Curso base

INTELIGENCIA ARTIFICIAL DESDE CERO

LECCIÓN 20
Sobreajuste y mejoras simples

Cómo reconocer cuándo el modelo memoriza los datos y qué ajustes sencillos pueden mejorar su desempeño real

Lección	20	**Tema**	Sobreajuste y mejoras simples
Nombre del participante	_____	**Fecha**	_____

Actividad central

Actividad principal: comparar desempeño aparente y desempeño real, identificar una posible señal de sobreajuste y proponer o probar una mejora simple.
Producto esperado: análisis comparado de entrenamiento y prueba con evidencia sencilla de sobreajuste y una explicación breve sobre qué ajuste ayudaría a buscar un modelo más confiable.

Propósito de la guía. Comprender qué es el sobreajuste, cómo detectarlo al comparar entrenamiento y prueba, y qué mejoras simples pueden ayudar a construir modelos más confiables. La meta es que el estudiante no se deje impresionar solo por un puntaje alto en entrenamiento, sino que aprenda a mirar el desempeño real con datos nuevos.

Idea central

Un modelo puede verse excelente durante el entrenamiento y, aun así, fallar cuando se enfrenta a datos nuevos. Ese problema se llama **sobreajuste**. En vez de aprender patrones generales, el modelo termina recordando detalles y ruido del conjunto de entrenamiento.

En esta lección no buscamos técnicas avanzadas. Buscamos desarrollar criterio. Si un modelo rinde muy bien en train pero claramente peor en test, debemos sospechar que no está generalizando bien. Luego podemos intentar mejoras sencillas, como limitar la complejidad del modelo o eliminar ruido innecesario.

¿Qué vamos a hacer en esta guía?

- Reconocer de manera intuitiva qué significa sobreajustar.
- Comparar entrenamiento y prueba para detectar una señal de alerta.
- Trabajar un primer ejemplo guiado de lectura e interpretación.
- Desarrollar un segundo ejemplo guiado con código sencillo.
- Usar prompts de IA generativa para comprender mejor los resultados.

Vocabulario mínimo para comenzar

Concepto	Interpretación sencilla
Sobreajuste	Ocurre cuando el modelo aprende demasiado pegado a los datos de entrenamiento y luego falla con casos nuevos.
Generalización	Capacidad del modelo para mantener un desempeño razonable con datos que no vio antes.
Ruido	Información poco útil o accidental que puede confundir al modelo.
Complejidad	Nivel de detalle o flexibilidad que tiene el modelo para ajustarse a los datos.
Mejora simple	Ajuste básico que busca un comportamiento más equilibrado entre train y test.

Ejemplo guiado 1: leer una señal de sobreajuste

Comencemos con una situación muy simple. Supongamos que un clasificador intenta predecir si un cliente comprará o no un producto. Al final del proceso obtenemos estas dos medidas:

Medida observada	Valor	Lectura breve
Accuracy en entrenamiento	0.98	Parece excelente.
Accuracy en prueba	0.74	Ya no se ve tan fuerte con datos nuevos.
Diferencia train - test	0.24	La brecha es grande y merece atención.

Lectura paso a paso

1. **Observe la brecha entre entrenamiento y prueba.** Si el modelo casi no se equivoca en train, pero baja claramente en test, no debemos celebrar todavía.
2. **Pregúntese qué pudo pasar.** A veces el modelo es demasiado complejo para la cantidad de datos disponible. Otras veces está aprendiendo detalles accidentales, no patrones estables.
3. **Formule una conclusión prudente.** Con estos resultados no basta decir: *el modelo es muy bueno*. Lo correcto es decir: *el modelo se ve muy fuerte en entrenamiento, pero su desempeño en prueba sugiere posible sobreajuste.*

4. **Piense en una mejora simple.** Por ejemplo: reducir la complejidad del modelo, revisar variables irrelevantes o comparar varias configuraciones pequeñas.

Prompts de IA generativa para acompañar el ejemplo 1

- Explícame con lenguaje para principiantes por qué un accuracy muy alto en entrenamiento no garantiza un buen modelo.
- Tengo accuracy 0.98 en train y 0.74 en test. Ayúdame a redactar una interpretación corta y prudente.
- Dime tres causas sencillas por las que un modelo puede memorizar datos en vez de generalizar.
- Proponme dos mejoras simples para un modelo que rinde muy bien en train pero empeora en test.

Ejemplo guiado 2: comparar un modelo muy flexible con uno más controlado

Ahora haremos un ejemplo práctico. Usaremos un conjunto de práctica incorporado en `scikit-learn` y compararemos dos árboles de decisión: uno sin límite de profundidad y otro más simple. La idea no es memorizar el código, sino comprender la lógica del experimento.

En una corrida típica, el modelo sin límite puede llegar a accuracy de entrenamiento igual a 1.00 y bajar en prueba; en cambio, una versión con profundidad limitada suele perder un poco de rendimiento en train, pero mejorar o estabilizar test. Esa es justamente la clase de equilibrio que buscamos.

Código base del experimento

```
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score

datos = load_iris()
X = datos.data
y = datos.target

X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.25, random_state=42, stratify=y
)

modelo_profundo = DecisionTreeClassifier(max_depth=None, random_state=42)
modelo_simple = DecisionTreeClassifier(max_depth=2, random_state=42)

modelo_profundo.fit(X_train, y_train)
modelo_simple.fit(X_train, y_train)
```

```
acc_train_profundo = accuracy_score(y_train, modelo_profundo.predict(
    X_train))
acc_test_profundo = accuracy_score(y_test, modelo_profundo.predict(X_test))

acc_train_simple = accuracy_score(y_train, modelo_simple.predict(X_train))
acc_test_simple = accuracy_score(y_test, modelo_simple.predict(X_test))

print("Modelo profundo - train:", round(acc_train_profundo, 2))
print("Modelo profundo - test:", round(acc_test_profundo, 2))
print("Modelo simple - train:", round(acc_train_simple, 2))
print("Modelo simple - test:", round(acc_test_simple, 2))
```

Lectura guiada del experimento

1. Ejecute el código y anote las cuatro medidas: train y test del modelo profundo, y train y test del modelo simple.
2. Compare la brecha interna de cada modelo. Si uno de ellos tiene train demasiado alto y test claramente menor, ese modelo está dando una señal de sobreajuste.
3. Decida cuál versión inspira más confianza. En aprendizaje automático para principiantes, suele ser mejor un modelo un poco menos brillante en entrenamiento, pero más estable en prueba.
4. Redacte la idea central con palabras sencillas: *lo importante no es impresionar con train, sino responder bien en datos nuevos.*

Modelo	Qué suele pasar	Lectura breve
Árbol muy profundo	Accuracy muy alto en train y menor en test	Puede memorizar detalles del conjunto de entrenamiento.
Árbol más simple	Train un poco menor pero test más equilibrado	Suele generalizar mejor o al menos ser más estable.

Prompts de IA generativa para acompañar el ejemplo 2

- Explícame como si fuera principiante la diferencia entre un árbol de decisión muy profundo y uno más simple.
- Ayúdame a interpretar estos cuatro resultados: train y test del modelo profundo, y train y test del modelo simple.
- Redáctame una conclusión corta sobre por qué limitar la profundidad puede ayudar a generalizar.
- Revisa este código y dime si realmente estoy comparando entrenamiento y prueba de forma correcta.

Mejoras simples que suelen ayudar

Mejora simple	Qué hace	Qué se espera observar
Limitar profundidad o complejidad	Evita que el modelo memorice demasiados detalles.	Train puede bajar un poco, pero test suele estabilizarse.
Reducir variables irrelevantes	Quita ruido y simplifica la señal.	Mejor interpretación y menos riesgo de ajuste excesivo.
Usar train/test correctamente	Separa aprendizaje y evaluación.	Medición más honesta del desempeño real.
Comparar varias configuraciones pequeñas	Permite elegir con criterio sin complicar el proceso.	Se identifica una versión más equilibrada.

Buenas prácticas para interpretar resultados

- Compare siempre entrenamiento y prueba; nunca mire solo una de las dos.
- No elija el modelo solo por el puntaje más alto en train.
- Prefiera explicaciones claras y prudentes antes que afirmaciones exageradas.
- Registre las mejoras probadas para justificar por qué una versión parece más confiable.

Mini ruta de trabajo sugerida

- Observe primero si existe una brecha entre train y test.
- Describa con palabras sencillas si la brecha parece pequeña o preocupante.
- Pruebe una mejora simple, como limitar la complejidad.
- Vuelva a comparar train y test después del ajuste.
- Escriba una conclusión breve sobre si hay evidencia de sobreajuste.

Producto esperado

El producto esperado en esta versión de la guía es un notebook corto y ordenado donde el estudiante compare entrenamiento y prueba, identifique si existe o no una señal de sobreajuste y explique qué mejora simple aplicó o aplicaría para buscar un modelo más confiable.

Lista de verificación

Revisión	Criterio
☐	Se comparó explícitamente el desempeño en entrenamiento y en prueba.
☐	Se identificó si existe una brecha que sugiera sobreajuste.
☐	Se explicó con palabras sencillas por qué un resultado alto en train no basta.
☐	Se probó o se propuso al menos una mejora simple.
☐	La conclusión final justifica si hay o no evidencia de sobreajuste.

Cierre

El sobreajuste es una de las primeras lecciones importantes del aprendizaje automático. Nos enseña que un modelo no vale por lo bien que se ve en el entrenamiento, sino por lo bien que responde cuando aparecen casos nuevos. Aprender a comparar train y test, y a buscar mejoras simples, es un paso clave para construir criterio y trabajar con IA de manera responsable.