

APUNTES DE INTRODUCCIÓN

Modelos predictivos

lo esencial

Tipos de problema, proceso, preparación de datos, algoritmos, evaluación, series temporales, interpretabilidad, librerías de Python, producción, ética y glosario

Autor: David M.
davmacsolutions.com

Octubre de 2026

ÍNDICE

1. Qué es un modelo predictivo	2
2. Tipos de problemas	3
3. El proceso de un proyecto predictivo	3
4. Preparar los datos	4
5. Entrenar y validar	5
6. Algoritmos principales	6
7. Evaluar un modelo	7
8. Series temporales	8
9. Interpretar el modelo	9
10. Del modelo al negocio	9
11. Librerías de Python	10
12. Un ejemplo completo en Python	12
13. Ética y marco legal	13
14. Errores frecuentes	13
15. Lista de comprobación	13
16. Glosario	14

1. QUÉ ES UN MODELO PREDICTIVO

Un modelo predictivo es una función aprendida a partir de datos históricos que estima un valor desconocido: cuánto se venderá la semana que viene, qué cliente se dará de baja o cuál es el precio adecuado para una habitación. Se construye con **aprendizaje supervisado**: se le muestran muchos ejemplos en los que ya se conoce la respuesta y el algoritmo aprende la relación entre las variables de entrada y el resultado.

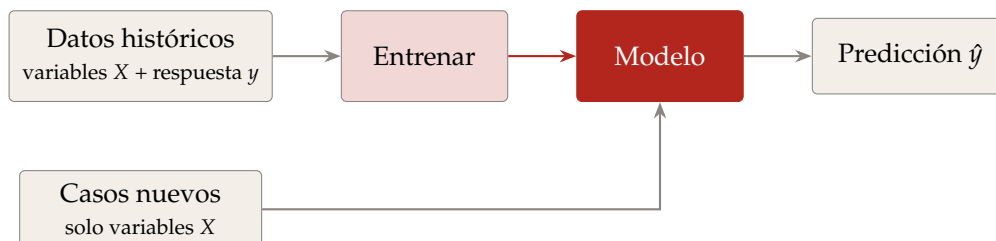


Figura 1. Aprendizaje supervisado: se entrena con el pasado, en el que se conoce la respuesta, y se aplica a casos nuevos.

1.1 Predecir no es lo mismo que explicar

Un modelo puede acertar mucho sin que sus variables sean la *causa* del resultado. Que los clientes que llaman a soporte se den más de baja no significa que llamar a soporte provoque la baja. Para decidir acciones hay que combinar el modelo con conocimiento del negocio y, cuando se pueda, con experimentos.

1.2 Vocabulario básico

Término	Significado
Variable objetivo (y)	Lo que se quiere predecir: ventas, baja sí/no, precio.
Variables predictoras (X)	La información disponible para predecir (<i>features</i>).
Observación	Cada fila del conjunto de datos: un cliente, un día, un producto.
Entrenamiento	Proceso en el que el algoritmo ajusta sus parámetros a los datos.
Inferencia	Usar el modelo ya entrenado para predecir casos nuevos.

2. TIPOS DE PROBLEMAS

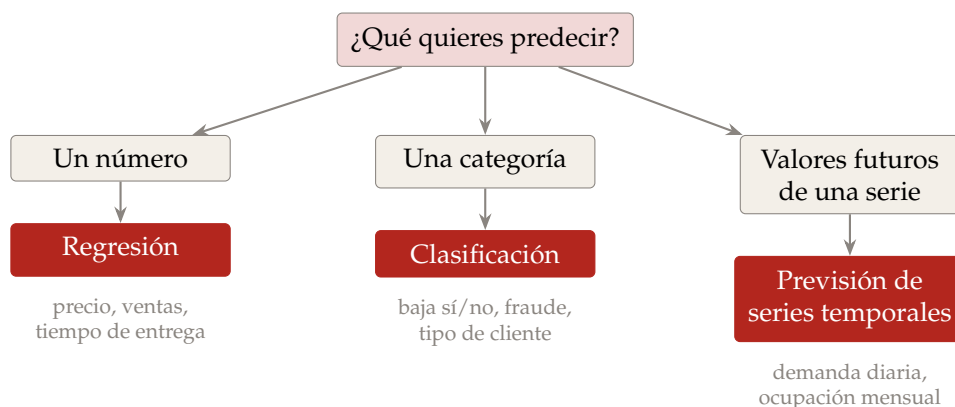


Figura 2. Qué tipo de modelo necesitas según lo que quieres predecir.

Regresión. Predice una cantidad continua. Ejemplo: el precio óptimo de un apartamento según ubicación, temporada y competencia.

Clasificación. Predice a qué grupo pertenece un caso. Puede ser binaria (sí/no) o multiclase. Normalmente devuelve una *probabilidad* que luego se convierte en decisión con un umbral.

Series temporales. Predice valores futuros a partir de su propia historia y de factores externos (festivos, promociones, clima). El orden en el tiempo importa.

Otros. Sistemas de recomendación, detección de anomalías o segmentación. La segmentación (*clustering*) es aprendizaje *no* supervisado: no hay respuesta conocida, se buscan grupos parecidos.

3. EL PROCESO DE UN PROYECTO PREDICTIVO

El algoritmo es solo una parte pequeña del trabajo. La mayor parte del tiempo se va en entender el problema y preparar los datos.

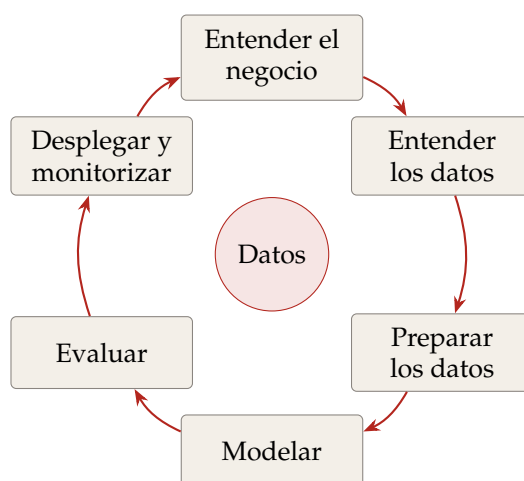


Figura 3. Ciclo de vida de un modelo, inspirado en la metodología CRISP-DM. Es iterativo: se vuelve atrás a menudo.

1. **Entender el negocio:** qué decisión se va a tomar con la predicción, cuánto cuesta equivocarse y cómo se medirá el éxito.
2. **Entender los datos:** qué hay, de dónde viene, con qué calidad y desde cuándo.
3. **Preparar los datos:** limpiar, unir fuentes y crear variables útiles.
4. **Modelar:** probar varios algoritmos, empezando por los más sencillos.
5. **Evaluar:** con datos que el modelo no ha visto y con métricas que tengan sentido para el negocio.
6. **Desplegar y monitorizar:** integrar el modelo en el día a día y vigilar que sigue funcionando.

Observación. Antes de construir nada, define una **línea base**: la predicción más sencilla posible, como «lo mismo que la semana pasada» o «la media». Si el modelo no la mejora claramente, no aporta valor.

4. PREPARAR LOS DATOS

4.1 Limpieza

- **Valores ausentes:** eliminar filas, imputar (media, mediana, valor más frecuente) o añadir una variable que indique que faltaba el dato.
- **Valores atípicos:** revisar si son errores o casos reales extremos antes de quitarlos.
- **Duplicados e incoherencias:** fechas imposibles, unidades mezcladas, categorías escritas de varias formas.

4.2 Ingeniería de variables

Crear buenas variables suele mejorar más un modelo que cambiar de algoritmo.

Técnica	Ejemplo
Variables de fecha	Día de la semana, mes, festivo, días hasta Navidad.
Agregados históricos	Compras de los últimos 30 días, ticket medio, días desde la última compra.
Retardos (<i>lags</i>)	Ventas de ayer y de hace una semana, para series temporales.
Ratios	Precio propio frente al precio medio de la competencia.
Codificar categorías	<i>One-hot</i> (una columna 0/1 por categoría) o codificación por la media del objetivo.
Escalado	Llevar las variables a rangos comparables; necesario en modelos lineales, k-NN o redes neuronales.

ATENCIÓN. Fuga de datos (*data leakage*): ocurre cuando el modelo se entrena con información que no estará disponible en el momento de predecir. Por ejemplo, usar «fecha de baja» para predecir si un cliente se dará de baja. El modelo parece excelente en las pruebas y fracasa en la realidad. Pregúntate siempre: ¿conocería este dato en el momento de la predicción?

5. ENTRENAR Y VALIDAR

Un modelo se evalúa siempre con datos que no ha visto durante el entrenamiento. Si se evalúa con los mismos datos, se mide su memoria, no su capacidad de predecir.

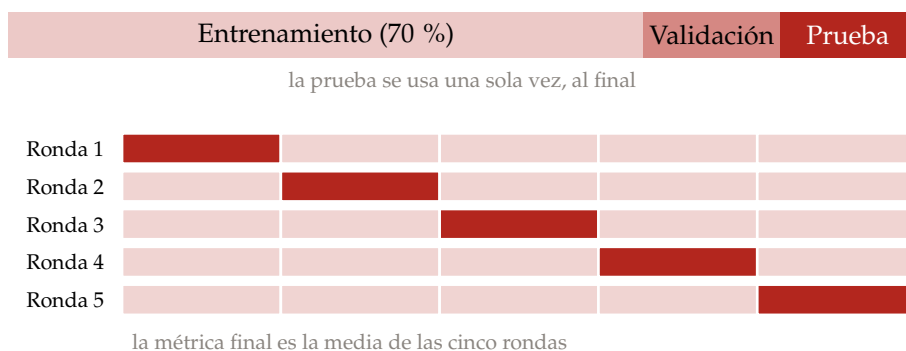


Figura 4. Arriba, la división clásica de los datos. Abajo, la validación cruzada con cinco particiones: cada bloque sirve una vez para validar.

Entrenamiento. Datos con los que el algoritmo aprende.

Validación. Datos para comparar modelos y ajustar hiperparámetros (la configuración del algoritmo, como la profundidad de un árbol).

Prueba. Datos reservados para estimar, una sola vez, cómo funcionará el modelo con datos nuevos.

Validación cruzada. Repite el entrenamiento y la validación con distintas particiones para obtener una estimación más estable, sobre todo con pocos datos.

5.1 Sobreajuste e infraajuste

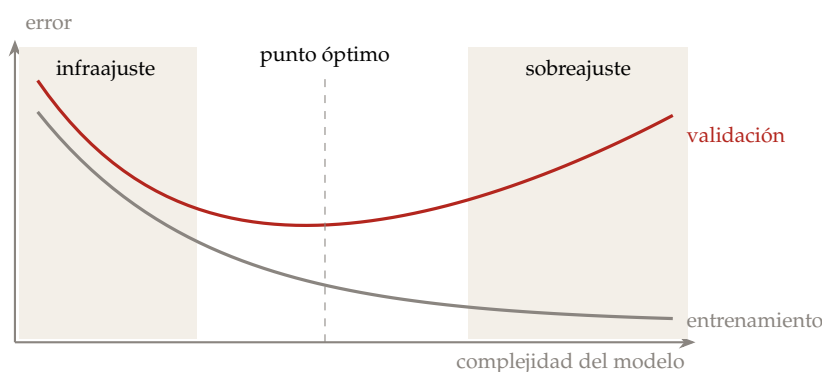


Figura 5. Al aumentar la complejidad, el error de entrenamiento siempre baja, pero el de validación vuelve a subir cuando el modelo empieza a memorizar ruido.

- **Infraajuste:** el modelo es demasiado simple y falla incluso con los datos de entrenamiento. Solución: más variables o un modelo más flexible.
- **Sobreaajuste:** el modelo memoriza el entrenamiento y falla con datos nuevos. Solución: más datos, menos variables, **regularización** (penalizar la complejidad), limitar la profundidad de los árboles o parar el entrenamiento antes.

6. ALGORITMOS PRINCIPALES

Algoritmo	Ventajas	Limitaciones
Regresión lineal	Muy interpretable, rápida, buena línea base.	Solo capta relaciones lineales salvo que se transformen variables.
Regresión logística	El estándar interpretable para clasificación binaria; da probabilidades.	Igual que la lineal: relaciones sencillas.
Árbol de decisión	Fácil de explicar con reglas «si... entonces».	Inestable y propenso al sobreajuste por sí solo.
Random forest	Muchos árboles promediados; robusto y con pocos ajustes.	Menos interpretable; modelos pesados.
Gradient boosting (XGBoost, LightGBM, CatBoost)	Suele ser el más preciso con datos tabulares.	Más hiperparámetros; puede sobreajustar si no se valida bien.
k vecinos (k-NN)	Intuitivo: predice como los casos más parecidos.	Lento con muchos datos; sensible a la escala.
Redes neuronales	Imprescindibles para imágenes, texto y audio.	Necesitan muchos datos; con tablas no suelen superar al boosting.

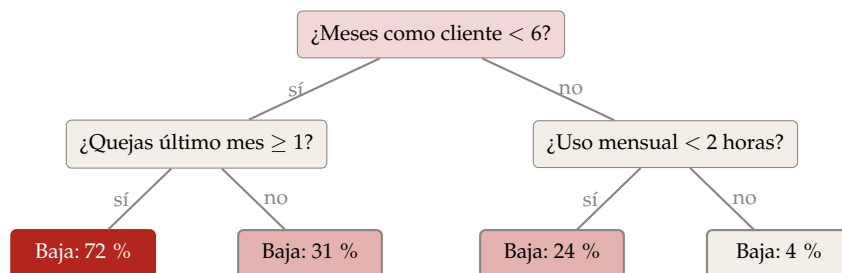
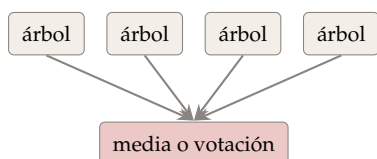


Figura 6. Un árbol de decisión para predecir la baja de clientes. Cada hoja indica la probabilidad estimada.

6.1 Modelos de conjunto

Los algoritmos más usados en la práctica combinan muchos árboles sencillos:

Bagging (random forest)



Boosting (XGBoost, LightGBM)

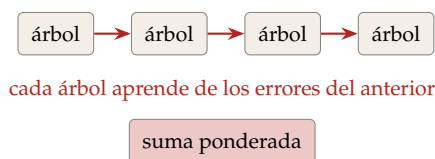


Figura 7. Bagging entrena árboles en paralelo y promedia; boosting los encadena y cada uno corrige los errores del anterior.

7. EVALUAR UN MODELO

7.1 Métricas de regresión

Métrica	Qué mide	Cuándo usarla
MAE	Error medio absoluto, en las mismas unidades que el objetivo.	Fácil de explicar: «nos equivocamos de media en 12 unidades».
RMSE	Raíz del error cuadrático medio; castiga más los errores grandes.	Cuando un error grande es mucho peor que varios pequeños.
MAPE	Error porcentual medio.	Para comparar productos de volúmenes distintos; falla con valores cercanos a cero.
R^2	Proporción de la variabilidad explicada (máximo 1).	Para una idea general, nunca como única métrica.

$$MAE = \frac{1}{n} \sum_{i=1}^n |y_i - \hat{y}_i| \quad RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2}$$

7.2 Métricas de clasificación

Todo parte de la **matriz de confusión**, que cruza lo que ocurrió con lo que predijo el modelo.

		PREDICCIÓN	
		predice baja	predice que sigue
REALIDAD	se dio de baja	80 verdaderos positivos	20 falsos negativos
	siguió	60 falsos positivos	840 verdaderos negativos

Figura 8. Matriz de confusión de un modelo de bajas sobre 1.000 clientes de prueba.

Métrica	Pregunta que responde	En el ejemplo
Exactitud (<i>accuracy</i>)	¿Qué proporción acierta en total?	920/1000 = 92 %
Precisión	De los que señala como baja, ¿cuántos lo son?	80/140 = 57 %
Exhaustividad (<i>recall</i>)	De las bajas reales, ¿cuántas detecta?	80/100 = 80 %
F1	Media armónica de precisión y exhaustividad.	67 %

ATENCIÓN. Con clases desequilibradas la exactitud engaña. Si solo el 2 % de las transacciones son fraude, un modelo que nunca detecta fraude acierta el 98 %. Usa precisión, exhaustividad, F1 o el área bajo la curva.

7.3 Umbral de decisión y curva ROC

Un clasificador devuelve una probabilidad. El **umbral** (por defecto 0,5) decide a partir de qué valor se actúa. Bajarlo detecta más casos a costa de más falsas alarmas. La curva ROC muestra ese equilibrio para todos los umbrales; el área bajo ella (**AUC**) resume la capacidad del modelo para ordenar los casos: 0,5 equivale al azar y 1 a la perfección.

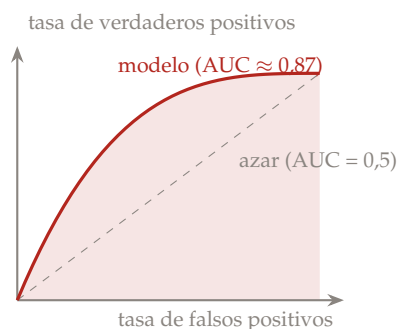


Figura 9. Curva ROC: cuanto más se acerca a la esquina superior izquierda, mejor separa el modelo.

Observación. Elige el umbral según el coste de cada error. Si retener a un cliente cuesta 10 € y perderlo cuesta 300 €, compensa avisar a muchos clientes aunque varios no se fueran a ir.

8. SERIES TEMPORALES

8.1 Componentes

- **Tendencia:** dirección a largo plazo (crecimiento, caída).
- **Estacionalidad:** patrones que se repiten (fines de semana, verano, Navidad).
- **Ruido:** variación aleatoria que no se puede predecir.

8.2 Métodos habituales

Método	Idea
Ingenuo y estacional ingenuo	Mañana igual que hoy, o igual que el mismo día de la semana pasada. La línea base obligatoria.
Media móvil y suavizado exponencial (ETS)	Promedios que dan más peso a los datos recientes; capturan tendencia y estacionalidad.
ARIMA / SARIMA	Modelos estadísticos clásicos basados en la autocorrelación de la serie.
Prophet	Librería que descompone tendencia, estacionalidades y festivos; sencilla de usar.
Modelos de aprendizaje automático	Gradient boosting con retardos, fechas y variables externas; muy eficaces con muchas series a la vez.

8.3 Validar sin mirar al futuro

En una serie temporal no se pueden mezclar los datos al azar: el modelo nunca debe entrenarse con datos posteriores a los que predice.

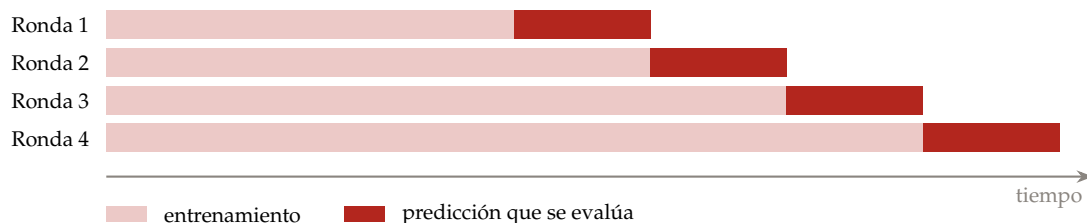


Figura 10. Validación con origen móvil (backtesting): cada ronda entrena con todo el pasado disponible y predice el periodo siguiente.

9. INTERPRETAR EL MODELO

Un modelo que nadie entiende es un modelo que nadie usa. Además, explicar las predicciones ayuda a detectar errores y sesgos.

Importancia de variables. Qué variables pesan más en el modelo en general.

Valores SHAP. Cuánto aporta cada variable a una predicción concreta: «a este cliente se le asigna un 70 % de baja sobre todo por sus tres quejas y por llevar solo dos meses».

Dependencia parcial. Cómo cambia la predicción al variar una sola variable, por ejemplo el precio.

Coeficientes. En modelos lineales, cada coeficiente indica cuánto cambia la predicción por cada unidad de la variable.

10. DEL MODELO AL NEGOCIO

10.1 Casos de uso habituales

Caso	Qué se predice	Tipo
Previsión de demanda	Unidades por producto y semana para ajustar el stock.	Series
Abandono de clientes	Probabilidad de baja para actuar antes.	Clasificación
Puntuación de clientes potenciales	Probabilidad de que un contacto compre.	Clasificación
Precios dinámicos	Precio que maximiza ingresos u ocupación.	Regresión
Fraude	Probabilidad de que una operación sea fraudulenta.	Clasificación
Mantenimiento predictivo	Probabilidad de avería de una máquina.	Clasificación
Valor de vida del cliente	Ingresos futuros esperados de cada cliente.	Regresión

10.2 Puesta en producción

- **Por lotes:** el modelo se ejecuta cada noche o cada semana y guarda las predicciones en una tabla. Es lo más sencillo y suficiente en la mayoría de casos.
- **En tiempo real:** una API devuelve la predicción al instante, por ejemplo para aprobar una operación.
- **Monitorización:** el mundo cambia y los modelos se degradan. Hay que vigilar la **deriva de datos** (*data drift*: las variables de entrada cambian) y la calidad de las predicciones, y reentrenar cuando empeoran.
- **Herramientas MLOps:** MLflow para registrar experimentos y versiones; plataformas como Databricks, AWS SageMaker, Azure Machine Learning o Google Vertex AI para entrenar, desplegar y vigilar modelos.

11. LIBRERÍAS DE PYTHON

Python es el lenguaje de referencia para crear modelos predictivos. Casi todo el trabajo se hace con un puñado de librerías gratuitas y de código abierto.

Producción	MLflow · FastAPI · joblib
Ajuste y explicación	Optuna · SHAP · imbalanced-learn
Series temporales	Prophet · statsforecast · Darts · sktime
Boosting	XGBoost · LightGBM · CatBoost
Modelos clásicos	scikit-learn · statsmodels
Visualización	matplotlib · seaborn · Plotly
Datos	pandas · NumPy · Polars · PySpark

Figura 11. La pila de librerías de un proyecto predictivo, de los datos a la producción.

Librería	Para qué sirve
pandas	Leer, limpiar, unir y transformar tablas (CSV, Excel, SQL). La base de todo.
NumPy	Cálculo numérico con matrices; está debajo de casi todas las demás.
Polars	Alternativa a pandas mucho más rápida para volúmenes grandes.
matplotlib, seaborn, Plotly	Gráficos para explorar datos y presentar resultados; Plotly los hace interactivos.
scikit-learn	La caja de herramientas principal: preparación de datos, decenas de algoritmos, validación cruzada, métricas y <i>pipelines</i> .
statsmodels	Modelos estadísticos con inferencia (coeficientes, p-valores), regresión clásica y ARIMA.
XGBoost, LightGBM, CatBoost	Gradient boosting de alto rendimiento. CatBoost maneja muy bien las variables categóricas; LightGBM destaca en velocidad.
Prophet	Previsión de series con tendencia, estacionalidades y festivos en pocas líneas.
statsforecast, Darts, sktime	Previsión de muchas series a la vez y comparación de métodos clásicos y de aprendizaje automático.
imbalanced-learn	Técnicas para clases desequilibradas (sobremuestreo, submuestreo).
Optuna	Búsqueda automática de los mejores hiperparámetros.
SHAP	Explicar qué variables impulsan cada predicción.
PyTorch, TensorFlow/Keras	Redes neuronales y aprendizaje profundo (imágenes, texto, audio).
PySpark (MLlib)	Datos y modelos distribuidos en un clúster, por ejemplo en Databricks.
MLflow	Registrar experimentos, métricas y versiones de los modelos.
FastAPI, joblib	Guardar un modelo entrenado y servirlo como API.

11.1 Instalar el entorno

```
python -m venv .venv          # entorno aislado para el proyecto
source .venv/bin/activate     # en Windows: .venv\Scripts\activate
pip install pandas scikit-learn lightgbm shap optuna prophet matplotlib jupyter
```

Observación. Trabaja siempre en un entorno virtual y guarda las versiones con `pip freeze > requirements.txt`. Así el modelo podrá reproducirse meses después en otro ordenador.

11.2 Pequeños ejemplos

```
# Gradient boosting con LightGBM
import lightgbm as lgb
modelo = lgb.LGBMRegressor(n_estimators=500, learning_rate=0.05)
modelo.fit(X_train, y_train, eval_set=[(X_val, y_val)])

# Prevision de una serie con Prophet (columnas ds = fecha, y = valor)
from prophet import Prophet
m = Prophet(yearly_seasonality=True, weekly_seasonality=True)
m.fit(ventas_diarias)
futuro = m.make_future_dataframe(periods=30)
prevision = m.predict(futuro)[["ds", "yhat", "yhat_lower", "yhat_upper"]]
```

```

# Explicar predicciones con SHAP
import shap
explicador = shap.TreeExplainer(modelo)
valores = explicador.shap_values(X_val)
shap.summary_plot(valores, X_val)

# Buscar hiperparametros con Optuna
import optuna
from sklearn.model_selection import cross_val_score

def objetivo(trial):
    params = {
        "num_leaves": trial.suggest_int("num_leaves", 15, 255),
        "learning_rate": trial.suggest_float("learning_rate", 0.01, 0.3, log=True),
    }
    return cross_val_score(lgb.LGBMRegressor(**params), X_train, y_train,
                           cv=5, scoring="neg_mean_absolute_error").mean()

estudio = optuna.create_study(direction="maximize")
estudio.optimize(objetivo, n_trials=50)
print(estudio.best_params)

# Guardar el modelo entrenado y cargarlo despues para predecir
import joblib
joblib.dump(modelo, "modelo_demanda.joblib")
modelo = joblib.load("modelo_demanda.joblib")

```

12. UN EJEMPLO COMPLETO EN PYTHON

Modelo de abandono de clientes con scikit-learn: preparación, entrenamiento, evaluación y predicción en un mismo flujo (*pipeline*), lo que evita aplicar transformaciones distintas al entrenar y al predecir.

```

import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.compose import ColumnTransformer
from sklearn.preprocessing import OneHotEncoder, StandardScaler
from sklearn.pipeline import Pipeline
from sklearn.ensemble import HistGradientBoostingClassifier
from sklearn.metrics import classification_report, roc_auc_score

df = pd.read_csv("clientes.csv")
X = df.drop(columns=["baja"])
y = df["baja"]

X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, stratify=y, random_state=42)

numericas = ["meses_cliente", "uso_mensual", "quejas_30d", "importe_medio"]
categoricas = ["plan", "region"]

prep = ColumnTransformer([
    ("num", StandardScaler(), numericas),
    ("cat", OneHotEncoder(handle_unknown="ignore"), categoricas),

```

```
] )

modelo = Pipeline([("prep", prep),
                   ("clf", HistGradientBoostingClassifier(max_depth=4))])
modelo.fit(X_train, y_train)

prob = modelo.predict_proba(X_test)[: , 1]
print("AUC:", round(roc_auc_score(y_test, prob), 3))
print(classification_report(y_test, prob >= 0.3)) # umbral elegido por negocio
```

Observación. El umbral de 0,3 del ejemplo no es arbitrario: se elige comparando el coste de una falsa alarma con el de una baja no detectada.

13. ÉTICA Y MARCO LEGAL

- **Sesgos:** un modelo aprende de datos históricos y puede reproducir discriminaciones pasadas. Revisa los resultados por grupos (sexo, edad, origen) antes de usarlo para decidir sobre personas.
- **RGPD:** el artículo 22 reconoce el derecho a no ser objeto de decisiones basadas únicamente en tratamientos automatizados con efectos jurídicos o similares, salvo excepciones; en esos casos hay que permitir la intervención humana y explicar la lógica aplicada.
- **Reglamento europeo de IA:** clasifica como de alto riesgo, con obligaciones reforzadas de documentación, supervisión humana y calidad de datos, sistemas como los que evalúan la solvencia de personas o seleccionan candidatos a un empleo.
- **Transparencia:** documenta con qué datos se entrenó el modelo, para qué sirve, sus limitaciones y cómo se ha validado.

14. ERRORES FRECUENTES

- Empezar por el algoritmo en lugar de por la decisión de negocio.
- No definir una línea base con la que comparar.
- Fuga de datos: usar información del futuro sin darse cuenta.
- Evaluar con los mismos datos de entrenamiento, o mezclar al azar una serie temporal.
- Fiarse de la exactitud con clases desequilibradas.
- Optimizar una métrica técnica que no se traduce en euros, horas o clientes.
- Desplegar el modelo y olvidarse de él.

15. LISTA DE COMPROBACIÓN

1. ¿Está claro qué decisión se tomará con la predicción y cuánto cuesta cada tipo de error?
2. ¿Hay datos históricos suficientes y fiables de la variable objetivo?
3. ¿Todas las variables estarán disponibles en el momento de predecir?

4. ¿He separado entrenamiento, validación y prueba (respetando el tiempo si es una serie)?
5. ¿El modelo mejora claramente una línea base sencilla?
6. ¿Las métricas elegidas reflejan el objetivo del negocio?
7. ¿Puedo explicar las predicciones más importantes?
8. ¿He revisado posibles sesgos y obligaciones legales?
9. ¿Hay un plan para desplegar, monitorizar y reentrenar el modelo?

16. GLOSARIO

AUC. Área bajo la curva ROC; mide la capacidad de ordenar casos (0,5 azar, 1 perfecto).

Backtesting. Evaluar un modelo de series simulando predicciones en el pasado.

Clasificación. Predicción de una categoría.

Deriva de datos. Cambio en los datos de entrada respecto a los de entrenamiento.

Fuga de datos. Uso en el entrenamiento de información no disponible al predecir.

Gradient boosting. Conjunto de árboles en el que cada uno corrige los errores del anterior.

Hiperparámetro. Configuración del algoritmo que se fija antes de entrenar.

Ingeniería de variables. Creación de variables predictoras a partir de los datos brutos.

Línea base. Predicción sencilla de referencia que el modelo debe superar.

LightGBM / XGBoost. Librerías de gradient boosting muy usadas con datos tabulares.

MAE / RMSE. Errores medios de una regresión, absoluto y cuadrático.

Matriz de confusión. Tabla que cruza clases reales y predichas.

MLOps. Prácticas para desplegar, versionar y vigilar modelos en producción.

pandas. Librería de Python para manipular tablas de datos.

Precisión / exhaustividad. Acierto de las alertas / proporción de casos reales detectados.

Random forest. Conjunto de árboles entrenados en paralelo y promediados.

Regresión. Predicción de un valor numérico.

Regularización. Penalización de la complejidad para evitar el sobreajuste.

SHAP. Método que reparte una predicción entre las variables que la explican.

scikit-learn. Librería principal de aprendizaje automático en Python.

Sobreajuste. Modelo que memoriza el entrenamiento y generaliza mal.

Umbral. Probabilidad a partir de la cual se toma la decisión.

Validación cruzada. Evaluación repetida con distintas particiones de los datos.