

SESIÓN FUNDAMENTOS DEL APRENDIZAJE DE MÁQUINA

CONTENIDOS:

- ¿Qué es el aprendizaje de máquina?
- Aplicaciones del aprendizaje de máquina.
- Etapas típicas de un problema de aprendizaje de máquina.
- Tipos de algoritmos de aprendizaje de máquina.
- Aprendizaje supervisado v/s no supervisado.
- Algoritmos de regresión: qué es una tarea de regresión.
- Algoritmos más utilizados.
- Algoritmos de clasificación: qué es una tarea de clasificación.
- Tipos de tarea de clasificación (binaria, multiclase, multietiqueta).
- Dimensiones y la maldición de la dimensionalidad.

El aprendizaje de máquina permite a las computadoras identificar patrones en los datos y tomar decisiones de manera autónoma. Esta disciplina se aplica en diversos campos, desde la predicción de tendencias hasta el reconocimiento de imágenes y voz. Existen diferentes tipos de aprendizaje, entre ellos el supervisado, que utiliza datos etiquetados para entrenar modelos en la resolución de problemas de clasificación y regresión, optimizando así la toma de decisiones basada en datos.

¿QUÉ ES EL APRENDIZAJE DE MÁQUINA?

El Aprendizaje de Máquina (*Machine Learning, ML*) es una rama de la inteligencia artificial que permite a las computadoras identificar patrones en los datos y tomar decisiones de manera autónoma sin ser programadas explícitamente. Para lograr esto, los algoritmos de aprendizaje de máquina se basan en modelos matemáticos y estadísticos que ajustan sus parámetros automáticamente según los datos que reciben. Este enfoque permite automatizar tareas complejas y hacer predicciones precisas sin intervención humana directa.

El aprendizaje de máquina se utiliza en aplicaciones como el reconocimiento de imágenes, procesamiento del lenguaje natural, análisis de datos y robótica, entre muchas otras áreas. Existen

diferentes tipos de aprendizaje, como el supervisado, no supervisado y por refuerzo, que se aplican según la disponibilidad y estructura de los datos.

APLICACIONES DEL APRENDIZAJE DE MÁQUINA

El aprendizaje de máquina ha revolucionado diversas industrias gracias a su capacidad para automatizar procesos y extraer información relevante a partir de grandes cantidades de datos. Algunas de sus principales aplicaciones son:

Aplicación	Descripción
Diagnóstico médico	Predicción de enfermedades a partir de datos médicos.
Reconocimiento de imágenes	Identificación de objetos en fotografías.
Análisis de sentimientos	Evaluación de opiniones en redes sociales.
Sistemas de recomendación	Sugerencias personalizadas en plataformas de streaming.
Vehículos autónomos	Control y navegación basada en sensores.

Ilustración 1 Principales aplicaciones del Aprendizaje Supervisado

Cada una de estas aplicaciones depende de modelos específicos que se entrenan con grandes volúmenes de datos para mejorar su desempeño y precisión.

ETAPAS TÍPICAS DE UN PROBLEMA DE APRENDIZAJE DE MÁQUINA

Antes de implementar un modelo de Machine Learning, es fundamental seguir una serie de pasos que garanticen su eficacia y precisión. Cada etapa del proceso tiene una función específica que contribuye a la construcción de un modelo óptimo.

1. Definición del Problema

Identificar qué se desea predecir o clasificar. Es crucial entender claramente el objetivo del proyecto.

- Ejemplo: Supongamos que queremos predecir el precio de las casas en una ciudad específica.

2. Recolección y Limpieza de Datos

Obtener datos relevantes y eliminar inconsistencias. Los datos deben ser precisos, completos y adecuados para el problema.

- Ejemplo: Recolectamos datos históricos de ventas de casas que incluyen características como el tamaño, número de habitaciones, ubicación, edad de la casa, etc. Luego limpiamos los datos eliminando registros incompletos o incorrectos.

3. División de Datos en Entrenamiento y Prueba

Separar los datos en conjuntos de entrenamiento y prueba para evitar sobreajuste y evaluar el rendimiento del modelo.

- Ejemplo: Dividimos los datos en un 80% para entrenamiento y un 20% para prueba. El conjunto de entrenamiento se usa para ajustar el modelo, mientras que el conjunto de prueba se utiliza para evaluar su desempeño.

4. Selección del Modelo Adecuado

Escoger entre clasificadores, regresores o redes neuronales según el tipo de problema.

- Ejemplo: Elegimos un modelo de regresión lineal ya que estamos tratando con un problema de predicción continua (precios de casas).

5. Entrenamiento del Modelo

Aplicar algoritmos de optimización para ajustar el modelo a los datos de entrenamiento.

- Ejemplo: Entrenamos el modelo de regresión lineal utilizando los datos de entrenamiento para encontrar la relación entre las características de las casas y sus precios.

6. Evaluación y Ajuste del Modelo

Medir el desempeño del modelo con métricas como MSE (Mean Squared Error) o F1-Score, y ajustarlo según sea necesario.

- Ejemplo: Evaluamos el modelo usando el conjunto de prueba y obtenemos un MSE de 15,000. Decidimos ajustar algunos parámetros o probar con otros modelos para mejorar la precisión.

7. Implementación y Monitorización

Desplegar el modelo en un entorno real y actualizarlo periódicamente para mantener su precisión.

- Ejemplo: Una vez satisfechos con el desempeño del modelo, lo implementamos en una aplicación web que predice el precio de una casa basado en sus características ingresadas por el usuario. Monitorizamos el modelo continuamente y lo actualizamos con nuevos datos para mejorar su precisión.

TIPOS DE ALGORITMO DE APRENDIZAJE DE MÁQUINA

Los algoritmos de Aprendizaje de Máquina (ML) se pueden clasificar en tres tipos principales:

Aprendizaje Supervisado

Este tipo de algoritmo se entrena utilizando un conjunto de datos etiquetados, es decir, datos en los que ya conocemos el resultado esperado. El objetivo es aprender a mapear entradas y salidas correctas. Ejemplos comunes de tareas de aprendizaje supervisado incluyen:

- Clasificación: como la identificación de correos electrónicos como "spam" o "no spam."
- Regresión: como la predicción de los precios de viviendas basado en características como el tamaño y la ubicación.

Aprendizaje No Supervisado

A diferencia del aprendizaje supervisado, en el aprendizaje no supervisado, los algoritmos trabajan con datos no etiquetados. Aquí, el objetivo es identificar estructuras o patrones subyacentes dentro de los datos. Ejemplos de tareas de aprendizaje no supervisado incluyen:

- Agrupación (Clustering): como la agrupación de clientes con características similares en segmentos.
- Reducción de dimensionalidad: como la simplificación de grandes conjuntos de datos mientras se retiene la información esencial.

Aprendizaje por Refuerzo

El aprendizaje por refuerzo se basa en el concepto de recompensa y castigo. Un agente aprende a tomar acciones en un entorno con el fin de maximizar una recompensa acumulativa. Algunos ejemplos incluyen:

- Entrenar un robot para navegar un laberinto a través de prueba y error.
- Juegos: como agentes que aprenden a jugar videojuegos ganando puntos o alcanzando metas.

APRENDIZAJE SUPERVISADO VS NO SUPERVISADO

Antes de entrar en la comparación entre aprendizaje supervisado y no supervisado, es esencial entender qué los define:

El Aprendizaje Supervisado utiliza datos de entrada y salida etiquetados, lo que significa que el modelo se entrena con pares de datos donde se conoce la respuesta correcta. Por otro lado, el Aprendizaje No Supervisado trabaja con datos sin etiquetas, tratando de encontrar estructuras o patrones intrínsecos en los datos sin supervisión directa.

Característica	Aprendizaje Supervisado	Aprendizaje No Supervisado
Definición	Utiliza datos de entrada y salida etiquetados.	Encuentra patrones en datos no etiquetados.
Ejemplo Común	Predicción de precios de viviendas basado en datos históricos.	Segmentación de clientes en un mercado según su comportamiento de compra.
Aplicaciones	Clasificación, regresión.	Agrupamiento (clustering), reducción de dimensionalidad, detección de anomalías.
Ventajas	Resultados precisos y predecibles.	Puede manejar grandes cantidades de datos no etiquetados, descubre patrones ocultos.
Desventajas	Requiere grandes cantidades de datos etiquetados, puede ser costoso y lento.	Resultados menos precisos, difícil de interpretar y evaluar la calidad de los modelos.
Algoritmos Comunes	Regresión lineal, árboles de decisión, máquinas de vectores de soporte (SVM), redes neuronales.	K-means, análisis de componentes principales (PCA), autoencoders.

Ilustración 2 Aprendizaje Supervisado vs Aprendizaje No Supervisado

ALGORITMOS DE REGRESIÓN

Una tarea de regresión busca predecir valores continuos en lugar de categorías discretas. Los algoritmos de regresión son fundamentales en el aprendizaje de máquina cuando el objetivo es estimar o prever resultados cuantitativos. Estos modelos intentan encontrar una relación entre las variables de entrada (predictoras) y la variable de salida (dependiente). A diferencia de los problemas de clasificación, donde el resultado es una categoría, en regresión el resultado es un valor numérico continuo.

ALGORITMOS MÁS UTILIZADOS EN REGRESIÓN

1. Regresión Lineal

- a. Descripción: La regresión lineal modela la relación entre una variable dependiente y una o más variables independientes utilizando una línea recta (para una sola variable) o un hiperplano (para múltiples variables).
- b. Aplicaciones: Predecir precios de viviendas, ventas en función de la inversión en publicidad, etc.

```
1 import numpy as np
2 import matplotlib.pyplot as plt
3 from sklearn.linear_model import LinearRegression
4
5 # Datos de ejemplo (tamaño de la casa en m² y precio en miles de USD)
6 X = np.array([50, 60, 70, 80, 90, 100]).reshape(-1, 1)
7 y = np.array([150, 160, 180, 200, 220, 240])
8
9 # Crear el modelo de regresión lineal y ajustarlo a los datos
10 model = LinearRegression()
11 model.fit(X, y)
12
13 # Predecir precios para nuevos datos
14 X_new = np.array([55, 65, 75, 85, 95]).reshape(-1, 1)
15 y_pred = model.predict(X_new)
16
17 # Visualizar los datos y la línea de regresión
18 plt.scatter(X, y, color='blue', label='Datos de entrenamiento')
19 plt.plot(X_new, y_pred, color='red', linestyle='--', label='Predicciones')
20 plt.xlabel('Tamaño de la casa (m²)')
21 plt.ylabel('Precio (miles de USD)')
22 plt.legend()
23 plt.show()
```

Ilustración 3 Script regresión lineal.

2. Regresión Polinómica

- Descripción: Extensión de la regresión lineal que modela relaciones no lineales al incluir términos polinómicos en las variables predictoras.
- Aplicaciones: Modelar el crecimiento de una planta en función de la cantidad de fertilizante y luz solar.

```
1 from sklearn.preprocessing import PolynomialFeatures
2
3 # Crear características polinómicas
4 poly = PolynomialFeatures(degree=2)
5 X_poly = poly.fit_transform(X)
6
7 # Ajustar el modelo de regresión polinómica
8 model = LinearRegression()
9 model.fit(X_poly, y)
10
11 # Predecir precios para nuevos datos
12 X_new_poly = poly.transform(X_new)
13 y_pred_poly = model.predict(X_new_poly)
14
15 # Visualizar los datos y la curva de regresión polinómica
16 plt.scatter(X, y, color='blue', label='Datos de entrenamiento')
17 plt.plot(X_new, y_pred_poly, color='red', linestyle='--', label='Predicciones Polinómicas')
18 plt.xlabel('Tamaño de la casa (m²)')
19 plt.ylabel('Precio (miles de USD)')
20 plt.legend()
21 plt.show()
```

Ilustración 4 Script regresión polinómica.

3. Regresión Logística

- Descripción: Utilizada para clasificación binaria, prediciendo probabilidades que pueden ser categorizadas en dos grupos. Aunque se llama regresión, se usa principalmente para problemas de clasificación.
- Aplicaciones: Determinar si un estudiante aprobará o no un examen basado en sus hábitos de estudio.

```
1 from sklearn.linear_model import LogisticRegression
2
3 # Datos de ejemplo (horas de estudio y aprobación del examen)
4 X = np.array([5, 10, 15, 20, 25, 30]).reshape(-1, 1)
5 y = np.array([0, 0, 0, 1, 1, 1]) # 0: no aprobado, 1: aprobado
6
7 # Crear y ajustar el modelo de regresión logística
8 model = LogisticRegression()
9 model.fit(X, y)
10
11 # Predecir probabilidades para nuevos datos
12 X_new = np.array([8, 12, 18, 22, 28]).reshape(-1, 1)
13 y_pred = model.predict_proba(X_new)[:, 1]
14
15 # Visualizar los datos y la curva logística
16 plt.scatter(X, y, color='blue', label='Datos de entrenamiento')
17 plt.plot(X_new, y_pred, color='red', linestyle='--', label='Probabilidades Predichas')
18 plt.xlabel('Horas de Estudio')
19 plt.ylabel('Probabilidad de Aprobar')
20 plt.legend()
21 plt.show()
```

Ilustración 5 Script regresión logística

4. Redes Neuronales para Regresión

- a. Descripción: Modelos más complejos capaces de capturar relaciones no lineales y patrones complejos en los datos. Las redes neuronales pueden tener múltiples capas y nodos, lo que les permite aprender representaciones avanzadas.
- b. Aplicaciones: Predecir precios de acciones basado en múltiples factores económicos y financieros.

```
1 from keras.models import Sequential
2 from keras.layers import Dense
3
4 # Crear el modelo de red neuronal
5 model = Sequential()
6 model.add(Dense(64, input_dim=1, activation='relu'))
7 model.add(Dense(64, activation='relu'))
8 model.add(Dense(1))
9
10 # Compilar y entrenar el modelo
11 model.compile(optimizer='adam', loss='mean_squared_error')
12 model.fit(X, y, epochs=100, verbose=0)
13
14 # Predecir precios para nuevos datos
15 y_pred = model.predict(X_new)
16
17 # Visualizar los datos y la predicción de la red neuronal
18 plt.scatter(X, y, color='blue', label='Datos de entrenamiento')
19 plt.plot(X_new, y_pred, color='red', linestyle='--', label='Predicciones Neuronal')
20 plt.xlabel('Tamaño de la casa (m²)')
21 plt.ylabel('Precio (miles de USD)')
22 plt.legend()
23 plt.show()
```


Ilustración 6 Script de redes neuronales para regresión.

ALGORITMOS DE CLASIFICACIÓN

Una tarea de clasificación asigna una etiqueta a una entrada, agrupando datos en categorías predefinidas. Es común en muchas aplicaciones del mundo real, como el filtrado de spam, la detección de fraudes y la segmentación de clientes.

Ejemplo de una Tarea de Clasificación

Identificación de correos electrónicos como "spam" o "no spam": Un modelo de clasificación puede analizar el contenido de un correo electrónico, identificando patrones comunes en los correos



etiquetados como spam por los usuarios. Basándose en estas características, puede predecir si un nuevo correo es spam o no.

TIPOS DE TAREAS DE CLASIFICACIÓN

Antes de adentrarnos en los tipos específicos, es crucial reconocer que las tareas de clasificación pueden variar en complejidad y en el número de categorías. Las tareas de clasificación asignan una etiqueta a una entrada, agrupando datos en categorías predefinidas. Estas tareas son fundamentales en muchas aplicaciones del aprendizaje de máquina, desde el diagnóstico médico hasta la detección de fraudes. Aquí se detallan los principales tipos de tareas de clasificación:

1. Clasificación Binaria

En la clasificación binaria, el objetivo es asignar una de dos posibles etiquetas a cada entrada. Esta es la forma más simple de clasificación y es ampliamente utilizada en diversas aplicaciones.

- **Descripción:** La clasificación binaria implica determinar si una instancia pertenece a una clase o a otra.
- **Ejemplo:** Clasificación de correos electrónicos como "spam" o "no spam".
- **Aplicaciones Comunes:** Detección de fraudes en transacciones financieras, diagnósticos médicos (por ejemplo, detección de cáncer basado en imágenes), y filtrado de spam en correos electrónicos.

2. Clasificación Multiclase

La clasificación multiclase extiende el concepto de la clasificación binaria a más de dos clases. En este caso, cada instancia se asigna a una de varias categorías posibles.

- **Descripción:** La clasificación multiclase asigna una etiqueta de entre varias posibles a cada instancia.
- **Ejemplo:** Clasificación de imágenes de animales en categorías como "perros", "gatos", "caballos", etc.
- **Aplicaciones Comunes:** Reconocimiento de escritura a mano (como en los dígitos del 0 al 9), clasificación de noticias en diferentes categorías temáticas, y etiquetado de productos en sitios de comercio electrónico.

3. Clasificación Multietiqueta

En la clasificación multietiqueta, una instancia puede pertenecer a múltiples clases simultáneamente. Este tipo de clasificación es útil cuando las categorías no son mutuamente excluyentes.

- **Descripción:** La clasificación multietiqueta permite que una instancia tenga varias etiquetas simultáneamente.
- **Ejemplo:** Etiquetado de un video con múltiples etiquetas como "deporte" y "aire libre".
- **Aplicaciones Comunes:** Análisis de sentimientos en redes sociales donde un texto puede expresar múltiples emociones (por ejemplo, alegría y sorpresa), y sistemas de recomendación donde un usuario puede interesarse por varias categorías de productos simultáneamente.

Tipo de Clasificación	Descripción	Ejemplo	Aplicaciones Comunes
Clasificación Binaria	Asigna una de dos posibles etiquetas a cada entrada.	Clasificación de correos electrónicos como "spam" o "no spam".	Detección de fraudes, diagnósticos médicos, filtrado de spam en correos.
Clasificación Multiclase	Asigna una etiqueta de entre varias posibles a cada entrada.	Clasificación de imágenes de animales en "perros", "gatos", "caballos", etc.	Reconocimiento de escritura a mano, clasificación de noticias, etiquetado de productos.
Clasificación Multietiqueta	Permite que una instancia tenga varias etiquetas simultáneamente.	Etiquetado de un video con "deporte" y "aire libre".	Análisis de sentimientos en redes sociales, sistemas de recomendación.

Ilustración 7 Comparativa tipos de clasificación

10. DIMENSIONES Y LA MALDICIÓN DE LA DIMENSIONALIDAD

La maldición de la dimensionalidad se refiere al problema que surge cuando el número de variables o características en un conjunto de datos aumenta significativamente, lo que provoca una explosión exponencial en el volumen del espacio de datos. Este fenómeno puede llevar a un sobreajuste y a una disminución en la eficiencia de los modelos de machine learning.

Técnicas de Mitigación:

- **Selección de Características:** Eliminar variables irrelevantes o redundantes.
- **Reducción de Dimensionalidad:** Uso de métodos como PCA (Análisis de Componentes Principales) o t-SNE para reducir el número de variables conservando la información clave.
- **Normalización y Escalado:** Transformar los datos para evitar que algunas variables dominen otras.
- **Aumento del Tamaño del Conjunto de Datos:** Recoger más datos para compensar el aumento de dimensiones.

```
1  from sklearn.decomposition import PCA
2  import matplotlib.pyplot as plt
3
4  # Datos de ejemplo con 3 dimensiones
5  X = np.array([
6      [2.5, 2.4, 3.1],
7      [0.5, 0.7, 1.8],
8      [2.2, 2.9, 2.7],
9      [1.9, 2.2, 2.3],
10     [3.1, 3.0, 3.6],
11     [2.3, 2.7, 2.5],
12     [2.0, 1.6, 2.2],
13     [1.0, 1.1, 1.5]
14 ])
15
16 # Aplicar PCA para reducir a 2 dimensiones
17 pca = PCA(n_components=2)
18 X_pca = pca.fit_transform(X)
19
20 # Visualizar los datos reducidos
21 plt.scatter(X_pca[:, 0], X_pca[:, 1])
22 plt.xlabel('Componente Principal 1')
23 plt.ylabel('Componente Principal 2')
24 plt.title('Reducción de Dimensionalidad con PCA')
25 plt.show()
```

Ilustración 8 Método PCA.