Clase 059 — Launch, monitoreo y mantenimiento de modelos

Parte: 1 — Machine Learning Clásico · Fuente: Géron, cap. 2 § Launch, Monitor, and Maintain Your System. · Duración estimada: 60 min.

Clase 059 — Launch, monitoreo y mantenimiento de modelos

> *Parte: 1 — Machine Learning Clásico · Fuente: Géron, cap. 2 § Launch, Monitor, and Maintain Your System. · Duración estimada: 60 min.*

Objetivo

Que el alumno entienda que entrenar el modelo es la mitad del trabajo: el resto es ponerlo en producción de forma segura, monitorearlo para detectar degradación (data drift, model drift) y mantenerlo vivo con un ciclo de retraining. Además, documentar el modelo con una Model Card para que terceros (compliance, negocio, usuarios) sepan qué hace, dónde falla y qué no hay que hacerle.

Resultados de aprendizaje

Al finalizar la clase, el alumno podrá:

1. Diseñar un pipeline de deploy mínimo (serialización con joblib, servicio detrás de una API, versionado del artefacto).
2. Distinguir data drift de model drift y elegir métricas para cada uno (PSI, KS, accuracy en holdout móvil).
3. Definir un retraining trigger (calendario fijo vs. trigger por drift vs. trigger por caída de KPI de negocio).
4. Comparar estrategias de release (canary, shadow deploy, A/B test) y elegir según riesgo.
5. Redactar una Model Card con secciones mínimas (uso previsto, métricas por subgrupo, limitaciones).

Temas

#	Tema	Por qué importa
1	Pipeline de deploy: joblib.dump, contenedo	El modelo serializado es el artefacto prod
2	Data drift (inputs cambian) vs. model drif	Se monitorean distinto; confundirlos te ll
3	Métricas de drift: PSI, KS-test, distancia	Cuantifican el cambio en distribución ante
4	Retraining: calendario, trigger por drift,	Cuándo reentrenar sin gastar de más ni que
5	Estrategias de release: shadow, canary, A/	Bajan el riesgo de un modelo malo en produ
6	Alertas y observabilidad	Logueo de inputs/outputs, dashboards, on-c
7	Model Cards y Datasheets	Documentación responsable del modelo y de
8	Governance y rollback	Versionar modelos como código; poder volve

Versión profundizada — 2026

El tema moderno que antes vivía como complemento dentro de esta clase ahora tiene su(s) clase(s) propia(s) con patrón completo, ejercicios y homework:

- Clase 053a — Model Cards y Responsible ML

Definiciones y características

Deployment

: Proceso de exponer un modelo entrenado como servicio consumible (REST, batch, edge). Incluye serialización del artefacto (joblib, pickle, ONNX), versionado y contenedorización.

Model drift (concept drift)

: La relación entre X e y cambia con el tiempo. Ejemplo: hábitos de compra post-pandemia. Se detecta midiendo accuracy/AUC sobre una ventana móvil con labels reales (cuando llegan).

Data drift (covariate shift)

: La distribución de los inputs X cambia, aunque la relación X→y siga estable. Se detecta sin necesidad de labels — comparás la distribución de cada feature en producción vs. la del set de entrenamiento (PSI, KS).

Retraining trigger

: Regla que decide cuándo reentrenar. Tres familias: (a) calendario fijo (cada N días), (b) drift detectado (PSI > 0.25), (c) caída de KPI de negocio (conversion baja > X%).

A/B test

: Dos versiones del modelo sirven tráfico en paralelo (e.g. 50/50). Se compara KPI de negocio con significancia estadística. Requiere volumen.

Shadow deploy

: El modelo nuevo recibe los mismos requests que el viejo pero sus predicciones no se devuelven al usuario — se loguean para comparar. Cero riesgo, ideal para validar en datos reales antes del switch.

Model card

: Documento estandarizado que describe modelo, uso previsto, métricas (overall + por subgrupo), datos, limitaciones y consideraciones éticas. Formalizado por Mitchell et al. (2019).

Governance

: Conjunto de políticas: quién aprueba un deploy, cómo se versiona el modelo, cómo se hace rollback, qué se loguea para auditoría. Equivalente del "code review" para modelos.

Dataset / recursos

Sintético: dos snapshots de un dataset tabular (mes 1 y mes 6) para simular drift. Plantilla Markdown de Model Card en model_card_template.md.

Ejercicios

1. Serializar y cargar. Entrená un RandomForestClassifier sobre Titanic. Guardalo con joblib.dump. Cargalo en otro notebook y verificá que predice idéntico.

2. Detectar data drift con PSI. Calculá Population Stability Index entre dos snapshots de la feature edad. Interpretá: PSI < 0.1 estable, 0.1-0.25 leve, > 0.25 drift significativo.

3. KS-test para drift. Aplicá scipy.stats.ks_2samp a la feature monto entre snapshot1 y snapshot2. ¿p-valor < 0.05?

4. Simular shadow deploy. Tenés modelo A (viejo) y B (nuevo). Pasá 1000 requests por ambos, logueá las predicciones y reportá tasa de desacuerdo.

5. Redactar una Model Card. Tomá tu mejor modelo de la Parte 1 hasta ahora y completá una model card

con las 7 secciones del complemento. Incluí al menos una métrica desagregada por subgrupo.

Homework verificable

Notebook que: (a) entrene un clasificador, (b) lo serialice con joblib, (c) simule un mes de tráfico productivo con una feature drifteada, (d) calcule PSI por feature, (e) dispare una alerta si PSI > 0.25, (f) entregue un archivo MODEL_CARD.md completo en la carpeta del homework.

Criterio de aceptación: El notebook detecta el drift inyectado artificialmente. La Model Card tiene las 7 secciones mínimas con valores reales (no placeholders), incluye al menos una métrica por subgrupo y declara explícitamente un out-of-scope use.

Errores comunes

Síntoma / mensaje	Causa y cómo arreglar
El modelo cargado con joblib.load predice	Cambió la versión de sklearn entre dump y
Reentreno cada semana "por las dudas" y el	Retraining sin trigger real introduce ruid
PSI da siempre 0 o inf	División por cero cuando un bin queda vací
El A/B test "no da significativo" después	Volumen insuficiente. Fix: hacé power anal
Model card con métrica única "accuracy = 0	Oculta sesgos por subgrupo y no dice para

Preguntas frecuentes

¿Cada cuánto hay que reentrenar?

No hay número mágico. Si el dominio es estable (físico, industrial), meses o años. Si es comportamiento humano online (e-commerce, fraude), semanas o incluso días. Lo correcto es dejar que el trigger lo decida (PSI, caída de KPI), no el calendario.

¿Diferencia práctica entre shadow deploy y canary?

Shadow: el modelo nuevo predice pero sus predicciones no se devuelven al usuario — cero riesgo, solo comparás. Canary: el modelo nuevo sí sirve tráfico real, pero a un % chico (1-5%) — bajo riesgo pero no nulo. Shadow para validar, canary para liberar gradualmente.

¿Model card o documentación técnica clásica?

Las dos. La doc técnica (README, API reference) es para desarrolladores. La model card es para stakeholders no técnicos: producto, legal, compliance, auditoría. Si tu modelo cae bajo EU AI Act, la model card no es opcional.

¿pickle o joblib o ONNX?

joblib para sklearn (más eficiente con arrays NumPy grandes que pickle puro). pickle para objetos Python genéricos. ONNX cuando necesitás portabilidad cross-language (servir un modelo Python desde un backend en C# o Java).

¿PSI o KS-test?

PSI es más interpretable para negocio (umbrales 0.1 / 0.25 son estándar de la industria de scoring crediticio) y opera sobre features categóricas/binneadas. KS-test es estadísticamente más riguroso para features

continuas y devuelve p-valor. En la práctica, equipos serios reportan ambas.

Referencias

- Géron, cap. 2 § Launch, Monitor, and Maintain Your System.
- Mitchell et al. (2019). Model Cards for Model Reporting. FAT* 2019.
- Gebru et al. (2018). Datasheets for Datasets.
- HuggingFace Model Cards guide.
- Google model-card-toolkit.
- EU AI Act — documentación técnica (Anexo IV).

Siguiente clase

Clase 060 — Model Cards y Responsible ML

Apéndice: notebook (primer bloque)

Primera celda ejecutable del notebook de la clase.

```
# Imports y configuración inicial
```

Archivos complementarios

- notebook.ipynb