento diseñadas para el desarrollo de **modelos avanzados y de aprendizaje profundo (Deep Learning)**, utilizadas tanto en investigación como en aplicaciones industriales.

5. Técnicas.

La **elección de la técnica y la herramienta** más adecuada depende del **tipo de problema a resolver**, la **cantidad y calidad de los datos disponibles** y el **nivel de complejidad** que se desee alcanzar.

En todos los casos, el objetivo es **seleccionar la técnica que mejor se adapte a la naturaleza de los datos** y que proporcione **resultados precisos, interpretables y eficientes**.

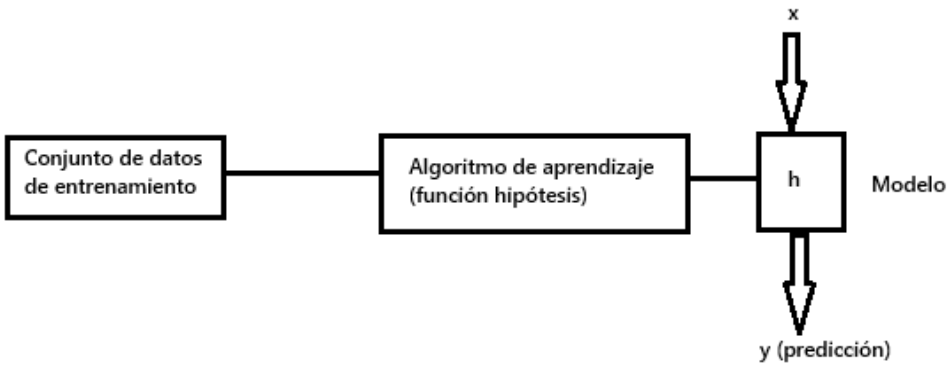
A. Aprendizaje supervisado.

A continuación, abordaremos las **técnicas de Machine Learning basadas en el aprendizaje supervisado**, analizando en qué consisten y cuáles son sus principales características.

El **aprendizaje supervisado** es uno de los enfoques más utilizados dentro del aprendizaje automático. En este tipo de sistemas, el modelo aprende a partir de **ejemplos etiquetados**, es decir, de conjuntos de datos donde se conoce tanto la entrada como la salida esperada.

Una de las definiciones más aceptadas lo describe de la siguiente manera:

"El aprendizaje supervisado es la tarea del aprendizaje automático que consiste en aprender una función que relaciona una entrada con una salida, a partir de pares de ejemplos entrada–salida."



Esta estructura es muy similar a la que ya analizamos al explicar qué es el **aprendizaje automático**, aunque aquí se presenta de forma más detallada.

En el aprendizaje supervisado, se parte de un **conjunto de datos de entrenamiento** que contiene ejemplos compuestos por **pares de entrada y salida** conocidos. Estos datos se proporcionan a un **algoritmo de aprendizaje**, el cual ajusta sus **parámetros internos** para encontrar una función que relacione correctamente las entradas con las salidas. El resultado de este proceso es el **modelo**, que representa el conocimiento adquirido por el sistema. Posteriormente, este modelo se utiliza para **realizar predicciones** o clasificaciones sobre nuevos datos, de los que ya no se conoce la salida esperada.

La principal característica que distingue a un **algoritmo de aprendizaje supervisado** es que **requiere ejemplos etiquetados** para poder aprender la relación entre las entradas y las salidas. En otras palabras, el modelo necesita un **conjunto de datos de entrenamiento** formado por **pares de entrada y salida conocidos**, que le permitan aprender cómo transformar una entrada en el resultado correcto.

A este conjunto se le denomina **conjunto de datos etiquetado**, ya que cada ejemplo incluye la información necesaria para identificar la respuesta esperada. Por ejemplo, en un problema de **detección de spam**, el modelo se entrenaría con un conjunto de correos electrónicos previamente clasificados como *legítimos* o *spam*. Durante el entrenamiento, el algoritmo analiza estos ejemplos y aprende los **patrones y características** que diferencian ambos tipos de correos, para poder aplicarlos posteriormente a mensajes nuevos.

En el ámbito del **aprendizaje automático**, el modelo que construimos también se conoce como **función hipótesis**. Esta función representa el conocimiento que el algoritmo ha adquirido durante el entrenamiento y permite **establecer la relación entre las entradas y las salidas**.

Lo que caracteriza al **aprendizaje supervisado** es que el algoritmo recibe un **conjunto de datos etiquetado**, es decir, un conjunto en el que un analista ha identificado previamente a qué categoría pertenece cada ejemplo. A partir de esos datos etiquetados, el **algoritmo matemático** aprende y **construye la función hipótesis**, es decir, el modelo que será capaz de realizar predicciones de forma automática.

Una vez entrenado, el modelo puede aplicarse a **nuevos datos no etiquetados**, por ejemplo, un correo electrónico recién recibido y, a través de la función hipótesis, **predecir la categoría correspondiente**, como determinar si ese mensaje es *spam* o *legítimo*.

Una vez comprendido que es el aprendizaje supervisado vamos a ver subcategorías. Dentro del aprendizaje supervisado hay dos subcategorías principales: **regresión y clasificación**, que se diferencian en el valor que intentan predecir, en este caso el tipo de valor que es la Y.

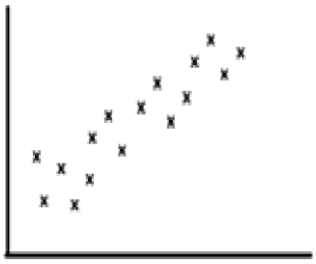
- **Regresión**: La regresión se utiliza cuando el objetivo es **predecir valores continuos**, es decir, resultados que pueden tomar un amplio rango de posibles valores. Este tipo de técnica se aplica, por ejemplo, a problemas como **estimar el precio de una vivienda** o **predecir el coste económico de un incidente de seguridad** a partir de ciertas características de entrada.

  En el caso de la vivienda, el valor a predecir (su **precio**) puede variar dentro de un rango continuo, por ejemplo: 50.000 €, 80.000 €, 150.000 € o 300.000 €. Por tanto, el modelo no clasifica en categorías, sino que **calcula un valor numérico aproximado**.

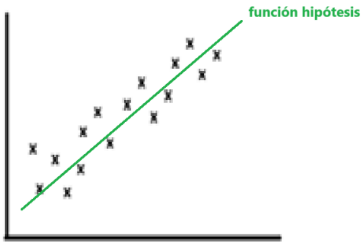
  Otro ejemplo puede encontrarse en el ámbito de la **seguridad informática**. Supongamos que queremos predecir el **coste estimado de un incidente** en función de determinadas variables, como el **número de equipos afectados**. Para entrenar el modelo, analizamos incidentes pasados y recopilamos datos de entrada (por ejemplo, el número de ordenadores comprometidos, representado como *x*) junto con el **coste real asociado a cada incidente** (la variable objetivo *y*). A partir de estos datos, el algoritmo aprende la relación entre ambas variables y puede **predecir el coste esperado** de futuros incidentes similares.

(x) #equipos afectados	(y) #coste en euros
2 15 …	1000 50000 …

Cogeremos esos datos etiquetados y los pasamos a una gráfica:



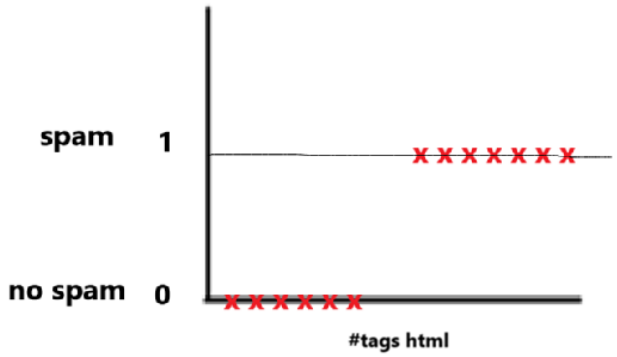
Y construiremos la función hipótesis. La función matemática será similar a esto:



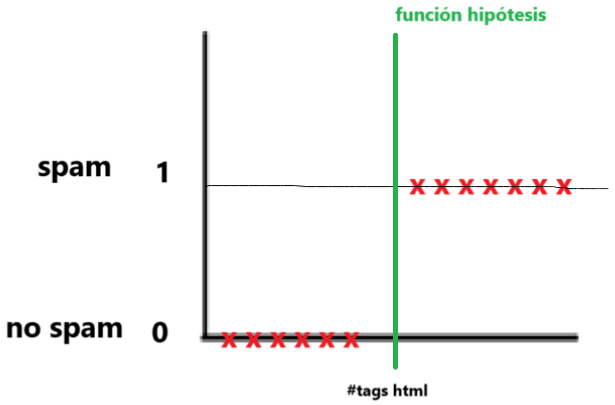
- **Clasificación**: La clasificación se utiliza cuando el objetivo es **predecir valores discretos**, es decir, resultados que pertenecen a un conjunto limitado y definido de categorías. A diferencia de la regresión, donde los valores posibles son continuos, en la clasificación el modelo **asigna cada ejemplo a una clase concreta**.

  Este tipo de técnica se aplica, por ejemplo, en el **problema de clasificar correos electrónicos** como *legítimos* o *spam*. En este caso, las posibles salidas están acotadas a dos categorías: *spam* (1) o *no spam* (0); no existen valores intermedios.

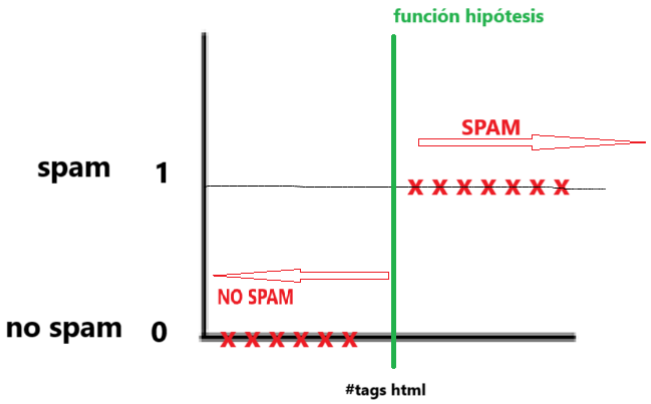
  De manera intuitiva, podemos entenderlo a través del ejemplo del **filtro de spam**. Para entrenar el modelo, primero debemos **extraer características relevantes** de los correos electrónicos (por ejemplo, el **número de etiquetas HTML** que contienen) y **asociarlas con su etiqueta correspondiente** (*spam* o *no spam*). A partir de esos datos etiquetados, el algoritmo aprende los patrones que diferencian ambos tipos de mensajes y, posteriormente, puede **clasificar automáticamente nuevos correos** según esas características.



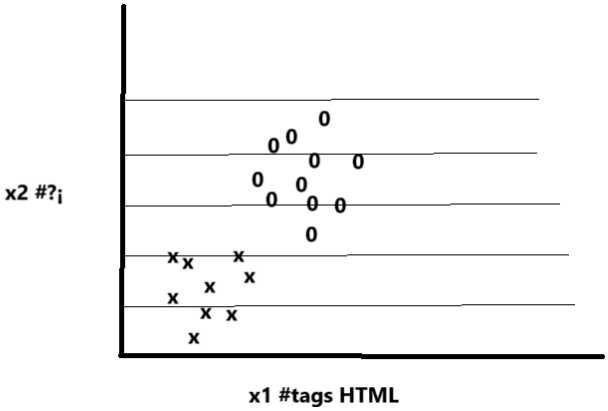
A medida que hay mayor número de tags es más probable que sea spam. En definitiva nos construirá una función hipótesis similar a la siguiente:



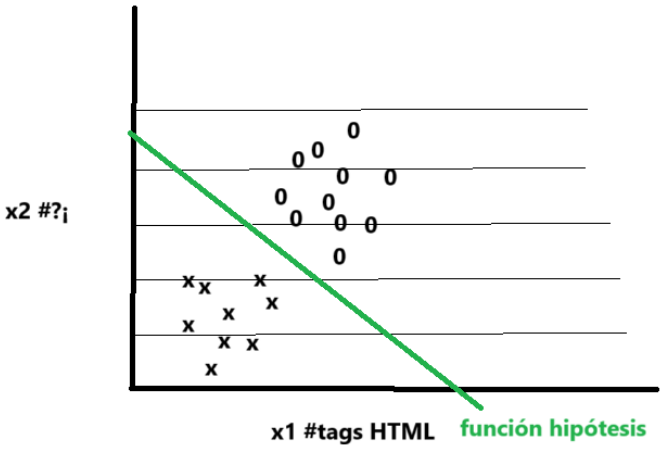
Todo lo que se encuentre a la derecha de la función será spam y lo que se encuentre a la izquierda será no spam:



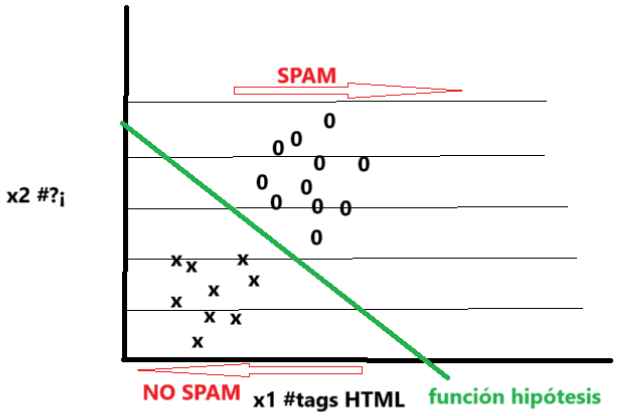
Si extrajese más características, por ejemplo añadiera al ejemplo anterior por ejemplo otros caracteres #?¡ (variable de entrada) se nos quedaría por ejemplo esta representación donde los 0 serán spam y los x no spam (sabemos que es spam o no porque los datos son etiquetados):



Construiremos la función hipótesis con esos datos:



Hemos construido por lo tanto un modelo que se adapta a esa tendencia y nos permite realizar predicciones.



Las técnicas de aprendizaje supervisado son las que más se utilizan en la actualidad.

B. Aprendizaje no supervisado.

Una de las definiciones más aceptadas del **aprendizaje no supervisado** es la siguiente:

"El aprendizaje automático no supervisado es la tarea de aprendizaje automático que consiste en inferir una función que describa la estructura interna de un conjunto de datos sin etiquetar, es decir, datos que no han sido previamente clasificados ni categorizados."

En esta definición aparecen tres conceptos clave que es importante comprender para **diferenciar las técnicas basadas en aprendizaje supervisado** de aquellas basadas en **aprendizaje no supervisado**.

- El **primer concepto** clave es que estas técnicas **no construyen explícitamente una función predictiva**, como ocurre en el aprendizaje supervisado, sino que **infieren una función** que les permite **descubrir relaciones o patrones ocultos** dentro de los datos. En lugar de ajustar parámetros para obtener una salida concreta, el objetivo es identificar la estructura subyacente del conjunto de datos.

- El **segundo aspecto** fundamental es precisamente ese: el **objetivo del aprendizaje no supervisado no es predecir resultados**, sino **describir y organizar los datos** que se le proporcionan. Estas técnicas buscan agrupar ejemplos similares, reducir dimensiones o detectar patrones sin que exista una "respuesta correcta" de referencia.

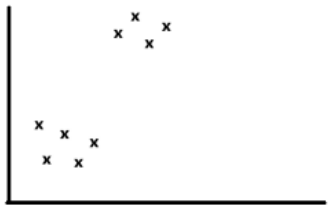
- Por último, la **tercera característica** distintiva es que los **datos de entrada no están etiquetados**. Es decir, el sistema no sabe de antemano a qué categoría pertenece cada ejemplo. Esta ausencia de etiquetas es la principal diferencia respecto al aprendizaje supervisado, donde el modelo aprende a partir de pares entrada–salida conocidos.

Estos algoritmos trabajan a partir de un **conjunto de datos sin etiquetar**, que se proporciona a un **algoritmo de aprendizaje automático**. El algoritmo, a partir de esos datos, **infiere una**

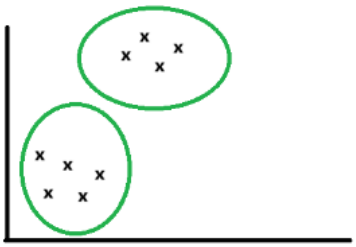
función, conocida también como **función hipótesis**, que describe las relaciones o patrones existentes entre los elementos del conjunto.

A diferencia del aprendizaje supervisado, el **objetivo principal de esta función no es realizar predicciones**, sino **descubrir y representar la estructura interna de los datos**. De esta manera, el modelo puede identificar similitudes, agrupamientos o tendencias sin disponer de información previa sobre las categorías o clases de los ejemplos analizados.

Imaginad que nuestra función de hipótesis tiene una tendencia similar a la que veis en el siguiente ejemplo:



El modelo **describe la estructura de los datos identificando similitudes entre los ejemplos del conjunto de entrenamiento**, basándose en **medidas de distancia o en otras características compartidas**. Como resultado, el algoritmo puede **agrupar los datos en distintos conjuntos o categorías**, por ejemplo, **dos grupos claramente diferenciados** que comparten rasgos comunes entre sí.



En este caso, el conjunto de datos se **divide en dos grupos principales**, donde los **ejemplos de cada grupo comparten características comunes entre sí**, mientras que **difieren significativamente de los ejemplos del otro grupo**.

Con esto hemos repasado las **características fundamentales de los algoritmos de aprendizaje no supervisado**, que conviene tener siempre presentes.

A continuación, vamos a ver un **ejemplo práctico** para afianzar estos conceptos:

Retomemos el caso del **filtro de spam** que analizamos anteriormente. Si aplicáramos un **algoritmo de aprendizaje supervisado**, dispondríamos de un conjunto de datos etiquetado similar al que se muestra a continuación:

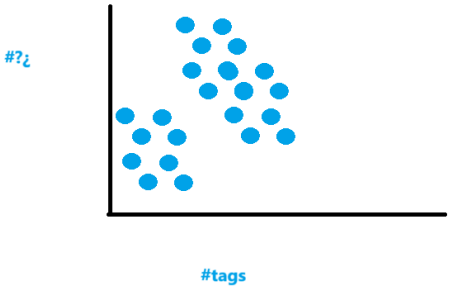
x_1 # ? ¡	x_2 # HTML	c Y
15	5	1
6	0	0
..	..	..

En el caso anterior, habíamos **extraído dos características** de nuestros correos electrónicos (x_1 y x_2) junto con una **etiqueta** (Y) que indicaba si cada mensaje era *spam* o *no spam*. Al proporcionar este conjunto de datos a un algoritmo supervisado, el sistema **aprendía una función** capaz de predecir, para un nuevo correo electrónico, si debía clasificarse como *spam* o como *legítimo*.

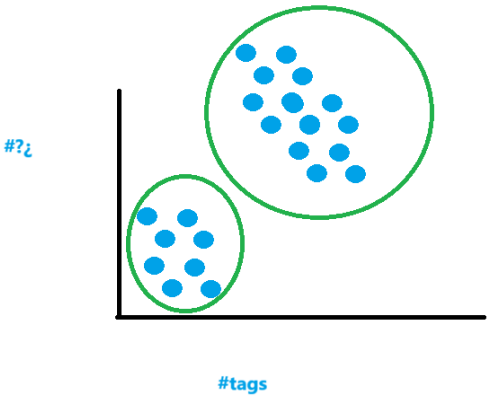
Sin embargo, en este nuevo escenario nos enfrentamos a un **conjunto de datos diferente**, ya que **no dispone de la columna *Y***. Esto significa que **no contamos con etiquetas** que indiquen si un correo es *spam* o *no spam*; únicamente disponemos de las **características de entrada** (x_1, x_2, etc.) a partir de las cuales el algoritmo deberá **descubrir por sí mismo la estructura o los posibles grupos existentes** dentro de los datos.

x_1 # ? ¡	x_2 # HTML
15	5
6	0
..	..

¿Cómo van a funcionar estos algoritmos no supervisados? Imaginad que yo represento este conjunto de datos de la tabla, todos de la misma forma porque yo no sé ahora cuáles son spam o no, ya que solo tengo las características de entrada:

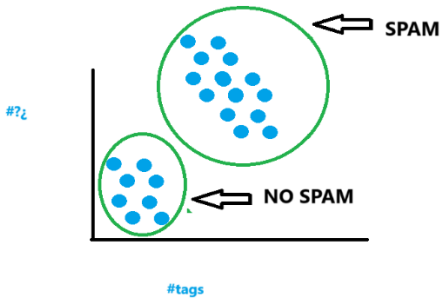


Entonces cuando yo le proporciono este conjunto de datos a un algoritmo basado en aprendizaje no supervisado, lo que nosotros podemos esperar es que coja estos datos y genere una función similar a esta (color verde) :



El algoritmo genera así una **función hipótesis** que representa la estructura inferida a partir de los datos. ¿Y qué significa esto exactamente? Que, en función de una serie de **métricas internas**, como la **similitud entre los datos**, la **distancia** o la **densidad de los grupos**, el algoritmo ha determinado que en el conjunto de datos existen **dos grandes subconjuntos o agrupaciones**.

El **objetivo principal** de este proceso es que, posteriormente, un **analista** examine la estructura obtenida por el algoritmo e **interprete los resultados**. En este ejemplo, tras analizar las agrupaciones generadas, el analista podría concluir que **uno de los grupos contiene principalmente correos de spam**, mientras que el otro **agrupa los mensajes legítimos**. De esta manera, el aprendizaje no supervisado permite **descubrir patrones ocultos en los datos**, incluso sin disponer de etiquetas previas.



¿Para qué sirven este tipo de algoritmos fundamentalmente? Fundamentalmente, se utilizan para **analizar grandes volúmenes de información** que serían **imposibles de etiquetar manualmente** debido a la cantidad de registros o ejemplos disponibles.

Estos algoritmos permiten **procesar los datos sin necesidad de intervención humana**, identificando automáticamente **estructuras, patrones o relaciones ocultas** entre los elementos del conjunto. A partir de la estructura descubierta, los analistas pueden **obtener información valiosa e intuiciones** que facilitan la comprensión del problema, la **definición de nuevas reglas** o incluso la **aplicación posterior de técnicas de aprendizaje supervisado**. En definitiva, el aprendizaje no supervisado actúa como una herramienta exploratoria que ayuda a **transformar datos masivos en conocimiento útil**.

6. Asociación entre técnicas, herramientas y tipos de sistemas.

Cada tipo de **aprendizaje automático** se vincula con **técnicas y herramientas específicas**, adaptadas a la naturaleza del problema y al tipo de datos con los que se trabaja:

- **Aprendizaje supervisado:** utiliza técnicas como la **regresión lineal**, los **árboles de decisión**, las **máquinas de vectores de soporte (SVM)** o las **redes neuronales**. Estas técnicas se implementan comúnmente mediante bibliotecas como **Scikit-learn** o **TensorFlow**, que permiten entrenar modelos predictivos a partir de datos etiquetados.
- **Aprendizaje no supervisado:** emplea métodos como **K-Means**, **PCA (Análisis de Componentes Principales)** o **algoritmos de agrupamiento jerárquico**, cuyo objetivo es descubrir patrones y estructuras ocultas en los datos. Herramientas como **Weka**, **Orange** y **Scikit-learn** resultan especialmente adecuadas para este tipo de análisis.

- **Aprendizaje semi-supervisado:** combina datos etiquetados y no etiquetados para mejorar la precisión del modelo. Se apoya en **modelos híbridos** y se implementa habitualmente con **Scikit-learn** o **TensorFlow**.
- **Aprendizaje por refuerzo:** se basa en técnicas como **Q-Learning** o **Deep Q-Networks (DQN)**, en las que el agente aprende mediante prueba y error, recibiendo recompensas o penalizaciones según sus acciones. Este tipo de aprendizaje se desarrolla principalmente con **TensorFlow** y **Keras**, que ofrecen potentes entornos para el diseño y entrenamiento de agentes inteligentes.

En conclusión, la **elección de la técnica y la herramienta** depende directamente del **tipo de aprendizaje automático** y de los **objetivos del problema a resolver**. Mientras que los enfoques **supervisados** se orientan a la **predicción y clasificación**, los **no supervisados** buscan **descubrir patrones** y **estructuras ocultas** en los datos. Los modelos **semi-supervisados** combinan ambos enfoques para aprovechar la información disponible de manera más eficiente, y los **algoritmos de refuerzo** se aplican cuando el sistema debe **aprender mediante la interacción con un entorno cambiante**.

Comprender esta asociación permite **seleccionar la estrategia más adecuada** y **optimizar los los resultados al aplicar técnicas de Machine Learning en contextos reales.**

7. Ejemplos prácticos de aplicabilidad.

A continuación se presentan algunos ejemplos de cómo aplicar distintas técnicas de aprendizaje automático a problemas reales, junto con un esquema de pseudocódigo que resume el proceso general.

Ejemplo 1: Clasificación de correos electrónicos (Aprendizaje supervisado)

Un modelo de aprendizaje supervisado puede entrenarse con un conjunto de correos electrónicos etiquetados como *spam* o *no spam*. Utilizando Scikit-learn y un algoritmo de regresión logística, el modelo aprende los patrones del texto (palabras frecuentes, longitud, presencia de enlaces, etc.) y posteriormente clasifica automáticamente nuevos mensajes.

```
Pseudocódigo:
# Entrenamiento del modelo
datos = cargar_datos("correos.csv")
X_train, Y_train = extraer_características_y_etiquetas(datos)
modelo = RegresiónLogística()
modelo.entrenar(X_train, Y_train)

# Predicción
nuevo_correo = extraer_características("mensaje_nuevo.txt")
resultado = modelo.predecir(nuevo_correo)

si resultado == 1:
   mostrar("Correo clasificado como SPAM")
sino:
   mostrar("Correo clasificado como NO SPAM")
```

Ejemplo 2: Segmentación de clientes (Aprendizaje no supervisado)

Mediante un enfoque de aprendizaje no supervisado con el algoritmo K-Means, un supermercado puede agrupar a sus clientes según su comportamiento de compra (frecuencia, gasto medio, tipo de productos adquiridos, etc.). Esto permite diseñar campañas de marketing personalizadas para cada grupo o clúster identificado.

```
Pseudocódigo:
# Agrupación de clientes
datos_clientes = cargar_datos("compras_clientes.csv")
modelo = KMeans(k=3)   # Crear 3 grupos de clientes
modelo.entrenar(datos_clientes)

# Resultados
grupos = modelo.obtener_grupos()
mostrar(grupos)
```

Ejemplo 3: Predicción de precios (Aprendizaje supervisado)

Con un modelo de regresión lineal en Scikit-learn, es posible predecir el precio de una vivienda a partir de variables como el tamaño, la ubicación o la antigüedad del inmueble. El modelo aprende la relación entre las características y el precio, permitiendo estimar el valor de nuevas propiedades.

```
Pseudocódigo:
# Entrenamiento del modelo
datos = cargar_datos("viviendas.csv")
X, Y = extraer_variables_y_precios(datos)
modelo = RegresiónLineal()
modelo.entrenar(X, Y)

# Predicción
nueva_vivienda = [tamaño=120, ubicación="centro", antigüedad=10]
precio_estimado = modelo.predecir(nueva_vivienda)
mostrar("Precio estimado:", precio_estimado)
```

Estos ejemplos ilustran cómo las distintas técnicas de aprendizaje automático (supervisadas y no supervisadas) pueden aplicarse para resolver problemas reales de clasificación, segmentación y predicción. La clave está en seleccionar el tipo de aprendizaje y el algoritmo adecuado según la naturaleza de los datos y el objetivo del análisis.