¿Qué es machine learning?

Machine learning (ml) o aprendizaje automático es una rama de la inteligencia artificial (ia) que permite a las máquinas aprender a partir de datos. En lugar de ser programadas explícitamente para realizar una tarea, las máquinas utilizan algoritmos que les permiten identificar patrones y tomar decisiones basadas en esos datos. La idea central detrás de ml es que, cuanto más datos tiene un sistema, mejor puede ajustarse y predecir resultados futuros.

Para entender mejor este concepto, podemos imaginar una máquina que aprende a predecir el precio de una casa. En lugar de escribir reglas específicas para cada posible casa (ubicación, tamaño, año de construcción, etc.), entrenamos un modelo con ejemplos reales de casas y sus precios. El modelo aprende de estos ejemplos y puede predecir el precio de una nueva casa basada en sus características, sin necesidad de que nosotros le indiquemos reglas fijas.

¿Cómo funciona machine learning?

El proceso de machine learning se basa en tres pasos fundamentales:

1. Entrenamiento: en este paso, se proporciona al modelo un conjunto de datos etiquetados. Estos datos contienen ejemplos que incluyen tanto la entrada (características) como la salida (resultados esperados). Usando estos datos, el modelo aprende a identificar relaciones entre las características y el resultado. Por ejemplo, en el caso de la predicción del precio de una casa, las características podrían incluir el tamaño, la ubicación y el número de habitaciones, mientras que el precio sería el resultado esperado.

2. Predicción: una vez que el modelo ha sido entrenado, se le presentan nuevos datos que no ha visto antes. El modelo utiliza lo que ha aprendido durante el entrenamiento para hacer predicciones. Siguiendo con el ejemplo de las casas, si le damos al modelo información sobre una nueva casa, puede predecir su precio basándose en las características aprendidas.

3. Evaluación: después de que el modelo realiza una predicción, es necesario evaluar su rendimiento. Se compara la predicción del modelo con los resultados reales para determinar cuán preciso es. Si las predicciones son buenas, el modelo se considera efectivo. Si no, puede ser necesario ajustar algunos parámetros o proporcionar más datos para mejorar el rendimiento.

Algoritmos básicos de machine learning

1. Regresión lineal

La regresión lineal es uno de los algoritmos más sencillos y utilizados en machine learning. Es una técnica estadística que se utiliza para modelar la relación entre una variable dependiente (lo que queremos predecir) y una o más variables independientes (las características).

¿Cómo funciona?

El objetivo de la regresión lineal es encontrar una línea recta (o plano, en el caso de varias variables independientes) que mejor se ajuste a los datos. Esta línea se conoce como la línea de regresión. La fórmula básica de la regresión lineal es:

$$y = b_0 + b_1 \cdot x_1 + b_2 \cdot x_2 + ... + b_n \cdot x_n$$

En términos simples, la regresión lineal busca encontrar los mejores valores para los coeficientes b_1,b_2,...bn para que la línea se ajuste lo mejor posible a los puntos de datos.

Ejemplo:

Imaginemos que queremos predecir el precio de una casa basándonos en su tamaño en metros cuadrados. La regresión lineal nos ayudará a encontrar una relación entre estas dos variables: tamaño y precio. Al alimentar al modelo con datos de varias casas (sus tamaños y precios correspondientes), la regresión lineal encontrará una línea que mejor represente la relación entre estas dos variables. Con esa línea, podremos predecir el precio de una nueva casa dada su superficie.

Si el tamaño de la casa es x y el precio es y, el modelo puede formular una ecuación como:

$$y = b_0 + b_1 \cdot x$$

Donde b_1 es el coeficiente que indica cuánto aumenta el precio por cada metro cuadrado adicional.

Ventajas y desventajas:

Ventajas:

- Es fácil de interpretar y entender.
- Funciona bien cuando la relación entre las variables es lineal.
- Requiere menos recursos computacionales que otros algoritmos más complejos.

Desventajas:

- No es adecuada cuando la relación entre las variables no es lineal.
- Es sensible a outliers (valores atípicos), lo que puede distorsionar los resultados.

2. Clasificación

La clasificación es uno de los problemas más comunes en machine learning y consiste en predecir una categoría o clase a la que pertenece una observación dada. A diferencia de la regresión, donde el objetivo es predecir un valor continuo (como el precio de una casa), en la clasificación el resultado es discreto, es decir, una etiqueta de clase.

¿cómo funciona?

El algoritmo de clasificación toma datos de entrada con sus características y aprende a asignar una etiqueta o clase a cada observación. Por ejemplo, si tenemos un conjunto de datos que contiene información sobre flores (como el tamaño de los pétalos y los sépalos), el objetivo de un algoritmo de clasificación puede ser determinar el tipo de flor (por ejemplo, "iris-setosa", "iris-versicolor", "iris-virginica").

En la clasificación, las salidas posibles pueden ser binarias (dos clases, como "sí" o "no") o múltiples (más de dos clases).

Ejemplo: clasificación binaria

Supongamos que queremos predecir si un correo electrónico es spam o no spam. En este caso, las características podrían ser la longitud del correo, la presencia de ciertas palabras clave, etc. El algoritmo aprenderá a partir de ejemplos previos (correos etiquetados como spam o no spam) y luego clasificará nuevos correos basándose en lo que ha aprendido.

Algoritmos comunes de clasificación:

Algunos de los algoritmos de clasificación más comunes incluyen:

Regresión logística: aunque su nombre sugiere lo contrario, la regresión logística es un algoritmo de clasificación binaria. Utiliza una función sigmoide que transforma los resultados en valores entre 0 y 1, que luego se utilizan para asignar una clase. Por ejemplo, si la probabilidad de que un correo sea spam es mayor que 0.5, se clasifica como spam.

Fórmula de la regresión logística:

$$P(y=1) = \frac{1}{1 + e^{-(b_0 + b_1 x_1 + b_2 x_2 + ... + b_n x_n)}}$$

Donde $P(y=1)$ es la probabilidad de que la observación pertenezca a la clase 1 (por ejemplo, spam), y eee es la constante de euler.

K-nearest neighbors (knn): este es un algoritmo simple pero eficaz que clasifica una nueva observación basándose en las clases de sus vecinos más cercanos en el espacio de características. El número de vecinos (*K*) se determina antes de comenzar, y el algoritmo asigna la clase más común entre esos vecinos.

Ejemplo de knn: imagina que tienes datos sobre varios pacientes con dos características: nivel de azúcar en sangre y edad. Si queremos predecir si un nuevo paciente tiene diabetes o no, knn buscaría los *K* pacientes más cercanos en términos de edad y nivel de azúcar, y asignaría la clase más común (diabetes o no diabetes) entre esos vecinos.

Ventajas y desventajas de la clasificación:

Ventajas:

- Puede utilizarse para una amplia gama de problemas (desde detección de spam hasta diagnóstico médico).
- Los resultados son fáciles de interpretar (por ejemplo, "sí" o "no").

Desventajas:

- Algunos algoritmos de clasificación pueden ser computacionalmente costosos, especialmente cuando se trabaja con grandes volúmenes de datos.
- La clasificación no siempre es precisa cuando las clases no están claramente separadas.

3. K-nearest neighbors (knn)

El k-nearest neighbors (knn) es un algoritmo intuitivo y fácil de entender. Se utiliza para clasificar datos en diferentes categorías o grupos. El principio de knn es sencillo: cuando llega un nuevo dato que no tiene una etiqueta (es decir, no sabemos a qué categoría pertenece), el algoritmo mira a sus "vecinos" más cercanos y decide a qué grupo pertenece basándose en lo que han hecho sus vecinos.

¿cómo funciona knn?

Imagina que tienes un grupo de puntos en un gráfico, y cada uno de esos puntos representa un objeto que ya conoces. Algunos puntos están etiquetados como "rojos" y otros como "azules". Ahora aparece un nuevo punto sin etiqueta. Lo que hace knn es buscar los puntos más cercanos a este nuevo punto y ver de qué color son. Si la mayoría de los puntos cercanos son rojos, entonces el nuevo punto probablemente también sea rojo.

Este número de vecinos cercanos que el algoritmo busca se llama K. Si K=3, el algoritmo mirará los tres puntos más cercanos. Si K=5, observará cinco vecinos. La idea es que el nuevo dato se clasifique según la mayoría de sus vecinos más cercanos.

Ejemplo práctico:

Imagina que quieres predecir si a una persona le gustará una película. En lugar de basarte solo en las características de esa persona, podrías usar knn para buscar a personas similares (vecinos) que ya han visto la película y han dado su opinión. Si la mayoría de las personas similares la calificaron positivamente, el algoritmo asumirá que a la nueva persona también le gustará.

Elección del número K:

El número de vecinos cercanos que seleccionas es importante:

Si eliges un número muy pequeño de vecinos (por ejemplo, 1 o 2), podrías tomar decisiones basadas en solo uno o dos datos, lo que puede no ser representativo y hacer que el algoritmo sea muy sensible a valores extraños.

Si eliges un número muy grande de vecinos, el algoritmo podría tomar en cuenta muchos puntos que no están realmente cerca, lo que puede llevar a decisiones incorrectas.

Ventajas y desventajas de knn:

Ventajas:

- Es fácil de entender y aplicar.
- No requiere un proceso complejo de entrenamiento; el trabajo se hace cuando se necesita clasificar un nuevo dato.
- Funciona bien para problemas pequeños y medianos.

Desventajas:

- Puede volverse lento cuando se tienen muchos datos, porque tiene que calcular la cercanía de cada nuevo dato a todos los puntos del conjunto.
- No siempre es eficaz cuando las características de los datos están en diferentes escalas, como comparar edad y altura, a menos que se normalicen.

4. Árboles de decisión

Los árboles de decisión son uno de los algoritmos más visuales y fáciles de entender en machine learning. Se utilizan tanto para clasificación como para regresión, y se basan en una serie de preguntas y respuestas, organizadas en un árbol de decisiones, para predecir el valor de una variable o la categoría a la que pertenece.

¿cómo funcionan los arboles de decisión?

Imagina un proceso donde se toman decisiones basadas en una serie de preguntas. Cada una de estas preguntas separa los datos en diferentes grupos. La idea es hacer preguntas que, paso a paso, nos lleven a la mejor respuesta posible. En un árbol de decisión, cada nodo representa una pregunta o condición, y cada rama representa una posible respuesta. El objetivo final es llegar a una hoja, que es la clasificación o predicción final.

Ejemplo práctico:

Supongamos que queremos predecir si una persona comprará un coche nuevo. Las preguntas que el árbol podría hacer podrían ser:

¿el salario de la persona es mayor de $30,000?

Si sí, la persona puede comprar el coche (continuamos con otra pregunta).

Si no, el árbol predice que la persona probablemente no lo comprará.

¿la persona tiene más de 25 años?

Si sí, seguimos preguntando, tal vez considerando si tiene hijos o su historial de crédito.

Si no, el árbol podría predecir que la persona no comprará el coche porque es joven y menos propensa a hacer grandes compras.

Cada una de estas preguntas forma un "nodo" del árbol, y las respuestas a cada pregunta nos llevan por diferentes caminos hasta que llegamos a una conclusión.

Construcción del arbol:

El algoritmo construye el árbol automáticamente, seleccionando las preguntas que dividen los datos de manera más efectiva. Esto se hace buscando las características que proporcionan la mayor "información" sobre los datos (es decir, que separan mejor las categorías).

Por ejemplo, si estamos clasificando a clientes que compran o no un producto, la primera pregunta que el algoritmo puede seleccionar es aquella que mejor separa a las personas que compran de las que no compran, como el salario.

Ventajas y desventajas de los árboles de decisión:

Ventajas:

- Fácil de entender y visualizar.
- No requiere que los datos sean escalados o normalizados.
- Puede manejar tanto problemas de clasificación como de regresión.

Desventajas:

- Puede volverse complejo y poco eficaz cuando se tiene un árbol demasiado profundo, lo que lleva a un fenómeno llamado sobreajuste (overfitting), donde el modelo aprende demasiado bien los detalles específicos de los datos de entrenamiento pero no generaliza bien con datos nuevos.
- Es sensible a pequeños cambios en los datos. Si los datos cambian un poco, el árbol puede tomar decisiones completamente diferentes.

5. Entrenamiento y evaluación de modelos

El proceso de entrenamiento de un modelo en machine learning es uno de los pasos más importantes. Es aquí donde el modelo "aprende" a partir de los datos. El siguiente paso, la evaluación, nos permite medir qué tan bien el modelo ha aprendido y si es capaz de hacer predicciones precisas con datos nuevos que no ha visto antes.

¿qué significa entrenar un modelo?

Entrenar un modelo es el proceso en el que un algoritmo de machine learning toma un conjunto de datos y ajusta sus parámetros internos para reconocer patrones en esos datos. El objetivo es que, una vez entrenado, el modelo pueda hacer predicciones precisas cuando se le presenten datos nuevos.

El conjunto de datos con el que entrenamos el modelo incluye tanto las características (los datos de entrada) como las etiquetas (los resultados que queremos predecir). El algoritmo utiliza estos ejemplos para aprender cómo están relacionadas las características con las etiquetas.

Etapas del entrenamiento:

División de los datos: para evaluar correctamente el rendimiento de un modelo, es común dividir el conjunto de datos en dos partes:

1. Datos de entrenamiento: este es el conjunto de datos que el modelo utilizará para aprender. Suele ser entre el 70% y el 80% del total de los datos.

2. Datos de prueba: este conjunto, que representa el 20%-30% restante, se reserva para evaluar el modelo después de haber sido entrenado.

3. Ajuste del modelo: durante el proceso de entrenamiento, el algoritmo ajusta sus parámetros internos (por ejemplo, los coeficientes en un modelo de regresión lineal o los nodos en un árbol de decisión) para minimizar el error en las predicciones que hace sobre los datos de entrenamiento.

4. Predicción: una vez que el modelo ha sido entrenado, puede hacer predicciones sobre nuevos datos. Estas predicciones se pueden comparar con los resultados reales (si los conocemos) para ver qué tan precisas son.

Evaluación del modelo:

Después de entrenar el modelo, es importante evaluar su rendimiento para asegurarnos de que es capaz de hacer buenas predicciones. Esto se hace utilizando los datos de prueba, que el modelo no ha visto durante el entrenamiento.

Algunas de las métricas más comunes para evaluar el rendimiento de un modelo incluyen:

1. Precisión: Es el porcentaje de predicciones correctas que hace el modelo. Se utiliza principalmente en problemas de clasificación. Por ejemplo, si un modelo predice correctamente 90 de 100 ejemplos, su precisión es del 90%.

2. Recall (Sensibilidad): Mide la capacidad del modelo para identificar correctamente los casos positivos. Por ejemplo, en un modelo de detección de enfermedades, el recall indica qué tan bien el modelo puede identificar a los pacientes enfermos.

3. Error Cuadrático Medio (MSE): Se utiliza en problemas de regresión. Es la medida promedio de la diferencia al cuadrado entre los valores predichos y los valores reales. Cuanto más bajo sea este valor, mejor es el rendimiento del modelo.

Ejemplo de evaluación:

Imagina que entrenamos un modelo para predecir si un cliente comprará o no un producto. Después de entrenar el modelo con datos históricos de compras, lo evaluamos con un conjunto de datos de prueba. Si el modelo predice correctamente el comportamiento de la mayoría de los clientes, podemos decir que tiene un buen rendimiento. Si el modelo se equivoca mucho, necesitaremos hacer ajustes en los parámetros o probar con un enfoque diferente.

Optimización del modelo:

Si el modelo no está funcionando como se esperaba, podemos hacer varios ajustes:

1. Ajuste de hiperparámetros: Muchos modelos tienen parámetros que podemos ajustar antes del entrenamiento, como el número de vecinos en KNN o la profundidad de un árbol de decisión. Estos ajustes se realizan para mejorar el rendimiento del modelo.

2. Aumento del conjunto de datos: En algunos casos, el modelo puede necesitar más datos para aprender mejor. Al aumentar la cantidad de datos de entrenamiento, el modelo puede generalizar mejor.

Flujo de trabajo de un proyecto de Machine Learning

Para aplicar machine learning, es importante entender cómo es el flujo de trabajo general en un proyecto típico. Este flujo puede desglosarse en los siguientes pasos:

Definición del problema:

Antes de comenzar, debes tener claro qué problema estás intentando resolver. ¿se trata de una predicción de un valor continuo (regresión) o de una clasificación en categorías? Ejemplos: predecir el precio de una casa (regresión) o clasificar un correo como spam o no spam (clasificación).

Recolección de datos:

El siguiente paso es obtener los datos que vas a utilizar. Los datos pueden provenir de diversas fuentes: archivos csv, bases de datos, apis, o datasets públicos en la web.

En esta etapa, también es crucial asegurarse de que los datos sean relevantes y representativos del problema que deseas resolver.

Exploración y preparación de los datos:

Una vez que tienes los datos, es fundamental explorarlos para comprender mejor su estructura. Debes identificar qué columnas (o variables) son importantes, revisar si hay valores faltantes o atípicos, y determinar si los datos necesitan ser transformados.

- Limpieza de datos: puede que tengas que eliminar o rellenar valores nulos (vacíos) y corregir inconsistencias en los datos.

- Transformación: algunas veces, los datos deben ser normalizados o escalados para que las características estén en una misma escala, lo cual es necesario para algunos algoritmos de machine learning.

División de los datos:

Es necesario dividir los datos en dos partes:

- Conjunto de entrenamiento: estos son los datos que el modelo utilizará para aprender.

- Conjunto de prueba: son los datos que se reservan para evaluar la capacidad del modelo de predecir correctamente sobre datos nuevos.

Selección del algoritmo:

Con base en el problema que deseas resolver, seleccionarás un algoritmo adecuado. Ya hemos explicado algunos algoritmos básicos como la regresión lineal, la clasificación y k-nearest neighbors. La selección del algoritmo dependerá del tipo de problema y de los datos que tengas.

Entrenamiento del modelo:

El entrenamiento es donde el modelo ajusta sus parámetros para aprender a predecir con precisión. Durante este proceso, el modelo analiza los datos de entrenamiento y encuentra patrones o relaciones entre las características.

Evaluación del modelo:

Una vez entrenado, el modelo debe ser evaluado para determinar qué tan bien funciona. Utilizamos el conjunto de prueba para ver si el modelo es capaz de predecir correctamente datos que no ha visto antes.

Existen métricas de evaluación como la precisión para clasificación, o el error cuadrático medio (mse) para problemas de regresión.

Optimización del modelo:

Si el modelo no tiene un buen rendimiento, puedes intentar ajustar los hiperparámetros (valores que controlan el comportamiento del algoritmo, como el número de vecinos en knn) o probar con un algoritmo diferente.

Otra opción es proporcionar más datos o mejorar la calidad de los datos existentes.

Implementación:

Una vez que el modelo está listo y tiene un buen rendimiento, el siguiente paso es implementarlo en un entorno real, donde pueda hacer predicciones sobre datos nuevos.