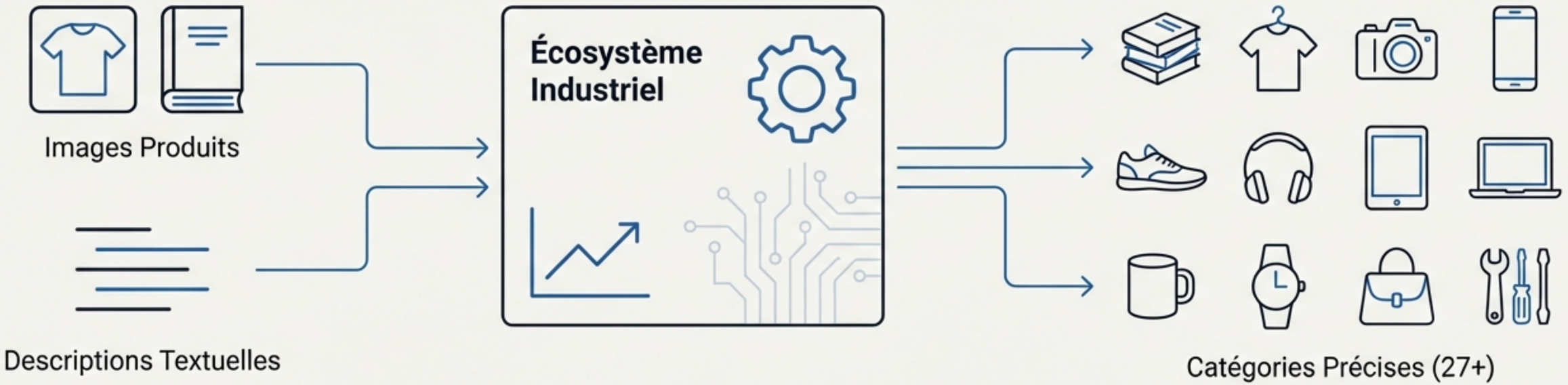
Industrialiser la classification de produits : une plateforme MLOps de bout en bout

Projet de classification multimodale pour l'écosystème e-commerce Rakuten



Le Défi

Classifier automatiquement des millions de produits dans 27 catégories précises en utilisant à la fois leurs descriptions textuelles et leurs images.

Notre Solution

Une plateforme MLOps complète et conteneurisée. Il ne s'agit pas seulement d'un modèle, mais d'un écosystème industriel pour entraîner, déployer, surveiller et maintenir le modèle en conditions opérationnelles.

Notre objectif : simuler un cycle de vie MLOps complet pour garantir la performance sur le long terme

Performance Multimodale

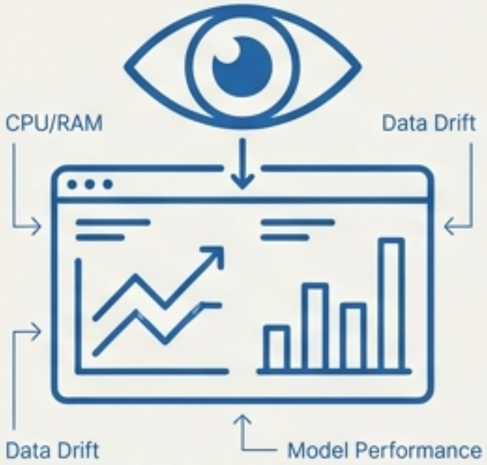
Améliorer la précision en combinant la puissance du NLP (texte) et de la Computer Vision (images).

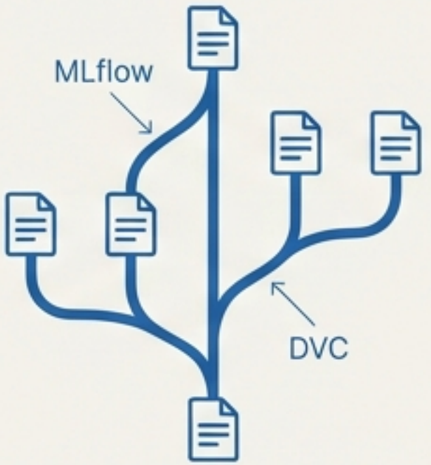
Robustesse & Maintenabilité

Automatiser intégralement les processus de réentraînement et de prédiction pour éliminer les interventions manuelles et garantir la fiabilité.

Observabilité “Full-Stack”

Surveiller non seulement la santé technique (CPU/RAM), mais surtout la qualité des données (Drift) et la performance métier du modèle.

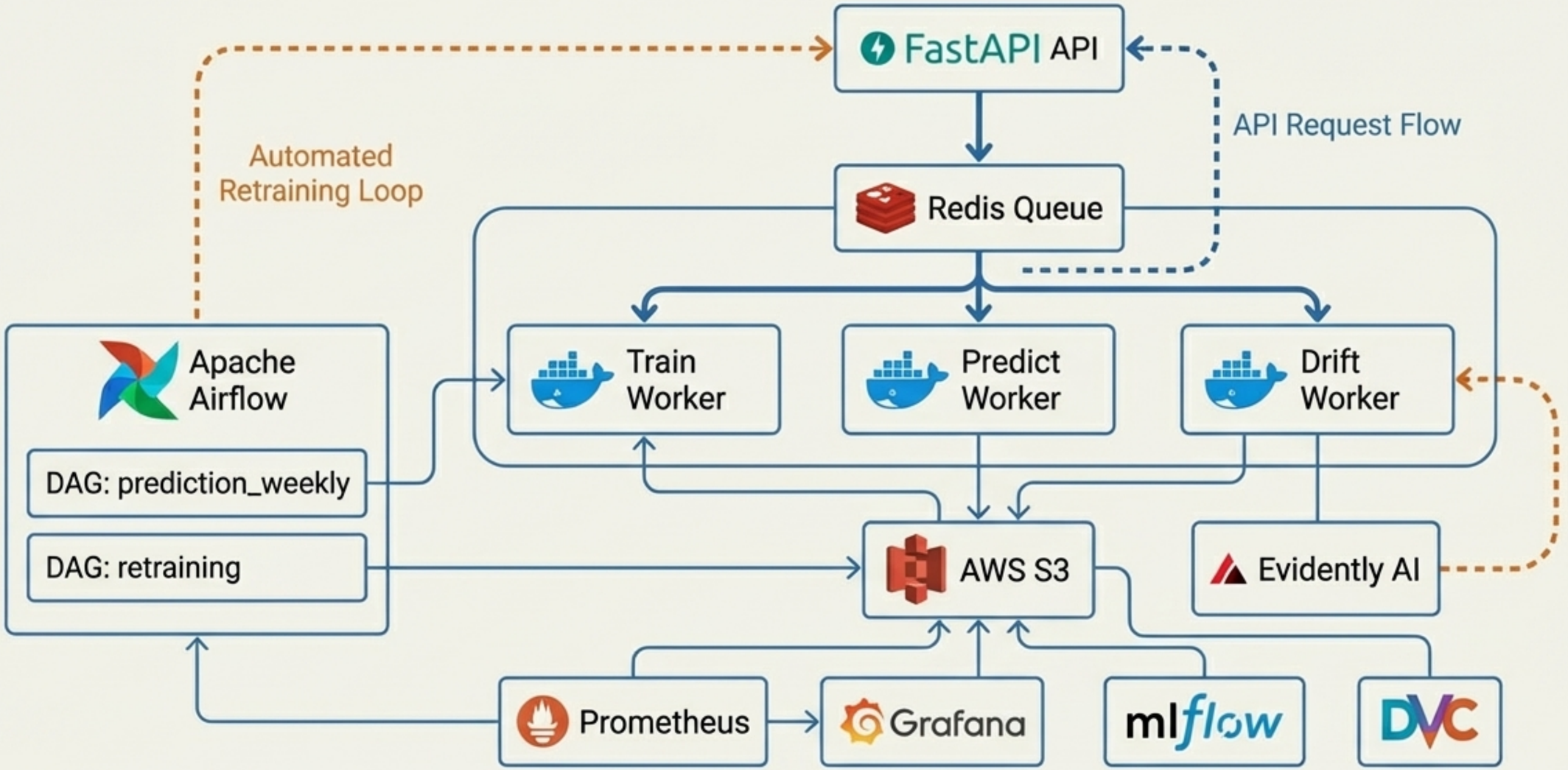




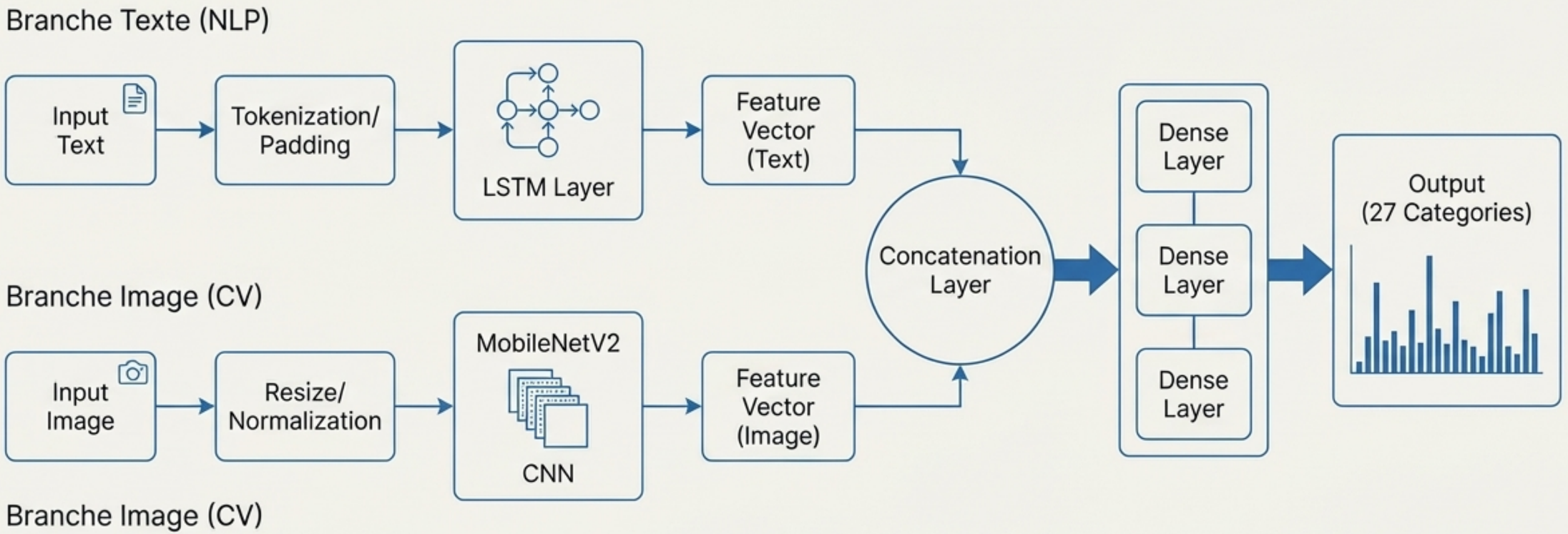
Reproductibilité Scientifique

Garantir que chaque expérience est traçable (MLflow) et que les données sont versionnées (DVC) pour des résultats fiables et auditables.

Notre réponse : un écosystème micro-services intégré et conteneurisé



Au cœur du système : un modèle de fusion pour une classification plus fine



Stack Technique: TensorFlow/Keras

Flexibilité: Support CPU et GPU (NVIDIA Container Toolkit)

Une stack technique moderne, choisie pour la modularité et l'automatisation

Infrastructure & Conteneurisation

Pour des environnements reproductibles et une architecture micro-services modulaire. Profils `cpu` et `gpu` pour la flexibilité.

Pour un stockage d'objets scalable et un déploiement cloud.

Orchestration & Serving

Pour orchestrer les pipelines complexes de prédiction et de réentraînement avec une logique conditionnelle.

Pour une API REST asynchrone, performante et résiliente, capable de gérer les tâches longues sans bloquer le serveur.

Monitoring & Observabilité

Pour le monitoring technique et la visualisation des métriques système en temps réel.

Pour le monitoring métier crucial : la détection de dérive des données (Data Drift).

Versioning & Tracking

mlflow DVC

Pour une traçabilité complète des expériences, des modèles et des données.

Le 'Continuous Training' : un système qui apprend et s'auto-corrige

1. SURVEILLER

Un DAG Airflow (`prediction_weekly`) s'exécute, générant un rapport de drift avec Evidently AI.

2. DÉTECTER

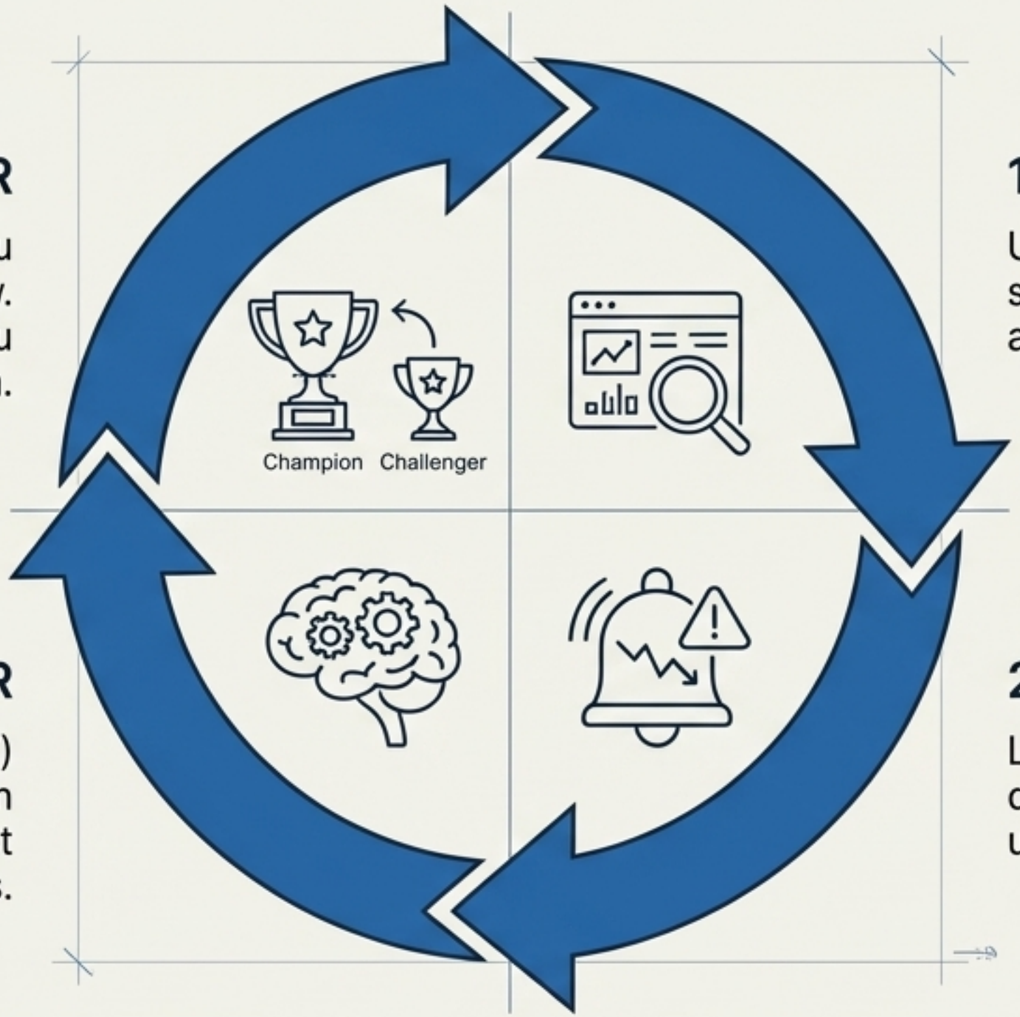
Le rapport est analysé. Si un drift de données ou de concept est avéré, une condition est remplie.

3. RÉENTRAÎNER

Un second DAG Airflow (`retraining`) est déclenché automatiquement. Un nouveau modèle 'Challenger' est entraîné sur les données fraîches.

4. DÉPLOYER

Le Challenger est comparé au 'Champion' en production via MLflow. S'il est plus performant, il est promu et devient le nouveau Champion.



Cette boucle d'automatisation est la clé pour garantir que le modèle reste performant face à l'évolution naturelle des données.

Nous avons simulé la vie réelle pour prouver la résilience du système

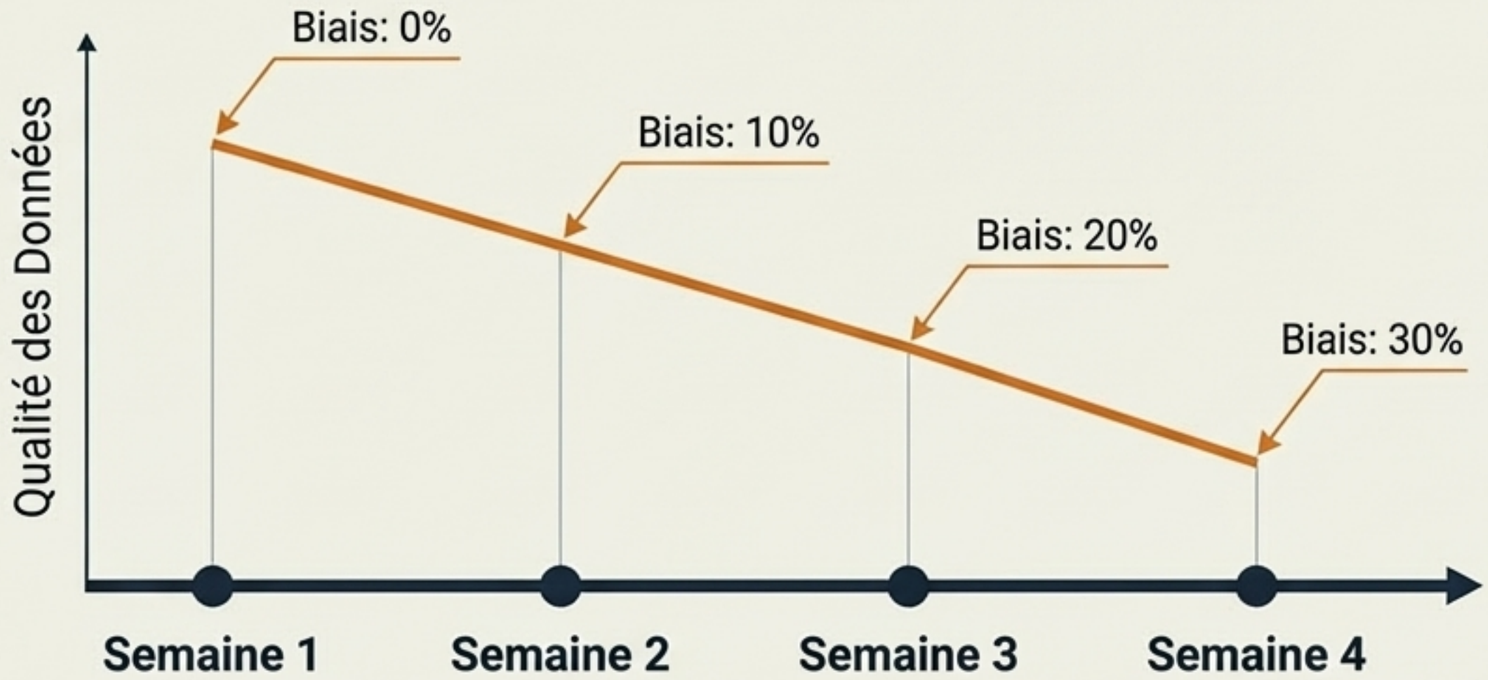
Scénario de Simulation

Hypothèse: Simulation de 4 semaines d'activité.

Mécanisme de Drift: Introduction d'une dérive progressive et contrôlée dans les données d'inférence.

Intensité: Le biais dans les données augmente de 0% (Semaine 1) à 30% (Semaine 4).

Implémentation: Utilisation d'un script dédié (`*manage_simulated_weeks.py*`) pour gérer la rotation des données et la synchronisation avec S3.



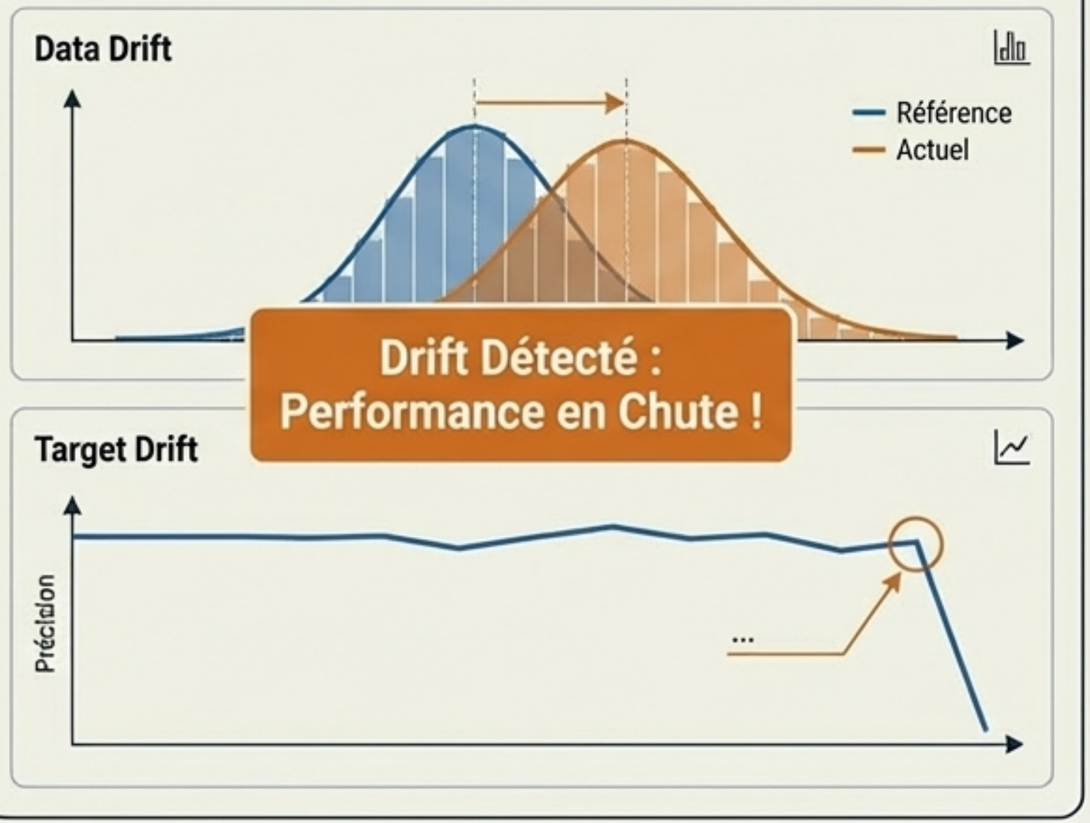
Objectif de la Démo

Prouver la capacité du système à :

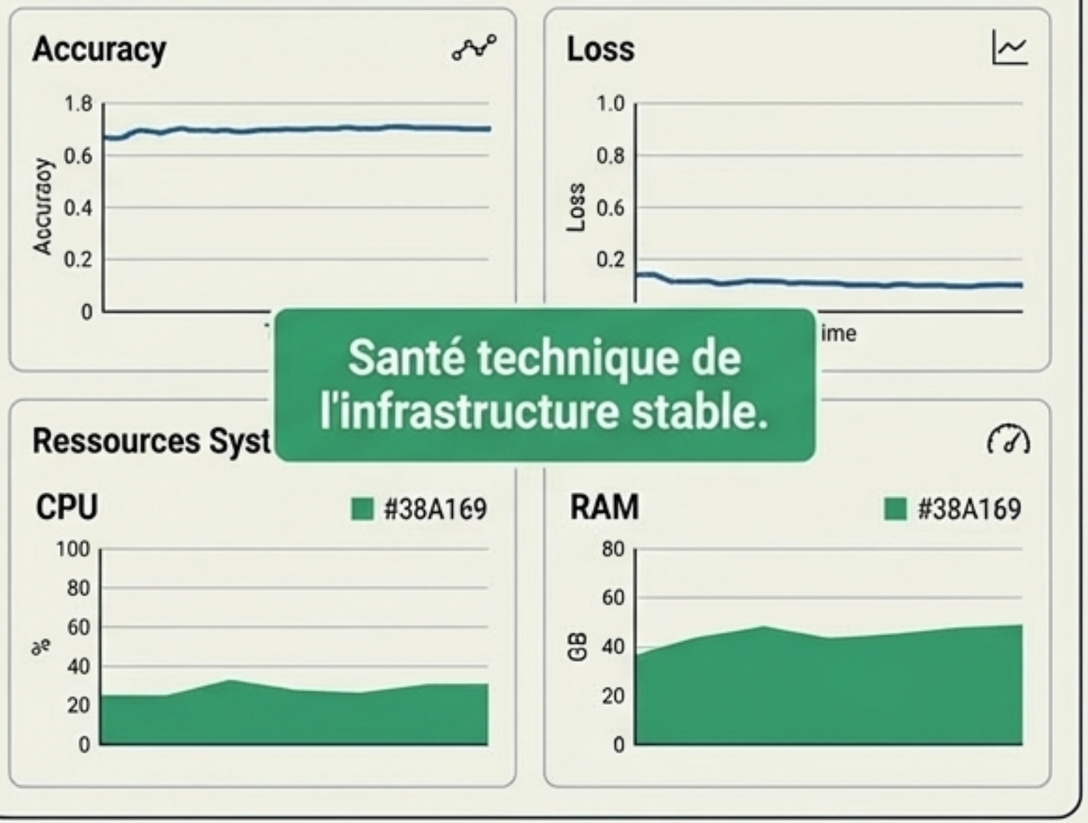
1. **Détecter** la dégradation.
2. **S'auto-corriger** sans intervention humaine.

L'observabilité en action : détection de la dégradation des performances

Rapport de Dérive



Monitoring Système

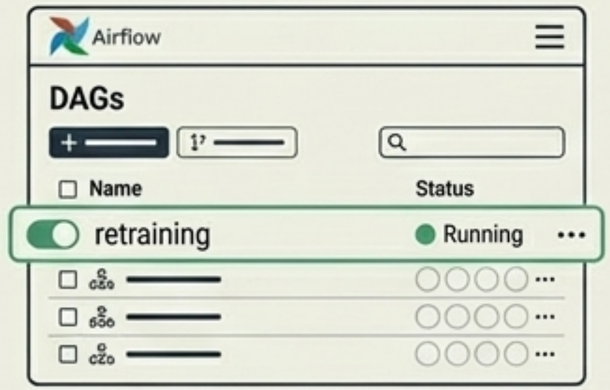


Le système distingue un problème de données/modèle d'un problème d'infrastructure.

Le système réagit : réentraînement et restauration automatiques de la performance

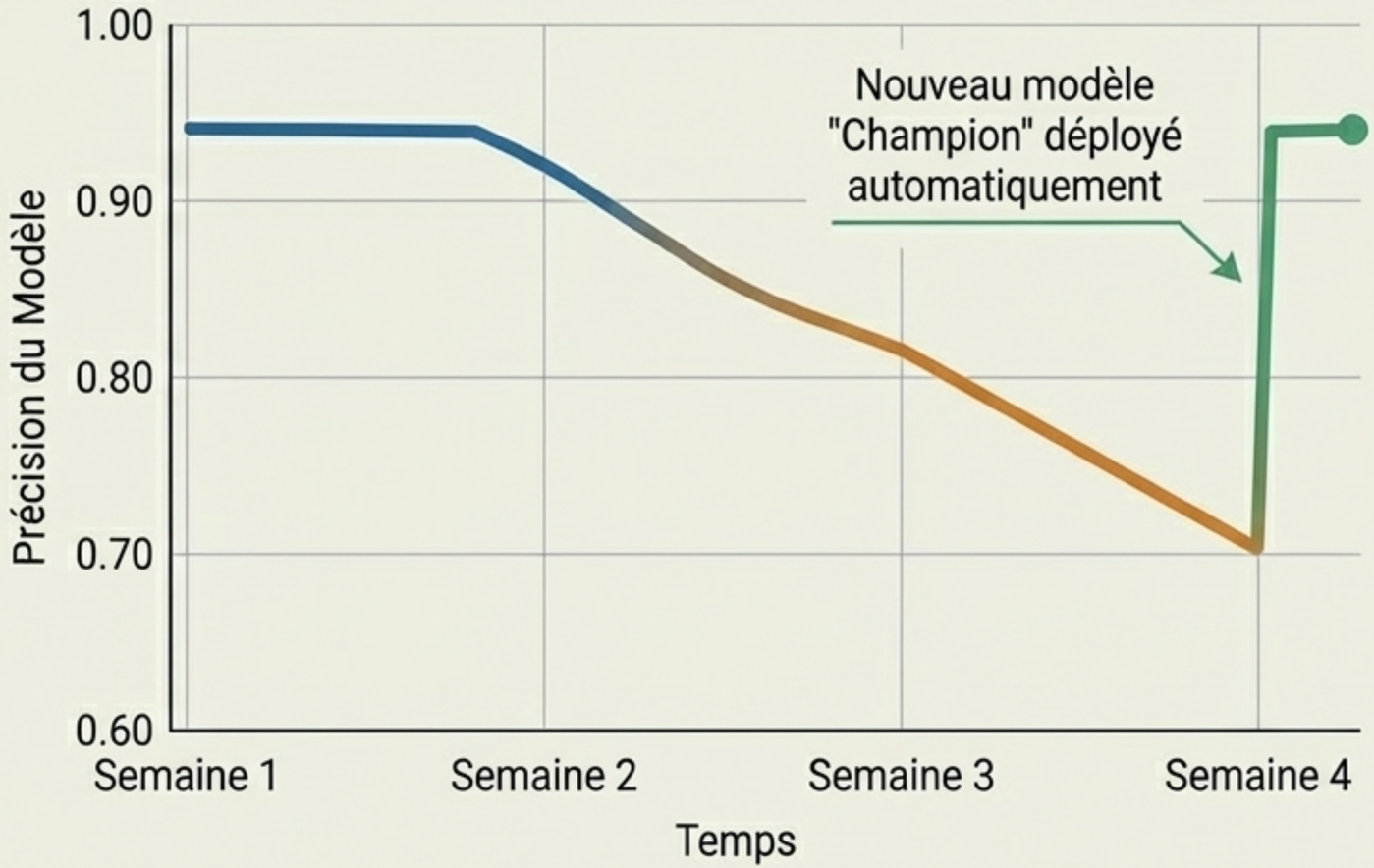
1. Alerte

Une notification est envoyée à la fin du cycle d'entraînement.



2. Déclenchement

Évolution de la Précision du Modèle



Les piliers de la plateforme : une conception axée sur la production

ROBUSTESSE

- Gestion des erreurs et `retries` automatiques automatiques dans Airflow.
- Découplage API/Worker via Redis pour une haute disponibilité.
- Fallback CPU/GPU pour s'adapter à l'environnement.

REPRODUCTIBILITÉ

- Environnements 100% conteneurisés avec Docker.
- Versioning intégral des données (DVC) et du code (Git) pour une traçabilité parfaite.

FLEXIBILITÉ

- API REST supportant l'inférence synchrone (temps réel) et asynchrone (batchs/entraînement).
- Architecture modulaire permettant de mettre à jour chaque composant indépendamment.

Une vision lucide : les axes d'amélioration pour une mise en production à grande échelle

Sécurité & Exposition

Problème Exposition directe de nombreux ports. Secrets gérés via `.env`.

Solution Intégrer un Reverse Proxy (Nginx, Traefik) pour centraliser l'accès et gérer le HTTPS. Utiliser un gestionnaire de secrets (Vault, Docker Secrets).

Tests & CI/CD

Problème Absence d'une pipeline CI/CD formalisée pour les tests d'intégration.

Solution Mettre en place des workflows GitHub Actions pour lancer les tests end-to-end à chaque `push`.

Scalabilité des Ressources

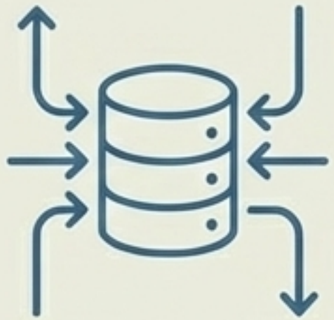
Problème L'exécution de tous les services sur une seule machine est gourmande. SQLite pour MLflow n'est pas scalable.

Solution Migrer la base MLflow vers PostgreSQL. Préparer une migration vers un orchestrateur de conteneurs distribué.

Perspectives d'évolution : vers une plateforme MLOps de calibre industriel

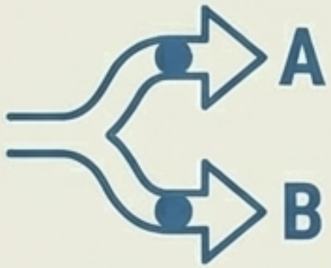
Déploiement Kubernetes (K8s)

Migrer de Docker Compose vers des manifestes Kubernetes/Helm pour bénéficier de l'auto-scaling, de la haute disponibilité et d'une gestion de ressources avancée.



Intégration d'un Feature Store

Centraliser la logique de transformation des features avec un outil comme Feast pour garantir la cohérence entre l'entraînement et l'inférence.



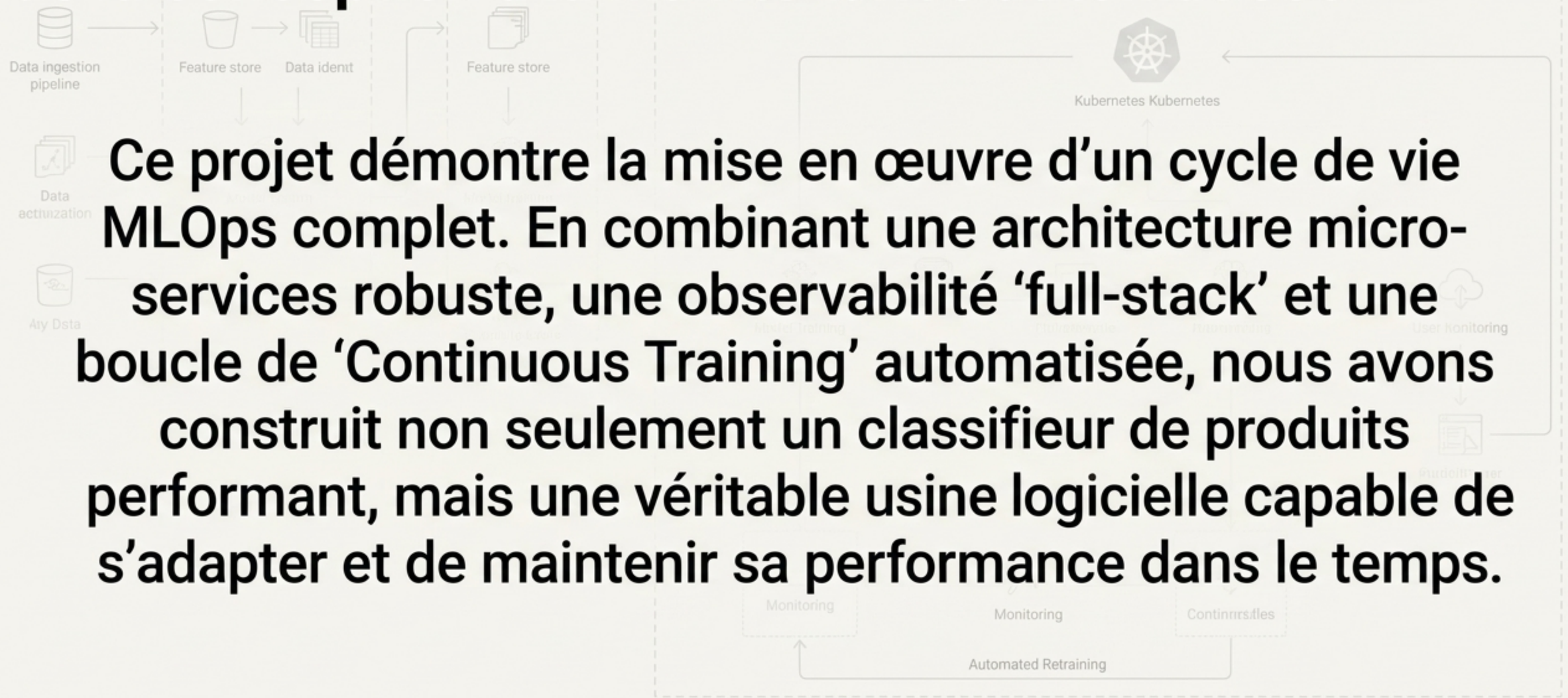
Stratégies de déploiement A/B

Implémenter des déploiements "Canary" ou "Blue/Green" pour tester un nouveau modèle sur une fraction du trafic avant un basculement complet.

Interface Utilisateur de Démonstration

Créer une application web simple (Streamlit ou React) pour permettre à un utilisateur final de tester la classification et de visualiser les résultats.

Plus qu'un modèle : un écosystème MLOps complet, de la conception à la maintenance automatisée



Ce projet démontre la mise en œuvre d’un cycle de vie MLOps complet. En combinant une architecture micro-services robuste, une observabilité ‘full-stack’ et une boucle de ‘Continuous Training’ automatisée, nous avons construit non seulement un classifieur de produits performant, mais une véritable usine logicielle capable de s’adapter et de maintenir sa performance dans le temps.