

Contenido

1. Introducción al Aprendizaje Supervisado	1
1.1. ¿Cómo funciona?.....	1
1.2. Tipos de problemas supervisados	2
1.3. Flujo de trabajo en el aprendizaje supervisado	3
1.4. Importancia del aprendizaje supervisado en la vida real	3
2. Preparación y Etiquetado de los Datos	4
2.1. Características (features) y las etiquetas (label).	4
2.2. Tipos de datos y cómo se representan	4
2.3. Etiquetado de los datos.....	4
2.4. Limpieza y preprocesamiento de los datos.	4
2.5. División del dataset.....	8
Ejemplo:Limpieza y preprocesamiento de los datos	9

1. Introducción al Aprendizaje Supervisado

El aprendizaje supervisado es una de las técnicas más utilizadas dentro de la **Inteligencia Artificial (IA)** y el **Machine Learning (ML)**. Su característica principal es que el modelo aprende a partir de un conjunto de datos que ya incluye la respuesta correcta para cada ejemplo. Esa respuesta se conoce como **etiqueta** o **target**.

En otras palabras, el aprendizaje supervisado consiste en enseñar a un algoritmo a reconocer patrones en los datos, proporcionándole ejemplos donde sabemos cuál es la solución. El objetivo es que, después de este proceso de entrenamiento, el modelo sea capaz de **predecir la etiqueta correcta para datos nuevos** que nunca ha visto.

1.1. ¿Cómo funciona?

Imagina que tenemos dos elementos fundamentales:

- **Características de entrada (features):** Son las variables que describen cada ejemplo. Por ejemplo, en el caso de una casa, podrían ser el tamaño, el número de habitaciones, la antigüedad o el barrio.
- **Salida o etiqueta (target):** Es el valor que queremos predecir. Siguiendo el mismo ejemplo, sería el **precio real** de la casa.

El modelo analiza muchos ejemplos donde conoce tanto las características como la etiqueta. A partir de ellos, aprende la relación entre las entradas y la salida. Cuando el aprendizaje es exitoso, el modelo puede estimar la etiqueta para nuevos casos.

Ejemplo práctico

Supongamos que queremos crear un modelo que prediga el precio de una vivienda. Para entrenarlo, le damos un conjunto de datos con información como:

- **Entradas (features):** tamaño en metros cuadrados, número de habitaciones, antigüedad, ubicación, etc.
- **Salida (target):** el precio real de cada casa.

Con suficientes ejemplos, el modelo detecta patrones: por ejemplo, que las casas más grandes y en barrios céntricos suelen tener precios más altos. Así, cuando le presentamos una casa nueva, puede estimar su precio basándose en lo que aprendió.

1.2. Tipos de problemas supervisados

En el aprendizaje supervisado, los problemas se dividen principalmente en dos grandes categorías: **clasificación** y **regresión**. La diferencia entre ambas radica en el tipo de salida que queremos predecir.

A) Clasificación

En los problemas de **clasificación**, el objetivo es **predecir una clase o categoría**. Esto significa que la salida del modelo será **discreta**, es decir, un conjunto limitado de opciones.

Ejemplos comunes:

- **Detección de spam:** decidir si un correo es *spam* o *no spam*.
- **Clasificación de imágenes:** identificar si una foto muestra un *gato*, un *perro* o un *pájaro*.
- **Diagnóstico médico:** determinar si un paciente tiene una enfermedad (*positivo*) o no (*negativo*).

¿Cómo funciona?

El modelo aprende a partir de ejemplos donde cada entrada está asociada a una categoría. Después, cuando recibe datos nuevos, asigna la clase más probable.

Algoritmos típicos para clasificación:

- **Regresión logística:** muy usada para problemas binarios (dos clases).
- **Árboles de decisión:** dividen los datos en ramas según sus características.
- **Random Forest:** combina muchos árboles para mejorar la precisión.
- **SVM (Máquinas de Vectores de Soporte):** busca el mejor límite entre clases.
- **Redes neuronales:** útiles para problemas complejos como reconocimiento de imágenes.

B) Regresión

En los problemas de **regresión**, el objetivo es **predecir un valor numérico**. La salida es **continua**, lo que significa que puede tomar infinitos valores dentro de un rango.

Ejemplos comunes:

- **Predecir el precio de un producto** (por ejemplo, una casa).
- **Calcular el tiempo de entrega de un pedido**.
- **Estimar la temperatura para mañana**.

¿Cómo funciona?

El modelo aprende la relación entre las características y el valor numérico esperado. Luego, cuando recibe datos nuevos, calcula una estimación.

Algoritmos típicos para regresión:

- **Regresión lineal:** el más sencillo, busca una relación lineal entre variables.
- **SVR (Regresión con Máquinas de Vectores de Soporte):** versión para valores continuos.
- **Random Forest Regressor:** similar al Random Forest, pero para valores numéricos.
- **Redes neuronales:** muy potentes para relaciones complejas.

1.3. Flujo de trabajo en el aprendizaje supervisado

Para resolver un problema supervisado, seguimos un **pipeline** o flujo de trabajo que asegura que el modelo se entrene correctamente:

1. **Recolectar datos:** obtener información relevante para el problema.
2. **Etiquetar los datos:** si no tienen etiqueta, debemos asignarla.
3. **Limpiar y preparar los datos:** eliminar errores, valores faltantes y normalizar.
4. **Dividir los datos:** en tres conjuntos:
 - **Entrenamiento (train):** para que el modelo aprenda.
 - **Validación (validation):** para ajustar parámetros.
 - **Prueba (test):** para evaluar el rendimiento final.
5. **Entrenar el modelo:** aplicar el algoritmo elegido.
6. **Evaluar y optimizar:** medir la precisión y mejorar el modelo.
7. **Probar con datos nuevos:** comprobar que funciona en casos reales.
8. **Usar para predicción real:** aplicar el modelo en producción.

Este proceso se conoce como **pipeline de Machine Learning**.

1.4. Importancia del aprendizaje supervisado en la vida real

El aprendizaje supervisado está presente en muchos sistemas que usamos a diario:

- **Recomendaciones personalizadas:** en Netflix, Spotify o Amazon.
- **Detección de fraude:** en transacciones bancarias.
- **Diagnóstico asistido:** en hospitales y clínicas.
- **Vehículos autónomos:** para reconocer señales y peatones.
- **Predicción de demanda:** en empresas para planificar producción.

En resumen, es una herramienta clave para **automatizar decisiones basadas en datos**, mejorando la eficiencia y reduciendo errores humanos.

2. Preparación y Etiquetado de los Datos

La preparación del **dataset** es una de las fases más críticas en el aprendizaje supervisado. Un modelo **solo será tan bueno como los datos que reciba**. Si los datos son incompletos, incorrectos o están mal organizados, el modelo aprenderá patrones erróneos y sus predicciones serán poco fiables.

2.1. Características (features) y las etiquetas (label).

Ya hemos visto en el punto Introducción al Aprendizaje Supervisado lo que son.

En resumen: **features → información para predecir | label → lo que queremos predecir.**

2.2. Tipos de datos y cómo se representan

- **Características numéricas:** Valores cuantitativos como altura, precio, velocidad, edad. Se pueden usar directamente en la mayoría de algoritmos.
- **Características categóricas:** Valores cualitativos como color, provincia, tipo de producto. **Importante:** los algoritmos no entienden texto, por lo que debemos convertir estas categorías en números mediante técnicas como One-Hot Encoding (crear columnas binarias para cada categoría).

2.3. Etiquetado de los datos

Para aplicar aprendizaje supervisado, **cada muestra del dataset debe tener su etiqueta correspondiente.**

Ejemplos:

- Un correo marcado como *spam* o *no spam*.
- Una radiografía etiquetada como *fractura* o *normal*.
- Un registro de ventas con el precio real del producto.

¿Qué pasa si los datos no están etiquetados?

- Hay que etiquetarlos manualmente (lo que implica un coste alto).
- Si no se pueden etiquetar, se debe usar aprendizaje **no supervisado**, pero eso no sirve para este tipo de problemas.

Problemas comunes al etiquetar:

- Etiquetas incorrectas o inconsistentes.
- Etiquetas ruidosas (marcadas por diferentes personas sin criterio común).
- Etiquetas incompletas.

2.4. Limpieza y preprocesamiento de los datos.

Antes de entrenar un modelo de aprendizaje automático, es imprescindible realizar un proceso de **limpieza y preprocesamiento** del dataset. La calidad de los datos influye directamente en el rendimiento del modelo: *datos sucios → modelos poco fiables*.

1) Eliminación de duplicados.

Los duplicados son registros repetidos que no aportan información nueva.

¿Por qué eliminarlos?

- Introducen sesgos en el entrenamiento.
- El modelo puede “memorizar” patrones artificiales.
- Afectan a métricas como precisión o error.

Ejemplo:

Si un cliente aparece varias veces con los mismos datos, el modelo puede darle más peso del que realmente tiene.

Acción habitual:

- Identificar filas idénticas.
- Eliminar duplicados completos o parciales según el caso.

2) Tratamiento de valores nulos (missing values).

Los valores nulos representan datos ausentes y son muy comunes en datasets reales.

Opciones para tratarlos:**a) Eliminación de filas o columnas.**

- Se usa cuando el porcentaje de valores nulos es muy alto.
- Riesgo: pérdida de información importante.

Ejemplo:

Eliminar una columna si más del 70% de sus valores están vacíos.

b) Imputación de valores.

Consiste en sustituir los valores nulos por valores estimados.

- **Media:** adecuada si los datos siguen una distribución normal.
- **Mediana:** preferible cuando hay outliers.
- **Moda:** habitual en variables categóricas.

Ejemplo:

Si falta la edad de un usuario → sustituir por la edad media del dataset.

c) Uso de algoritmos que toleran valores nulos.

Algunos modelos, como ciertos árboles de decisión, pueden manejar valores faltantes sin imputación previa.

3) Normalización y escalado.

Las variables numéricas pueden estar en rangos muy diferentes, lo que afecta a muchos algoritmos.

Problema:

Una variable con valores grandes puede dominar sobre otras.

Ejemplo:

- Edad: 0–100

- Ingresos: 0–100.000

El modelo dará más importancia a los ingresos si no se escalan.

Técnicas habituales:

- **Normalización (Min-Max Scaling):** transforma los valores de una variable para que todos queden dentro de un **rango fijo**, normalmente **[0, 1]**.

Valor original	Valor normalizado
20	0.00
30	0.25
40	0.50
60	1.00

(suponiendo min = 20 y max = 60)

¿Cuándo se recomienda?

- Cuando los datos **no siguen una distribución normal**.
- Cuando se usan modelos basados en **distanacias**.
- En **redes neuronales** (mejora la convergencia).
- Cuando se necesita un rango fijo (por ejemplo, [0,1]).

Modelos sensibles al Min-Max

- KNN.
- Redes neuronales.
- SVM.
- Algoritmos basados en distancia.

- **Estandarización (Standard Scaling):** La **estandarización** transforma los datos para que tengan:
 - **Media = 0**
 - **Desviación típica = 1**

No fija un rango concreto, pero centra y escala los datos.

Valor original	Valor estandarizado
1.000 €	-1.20
2.000 €	-0.40
3.000 €	0.00
4.000 €	0.40
5.000 €	1.20

Características principales

- Menos sensible a outliers que Min-Max.
- Adecuada si los datos siguen una **distribución aproximadamente normal**.
- Facilita la interpretación estadística.

¿Cuándo se recomienda?

- Cuando los datos siguen (o se aproximan a) una distribución normal.
- En modelos que asumen datos centrados.
- En algoritmos lineales.

Modelos sensibles a la estandarización

- Regresión lineal.
- Regresión logística.
- SVM.
- PCA.
- Redes neuronales.

4) Codificación de variables categóricas

Los modelos trabajan con números, por lo que las variables categóricas deben transformarse.

Métodos principales:

a) Label Encoding.

- Asigna un número a cada categoría.
- Riesgo: el modelo puede interpretar un orden inexistente.

Ejemplo:

Rojo → 0

Verde → 1

Azul → 2

b) One-Hot Encoding.

- Crea una columna por cada categoría.
- Evita relaciones de orden artificiales.

Ejemplo:

Color = Rojo → (1,0,0)

Color = Verde → (0,1,0)

Recomendado para la mayoría de modelos.

5) Detección de outliers (valores atípicos)

Los outliers son valores extremos que se alejan significativamente del resto de los datos.

Problemas que generan:

- Distorsionan medias y desviaciones.
- Afectan negativamente a modelos de regresión.
- Pueden aumentar el error del modelo.

Ejemplos de detección:

- Rango intercuartílico (IQR).
- Z-score.
- Análisis visual (boxplots).

Opciones de tratamiento:

- Eliminarlos si son errores.
- Limitar su valor (clipping).
- Usar modelos robustos a outliers.

El preprocessamiento es una fase **crítica** del pipeline de Machine Learning. Un modelo avanzado **no compensará** unos datos mal tratados. En muchos proyectos reales, esta fase consume **más tiempo que el entrenamiento del modelo**.

2.5. División del dataset.

Antes del entrenamiento, los datos se dividen en tres conjuntos:

Conjunto	Función
Train	Para que el modelo aprenda
Validation	Para ajustar hiperparámetros
Test	Para comprobar el rendimiento real

Proporciones frecuentes:

- 70% / 15% / 15%
- 60% / 20% / 20%

2.5.1. Estratificación

En muchos problemas de **clasificación**, los datasets presentan **clases desbalanceadas**, es decir, una clase aparece con mucha mayor frecuencia que otra.

Ejemplo típico:

- Detección de fraude
 - 99% → transacciones normales
 - 1% → transacciones fraudulentas

Si se divide el dataset de forma aleatoria sin control, puede ocurrir que:

- El conjunto de entrenamiento tenga muy pocos ejemplos de la clase minoritaria.
- El conjunto de test **no contenga apenas (o ningún) caso** de la clase minoritaria.

Esto provoca evaluaciones **engañosas** y modelos poco útiles.

¿Qué es la estratificación?

La **estratificación** es una técnica que garantiza que **la proporción de clases** de la variable objetivo se mantenga **similar** en todos los subconjuntos del dataset: de entrenamiento, de validación y de prueba. De esta forma, cada conjunto es representativo del dataset original.

¿Por qué es importante la estratificación?

- Evita que una clase minoritaria desaparezca en el conjunto de test.
- Permite una evaluación más justa del modelo.
- Reduce la variabilidad de las métricas entre particiones.
- Es imprescindible en datasets muy desbalanceados.

Ejemplo: Limpieza y preprocesamiento de los datos

1) Eliminación de duplicados

Dataset original

Nombre	Edad	Ciudad
Ana	20	Madrid
Luis	22	Sevilla
Ana	20	Madrid

- ⊕ Problema: Ana aparece dos veces con los mismos datos.
- ⊕ Qué puede pasar: El modelo “cree” que hay dos Anas, cuando solo hay una.
- ⊕ Acción: Eliminar la fila duplicada.

Dataset limpio:

Nombre	Edad	Ciudad
Ana	20	Madrid
Luis	22	Sevilla

2) Tratamiento de valores nulos (missing values)

Ejemplo de valores nulos

Nombre	Edad	Nota
Ana	20	8
Luis		7
Marta	22	

a) Eliminación de filas

Si falta mucha información en una fila: Eliminar la fila de Luis si no sabemos su edad y es importante.

Nombre	Edad	Nota
Ana	20	8
Marta	22	

b) Imputación de valores

Imputación con la media. Edades conocidas: 20 y 22.

$$\oplus \text{ Media} = (20 + 22) / 2 = 21$$

Sustituimos la edad de Luis:

Nombre	Edad	Nota
Ana	20	8
Luis	21	7
Marta	22	

Imputación con la moda (categórica)

Alumno	Curso
Ana	1º
Luis	X
Marta	1º

⊕ Curso más repetido = 1º

Alumno	Curso
Ana	1º
Luis	1º
Marta	1º

3) Normalización y escalado

Dataset sin escalar

Persona	Edad	Ingresos
Ana	20	20.000
Luis	40	80.000

- ⊕ Problema: 80.000 es muchísimo mayor que 40 → el modelo “solo mira” ingresos.
- Normalización (Min-Max)

Escalamos todo entre 0 y 1.(Escalar es como poner todas las variables en la misma regla.)

Persona	Edad (esc.)	Ingresos (esc.)
Ana	0.0	0.0
Luis	1.0	1.0

⊕ Ahora edad e ingresos pesan por igual.

4) Codificación de variables categóricas

Variable categórica original

Persona	Color favorito
Ana	Rojo
Luis	Verde
Marta	Azul

a) Label Encoding (muy básico)

Color	Código
Rojo	0
Verde	1
Azul	2

⊕ Problema: El modelo puede pensar: Azul > Verde > Rojo (y no es verdad).

b) One-Hot Encoding (recomendado)

Persona	Rojo	Verde	Azul
Ana	1	0	0
Luis	0	1	0
Marta	0	0	1

- ⊕ No hay orden, solo presencia o ausencia.

5) Detección de outliers (valores atípicos)**Dataset de salarios**

Persona	Salario
Ana	1.200
Luis	1.300
Marta	1.250
Pedro	50.000

- ⊕ Problema evidente: Pedro gana muchísimo más que el resto.
- ⊕ Qué puede pasar: La media se dispara y el modelo se distorsiona.

Opciones sencillas

- Eliminar el outlier (si es un error): Eliminar a Pedro si es un dato incorrecto.
- Limitar el valor (clipping): Cambiar 50.000 → 2.000 (máximo razonable).