edquest

Day 5 Agenda

Git, DVC, Feature Store & MLOps

- Recap
- MLOps lifecycle: code, data, models
- Git basics for ML
- DVC for data/model versioning
- Feast for feature storage and retrieval
- Quiz

MLOps

MLOps (Machine Learning Operations) is the practice of automating and managing the end-to-end lifecycle of machine learning models — from development to deployment and monitoring.

MLOps

Imagine a chef running a restaurant.

- The chef is your data scientist — experimenting with new recipes (ML models) in the kitchen (development environment).
- Once a dish is perfect, they pass it on to the kitchen staff (MLOps engineers) to make sure it gets cooked consistently for every customer — no surprises, no undercooked pasta.
- The kitchen staff then sets up a system that automates how each ingredient is added, how long it cooks, and how it's plated — like deploying the ML model to production.
- Meanwhile, the manager (monitoring system) is watching customer reactions — are they enjoying the food, or is someone silently crying into their risotto?
- If something's off — say the pasta keeps arriving cold — the manager gives feedback, and the recipe or process gets fixed.

edquest

MLOps Levels

MLOps level 0: Manual process

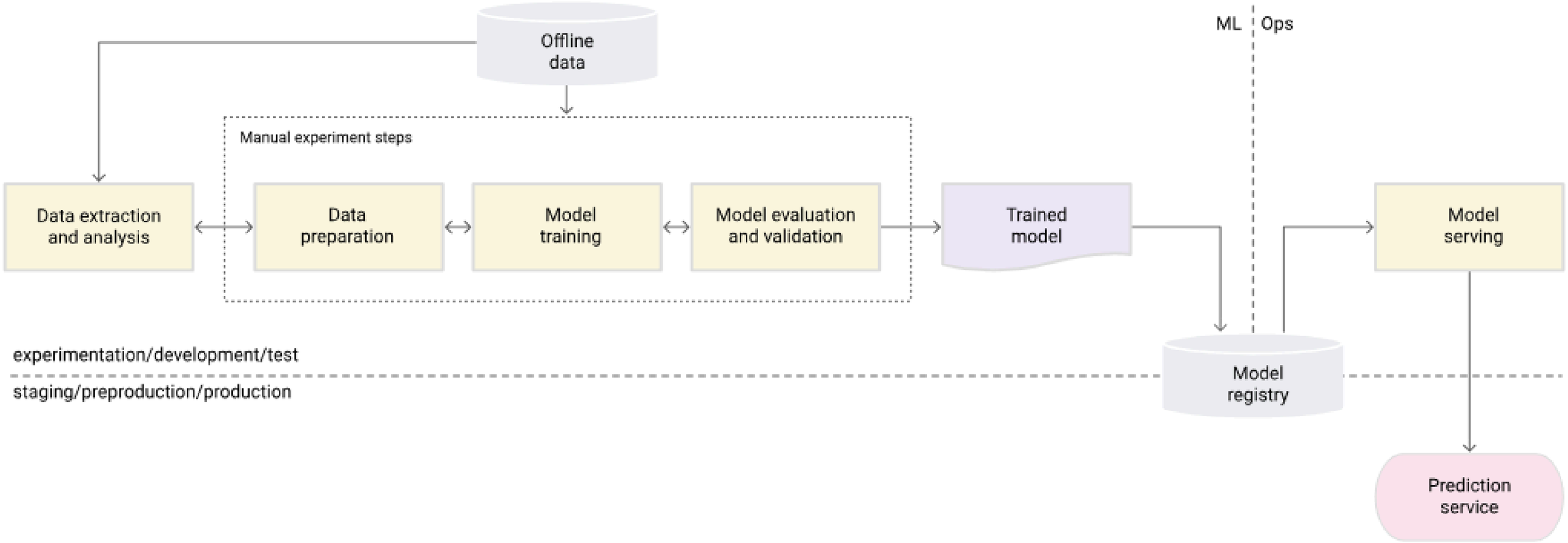
MLOps level 1: ML pipeline automation

MLOps level 2: CI/CD pipeline automation

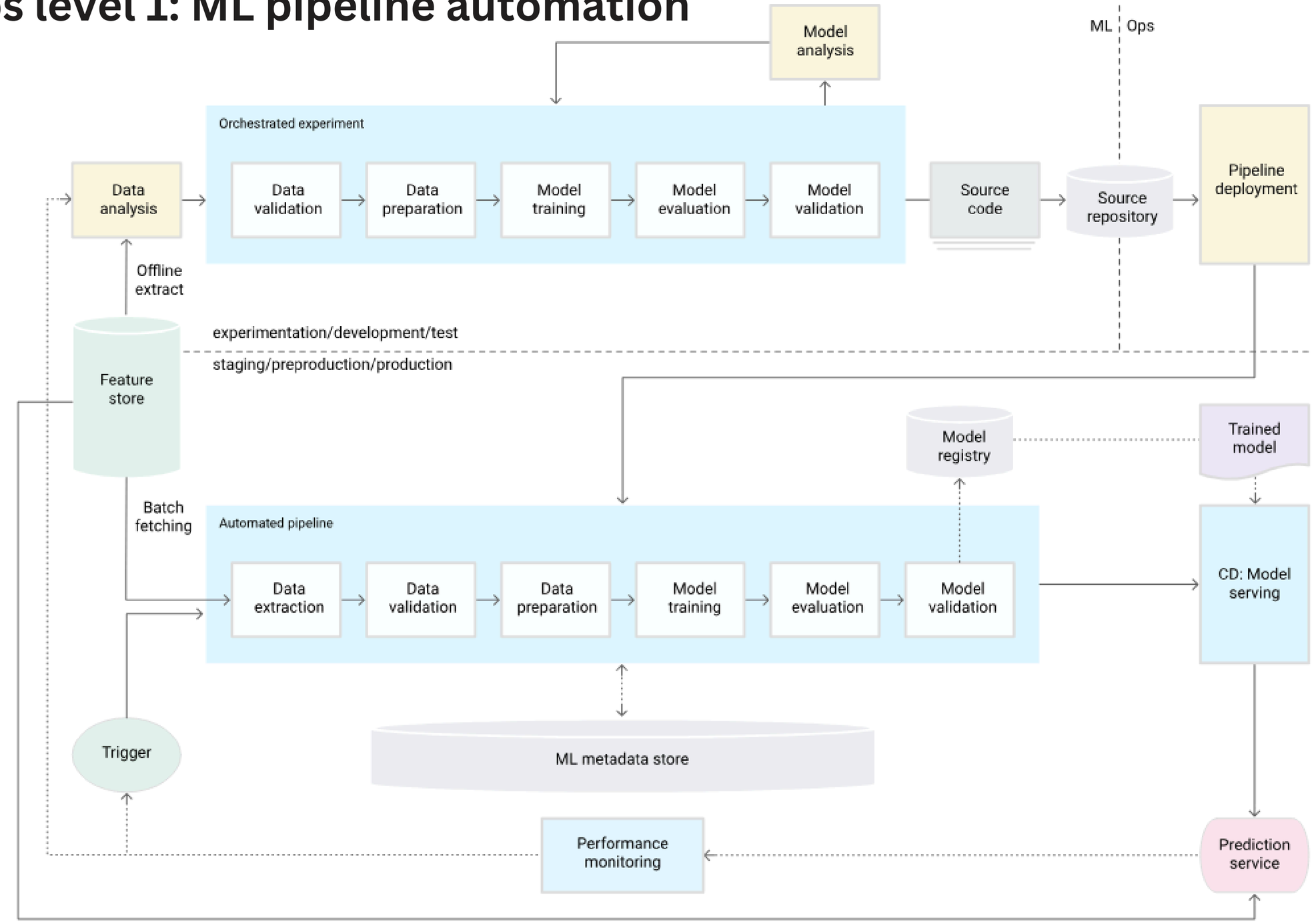
edquest

MLOps level 0: Manual process

edquest

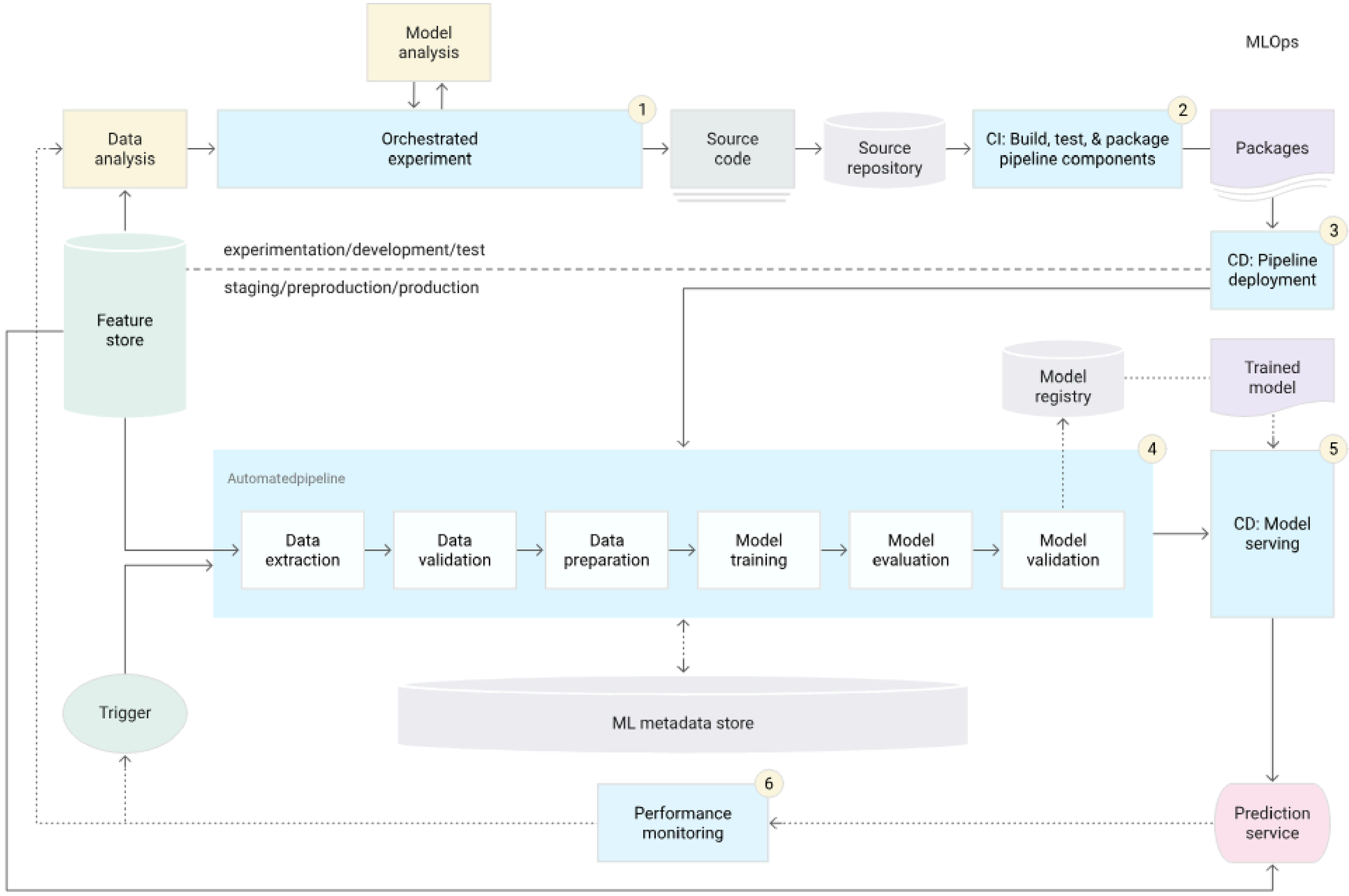


MLOps level 1: ML pipeline automation



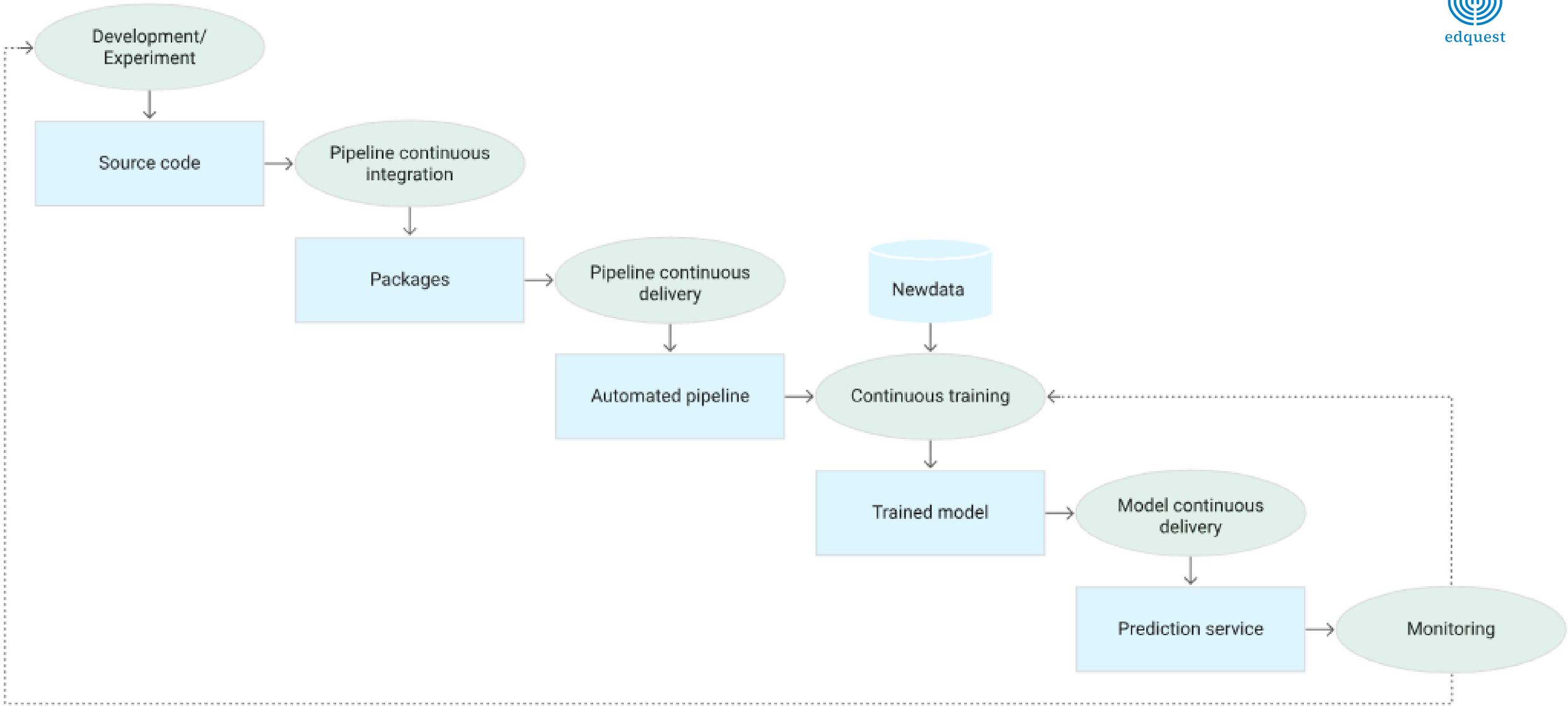
edquest

MLOps level 2: CI/CD pipeline automation



Stages of the ML CI/CD automation pipeline:

edquest



Stages of the ML CI/CD automation pipeline (With Example):

edquest

Stage	Description	Example
Data Ingestion & Validation	Collect and validate raw input data	Pull customer logs from S3 and check for missing values
Model Training	Train the ML model with labeled data	Train a churn prediction model using user activity data
Model Validation & Testing	Evaluate model performance on test data	Run accuracy, precision, recall checks on validation set
Model Packaging	Bundle model and dependencies for deployment	Package model in a Docker container for reproducible deployment
Model Deployment	Deploy model into a production environment	Expose model via a Docker container end point
Monitoring & Feedback Loop	Track model performance post-deployment	Detect model drift and retrain if predictions start degrading

edquest

Git

Why Git ?

Git helps manage and track changes in your machine-learning code and project files.

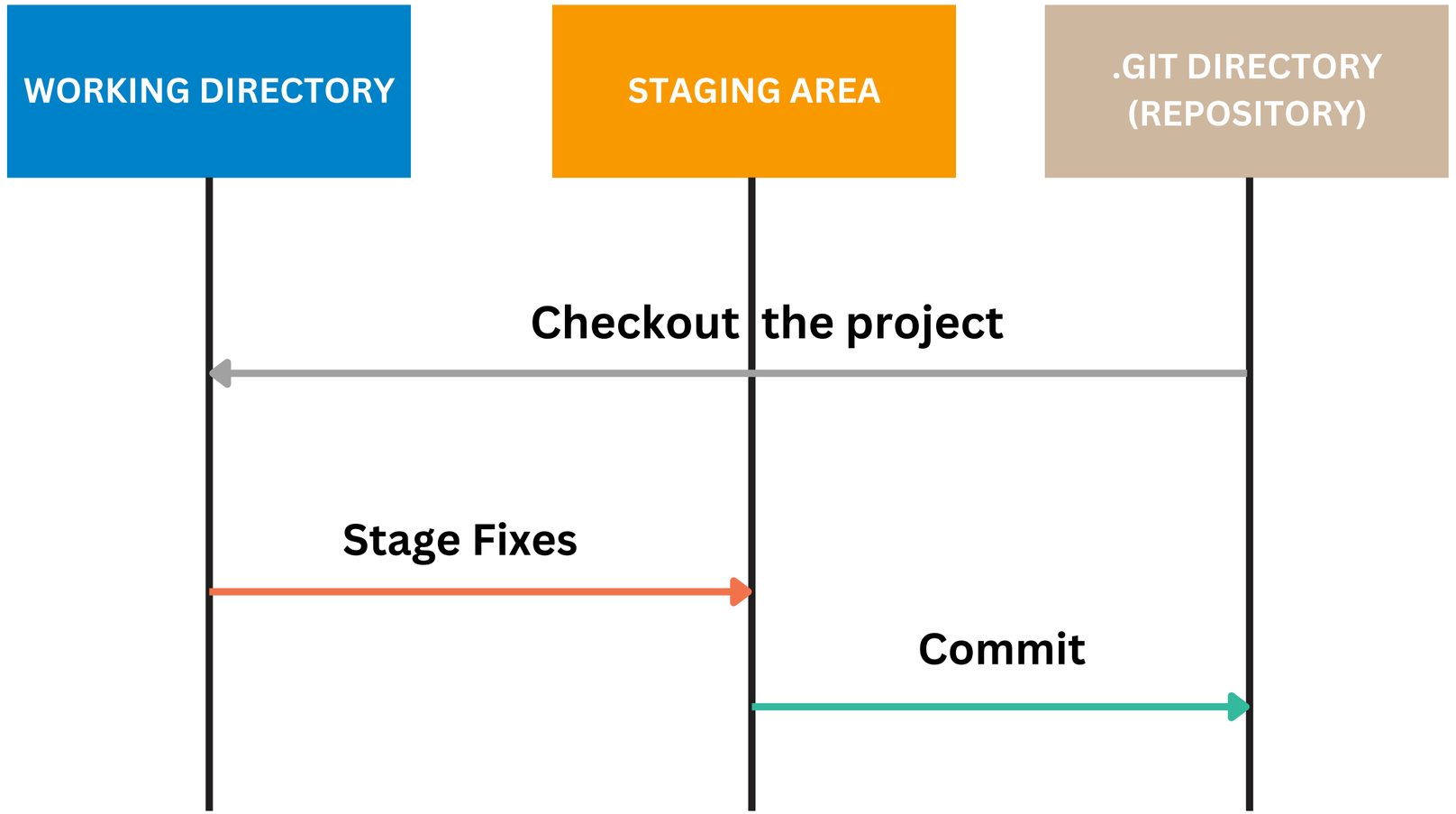
It keeps a complete history of your experiments, scripts, notebooks, and configurations — including who made the change, when, and why (via commit messages).

In ML projects, Git is especially useful to:

- Track different versions of code and scripts
- Collaborate with teams while keeping your work organized
- Revert back to earlier working versions if something breaks

edquest

States in Git



edquest

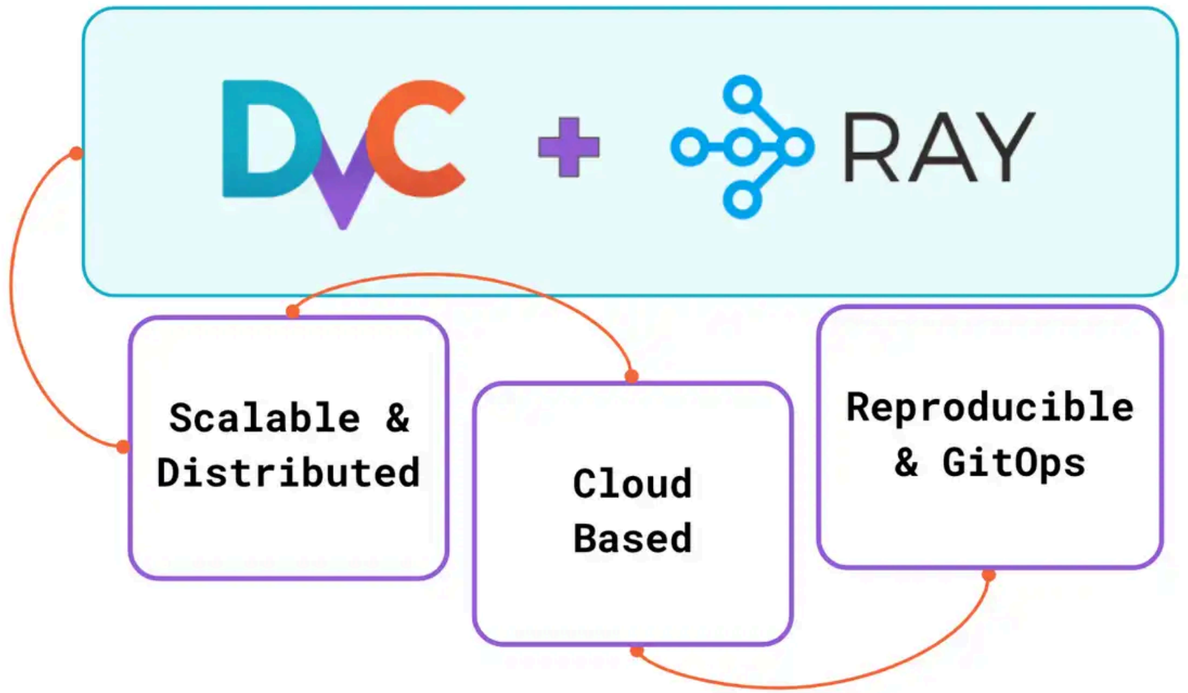
Git Workshop

DVC for Data/Model versioning

Data Version Control

edquest

Part 2



Why DVC

Even with all the success we've seen in machine learning, especially with deep learning and its applications in business, data scientists still lack best practices for organizing their projects and collaborating effectively. This is a critical challenge: while ML algorithms and methods are no longer tribal knowledge, they are still difficult to develop, reuse, and manage.

edquest

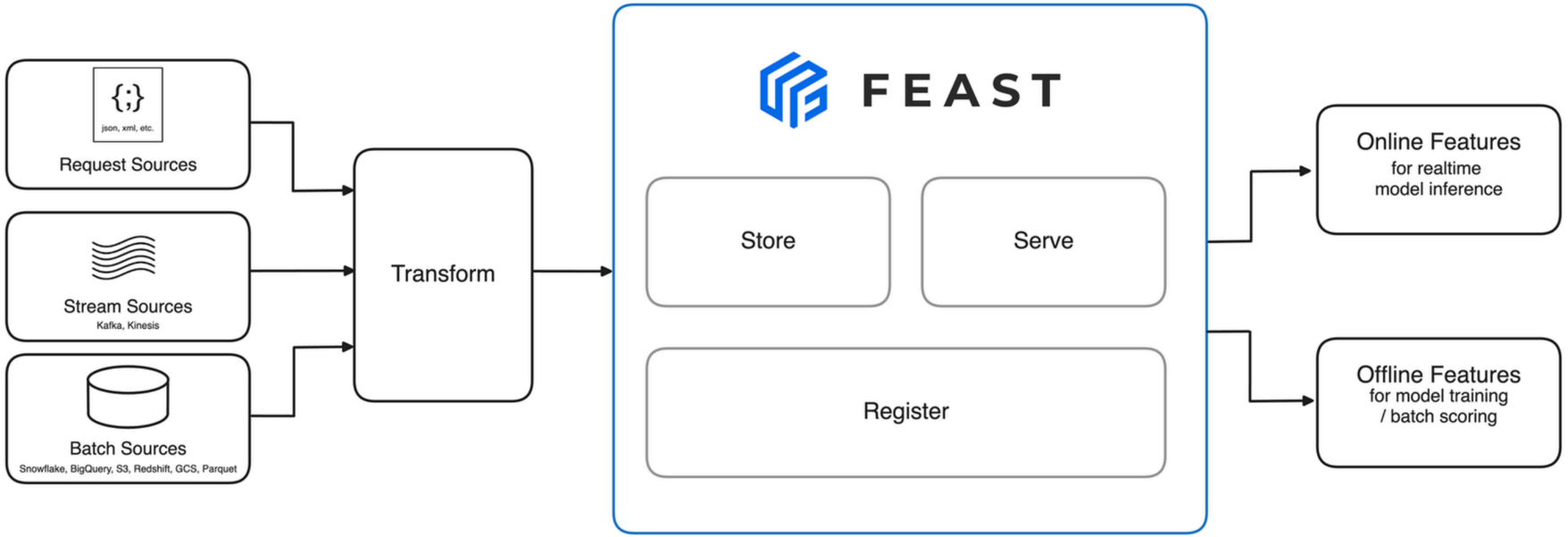
DVC Use Cases

- track and save data and machine learning models the same way you capture code;
- create and switch between versions of data and ML models easily;
- understand how datasets and ML artifacts were built in the first place;
- compare model metrics among experiments;
- adopt engineering tools and best practices in data science projects;

edquest

Feast for feature store and retrieval

edquest



Feast

Feast is an open-source feature store for managing and serving ML features in batch and real-time.

edquest

For Data Scientists

- Easily define, store, and retrieve features for training & inference.
- Focus on feature engineering, not infrastructure.

For MLOps Engineers

- Connects to existing infrastructure (DBs, microservices, orchestration tools).
- Ships features to production via a simple SDK.

For Data Engineers

- Centralized catalog for feature definitions.
- Supports both online and offline data stores with unified access.

For AI Engineers

- Enables scalable pipelines and richer data integration.
- Helps fine-tune and optimize model performance.

FEAST Use Cases

Feast Use Case: NLP / RAG / Information Retrieval

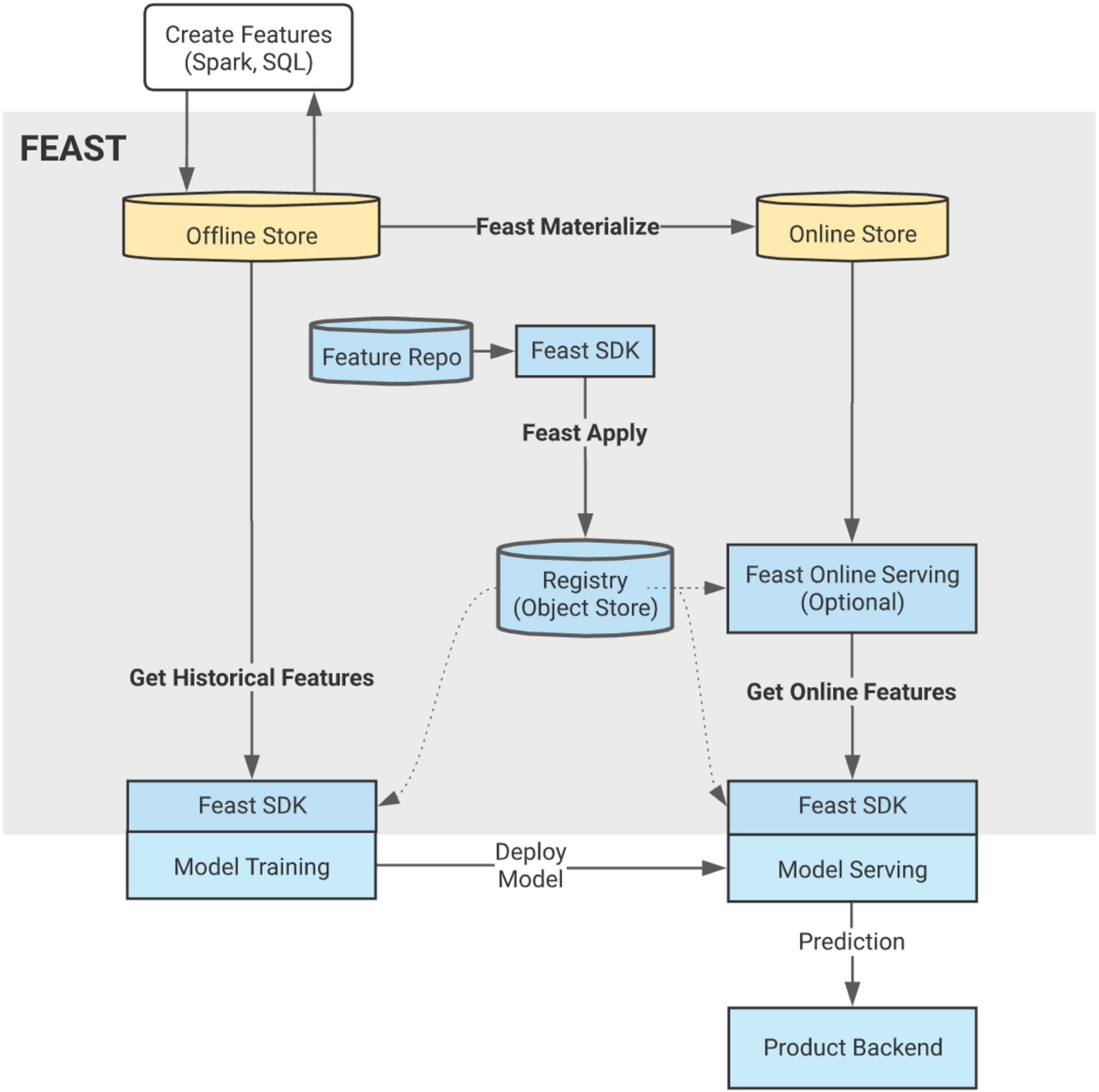
- Vector Storage: Store and index text embeddings for similarity search
- Document Metadata: Link embeddings with metadata for contextual retrieval
- Scaling Retrieval: Serve embeddings with low latency in real-time applications
- Versioning: Track changes in embedding models and document sets

Feast Use Case: Recommendation Engines

- Feature Management: Store user behavior, item features, and interaction data
- Low-Latency Serving: Power real-time, dynamic recommendations
- Point-in-Time Correctness: Ensure consistent training/serving data, prevent leakage
- Feature Reuse: Share feature definitions across multiple recommendation models

edquest

FEAST Architecture



edquest

Why Feast use cases matter ?

Feast solves one of the biggest pain points in machine learning: managing and serving feature data reliably and efficiently.

Whether you're building a search engine, chatbot, or recommendation system, the quality and freshness of your feature data directly impacts model performance. Feast acts as a central hub that ensures your ML models always get the right features at the right time — whether in training or real-time production.

By supporting use cases like NLP/RAG and recommendation engines,
Feast enables:

- Faster experimentation for data scientists
- Cleaner, more maintainable pipelines for engineers
- Low-latency, production-grade performance
- Feature consistency between training and inference
- Reusability and versioning to scale systems over time

edquest

edquest

Quiz

edquest

Q1. In the MLOps lifecycle, the steps code, data, model should be versioned to ensure ___ and reproducibility across teams.

A. collaboration

B. automation

C. simplicity

D. visualization

Explanation: Reproducibility and collaboration are essential in MLOps and are supported through proper versioning.

edquest

Q2. Git is commonly used in ML workflows to track changes in ____ and experiment logic.

A. model outputs

B. training datasets

C. code and scripts

D. hardware dependencies

Explanation: Git excels at version-controlling code, which forms the basis of ML experiments.

edquest

Q3. Git is not ideal for large binary files like datasets or models because it stores changes by ___.

A. overwriting

B. chunking

C. appending

D. deltas

Explanation: Git stores changes as deltas (differences), which is inefficient for large binaries—hence tools like DVC are needed.

edquest

Q4. DVC helps solve the limitations of Git in ML by versioning large data and model files using ___ storage.

A. container

B. remote

C. cloud

D. local

Explanation: DVC links files in Git to versions stored in remote storage (like S3, GCS), enabling scalable versioning.

edquest

Q5. The command dvc add data.csv creates a .dvc file which tracks the ___ of the data file.

A. size

B. content hash

C. name

D. metadata

Explanation: DVC uses a content hash to track changes in large files.

edquest

Q6. When using DVC, dvc push is used to upload data or models to ____ storage so they can be shared across machines.

A. GitHub

B. Local

C. Remote

D. Cache

Explanation: DVC separates code (via Git) and data (via remote storage), making it easy to share across environments.

edquest

Q7. Feast is used in MLOps to store and serve ___ to models in both training and production environments.

A. predictions

B. hyperparameters

C. raw data

D. features

Explanation: Feast (Feature Store) centralizes feature definitions and makes them reusable across ML stages.

edquest

Q8. A key advantage of using a feature store like Feast is that it ensures ___ between training and serving features.

A. duplication

B. consistency

C. compression

D. normalization

Explanation: Feature stores eliminate training-serving skew by maintaining consistent transformations.

edquest

Q9. In Feast, features are often grouped into entities (e.g., user_id, item_id) to support ___ feature retrieval.

A. static

B. versioned

C. point-in-time

D. streaming

Explanation: Point-in-time retrieval ensures the model sees only features that were available at prediction time.

edquest

Q10. A complete MLOps system integrates Git for code, DVC for data/model versioning, and Feast for features, enabling automated and reproducible ___ pipelines.

A. deployment

B. CI/CD

C. ETL

D. ML

Explanation: MLOps pipelines benefit from integrating versioning and data/feature consistency for scalable ML workflows.