Clase 202 — Monitoreo: data drift, model drift, alertas

Parte: 4 — MLOps · Fuente: Huyen cap. 8 + Evidently AI docs + NannyML + Gama et al., A Survey on Concept Drift Adaptation (ACM, 2014). Duración estimada: 80 min.

Clase 202 — Monitoreo: data drift, model drift, alertas

> *Parte: 4 — MLOps · Fuente: Huyen cap. 8 + Evidently AI docs + NannyML + Gama et al., A Survey on Concept Drift Adaptation (ACM, 2014). Duración estimada: 80 min.*

Objetivo

Detectar antes que el negocio: (a) data drift (la distribución de features cambió), (b) prediction drift (la distribución de predicciones cambió), (c) concept drift (la relación X→y cambió). Configurar alertas que avisen antes de que la métrica de negocio caiga, no después.

Resultados de aprendizaje

Al finalizar, el estudiante podrá:

- Distinguir data drift (P(X) cambia), prediction drift (P(ŷ) cambia), concept drift (P(y|X) cambia) y elegir el test correcto para cada uno.
- Detectar drift con tests estadísticos: PSI (Population Stability Index), K-S (continuas), chi-cuadrado (categóricas), Wasserstein (más sensible que K-S en colas).
- Usar Evidently AI para generar reportes HTML con todos los tests + visualizaciones, y NannyML para estimar performance sin labels en producción (CBPE).
- Configurar alertas en Grafana / CloudWatch / Slack vía webhook cuando el drift score supera el umbral.
- Reconocer cuándo el "drift" es ruido (re-test con bonferroni) vs señal (acción).

Temas

#	Tema	Por qué importa
1	3 tipos de drift y por qué importan distin	Confundirlos lleva a "rentrenar" sin neces
2	PSI: umbral 0.1 / 0.2	El estándar de la industria; simple, inter
3	K-S, chi-cuadrado, Wasserstein	Tests con propiedades distintas — elegir s
4	Performance estimation sin labels (CBPE)	Cómo saber si el modelo empeora antes de t
5	Reference window vs current window	El qué se compara con qué.
6	Alertas: umbral + cooldown + canal	Sin diseño → alert fatigue.

Definiciones y características

- Data drift (covariate shift): P_train(X) ≠ P_prod(X). Ej: en training los users tenían 18-65 años, en prod aparecen muchos >70. El modelo todavía es válido si la relación X→y no cambió, pero las predicciones serán menos confiables en zonas no vistas.
- Prediction drift: P_train(ŷ) ≠ P_prod(ŷ). La distribución de salidas cambió. A veces es síntoma de data drift, a veces es legítimo (estacionalidad). No siempre malo.
- Concept drift (real drift): P(y|X) cambió — la relación entre features y target. Ej: durante COVID, "salir a la calle" pasó de actividad normal a anómala. Aún sin data drift, el modelo deja de servir. Este es el más grave, y el más difícil de detectar sin labels.
- PSI (Population Stability Index): Σ (p_i - q_i) * ln(p_i / q_i) sobre bins. Umbrales clásicos: <0.1 estable,

0.1-0.2 cambio menor, >0.2 shift significativo (acción).
- K-S test (Kolmogorov-Smirnov): estadístico = sup|F_ref(x) - F_cur(x)|. p-value bajo → distros distintas. Sensible al centro de la distribución.
- Chi-cuadrado: para variables categóricas. Compara frecuencias observadas vs esperadas.
- Wasserstein distance (Earth Mover's Distance): "cuánta tierra hay que mover" para transformar una distro en otra. Más sensible que K-S a diferencias en cola.
- Reference window: muestra del periodo de training (o un slice estable previo).
- Current window: muestra del periodo a evaluar (último día, última semana). Tamaño del window afecta sensibilidad (chico → ruidoso, grande → lento).
- CBPE (Confidence-Based Performance Estimation, NannyML): estima accuracy o AUC de producción usando solo las probabilidades del modelo (no labels). Asume bien-calibrado.

Dataset / recursos

- Dataset training: California Housing — usado como reference.
- Dataset "producción": California Housing con shift sintético (escalar MedInc × 1.5 para simular inflación, o filtrar HouseAge > 30 para simular nuevo segmento).
- Librerías: evidently, scipy, nannyml, scikit-learn.

Ejercicios

1. PSI manual: bineá MedInc en 10 deciles (con bordes del reference). Calculá p_ref, p_cur y aplicá la fórmula PSI. Verificá que sin shift PSI < 0.05; con shift × 1.5 PSI > 0.5.
2. K-S vs Wasserstein: agregale outliers a 5% de la muestra de producción (multiplicá esos por 100). K-S podría no detectar (la mediana no cambió mucho); Wasserstein sí. Reproducí ambos casos.
3. Reporte Evidently: Report(metrics=[DataDriftPreset()]) sobre reference vs current. Guardalo como HTML, abrilo, identificá qué features driftearon y con qué test.
4. Concept drift sin labels (CBPE): con NannyML, fit el estimador sobre reference + predicciones. Aplicalo a current. Compará estimated_accuracy vs actual_accuracy (calculable porque tenés y en este ejercicio).
5. Alerta: escribí un script que (a) calcule PSI por feature, (b) si alguna >0.2 emite un POST a un webhook Slack/Discord, (c) usá cooldown de 4 h con un archivo last_alert.txt para evitar spamming.

Homework verificable

Sistema de monitoreo que produce:

1. Job diario (Cron / Airflow / GH Actions schedule) que toma el snapshot de producción de las últimas 24 h y lo compara con el reference window.
2. Reporte Evidently HTML guardado en S3 / GCS (timestamped).
3. PSI dashboard en Grafana o Streamlit que muestra evolución por feature en los últimos 30 días.
4. Alerta a Slack si: PSI > 0.2 en cualquier feature, O drift score global > umbral, O CBPE estimated_accuracy cayó >5 puntos.
5. Runbook (runbook.md) con 3 escenarios: (a) drift de feature legítimo (cambio negocio) → retrainings, (b) drift por bug en pipeline → fix upstream, (c) concept drift → A/B test modelo nuevo.

Criterio de aceptación: simular un shift introducido a propósito (multiplicar feature × 2) dispara la alerta en <24 h. El runbook tiene pasos accionables, no descripciones genéricas.

Errores comunes

Síntoma / mensaje	Causa y cómo arreglar
PSI = 0.3 pero el modelo sigue bien	Es data drift sin concept drift — la distr
Tests dan "drift" en TODO siempre	n muy grande → cualquier diferencia es "si
Alerta dispara 50 veces / día	Falta cooldown y/o el threshold es muy agr
Reference window se actualiza cada deploy	Si el reference cambia con cada release, p
Detecto drift pero no sé qué hacer	Falta runbook. Fix: para cada feature crít
CBPE da accuracy muy distinta a la real	El modelo está mal calibrado — CBPE asume

Preguntas frecuentes

¿Cada cuánto retrainear?

Hay 3 estrategias: (1) schedule fijo (cada lunes) — simple pero ciego. (2) trigger por drift (PSI > 0.3) — reactivo. (3) trigger por performance degradation (CBPE estima accuracy caída >X%) — mejor. La industria converge a (3) con fallback a (2). Ver Clase 203.

¿PSI o KL divergence?

PSI es simétrica (PSI(A,B) = PSI(B,A)) y robusta; KL no. PSI domina en la industria financiera; ML moderno usa también Wasserstein o Earth Mover's Distance. Para empezar: PSI.

¿Mido drift por feature o multivariado?

Por feature es interpretable ("MedInc driftó"). Multivariado (PCA + drift sobre componentes, o Maximum Mean Discrepancy) capta correlaciones nuevas. Ideal: ambos, pero un drift de feature suele ser suficiente para disparar investigación.

¿Cómo evito alertas en feriados/temporadas?

(1) Reference window por época: comparar diciembre 2026 vs diciembre 2025, no vs julio. (2) Modelar estacionalidad explícitamente (incluir month/day_of_week como feature). (3) Suprimir alertas en feriados conocidos.

¿Evidently vs NannyML vs Whylogs vs Arize?

Evidently: open-source, reportes HTML excelentes. NannyML: open-source, especialista en CBPE (estimar performance sin labels). Whylogs: librería de profiling minimalista de WhyLabs. Arize: SaaS comercial, dashboards e integraciones empresariales. Para empezar: Evidently + NannyML.

¿Y si no tengo labels en producción?

Es lo normal en muchos casos. Estrategias: (1) CBPE (NannyML) — estima con probas. (2) Proxy labels — métricas de negocio downstream (CTR, conversión). (3) Active learning — etiquetar un subset random para validar. (4) Shadow scoring — guardás predicciones + features y cuando llegan labels (días después), evaluás.

Referencias

- Huyen, Chip. Designing Machine Learning Systems (O'Reilly, 2022), cap. 8 — Monitoring.
- Evidently AI — reportes y tests.

- NannyML — CBPE y DLE.
- Gama et al., A Survey on Concept Drift Adaptation (ACM Computing Surveys, 2014) — taxonomía clásica.
- Monitoring ML Models: Theory (Tagliabue) — visión práctica.

Siguiente clase

Clase 203 — Reentrenamiento programado

Apéndice: notebook (primer bloque)

Primera celda ejecutable del notebook de la clase.

```
import numpy as np, pandas as pd
from scipy import stats
from sklearn.datasets import fetch_california_housing

X, y = fetch_california_housing(return_X_y=True, as_frame=True)
rng = np.random.default_rng(42)

# Reference = primer 60%, Current = último 40% con shift
n = len(X)
ref = X.iloc[:int(n * 0.6)].copy()
cur = X.iloc[int(n * 0.6):].copy()

# Shift sintético: 'MedInc' inflación + 'HouseAge' segmento nuevo
cur['MedInc'] = cur['MedInc'] * 1.5
cur = cur[cur['HouseAge'] >= 25]
print('ref:', ref.shape, '| cur:', cur.shape)
```

Archivos complementarios

- notebook.ipynb