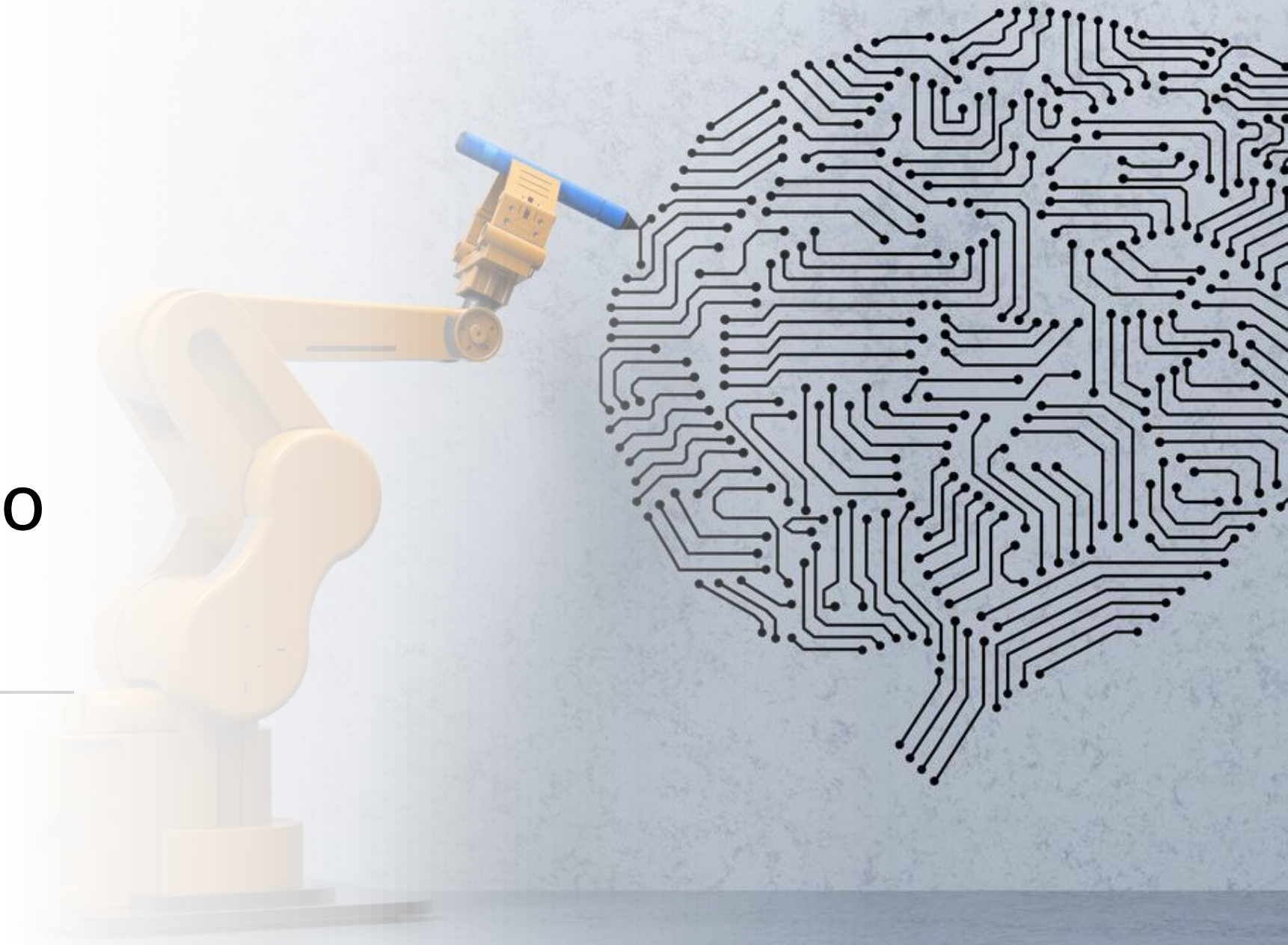
Reconocimiento de Patrones

Clase 7

Evaluación de modelos

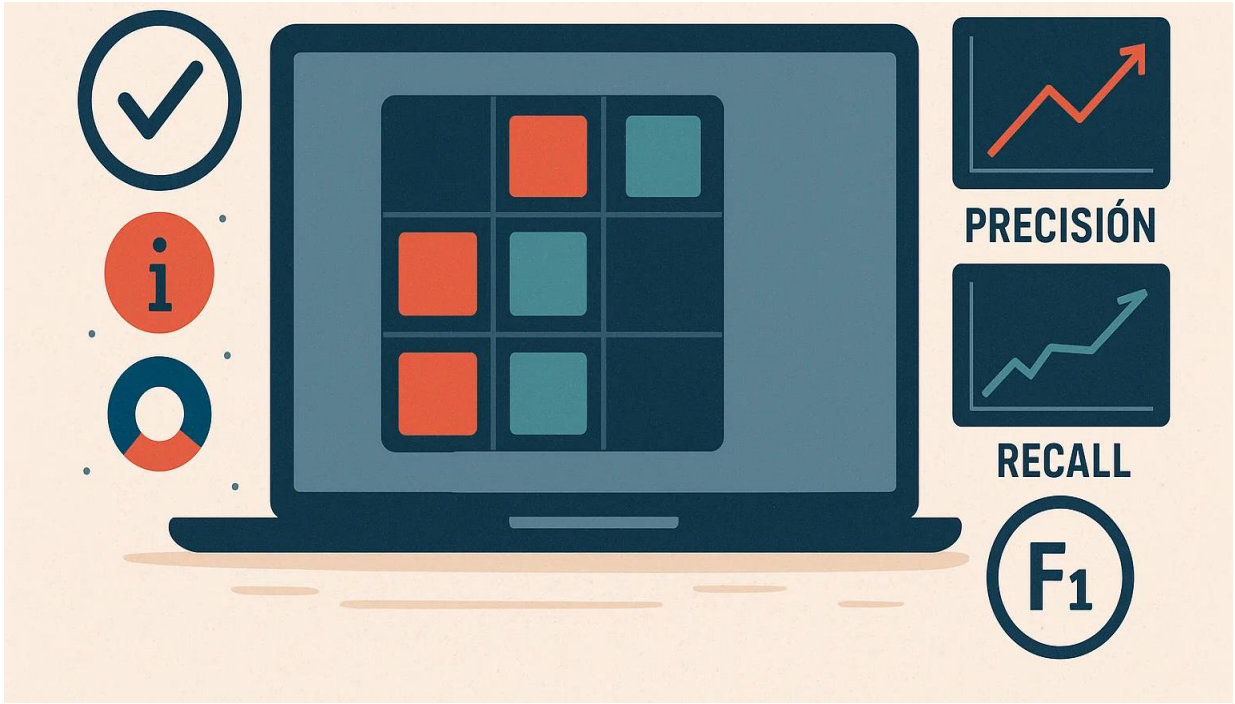


Antes de empezar

- ¿Dudas de la clase anterior?

¿Cómo sabemos si un clasificador es realmente "bueno"?

- Entrenar un modelo no significa que funcione bien.
- Necesitamos medir:
  - Qué tan correcto es.
  - Qué tan confiable es.
  - Qué tan bien generaliza.
- Construir el modelo es solo el 50% del trabajo; el resto es evaluarlo correctamente.



- **Problema:** Detección de cáncer de mama. 950 pacientes sanas, 50 con cáncer.

- **Modelo A:** Siempre predice "sano". Accuracy = 950/1000 = **95%** ✓
  Resultado real: No detecta ni un solo caso de cáncer. 50 personas mueren. ✗

- Necesitamos métricas que capturen lo que realmente importa.

Matriz de confusión y metricas fundamentales

- Una matriz de confusión resume las predicciones de un clasificador.

	Pred. Positivo	Pred. Negativo
Real Positivo	TP Verdadero Positivo	FN Falso Negativo
Real Negativo	FP Falso Positivo	TN Verdadero Negativo

Pensemos en un sistema de alarma contra incendios:
TP: Hay fuego y la alarma suena. (Excelente).
TN: No hay fuego y la alarma está callada. (Excelente).
FP (Error Tipo I): No hay fuego, pero la alarma suena. Nos despertó a todos por nada. Es una *falsa alarma*.
FN (Error Tipo II): ¡Hay fuego, pero la alarma NO suena! Este error es catastrófico.

- **Accuracy (Exactitud)**

Es la proporción de predicciones correctas sobre el total de casos.

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}$$

- **Precisión (Valor Predictivo Positivo)**

De todo lo que el modelo **dijo que era positivo**, ¿qué porcentaje realmente lo era?

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

- **Recall / Sensitivity (Sensibilidad / Exhaustividad)**

De todo lo que **realmente era positivo** en la realidad, ¿cuánto logró capturar o "pescar" el modelo?

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

- **Specificity (Especificidad)**

De todo lo que **realmente era negativo**, ¿cuánto logró identificar el modelo como tal?

$$\text{Specificity} = \frac{\text{TN}}{\text{TN} + \text{FP}}$$

- **F1-Score**

Es la media armónica entre la *Precision* y el *Recall*. Penaliza severamente los valores extremos. Si la *Precision* es 1.0 y el *Recall* es 0.0, la media aritmética sería 0.5, pero el F1-Score será 0.0. Exige que ambas métricas sean buenas.

$$\text{F1-Score} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} = \frac{2\text{TP}}{2\text{TP} + \text{FP} + \text{FN}}$$

- **Balanced Accuracy (Exactitud Balanceada)**

Calcula el promedio del *Recall* de cada una de las clases.

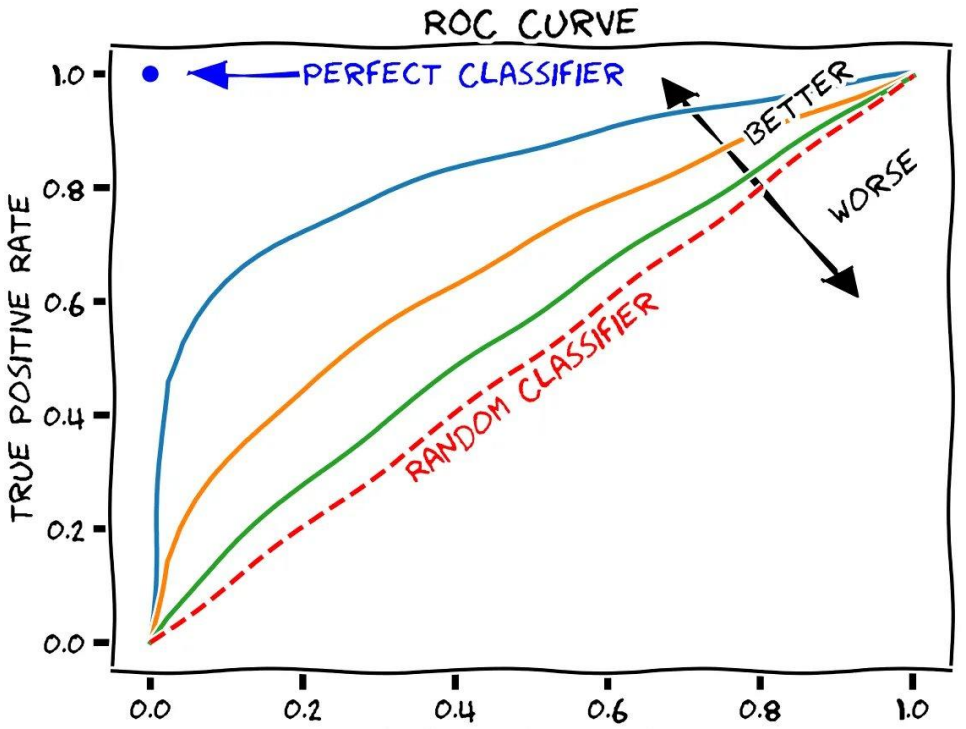
$$\text{Balanced Accuracy} = \frac{\text{Sensitivity} + \text{Specificity}}{2}$$

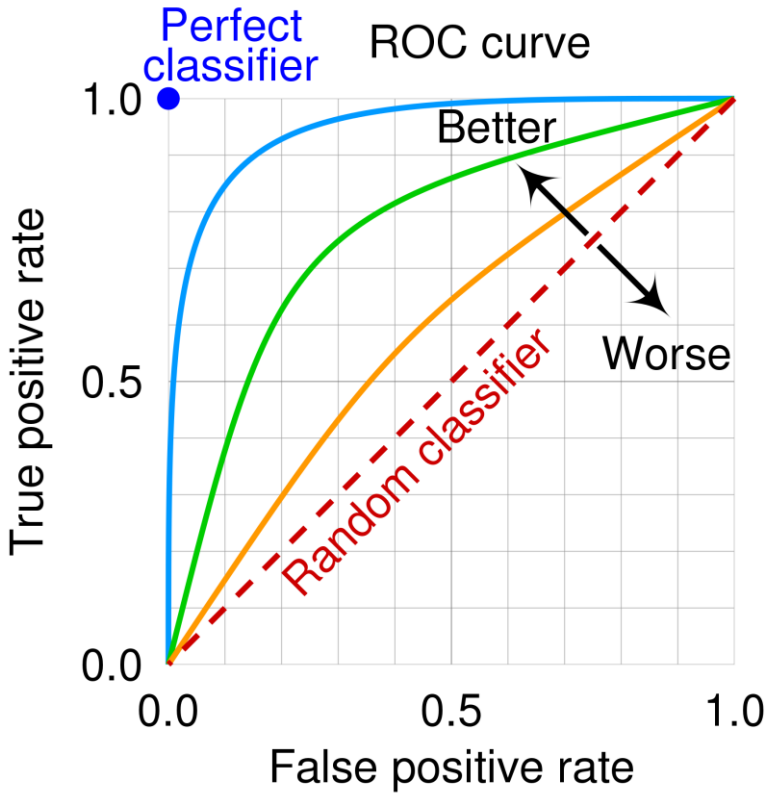
	Pred. Spam	Pred. No spam
Real Spam	30	5
Real No spam	5	60

Accuracy	Precision	Recall	F1-score
90.0%	85.7%	85.7%	85.7%

Curvas ROC (Receiver Operating Characteristic) y AUC

- Los clasificadores probabilísticos devuelven un valor continuo entre 0.0 y 1.0.
- Por defecto usamos un umbral (threshold) de $\theta = 0.5$. ¿Qué pasa si lo movemos?
- La idea es evaluar el rendimiento a través de todos los thresholds posibles.





- Eje X: FPR = FP/(FP+TN) = 1 - Specificity
- Eje Y: TPR = TP/(TP+FN) = Recall / Sensitivity

Cada punto es un threshold diferente (de 1.0 a 0.0)

Area Under the Curve (AUC)

- AUC = 1.0: Clasificador perfecto
- AUC = 0.9–1.0: Excelente
- AUC = 0.7–0.9: Bueno
- AUC = 0.5–0.7: Regular
- AUC = 0.5: Aleatorio (diagonal)
- AUC < 0.5: Peor que azar (invertir predicción)

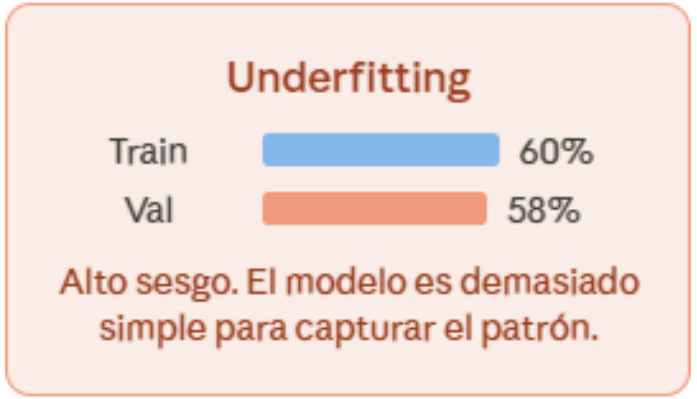
Overfitting, underfitting y el trade-off bias-variance

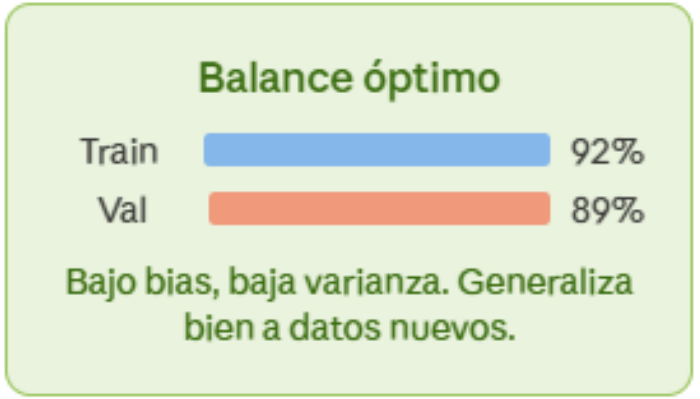
Bias (Sesgo)

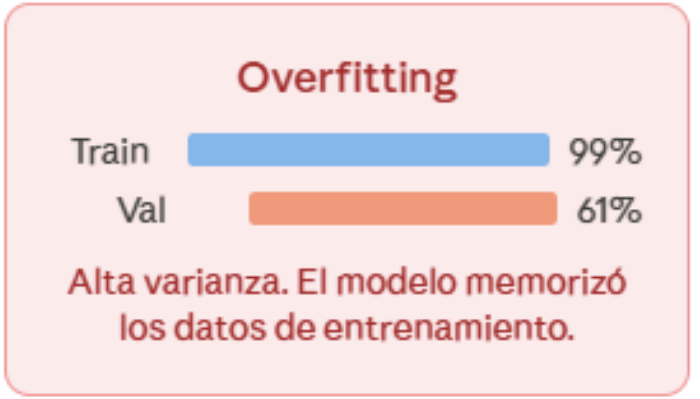
- **Definición:** Error sistemático del modelo por suposiciones incorrectas o simplificación excesiva.
- **Analogía:** Un arquero que siempre tira hacia la izquierda, independientemente de la diana. Consistente pero inexacto.
- **Síntoma:** Alto error tanto en train como en validation.

Variance (Varianza)

- **Definición:** Sensibilidad excesiva a fluctuaciones en los datos de entrenamiento. El modelo aprende el ruido.
- **Analogía:** Un arquero que dispara en todas direcciones según el viento. Impredecible con datos nuevos.
- **Síntoma:** Bajo error en train, alto error en validation.







Al evaluar el error:

Si train ≈ val (ambos altos): Underfitting / High bias
Solución: modelo más complejo, más features, menos regularización

Si train ≈ val (ambos bajos): Balance óptimo
Seguir iterando para mejorar
Verificar que train set sea representativo

Si train » val (train bajo) : Overfitting / High variance
Solución: más datos, regularización, dropout, early stopping

Ejemplo: Imaginen que están estudiando para su examen de calculo

Caso 1 (Underfitting): Solo se memorizan una fórmula para todo. Llegan al examen y reprueban porque la fórmula no aplica a la mayoría de los problemas.

El modelo no tuvo la capacidad de aprender la complejidad subyacente (Alto Sesgo).

Caso 2 (Overfitting): Se consiguen los exámenes de los últimos 5 años y se memorizan las respuestas exactas con todo y puntos y comas. Llegan a clase, les cambian un número en el examen, y colapsan.

No aprendieron a resolver el problema, memorizaron los datos de entrenamiento (Alta Varianza).

Detección de fraude

0.1%

de transacciones son fraudulentas

Diagnóstico de cáncer

1–5%

de pacientes tienen la enfermedad

Intrusiones de red

0.5%

del tráfico es malicioso

Datasets desbalanceados

Estrategias de manejo

- Elegir metricas adecuadas
- Ajustar pesos de clases (class_weight)
- Técnicas de resampling
- Ajustar el threshold de decisión

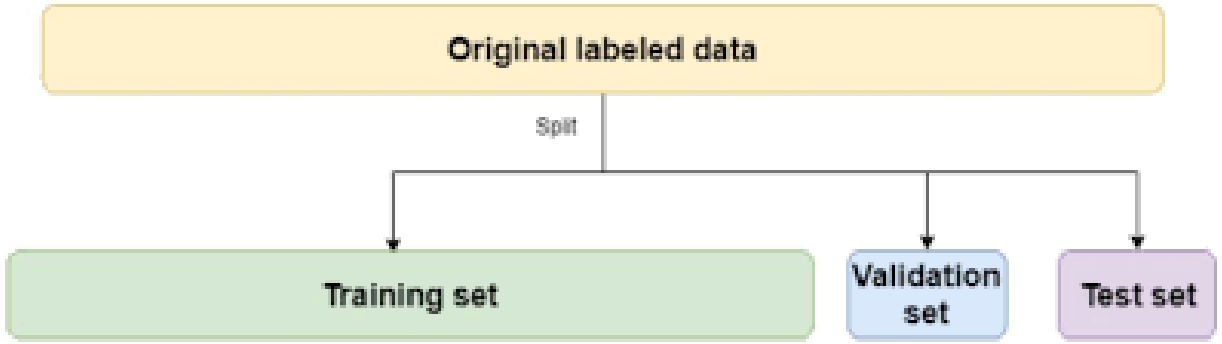
Protocolos de Validación y Data Leakage

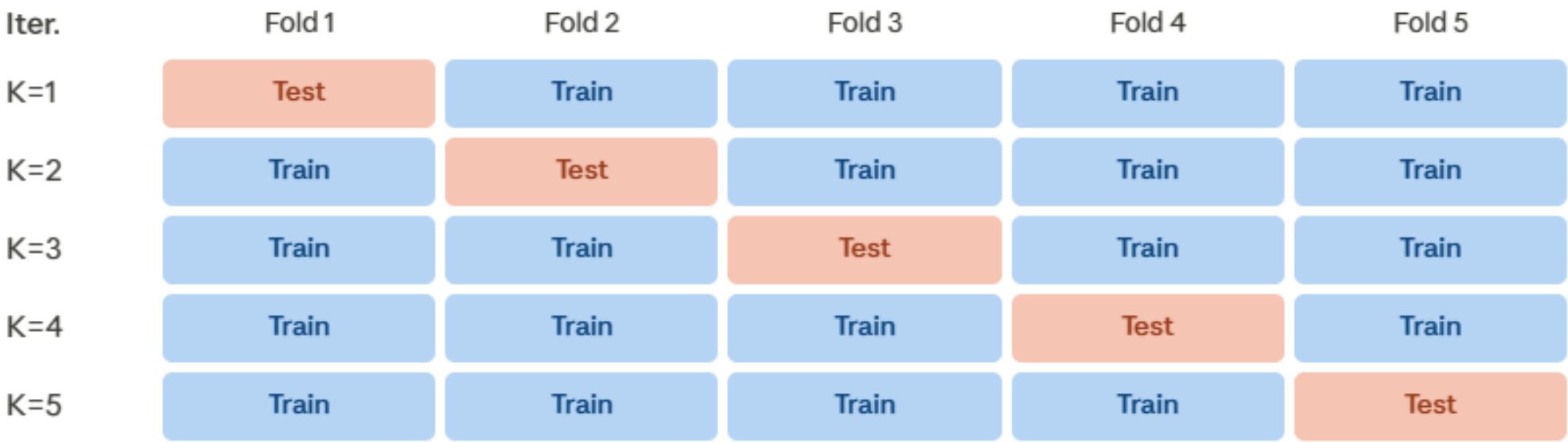
Para evaluar un modelo, jamás debemos usar los mismos datos con los que entrenó. Sería como darle a un alumno el mismo examen que resolvieron juntos en clase. Para solventar este problema tenemos distintas estrategias de validación de modelos.

- Hold-out
- K-fold cross validation
- Stratified K-fold

- **Hold-out**

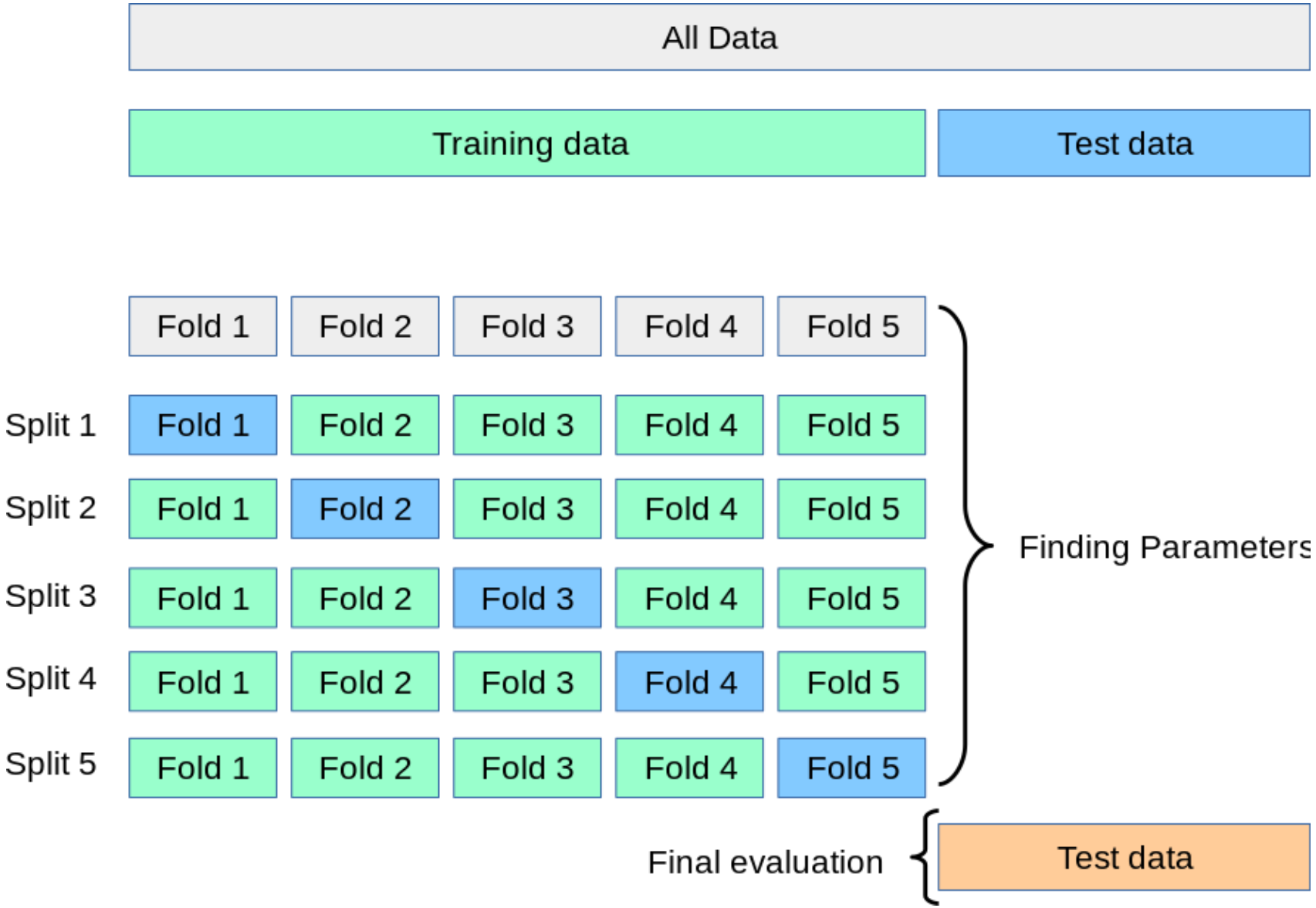
Dividir una sola vez el dataset, típicamente en tres conjuntos entrenamiento/validación/prueba. Pero puede ser muy sensible a cómo se hace la partición





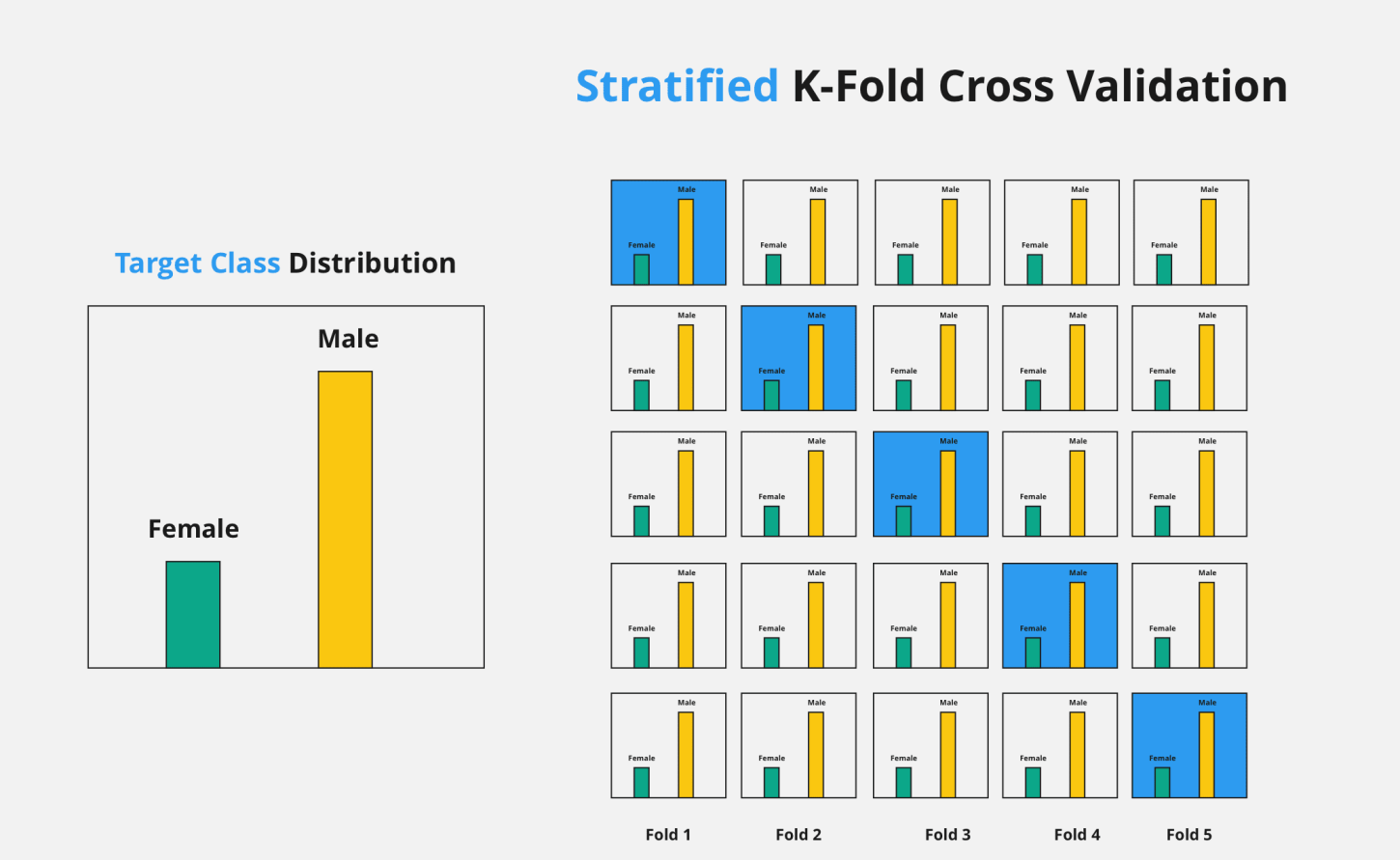
- **K-fold cross validation**

Dividir en K particiones iguales. Entrenar K veces, cada vez dejando una partición diferente como validación. Cada dato se usa tanto para entrenamiento como para validación, haciendo estimaciones más robustas, pero es K veces más costoso computacionalmente


All Data
Training data
Test data
Fold 1
Fold 2
Fold 3
Fold 4
Fold 5
Split 1
Split 2
Split 3
Split 4
Split 5
Finding Parameters
Final evaluation
Test data

- **Stratified K-Fold**

Crucial para datos desbalanceados. Asegura que cada uno de los K bloques contenga exactamente la misma proporción de clases que el dataset original



El pecado capital: Data Leakage (Fuga de Datos)

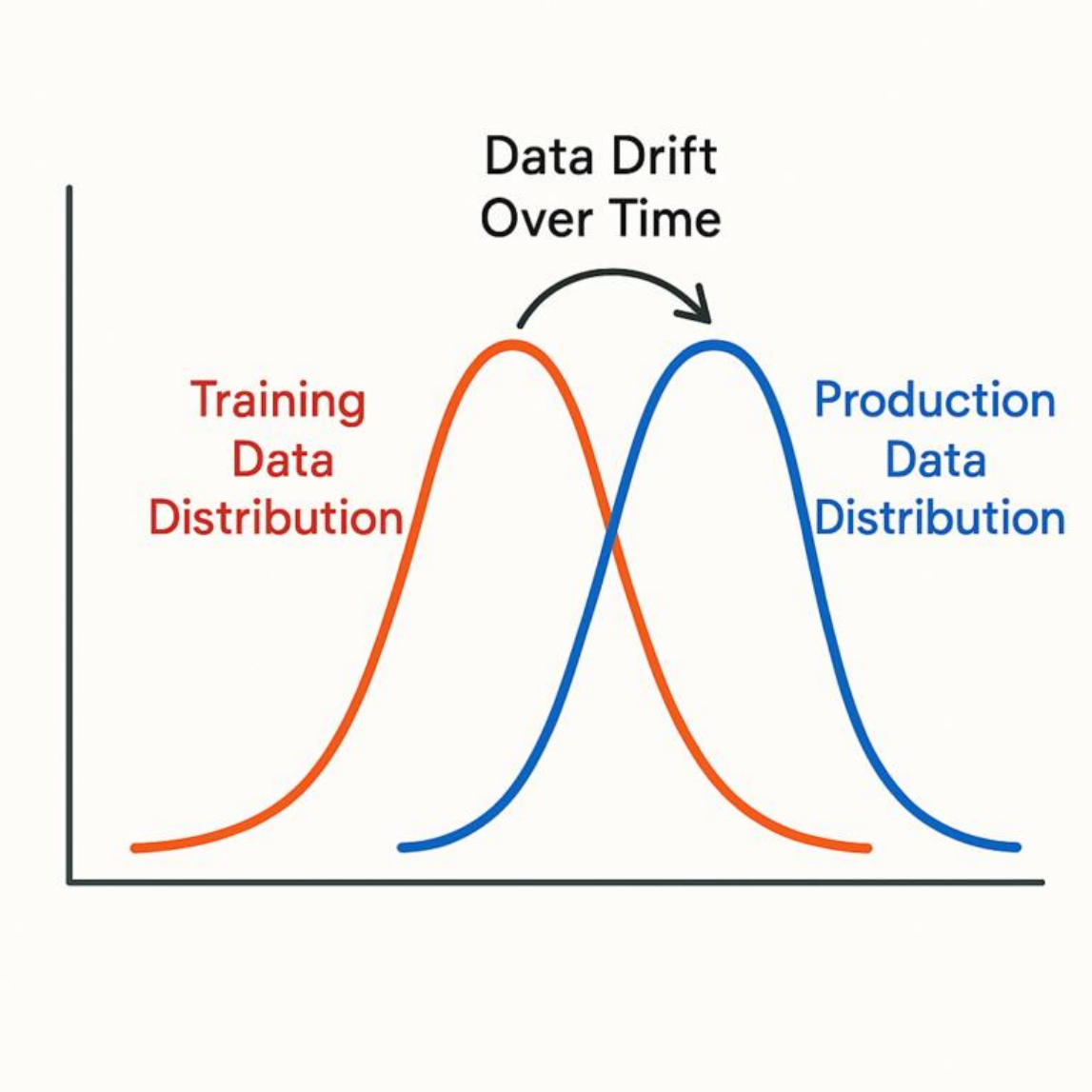
Cuando información del conjunto de validación/test "contamina" el entrenamiento, generando métricas artificialmente optimistas que no se reproducen en producción.

Tipos comunes:

- Leakage temporal: Usar datos futuros para predecir el pasado.
- Leakage de preprocesamiento: Aplicar normalización/PCA con estadísticas del test set incluidas.
- Leakage de target: Incluir como feature una variable derivada del target.

Data Drift

- Es un problema que ocurre cuando las características estadísticas de los datos que el modelo ve en producción cambian con respecto a los datos con los que fue entrenado. En esencia, el mundo cambia, pero el modelo no.
- El modelo no falla de golpe — se degrada silenciosamente

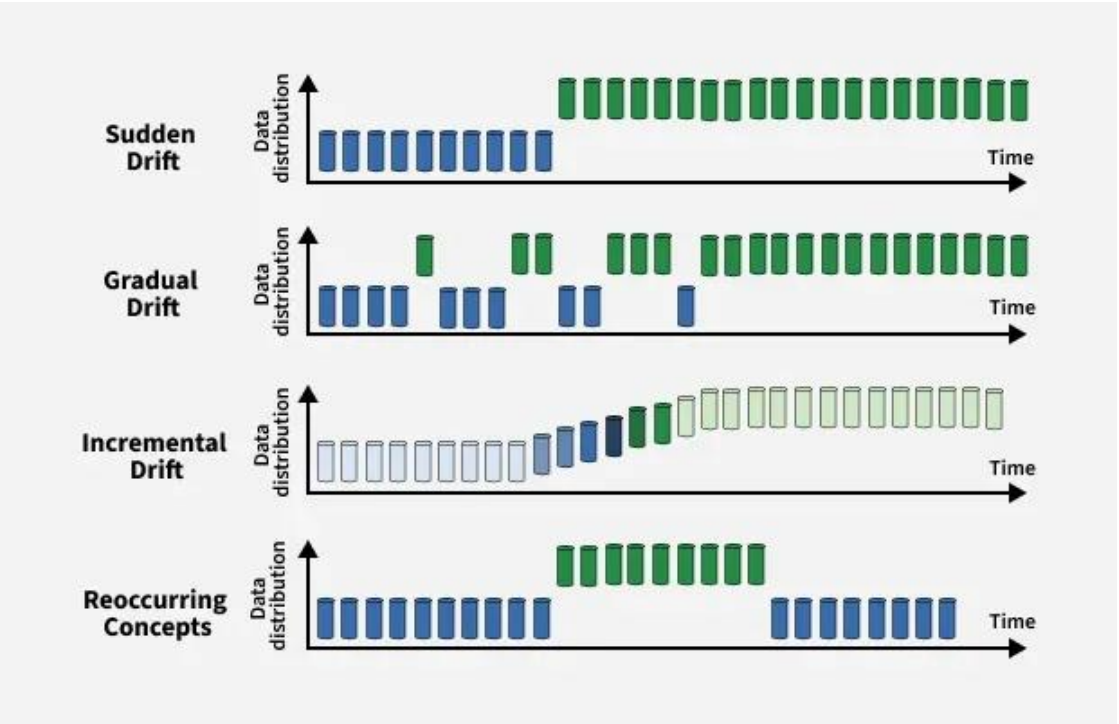


- **Cómo detectarlo**

La detección se basa en monitorear estadísticas de los datos en producción y compararlas contra una distribución de referencia (normalmente los datos de entrenamiento).

- **Cómo solucionarlo**

Se actualiza el modelo periódicamente alimentándolo con datos recientes para que recupere su capacidad de predicción.



Conclusiones finales

- La evaluación correcta es esencial, no existe una métrica universal.
- Validar correctamente evita falsas conclusiones.
- El objetivo real es la **generalización**.
- La evaluación de modelos es mucho más que calcular accuracy. Implica: comprender errores, costos y el contexto.
- No solo es entrenar modelos, sino saber cuando confiar en ellos.
- La próxima vez que entrenen un modelo, la primera pregunta es "¿qué tipo de errores comete, con qué frecuencia, y en qué contexto eso importa?"