Business Analytics & Data Science

Día 8: Clasificación y Cierre del Curso

EAE Business School Barcelona

11 de febrero de 2026

Plan del Día 8 (Último Día!)

Primera Parte (9:00-11:00)

1. De regresión a clasificación
2. Regresión logística y función sigmoide
3. Métricas de clasificación

Segunda Parte (11:30-13:30)

1. Ejemplo completo: predicción de cancelaciones
2. Síntesis del curso
3. Recursos para continuar aprendiendo

Primera Parte: De Regresión a Clasificación

Regresión vs Clasificación

Regresión (Días 6-7):

- Predecir valor numérico continuo
- Ejemplos: precio, temperatura, ventas
- Output: número (ej: 325000€)

Clasificación (Hoy):

- Predecir categoría discreta
- Ejemplos: spam/no spam, aprobado/suspendido, cancelará/no cancelará
- Output: clase (ej: "Cancelará")

Clasificación Binaria

Clasificación binaria = 2 clases posibles

Ejemplos:

- Email: spam (1) o no spam (0)
- Transacción: fraude (1) o legítima (0)
- Reserva hotel: cancelada (1) o no cancelada (0)
- Cliente: se irá (1) o se quedará (0)

Codificación típica: Clase positiva = 1, Clase negativa = 0

Regresión logística

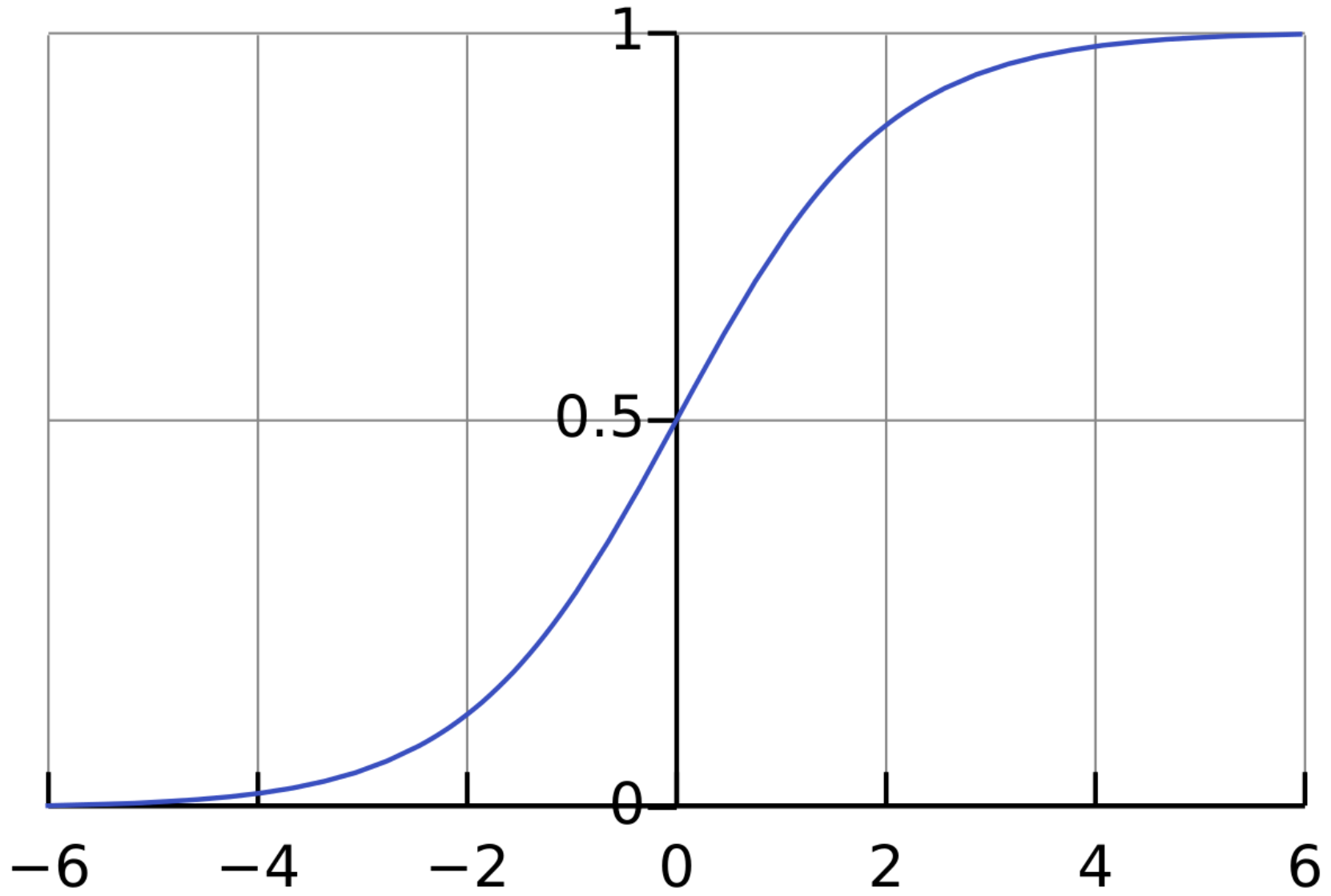
Nos gustaría aplicar una **regresión lineal**, pero nos daría valores en $(-\infty, \infty)$

Para clasificación necesitamos **probabilidades** en el intervalo (0, 1)

Solución:

- Hacemos primero una regresión lineal
- Después, mapeamos el intervalo $(-\infty, \infty)$ al intervalo $(0, 1)$
- Una vez obtenidas probabilidades, aplicamos un **umbral de decisión** para predecir la clase

Por lo tanto, nos hace falta una función que mapee la recta al intervalo $(0, 1)$


1
0.5
0
−6
−4
−2
0
2
4
6

Función sigmoide

También llamada función *logística*:

$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

Definición de la regresión logística

$$z = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \dots$$

$$p = \sigma(z)$$

Consideraciones prácticas

Para entrenar una regresión logística las consideraciones a tener en cuenta son las mismas que con una regresión lineal:

- Usar one-hot encoding para variables categóricas
- El código es muy parecido, con algunas sutilezas extra

Predecir Probabilidades

```python
from sklearn.linear_model import LogisticRegression

model = LogisticRegression()
model.fit(X_train, y_train)

# Probabilidades
probas = model.predict_proba(X_test)  # (n_samples, 2)
proba_positiva = probas[:, 1]  # Probabilidad de clase 1

print(f"Probabilidad de cancelación: {proba_positiva[0]:.2f}")
```

Umbral de Decisión

Problema: ¿Cómo convertir probabilidad → clase?

Solución: Umbral (por defecto = 0.5)

```
# Predicción de clase (umbral 0.5)
predicciones = model.predict(X_test)

# Equivalente a:
predicciones = (proba_positiva >= 0.5).astype(int)
```

El umbral debe ser ajustado según el contexto:

- Detectar fraude: umbral bajo (0.3) → más sensible
- Spam: umbral alto (0.7) → más conservador

Esto tiene que ver con los errores de tipo I y II; el umbral de decisión permite hacer un trade-off entre ambos.

Matriz de Confusión

Matriz 2×2 que muestra todas las combinaciones:

	Predicho: 0	**Predicho: 1**
Real: 0	TN (✓)	FP (❌)
Real: 1	FN (❌)	TP (✓)

- **TP** (True Positive): Correctamente predicho como 1
- **TN** (True Negative): Correctamente predicho como 0
- **FP** (False Positive): Error tipo I
- **FN** (False Negative): Error tipo II

Métricas de Clasificación

Accuracy:

$$\frac{TP + TN}{Total}$$

- "¿Qué porcentaje acertamos?"
- Problemática si clases desbalanceadas

Precision:

$$\frac{TP}{TP + FP}$$

- "De los que predijimos como positivos, ¿cuántos lo son?"
- Importante cuando FP es costoso

Recall (Sensibilidad):

$$\frac{TP}{TP + FN}$$

- "De los positivos reales, ¿cuántos detectamos?"
- Importante cuando FN es costoso

F1-Score

$$F1 = 2 \times (Prec \times Rec) / (Prec + Rec)$$

- Es la media armónica entre precision y recall
- Es mucho más robusta que accuracy para problemas con clases no balanceadas

Trade-off Precision vs Recall

Precision alta → Pocas falsas alarmas, pero podemos perder positivos
Recall alto → Detectamos todos los positivos, pero más falsas alarmas

Ejemplo: Detector de cáncer

- Recall alto (no perdernos ningún caso) → más FP (más pruebas innecesarias)
- Precision alta (menos falsas alarmas) → más FN (casos perdidos)

Calcular Métricas en Python

```
from sklearn.metrics import (
  confusion_matrix, accuracy_score, precision_score, recall_score, f1_score
)

# Matriz de confusión
cm = confusion_matrix(y_test, y_pred)
print(cm)
```

```python
# Métricas
print(f"Accuracy: {accuracy_score(y_test, y_pred):.3f}")
print(f"Precision: {precision_score(y_test, y_pred):.3f}")
print(f"Recall: {recall_score(y_test, y_pred):.3f}")
print(f"F1: {f1_score(y_test, y_pred):.3f}")
```

Visualizar Matriz de Confusión

```
import plotly.express as px

cm = confusion_matrix(y_test, y_pred)

fig = px.imshow(cm,
                text_auto=True,
                labels=dict(x="Predicho", y="Real"),
                x=['No Cancela', 'Cancela'],
                y=['No Cancela', 'Cancela'],
                title='Matriz de Confusión')
fig.show()
```

🔬 Ahora al notebook

Ejercicio Regresión Logística (45 minutos)

Segunda Parte: Ejemplo Completo

Caso: Predicción de Cancelaciones

Contexto de negocio: Hotel quiere predecir qué reservas se cancelarán

¿Por qué es importante?

- Overbooking estratégico
- Enviar recordatorios a clientes con alto riesgo
- Optimizar revenue management

Datos: hotel_bookings.csv
Target: `is_canceled` (0 o 1)
Features: lead_time, previous_cancellations, deposit_type, etc.

Workflow Completo

1. **Explorar datos** (EDA)
2. **Limpiar y preparar** (missing values, encoding)
3. **Train/test split**
4. **Feature engineering** (si necesario)
5. **Entrenar modelo** (LogisticRegression)
6. **Evaluar** (accuracy, precision, recall, matriz de confusión)
7. **Interpretar coeficientes** (qué features importan)
8. **Ajustar umbral** (según costos de negocio)
9. **Comparar con baseline** (predecir siempre la clase mayoritaria)

Síntesis de lo visto hasta ahora

El Viaje Completo

Fundamentos

- Python: variables, listas, funciones
- pandas: DataFrames, filtrado, inspección
- Limpieza: missing values, duplicados
- Visualización: Plotly Express

Análisis

- Estadística descriptiva: mean, std, percentiles
- Probabilidad: distribución normal
- Inferencia: intervalos de confianza, t-tests
- Decisiones basadas en datos

El Viaje Completo (cont.)

Machine Learning

- Workflow ML: datos → features → modelo → evaluación
- Regresión lineal: simple y múltiple
- Métricas: MAE, RMSE, R^2
- Feature engineering, one-hot encoding
- Clasificación: regresión logística
- Métricas: accuracy, precision, recall

Habilidades Adquiridas

- ✅ Manipular y limpiar datos con pandas
- ✅ Crear visualizaciones interactivas
- ✅ Calcular estadísticas e interpretar resultados
- ✅ Realizar tests de hipótesis
- ✅ Construir modelos de ML (regresión y clasificación)
- ✅ Evaluar modelos con métricas apropiadas
- ✅ Conectar análisis técnico con preguntas de negocio

Estas habilidades son la base del análisis de datos moderno

Lo Que NO Hemos Cubierto (y está bien!)

Este curso es una **introducción**; hemos dado algunas pinceladas superficiales sin ahondar demasiado.
Por ejemplo, no hemos cubierto:

- Modelos avanzados (Random Forest, XGBoost, redes neuronales)
- Deep learning
- Series temporales
- Procesamiento de lenguaje natural (NLP)
- Validación cruzada, Grid search
- Feature selection avanzada
- Deployment de modelos

¡Hay mucho más por aprender! Esto es solo el principio

Recursos para Continuar Aprendiendo

Cursos online:

- Kaggle Learn (gratis, práctico)
- Fast.ai (deep learning práctico)
- Coursera: Machine Learning (Andrew Ng)

Libros:

- "Python for Data Analysis" - Wes McKinney
- "Hands-On Machine Learning" - Aurélien Géron
- "Introduction to Statistical Learning" (gratis online)

Consejos Finales

1. **Practicar es clave**: Leer no basta, programad
2. **Empezar simple**: No necesitáis redes neuronales para todo
3. **Entender los datos primero**: EDA antes de modelar
4. **Interpretar resultados**: Números sin contexto no sirven
5. **Ser críticos**: No todo problema necesita ML
6. **Comunidad**: Uníos a meetups, conferencias (PyData, Python Barcelona)
7. **Seguid curiosos**: La tecnología evoluciona constantemente

¡Gracias!

Ha sido un placer enseñaros

- Habéis hecho un gran progreso en 3 semanas
- Recordad: el aprendizaje es un viaje continuo
- No dudéis en seguir explorando y practicando

Conectemos: LinkedIn, GitHub, Python Barcelona meetups