

---

## Carga de datos

```
wine = load_wine()
X = pd.DataFrame(wine.data, columns=wine.feature_names)
y = pd.Series(wine.target)
```

### ✦ Qué pasa aquí

- Cargamos un dataset real (ya curado, como Iris)
- $x$  = features (variables de entrada)
- $y$  = target (clase del vino)

### 📐 Interpretación geométrica:

- Si hay  $n$  columnas  $\rightarrow$  espacio  $\mathbb{R}^n$
- Cada fila  $\rightarrow$  un punto en ese espacio

## 2 Exploración básica

```
X.info()
y.value_counts()
```

### ✦ Para qué sirve

- Ver si hay valores faltantes
- Ver cuántas variables tenemos
- Ver si las clases están balanceadas

### 👉 Esto conecta con:

- calidad de datos
- viabilidad del problema

## 3 División train / test

```
train_test_split(...)
```

### ✦ Por qué es clave

- Entrenamos con una parte

- Probamos con datos que el modelo **no vio**
- Simula datos “del mundo real”

⚠ Si evalúas con los mismos datos que entrenas → te engañas.

## 4 Escalado de datos

`StandardScaler()`

### 📌 Qué hace

- Lleva las variables a una escala comparable
- Media  $\approx 0$ , desviación  $\approx 1$

### 📌 Por qué es necesario

- Modelos como:
  - Regresión logística
  - KNN
  - SVM

dependen de distancias

⚠ Importantísimo:

- El scaler se ajusta **solo con train**
- Luego se aplica a test

(evita fuga de información)

## 5 Entrenamiento del modelo

`model.fit(X_train_scaled, y_train)`

### 📌 Qué significa entrenar

- Resolver un problema de optimización
- Ajustar parámetros para separar clases
- Aprender fronteras en  $\mathbb{R}^n$

### 📌 Por qué empezamos con uno simple

- Más barato

- Más interpretable
- Menos riesgo de overfitting

## 6 Evaluación

```
accuracy_score  
classification_report
```

### ✦ Qué medimos

- Accuracy → qué tanto acierta
- Precision / Recall / F1 → calidad por clase

### ✦ Por qué no solo accuracy

- Puede ocultar errores graves
- Especialmente con clases desbalanceadas

## 7 Validación cruzada

```
cross_val_score(...)
```

### ✦ Qué hace

- Parte los datos de entrenamiento varias veces
- Entrena y evalúa en distintos subconjuntos

### ✦ Para qué sirve

- Detectar sobreentrenamiento
- Ver si el modelo es estable
- Mejorar generalización

👉 Es como “estresar” al modelo