Авторы:

Нджака Манье Джордана Р Сесиль,

Иванова Ольга Николаевна, канд. наук.

Учреждение: ЧелГУ

РАЗРАБОТКА ПРОГРАММНОГО КОМПЛЕКСА ДЛЯ ДЕТЕКТИРОВАНИЯ ДРИФТА ДАННЫХ И АДАПТИВНОГО ПЕРЕОБУЧЕНИЯ МОДЕЛЕЙ

Аннотация

В данной статье предлагается унифицированная платформа для поддержки моделей машинного обучения (ML) в производственной среде, объединяющая обнаружение дрейфа данных, автоматическое переобучение, управление жизненным циклом модели и операционный API. Ядро платформы реализует три статистических теста для оценки дрейфа: PSI (Population Stability Index) для бинарных распределений числовых и категориальных признаков, критерий Колмогорова-Смирнова (KS) для числовых признаков и критерий χ^2 для категориальных признаков. Превышение настраиваемых пороговых значений политики запускает контролируемые действия : от логирования до автоматического переобучения модели на объединенном наборе данных «обучающая + текущая». Обученные артефакты версионируются, записываются в реестр моделей и автоматически или вручную продвигаются по этапам жизненного цикла (разработка, тестирование, производство, архивирование). Система реализована как сервис на FastAPI с панелью управления SPA на React и MUI, а для хранения данных (реестр запусков, эксперименты, версии моделей, происхождение данных) используется SQLite. Экспериментальное тестирование проводилось на двух наборах

данных; Данные о доходах взрослого населения (14 признаков, бинарная классификация доходов) и данные о мошенничестве с кредитными картами на Kaggle (29 числовых признаков, высокий дисбаланс классов) - в нескольких сценариях стабильности и явного дрейфа. Платформа демонстрирует полный цикл MLOps на легковесном и воспроизводимом стеке и служит эталонной реализацией для образовательных и прикладных задач.

Ключевые слова: MLOps , дрейф данных, PSI, критерий Колмогорова-Смирнова, критерий хи-квадрат, автоматическое переобучение, реестр моделей, жизненный цикл модели, FastAPI , React, SQLite

Annotation

This article proposes a unified platform for supporting machine learning (ML) models in a production environment, combining data drift detection, automatic retraining, model lifecycle management, and an operational API. The core of the platform implements three statistical tests for assessing drift: PSI (Population Stability Index) for binary distributions of numerical and categorical features, Kolmogorov-Smirnov criterion (KS) for numerical features and criterion $\chi 2$ for categorical features. Exceeding configurable policy thresholds triggers controlled actions: from logging to automatic retraining of the model on the combined training + current dataset. Trained artifacts are versioned, recorded in the model registry, and automatically or manually advanced through the life cycle stages (development, testing, production, archiving). The system is implemented as a FastAPI service with a SPA control panel on React and MUI, and SQLite is used to store data (launch registry, experiments, model versions, data origin). Experimental testing was conducted on two datasets; Data on adult income (14 features, binary income classification) and data on credit card fraud on Kaggle (29 numerical features, high class imbalance) - in several scenarios of stability and apparent drift. The platform demonstrates the full MLOps cycle on a

lightweight and reproducible stack and serves as a reference implementation for educational and applied tasks.

Keywords: MLOps , data drift, PSI, Kolmogorov-Smirnov criterion, chi-square criterion, automatic retraining, model registry, model lifecycle, FastAPI, React, SQLite

1. Введение

Качество модели машинного обучения , подтвержденное на этапе обучения, не гарантирует ее стабильность в работе. После развертывания в производственной среде модель сталкивается с изменениями, характером входных данных: сдвигом в распределении признаков (concept drift), изменением целевой переменной (concept drift), сезонностью и техническими артефактами сбора данных. В совокупности эти эффекты приводят к постепенному или резкому ухудшению качества, часто без очевидных сигналов от самой модели. Обеспечение надежности системы машинного обучения в работе является центральной задачей в современной практике MLOps [1, 2].

Промышленные решения для различных этапов жизненного цикла модели существуют и активно используются ; NannyML используются для мониторинга дрейфа, MLflow для регистрации и продвижения моделей, Weights & Biases для отслеживания экспериментов, а Apache Airflow и Prefect для оркестрации.

Однако типичной проблемой является фрагментация: инструменты охватывают разные этапы, интегрируются через разрозненные API и часто требуют значительных инженерных усилий.

Усилия по созданию согласованного процесса. Для учебных проектов, прототипов и небольших команд такой подход избыточен, и результат сложно представить это в виде единого наглядного картины.

В рамках этого проекта мы предлагаем и реализуем единую минималистичную платформу, которая объединяет ключевые компоненты MLOps в единый, согласованный поток:

● Обнаружение дрейфа данных с использованием PSI, KS и χ^2;

● Оценка правил политики и пороговых значений на основе агрегированных сигналов дрейфа;

● автоматическое переобучение модели при срабатывании политики ;

● Версионирование артефактов модели , регистрация экспериментов, продвижение версий на протяжении всех этапов жизненного цикла;

● Управление данными : блокировка версий обучающих выборок и связанных с ними базовых снимков;

● Единый REST API и панель управления для мониторинга, выполнения проверок, анализа и управления.

Цель данной работы не состоит в замене промышленных MLOps. Цель состоит в том, чтобы предложить решения, но при этом продемонстрировать
полную и воспроизводимую эталонную реализацию, в которой архитектурные решения и логика интеграции четко обозначены . Платформа ориентирована на прикладные исследования и демонстрацию принципов.

MLOps как часть академических и дипломных работ. Экспериментальная часть охватывает два разных набора данных (Adult Census Income и Kaggle Credit Card Fraud) и несколько сценариев: от «спокойного» случайного

разбиения до явного сдвига в разбиении транзакций по возрасту и времени D1/D2/D3, что соответствует реалистичному проявлению дрейфа.

Вклад данной работы включает в себя:

(1) описание архитектуры единой платформы, объединяющей мониторинг, оркестровку, переобучение и реестр; (2) формализованные пороговые политики на основе трех статистических тестов; (3) сквозной протокол происхождения данных, экспериментов и моделей;(4) воспроизводимую развертываемую конфигурацию (Docker Compose), подходящую для демонстрационных и образовательных целей.

2. Обзор предыдущих работ

Обнаружение дрейфа данных это классическая проблема на стыке статистики и машинного обучения. Индекс стабильности популяции PSI [3] широко используется в промышленной практике, особенно в банковском и страховом секторах, поскольку интуитивно интерпретируется как накопленная дивергенция бинарных распределений. Для числовых признаков часто используется критерий Колмогорова-Смирнова [4], который сравнивает эмпирические функции распределения; для категориальных признаков используется критерий χ^2 на таблице сопряженности [5]. В современных библиотеках (Evidently AI, NannyML , Alibi Detect) перечисленные тесты объединены в набор «стандартных» методов и дополнены более сложными подходами, включая те, которые основаны на плотностях и эталонных/текущих моделях классификации [2, 6].

На уровне оркестровки и автоматизации рабочих процессов доминирует Apache. Airflow [7] и Prefect [8] это универсальные системы для построения направленных ациклических графов задач (DAG). Они обеспечивают

планирование, повторные попытки, мониторинг и масштабирование, но для их полного использования требуется отдельное хранилище метаданных , планировщик и исполнители, что представляет собой значительную инфраструктурную нагрузку для небольших проектов.

Для управления жизненным циклом модели де-факто стандартом является MLflow Model Registry [9], который поддерживает эксперименты, версии моделей, этапы (Staging / Production / Archived) и интеграцию с различными инструментами логирования. Современные исследования в области ответственного ИИ (Responsible AI) также подчеркивают важность прослеживаемости — от версий обучающих данных и базовых профилей распространения до артефакта модели , развернутого в производственной среде [10]. Именно прослеживаемость преобразует отдельные технические компоненты в систему, управляемую инженерами, и позволяет проводить аудит решений, принимаемых моделью.

Настоящая работа не претендует на то, чтобы внести вклад непосредственно в область методов статистического выявления дрейфа. Её вклад лежит в архитектурной плоскости: продемонстрировать, как классические тесты, пороговые политики, автоматизированное переобучение, реестры моделей и отслеживание происхождения данных можно объединить в компактную, целостную и воспроизводимую систему ; одновременно понятную и расширяемую.

3. Архитектура платформы

3.1 Общая схема

Платформа состоит из пяти взаимосвязанных модулей :

1. Модуль обнаружения дрейфа (drift_detection) : статистические тесты, базовый профиль, отчет;
2. Модуль оркестровки : сценарии загрузки данных, политика пороговых значений, действия (ведение журнала, оповещения веб-перехватчика , переобучение), хранилище выполнения;
3. Модуль переобучения : сборка объединенной выборки, обучение конвейера sklearn , оценка, сохранение артефакта , регистрация;
4. Модуль жизненного цикла : реестр экспериментов и моделей, этапы, производственный индекс;
5. Модуль управления данными (data_management): версии набора данных, снимки базового профиля, исходные записи о переобучении.

Над этими модулями развернуты два интерфейса : унифицированный REST API, построенный на FastAPI , и SPA-панель управления, построенная на React с использованием MUI-компонентов. Платформа упакована в Docker. Она включает API , планировщик и фронтенд-сервисы, что позволяет воспроизводить развертывание на любом хосте.

3.2 Обнаружение дрейфа: базовый профиль и отчет

Профиль базовой линии фиксирует структуру обучающей выборки, используемой для построения интервалов PSI. Для каждой числовой характеристики границы квантилей рассчитываются на этапе инициализации (по умолчанию — 10 границ квантилей) , которые затем фиксируются и сохраняются в формате JSON. Это гарантирует, что последующие сравнения «текущего значения с эталонным» будут выполняться в соответствии с одним разделом и будут согласованными между различными запусками мониторинга.

отчет о дрейфе (DriftReport) на основе предопределенного профиля. Для числовых характеристик рассчитывается PSI на фиксированных интервалах, а также вычисляется двухвыборочный тест Колмогорова-Смирнова с двусторонним p-значением.
Для категориальных характеристик рассчитывается PSI на основе долей категорий, а критерий χ^2 вычисляется на таблице сопряженности, где редкие категории объединены в категорию «другие » (по умолчанию минимальное ожидаемое значение равно 5) для удовлетворения условий применимости теста. Каждому знаку присваивается интерпретация:

● psi_band (стабильный, умеренный, высокий): PSI < 0,10 - стабильный;

$0{,}10 \le PSI < 0{,}25$ – умеренный; $PSI \ge 0{,}25$ – высокий;

● ks_interpretation (нет сигнала , значимо) при $\alpha = 0{,}05$;

● chi 2_interpretation (no_signal , significant) с $\alpha = 0{,}05$.

Сводка отчёта содержит агрегированные данные счетчиков : n_features_high_psi , n_numeric_ks_significant , n_categorical_chi2_significant, а также количество строк и знаков V ссылки и текущее значение. Отчет сериализуется в JSON и сохраняется вместе с запуском оркестрации в SQLite.

3.3 Политика и действия

Решение о необходимости вмешательства принимается политикой пороговых значений DriftThresholdPolicy. По умолчанию политика срабатывает, если выполняется хотя бы одно из следующих условий (строгое превышение):

● n_features_high_psi > 0 (ни один параметр не должен попадать в диапазон «высокого» давления);

● n_numeric_ks_significant > 2 (допускается до двух значимых KS в качестве ожидаемого статистического шума при множественном тестировании);

● n_categorical_chi2_significant > 3 (аналогично для χ^2).

Пороговые значения оправданы двумя соображениями: во-первых, использованием широко цитируемых «рабочих» диапазонов PSI (0,10/0,25) из практики моделирования рисков [3]; во-вторых, компенсацией ожидаемого числа ложных срабатываний на уровне значимости

$\alpha = 0,05$: при наличии N признаков и отсутствии дрейфа математическое ожидание числа «значимых» тестов равно αN, что оправдывает использование счетчиков вместо реакции на один значимый результат.

выполняется цепочка действий (конвейер действий):

1. LogAction : структурированное ведение журнала причин срабатывания триггеров;

2. WebhookAlertAction: опциональный POST-запрос к внешнему веб-хуку (интеграция со Slack). PagerDuty и т. д.);

3. RetrainPipelineAction: запуск пайплайна переобучения на объединенной выборке (см. раздел 3.4).

Каждый запуск оркестровки , успешный или неудачный , сохраняется в SQLite (orchestration.db) в виде записи, содержащей идентификатор, метку времени, сценарий, флаг запуска, список причин и сводку отчета о расхождениях в формате JSON. Это обеспечивает возможность аудита истории мониторинга.

3.4 Переобучение и реестр моделей

Пайплайн переобучения (RetrainPipelineAction) объединяет

подвыборки как « reference » и « current », обучает конвейер scikit-learn и оценивает его на отложенной тестовой выборке. Для набора данных (Adult) используется модель `HistGradientBoostingClassifier` (с параметрами `max_depth=6`, `learning_rate=0.08`, `max_iter=200` и `random_state=42`); числовые признаки передаются «напрямую», а категориальные подвергаются One-Hot-кодированию. Для набора данных *Fraud* используется та же модель, но без категориальных признаков (все 29 признаков являются числовыми). Разбиение обучение/тест (0.8/0.2) выполняется со стратификацией по классам. В качестве основной метрики качества выбрана Macro-F1; также вычисляются точность (accuracy), `precision_macro` и `recall_macro`.

После обучения артефакт сохраняется в папке artifacts/models/ model_v. N.joblib , и версия N увеличивается за счет реестра моделей (ModelRegistry , файл registry.json). После этого выполняется правило повышения: если макрос F1 новой модели не хуже, чем у текущего «чемпиона» (champion.json), новая модель автоматически повышается.

Продвигается и становится текущей моделью, находящейся в производстве. В противном случае модель остается на стадии (development).

3.5 Жизненный цикл модели

Модуль жизненного цикла (lifecycle) поддерживает две сущности в базе данных жизненного цикла (lifecycle.db) . :

● Эксперименты (experiments): название, параметры обучения, метрики, сценарий, необязательный SHA-хеш Git, примечания;

● Версии модели (model_versions) : номер версии, ссылка на эксперимент, путь к артефакту , этап, метрики.

Поддерживаются этапы разработки - тестирования - производства - архивирования с ограниченным количеством переходов: версия, назначенная для производства, автоматически переводит предыдущую «производственную» модель в архив .

Указатель на производственную версию хранится в отдельной записи настроек (production_model_id) , к которой обращается API вывода для загрузки текущей модели.

После каждого переобучения выполняется функция sync_from_retrain : в файле lifecycle.db атомарно создаются запись об эксперименте и запись о версии модели ; если была создана новая модель, обновляется указатель на производственную модель.

3.6 Управление данными и провенанс

Модуль управления данными (data_management) обеспечивает отслеживаемость обучающих и мониторинговых данных. Версия набора данных (dataset_versions) определяется его содержимым: для DataFrame, Хэш SHA-256 вычисляется на основе нормализованной сериализации, что обеспечивает дедупликацию одинаковые байты дают одинаковый идентификатор. Снимки базового профиля (baseline_snapshots) записывают путь и хэш файла baseline_profile.json , а также краткое описание распределения

Основные характеристики. Таблица training_provenance связывает конкретное переобучение с версией обучающего набора данных, снимком базового профиля, идентификатором эксперимента в жизненном цикле и номером версии модели.

3.7 Единый API и дашборд

REST API (FastAPI) предоставляет сквозной интерфейс для всех компонентов:

● GET / api / overview - агрегированные KPI и данные о последнем запуске;

● GET / api /orchestration/ runs ?limit =N ; history launches ;

● POST / api /orchestration/ check-once ?scenario =... ; начать отклонение проверки;

● POST / api /retraining/run ; запуск переобучения вручную ;

● GET / api /lifecycle/ models| experiments , POST / api /lifecycle/

promote , GET / api /lifecycle/production - registry and stages;

● GET / api / data/ datasets|baselines|provenance - provenance ;

● POST / api / inference / predict — вывод результатов на основе текущей модели.

Панель управления (React + MUI) состоит из пяти страниц: Обзор (KPI, счетчики отклонений, запуск валидации), Рабочие процессы (история запусков, триггер валидации), Модели (реестр, метрики, прогресс этапов), Данные (версии наборов данных и происхождение) и Вывод (JSON-форма для валидации текущей модели). Панель управления использует только публичный REST API и может использоваться независимо.

4. Экспериментальная часть

4.1 Наборы данных

База данных Adult Census Income (UCI) содержит демографические характеристики и целевую переменную «доход > 50 тыс.». Полный набор включает 48 842 записи; после стандартного кодирования выделяются шесть числовых признаков (возраст, fnlwgt , education.num , capital.gain ,

capital.loss , hours.per.week) и восемь категориальных (workclass , education, marital.status , occupation, relationship, race, sex, native.country). Этот набор используется для демонстрации категориального дрейфа (например, по роду занятий) и смешанного случая.

Kaggle Credit Card Fraud содержится 284 807 транзакций, 492 из которых являются мошенническими (доля положительного класса ≈ 0,17%). Признаки V1…V28 были получены с использованием преобразования PCA для анонимизации; Amount это сумма транзакции, а Time это порядковое время. Для экспериментов по временному дрейфу набор данных был разделен по времени на три окна равной длины D1, D2 и D3, что соответствует последовательности «прошлое – будущее» в транзакционных системах и обеспечивает реалистичный сигнал дрейфа.

4.2 Сценарии мониторинга

В ходе экспериментов были рассмотрены шесть сценариев:

- random_holdout : эталонный и текущий сигналы получаются путем случайного разделения группы Adult (test_size = 0.3, random_state = 42); ожидается минимальный/тихий дрейф сигнала;
- age_shift : текущий набор данных состоит из строк "Взрослые" с возрастом ≥ 40 лет; ожидается выраженное изменение числовых и категориальных характеристик;

- incoming_csv : текущий поток считывается из внешнего CSV-файла, что имитирует фактический поток входящих данных;
- fraud_d1_vs_d2 / fraud_d2_vs_d3 / fraud_d1_vs_d3 : сравнение временных окон мошенничества, ожидается монотонное увеличение сигнала дрейфа с увеличением временного расстояния .

Для каждого сценария устанавливается базовый профиль: для кейса Adult на основе эталонного значения из случайной выборки (random_holdout) ; для кейса (Fraud), на основе окна D1. Это отражает практический сценарий, в котором базовый профиль « замораживается» на этапе обучения и не пересчитывается при каждом запуске.

4.3 Условия воспроизводимости

Все эксперименты проводились с параметром random_state = 42 для разбиения данных и модели. Обучение и вывод результатов выполнялись как локально (Python 3.11, sciki -learn 1.4, FastAPI 0.110), так и в контейнере Docker. Compose (API, планировщик и фронтенд-сервисы) обеспечивает надежную воспроизводимость. SQLite используется для хранения данных (orchestration.db, lifecycle.db, data_management.db) в общем томе ./artifacts.

5. Результаты

5.1 Сценарий random_holdout (Adult)

При «спокойном» случайном разбиении суммарные счетчики дрейфа остаются в пределах пороговых значений: PSI всех признаков находится в стабильном диапазоне (<0,10), а отдельные тесты Колмогорова-Смирнова и χ^2 могут давать изолированные значимые результаты (обычно не более 1-2 из 6 числовых и не более 1 из 8 категориальных). Политика не срабатывает, и автоматическое переобучение не запускается . Это соответствует ожидаемому статистическому шуму при $\alpha = 0{,}05$ и подтверждает, что

пороговые значения политики откалиброваны достаточно хорошо : система не реагирует на случайные флуктуации.

5.2 Сценарий age_shift (Adult)

При построении current из рядов Adult с age ≥ 40 наблюдается выраженный сдвиг по признаку age (PSI существенно выше 0.25), смещения проявляются также в связанных признаках (education.num,hours.per.week, marital.status). Счётчик n_features_high_psi переходит через порог 0, что запускает пороговую политику,WebhookAlertAction (если настроен) и RetrainPipelineAction.

Пайплайн переобучения строит новую версию модели; в зависимости от «совместимости» объединённой выборки с исходной задачей macro-F1 может оставаться на уровне чемпиона или незначительно снижаться ; обе ситуации корректно обрабатываются правилом продвижения: новая версия фиксируется в реестре (development), но указатель production не изменяется, если метрика хуже. Тем самым платформа демонстрирует важное свойство: срабатывание триггера переобучения не означает автоматическое развёртывание худшей модели.

5.3 Темпоральные сценарии (Fraud)

При сравнении окон D 1, D 2, D 3 наблюдается монотонное увеличение.

На сравнении окон D1-D2-D3 характерно монотонное усиление сигнала дрейфа. Для численных признаков часть V-признаков попадает в диапазоны moderate и high по PSI, KS-тесты фиксируют значимые различия.

Триггер политики срабатывает, когда превышен порог n_features_high_psi = 0 .

Переобучение выполняется на объединенной размеченной выборке D1 + D2 (сценарий fraud_retrain_d1_d2 на стороне API), что соответствует реалистичному сценарию «обучение на новых подтвержденных транзакциях».

Полученные макрометрики F1 сохраняются в файле lifecycle.db и отображаются в разделе «Модели» на панели мониторинга.

5.4 Операционные метрики и интерфейс

Эндпоинт GET /api/ops/stats агрегирует историю запусков в operational KPI : total_runs , triggered_runs , ok_runs , trigger_rate.

Например, на контрольном прогоне из 10 запусков при чередовании random_holdout/age_shift наблюдалась ожидаемая разница: на random_holdout политика не срабатывает, на age_shift срабатывает стабильно.

На странице Overview отображаются четыре KPI (Drift score - средний PSI, Workflow Runs, Model Versions, Production Row), а также столбчатая диаграмма счётчиков (High PSI, KS Significant, Chi² Significant) последнего запуска.

Таблица. Качество модели после переобучения (стратифицированный holdout 20%)

Сценарий	accuracy	macro-F1	precision_macro	recall_macro	n_train	n_test	Продвинуто

							в production
Random_holdout (Adult)	0.866	0.819	0.823	0.815	39 074	9 768	**Да** (= чемпион)
Age_shift (Adult)	0.866	0.818	0.821	0.815	43 075	10 769	**Нет** (f1_macro_below_champion)
Fraud_retrain_d1_d2	0.9994	0.917	0.953	0.886	151 898	37 974	**Да**

Правило продвижения: новая модель становится production только если macro-F1 ≥ macro-F1 чемпиона.

6. Обсуждение

Репрезентативность тестов. PSI, KS и χ^2 это классические инструменты с известными ограничениями. PSI чувствителен к выбору интервалов и наличию редких категорий; KS предполагает непрерывные распределения и может давать ложноположительные результаты при очень больших выборках; χ^2 требует минимальных ожидаемых частот, что решается на платформе путем объединения редких категорий в категорию (other)**.**

В совокупности три теста дают разнообразный и поддающийся интерпретации сигнал, пригодный для промышленной автоматизации. Более сложные методы, в частности, детекторы дрейфа, основанные на

классификаторах опорного/токового сигнала, остаются перспективным направлением для дальнейшего развития.

Выбор пороговых значений. Пороговые значения политики являются эвристическими. В реальных условиях их следует калибровать с использованием исторических данных за стабильный период: учитывать естественный уровень шума счетчиков и устанавливать пороговые значения выше него. Альтернативным вариантом является использование процедур контроля частоты ложных срабатываний (например Benjamini–Hochberg) для p-значения, что снижает зависимость от количества признаков.

Автоматическое переобучение. Предложенная политика «переобучение по триггеру и продвижение только в том случае, если макрос - F1 не ухудшается» является осторожной и обладает важным свойством: ни при каких обстоятельствах производственная модель не ухудшается по отношению к выбранной метрике. Ограничением подхода является эффект Симпсона на несбалансированных данных: Macro-F1 может не ухудшаться, но производительность на редком классе может пострадать. В качестве расширения платформа поддерживает указание произвольной основной метрики в конфигурации переобучения.

Сравнение с существующими инструментами. Платформа не предназначена для замены промышленных систем (MLflow , Airflow, Evidently AI). Ее цель это обеспечить согласованность и продемонстрировать логику интеграции на минимальном стеке. Для крупномасштабных коммерческих MLOps естественным путем миграции является замена

Отдельные модули: оркестровка Airflow/Prefect, жизненный цикл MLflow Registry, управление данными DVC/ LakeFS ; при этом бизнес-логика политик и действий остается на платформе в качестве слоя композиции.

Ограничения работы. В текущей реализации обучение ограничено двумя табличными наборами данных и одним семейством моделей (gradient boosting).

Для текстовых и графических данных модуль обнаружения дрейфа (эмбеддинг-дрейф) и соответствующие конвейеры обучения необходимо адаптировать. Кроме того, политика представлена в виде счетчиков с фиксированными пороговыми значениями; перспективным направлением является обучаемая политика, учитывающая историческую динамику запусков.

7. Заключение

В проекте предлагается унифицированная платформа для мониторинга и поддержки моделей машинного обучения, реализующая сквозной процесс: обнаружение дрейфа (PSI , KS, χ^2), политика пороговых значений, контролируемые действия (логирование, оповещения, переобучение), версионирование артефактов, отслеживание жизненного цикла, происхождение данных и экспериментов. Платформа полностью поддерживает REST.

API на FastAPI и панель управления на React, упакованы в Docker Compose и используют SQLite в качестве общего хранилища.

Экспериментальное тестирование на наборах данных Adult Census Income и Kaggle Credit Card Fraud в шести сценариях (random_holdout , age_shift , incoming_csv , fraud_d1_vs_d2 / d2_vs_d3 / d1_vs_d3) подтверждает корректную работу политики: в «спокойных» сценариях триггер не

срабатывает, но в сценариях с явным сдвигом он срабатывает, что приводит к контролируемому переобучению и подтвержденной записи в реестре.

Внедрение новой версии в производство осуществляется только в том случае, если основной показатель (macro-F1) не ухудшается, что обеспечивает безопасность обновления модели.

Направления будущих исследований:

(1) обучаемые стратегии, учитывающие динамику метрик и бизнес-издержки ошибок; (2) методы обнаружения дрейфа для нетабличных данных (дрейф встраивания в тексты и изображения); (3) интеграция с промышленными системами оркестровки и реестрами моделей без изменения бизнес-логики платформы; (4) расширение политических действий (поэтапное развертывание, подтверждение с участием человека).

СПИСОК ЛИТЕРАТУРЫ

1. Sculley D., Holt G., Golovin D. et al. Hidden Technical Debt in Machine Learning Systems // Proc. NeurIPS. — 2015. — P. 2503–2511.

2. Huyen C. Designing Machine Learning Systems. O'Reilly Media, 2022. — 386 p.

3. Siddiqi N. Credit Risk Scorecards: Developing and Implementing Intelligent Credit Scoring. — Wiley, 2005. — 196 p.

4. Massey F. J. The Kolmogorov–Smirnov Test for Goodness of Fit // Journal of the American Statistical Association. 1951. ; Vol. 46, № 253. — P. 68–78.

5. Pearson K. On the Criterion that a Given System of Deviations from the Probable in the Case of a Correlated System of

Variables is Such That It Can Be Reasonably Supposed to Have Arisen from Random Sampling // Philosophical Magazine. —

6. — Vol. 50, № 302. — P. 157–175.

7. Rabanser S., Günnemann S., Lipton Z. C. Failing Loudly: An Empirical Study of Methods for Detecting Dataset Shift // Proc. NeurIPS. — 2019. ; P. 1396–1408.

8. Apache Software Foundation. Apache Airflow: An Open-source Platform to Programmatically Author, Schedule and Monitor Workflows. — 2024. — URL: https://airflow.apache.org.

9. Prefect Technologies. Prefect: The New Standard in Dataflow Automation. — 2024. — URL: https://www.prefect.io.

10. Zaharia M., Chen A., Davidson A. et al. Accelerating the Machine Learning Lifecycle with MLflow // IEEE Data Engineering Bulletin. — 2018. — Vol. 41, № 4. — P. 39–45.

11. Amershi S., Begel A., Bird C. et al. Software Engineering for Machine Learning: A Case Study // Proc. ICSE-SEIP. — 2019. — P. 291–300.