



MLOPS 101

BY: CORNELLIUS YUDHA WIJAYA

 **LinkedIn: Cornelius Yudha Wijaya**

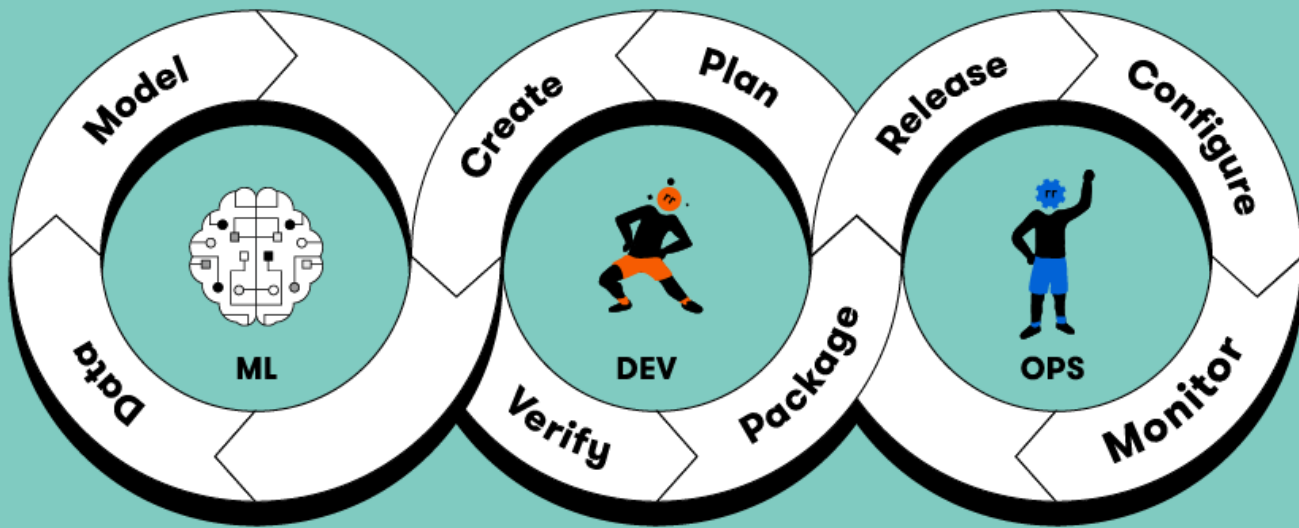
 **Medium: @cornelliusyudhawijaya**



Learning Agenda

- What is MLOps
- What are the component of MLOps
- What were MLOps Principles
- MLOps Example in Companies

What is MLOps?



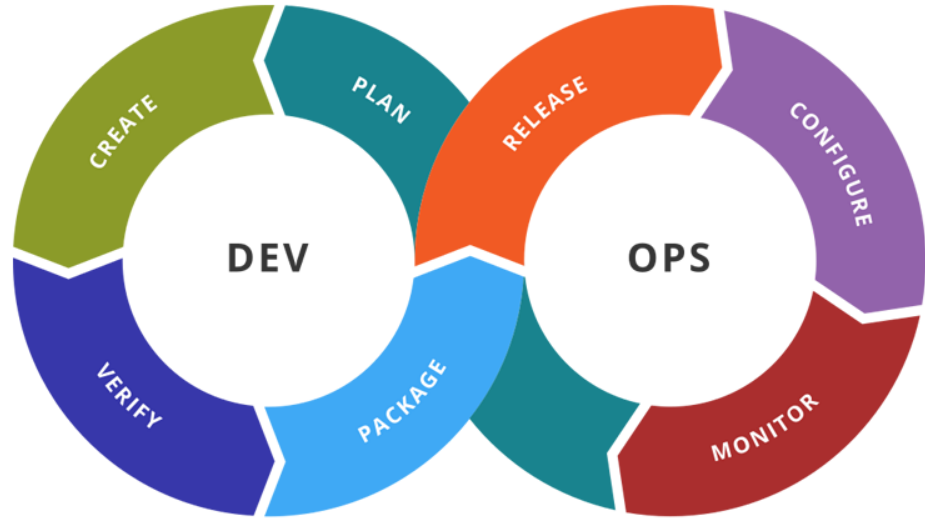
MLOps (machine learning operations) stands for the collection of techniques and tools for the deployment of ML models in production.

It contains the combination of:

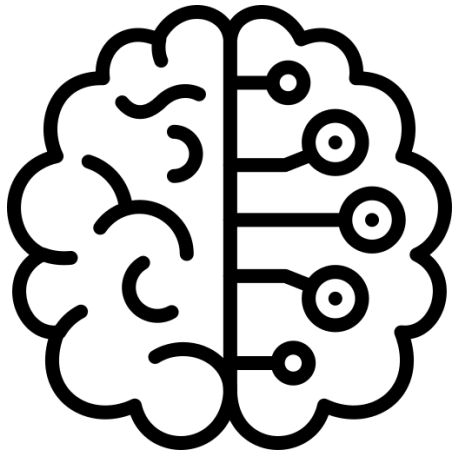
- DevOps
- Machine Learning.

In MLOps, we strive to avoid “*technical debt*” on machine learning applications

DevOps and Machine Learning



Machine Learning



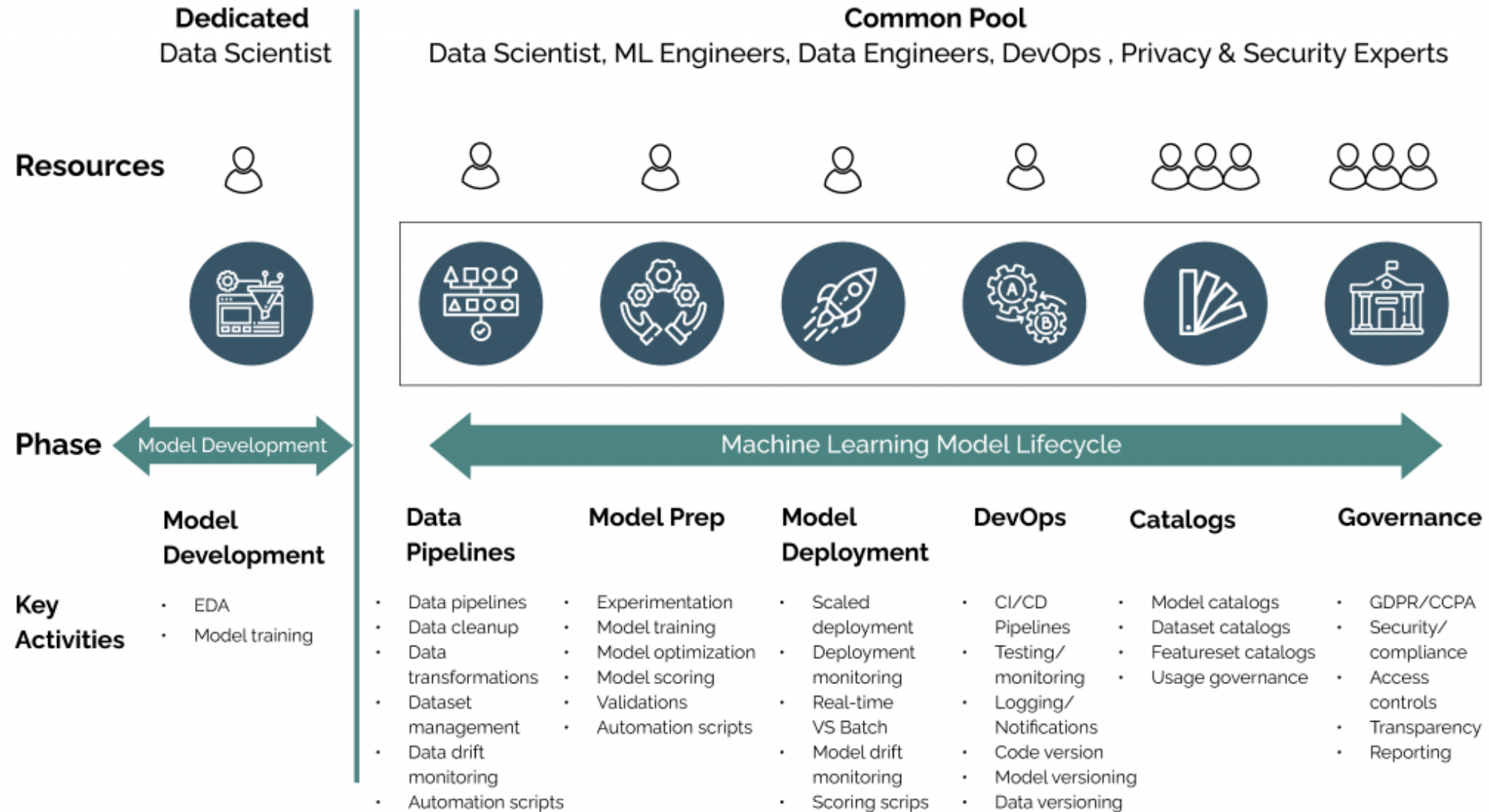
DevOps stands for a set of practices with the main purpose to minimize the needed time for a software release, reducing the gap between software development and operations. Two main principles of DevOps:

- **Continuous Integration (CI):** Principles to integrate code written by developer teams at frequent intervals
- **Continuous Delivery (CD):** Principles to constantly deliver new version of the software under development to be installed for testing, evaluation and then production

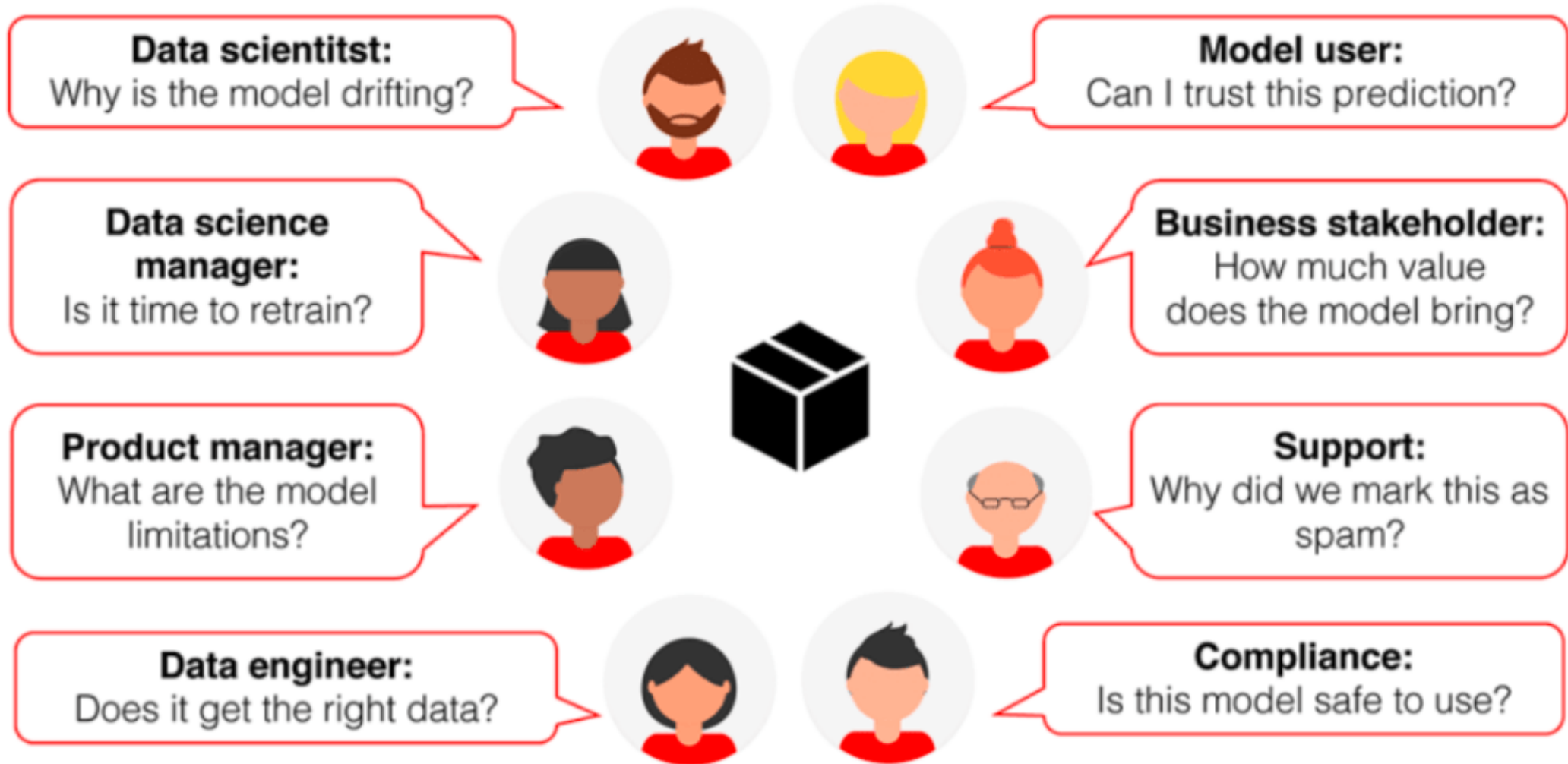
MLOps introduce a new practice, in addition to CI and CD which are:

- **Continuous Training (CT)** that aims to automatically retrain the model where needed.
- **Continuous Monitoring (CM)** concerns with monitoring production data and model performance metrics, which are bound to business metrics.

Machine Learning Operations (MLOps)



Roles in MLOps Project



MLOps Principle: *Iterative-Incremental Development*

The complete MLOps process includes three iterative phases:

- Designing the ML-powered application
- ML Experimentation and Development
- ML Operations

All three phases are interconnected and influence each other. For example, the design decision during the design stage will propagate into the experimentation phase and finally influence the deployment options during the final operations phase.

Designing the ML-powered application

It is phase for *business understanding, data understanding* and *designing the ML-powered software*.

We try to identify :

- Potential user,
- ML solution to solve the problem,
- Assess the further development of the project.

Mostly, we would act within two categories of problems - either increasing the productivity of the user or increasing the interactivity of the application.

ML Experimentation and Development

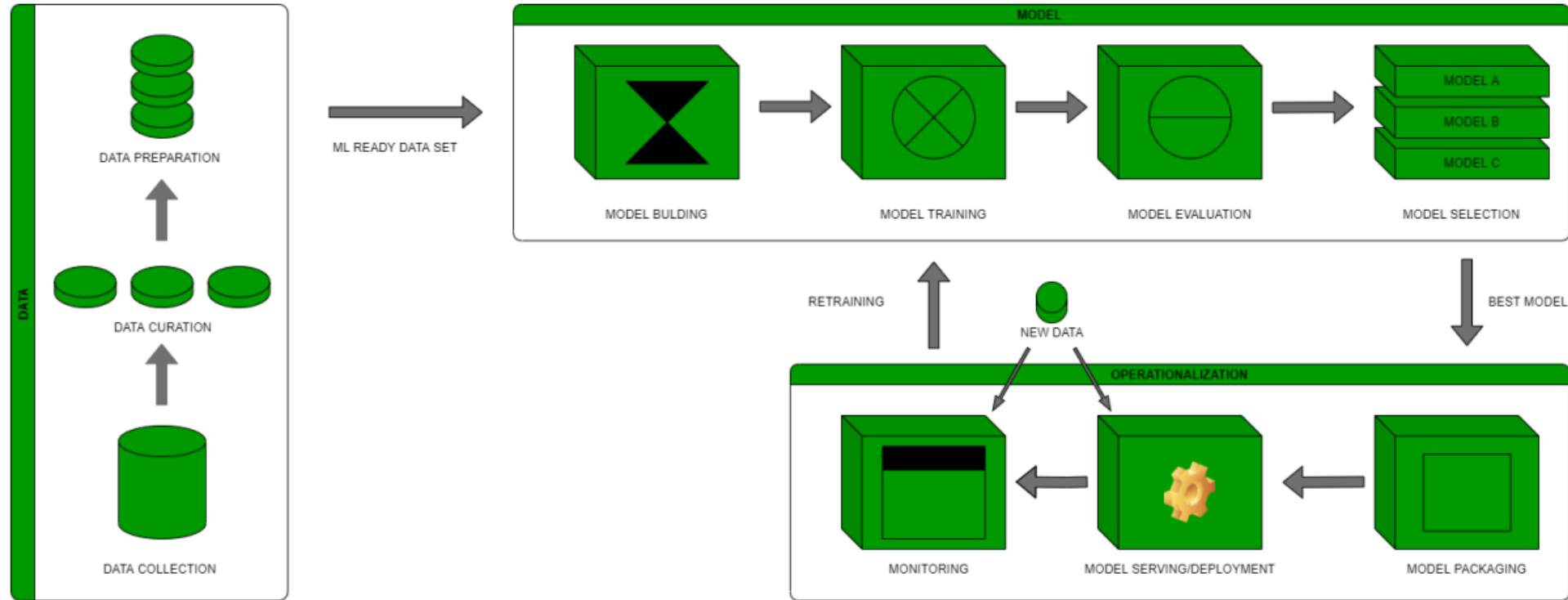
This is the phase of the *Proof-of-Concept for ML Model*. The primary goal in this phase is to deliver a stable quality ML model to run model in the production.

In this phase, we can experiment to identifying or polishing the suitable ML algorithm for our problem, data engineering, and model engineering.

ML Operations

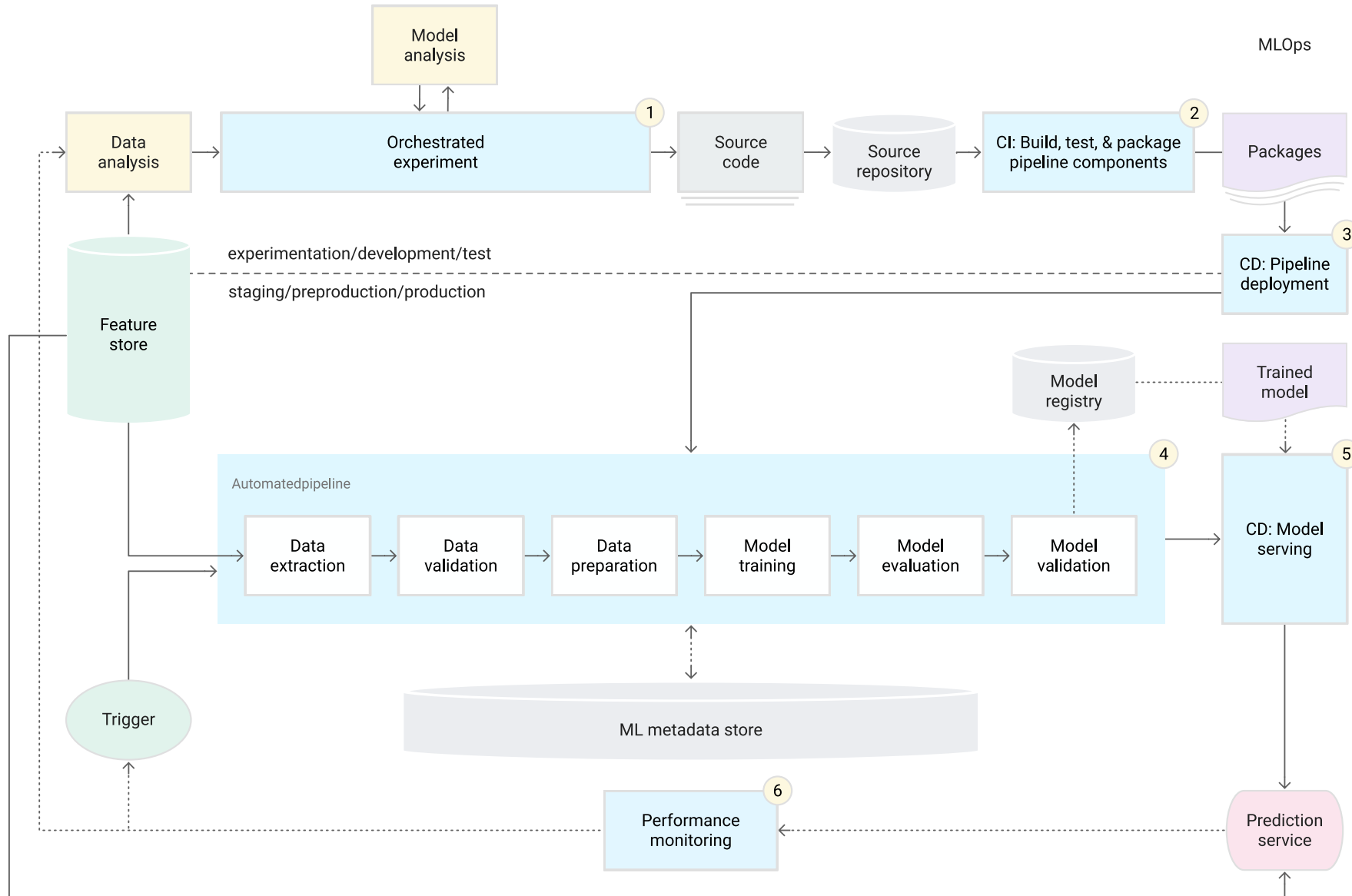
It is the phase is to deliver the previously developed ML model in production by using established DevOps practices such as testing, versioning, continuous delivery, and monitoring.

MLOps Pipeline



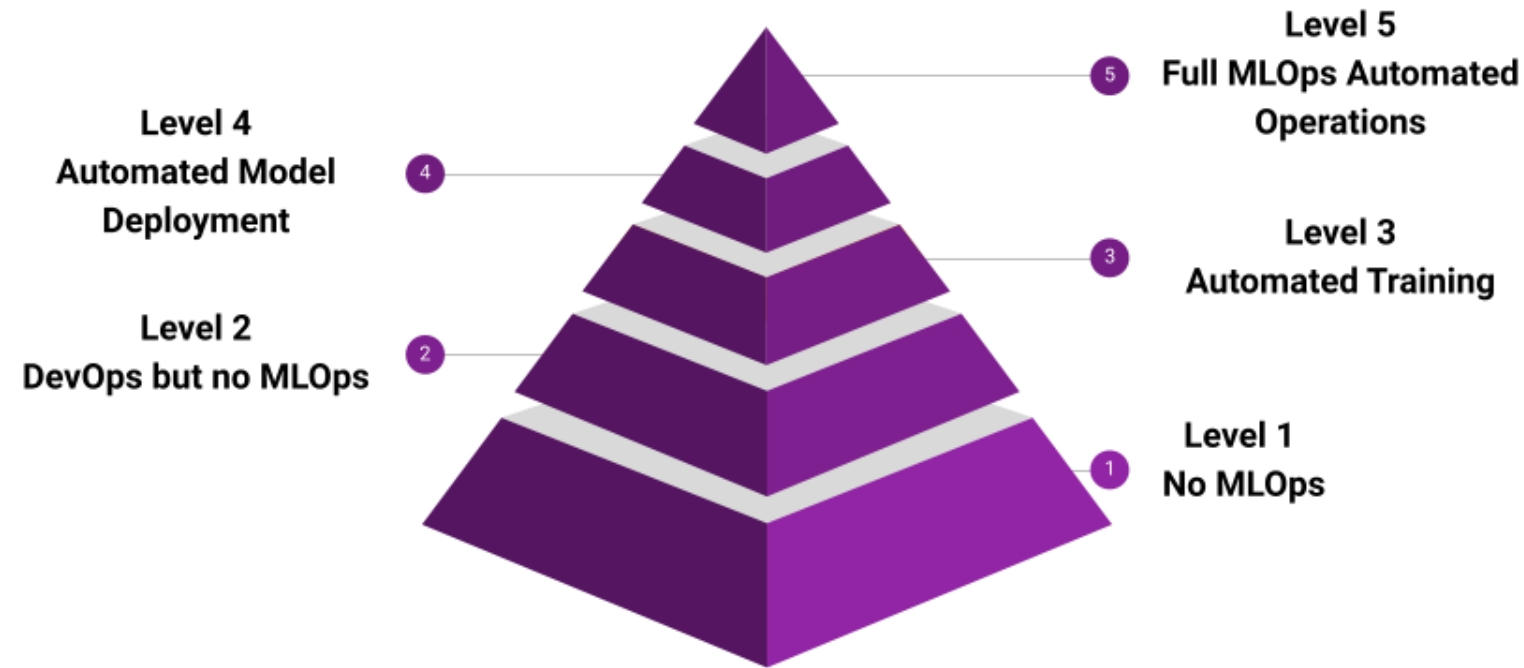
MLOps could be described as a software engineering approach in which an interoperable team produces machine learning applications based on code, data and models in small, secure new versions that can be replicated and delivered reliably at any time, in short custom cycles (Symeonidis et al 2022).

MLOps Pipeline



1. Development and Experimentation
2. Pipeline Continuous Integration
3. Pipeline Continuous Delivery
4. Automated Triggering
5. Model Continuous Delivery
6. Monitoring

MLOps Maturity



Source: [Microsoft Azure](#)

Level	Technology
1 – No MLOps	<ul style="list-style-type: none">• Manual builds and deployments• Manual testing of model and application• No centralized tracking of model performance• Training of model is manual
2 – DevOps but No MLOps	<ul style="list-style-type: none">• Automated builds• Automated tests for application code
3 – Automated Training	<ul style="list-style-type: none">• Automated model training• Centralized tracking of model training performance• Model management
4 – Automated Model Deployment	<ul style="list-style-type: none">• Integrated A/B testing of model performance for deployment• Automated tests for all code• Centralized tracking of model training performance
5 – Full MLOps Automated operations	<ul style="list-style-type: none">• Automated model training and testing• Verbose, centralized metrics from deployed model

MLOps Stages on the Automation Pipeline

MLOps Stage	Output of the Stage Execution
Development & Experimentation (ML algorithms, new ML models)	Source code for pipelines: Data extraction, validation, preparation, model training, model evaluation, model testing
Pipeline Continuous Integration (Build source code and run tests)	Pipeline components to be deployed: packages and executables.
Pipeline Continuous Delivery (Deploy pipelines to the target environment)	Deployed pipeline with new implementation of the model.
Automated Triggering (Pipeline is automatically executed in production. Schedule or trigger are used)	Trained model that is stored in the model registry.
Model Continuous Delivery (Model serving for prediction)	Deployed model prediction service (e.g. model exposed as REST API)
Monitoring (Collecting data about the model performance on live data)	Trigger to execute the pipeline or to start a new experiment cycle.

MLOps Setup

MLOps Setup Components	Description	Tools Example
Source Control	Versioning the Code, Data, and ML Model artifacts.	Git, Comet
Test & Build Services	Using CI tools for (1) Quality assurance for all ML artifacts, and (2) Building packages and executables for pipelines.	PyTest, Make
Deployment Services	Using CD tools for deploying pipelines to the target environment.	KubeFlow, Sagify
Model Registry	A registry for storing already trained ML models.	DVC, AWS S3
Feature Store	Preprocessing input data as features to be consumed in the model training pipeline and during the model production.	AWS Feature Store, Feast
ML Metadata Store	Tracking metadata of model training, for example model name, parameters, training data, test data, and metric results.	DVC, Neptune
ML Pipeline Orchestrator	Automating the steps of the ML experiments.	Kedro, Flyte, DVC

MLOps Tools

DATA



DEVELOPMENT



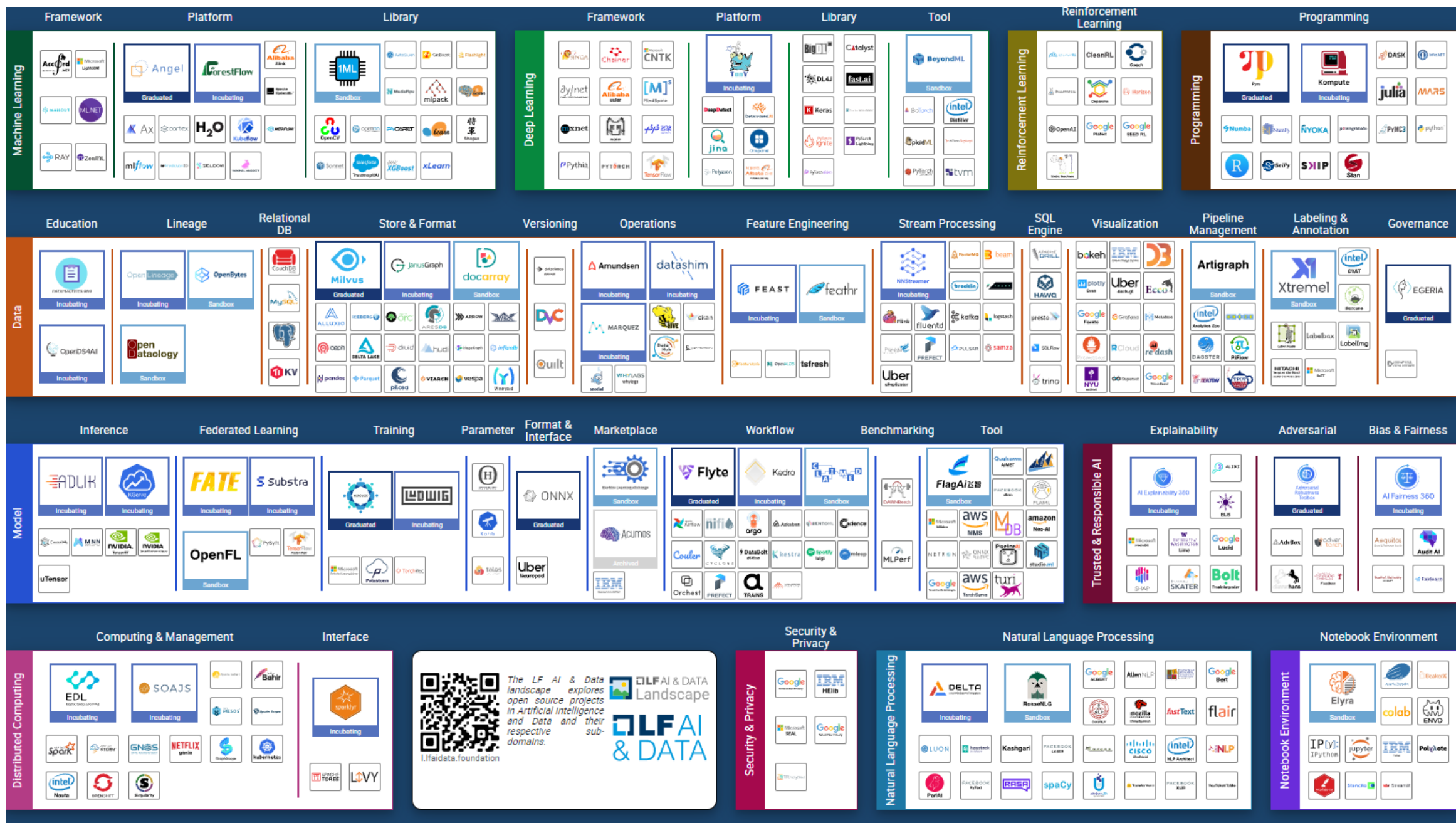
TUNING / EXPERIMENTATION



DEPLOYMENT / MONITORING



ML/AI Tools



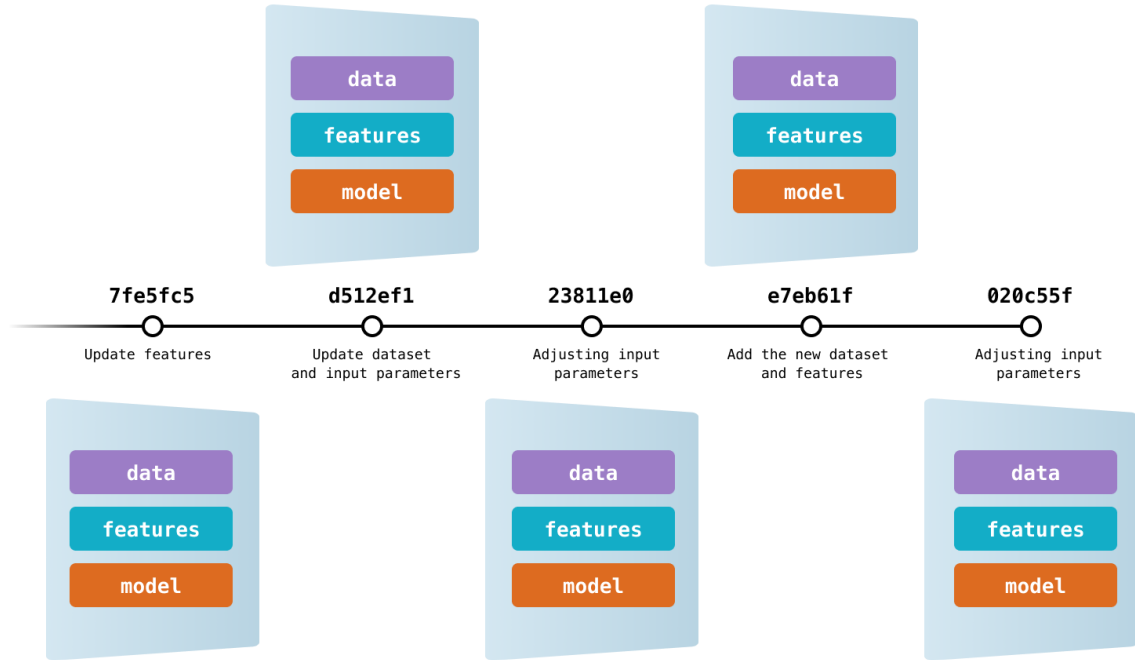
MLOps Principle: Automation

The level of automation of the Data, ML Model, and Code pipelines determines the maturity of the ML process.

Automate the complete ML-workflow steps without any manual intervention with certain triggers is the aim.

Data	ML Model	Code
1) Data transformation 2) Feature creation and manipulation	1) Data engineering pipeline 2) ML model training pipeline 3) Hyperparameter/Parameter selection	1) ML model deployment with CI/CD 2) Application build

MLOps Principle: Versioning

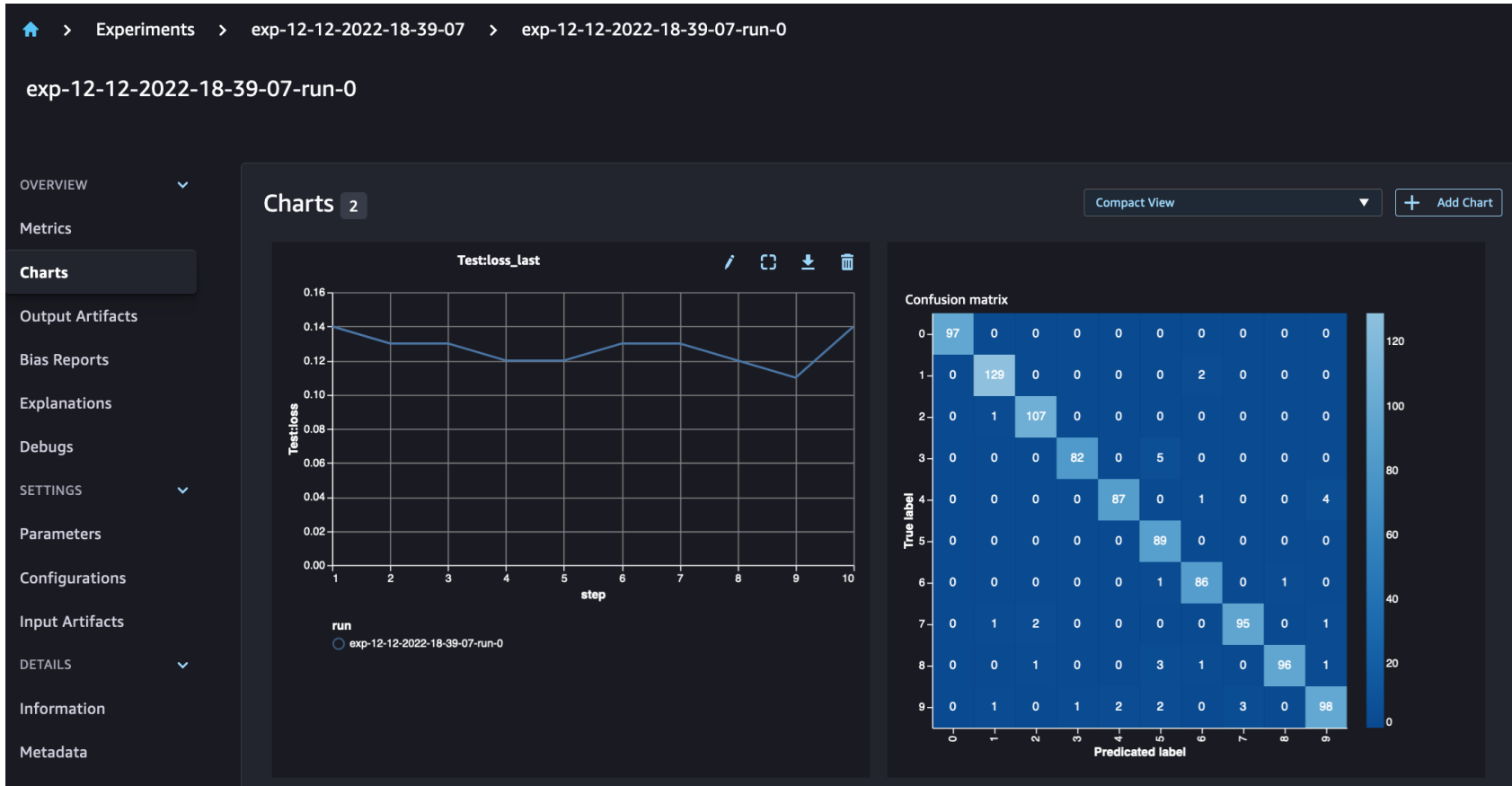


Explicit versioning allows for repeatability in experiments, enables comparisons, and prevents confusion. That is why it's important to versioning the ML training scrips, ML models and data sets for model.

Data	ML Model	Code
<ul style="list-style-type: none">1) Data preparation pipelines2) Features store3) Datasets4) Metadata	<ul style="list-style-type: none">1) ML model training pipeline2) ML model (object)3) Hyperparameters4) Experiment tracking	<ul style="list-style-type: none">1) Application code2) Configuration

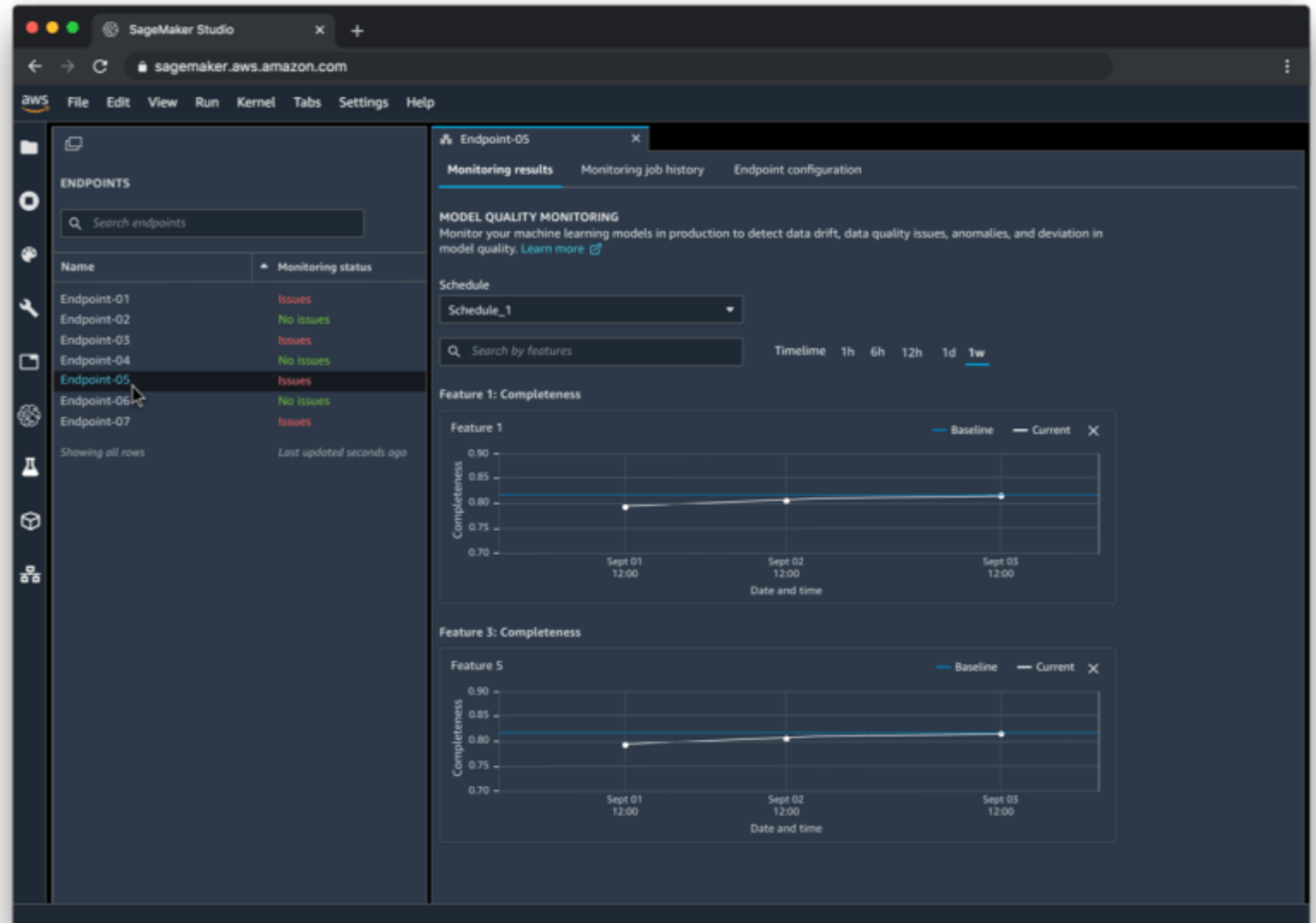
MLOps Principle: Versioning (Experiment Tracking)

In ML development, multiple experiments on model training can be executed in parallel before making the decision what model will be promoted to production. That is why it's important to track the experiment we have done.



MLOps Principle: Versioning (Model Management)

Model Management is there to ensure that ML models are consistent and all business requirements.



MLOps Principle: Testing

Testing or validation in MLOps refer to the activity to detect unexpected changes in data or infrastructure. There are three scopes withing testing principle:

- **Tests for data and features**
- **Tests for model development**
- **Tests for ML infrastructure**

Data	ML Model	Code
1) Data Validation (error detection) 2) Feature creation unit testing	1) Model specification is unit tested 2) ML model training pipeline is integration tested 3) ML model is validated before being operationalized 4) ML model staleness test (in production) 5) Testing ML model relevance and correctness 6) Testing non-functional requirements (security, fairness, interpretability)	1) Unit testing 2) Integration testing for the end-to-end pipeline

MLOps Principle: Testing (Data and Features)

Testing for data and features were done to validate the input and output data to comply with our current requirements. The activity including, but not limited to:

- Data validation: Check for data and features schema/domain.
- Features importance test: Check whether new features add a predictive power
- Policy-Compliant Test: Features and data pipelines should be policy-compliant
- Feature creation code unit test: Check bug in the code

MLOps Principle: Testing (Model Development)

Testing for data and features were done to detecting ML-specific errors. The activity including, but not limited to:

- ML Routines: What routines should exist in the pipeline (e.g. A/B Testing, accuracy metrics output, etc.)
- Model staleness test: Check how updated is the model
- Model Performance Validation: Check if the model performance still meet the requirements
- Model Fairness testing: Check if the model bias toward certain group or not

MLOps Principle: Testing (ML infrastructure)

Testing for ML Infrastructure were to ensure the machine learning pipeline more robust and resilient. The activity including, but not limited to:

- Integration testing: Test model pipeline to check ML model performs as expected
- Stress testing: Test if the model having an error during training or prediction

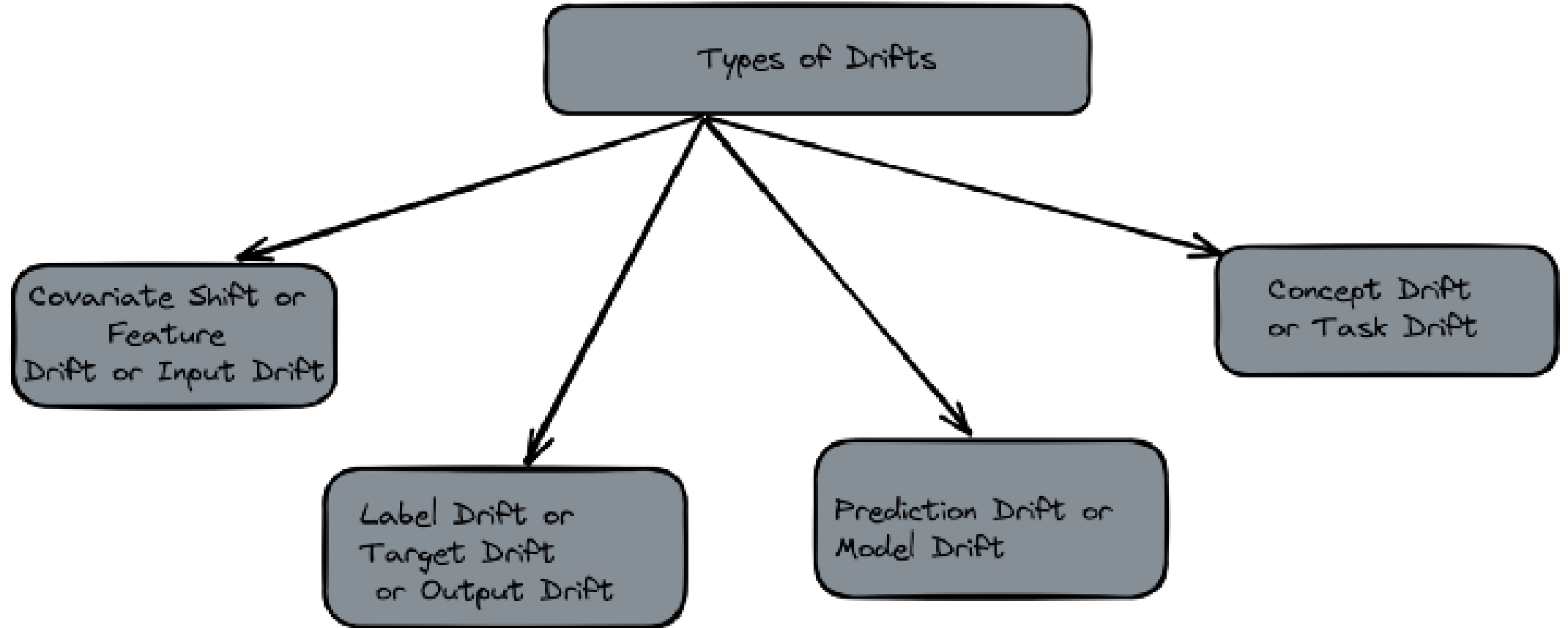
MLOps Principle: Monitoring

Monitoring were important to assure that the ML model performs as expected. For ML systems, it is important to monitor serving systems, training pipelines, and input data. Here are some reference for what to monitor:

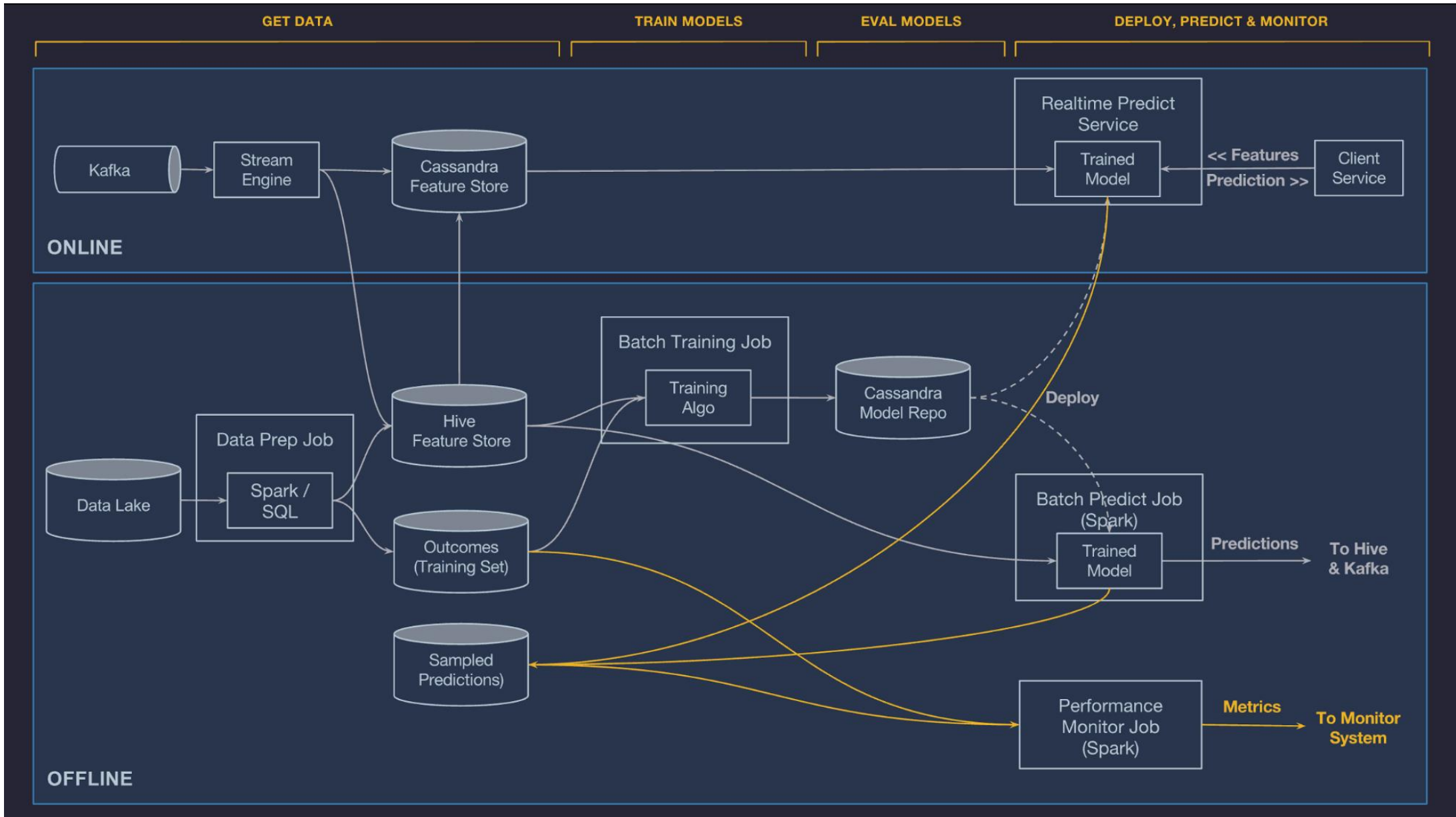
- Monitor Dependency Change
- Monitor Data invariants in development and production inputs
- Monitor for Data Development/Production Skew
- Monitor the numerical stability of the ML model
- Monitor computational performance of an ML system
- Monitor how *stale* the system in production is
- Monitor ML degradation on production data

Data	ML Model	Code
1) Data distribution changes (training vs. serving data) 2) Development vs Production features	1) ML model decay 2) Numerical stability 3) Computational performance of the ML model	1) Predictive quality of the application on Production data

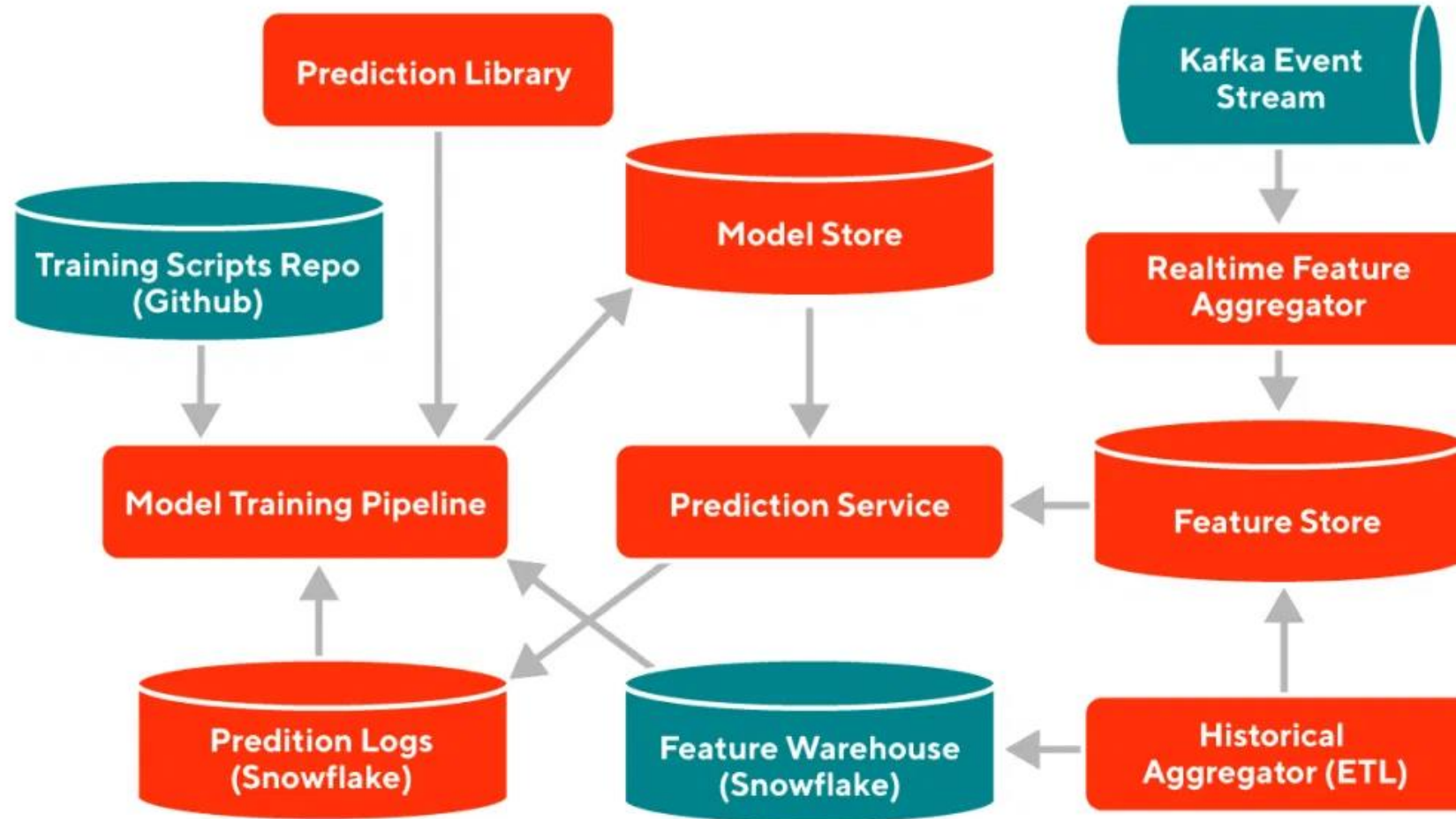
MLOps Principle: Monitoring (Drift)



MLOps Example Pipeline: Uber



MLOps Example Pipeline: DoorDash



MLOps Best Practices

MLOps Best Practices	Data	ML Model	Code
Documentation	<ul style="list-style-type: none">1) Data sources2) Decisions, how/where to get data3) Labelling methods	<ul style="list-style-type: none">1) Model selection criteria2) Design of experiments3) Model pseudo-code	<ul style="list-style-type: none">1) Deployment process2) How to run locally
Project Structure	<ul style="list-style-type: none">1) Data folder for raw and processed data2) A folder for data engineering pipeline3) Test folder for data engineering methods	<ul style="list-style-type: none">1) A folder that contains the trained model2) A folder for notebooks3) A folder for feature engineering4) A folder for ML model engineering	<ul style="list-style-type: none">1) A folder for bash/shell scripts2) A folder for tests3) A folder for deployment files (e.g Docker files)

10 MLOps Best Practices Tips

- Naming conventions
- Code quality checks
- Experiment — and track your experiments!
- Data validation
- Model validation across segments
- Resource utilization: remember that your experiments cost money
- Monitor predictive service performance
- Think carefully about your choice of ML platforms
- Open communication lines are important
- Score your ML system periodically

References

- ml-ops.org
- [MLOps - Definitions, Tools and Challenges](#)
- [MLOps Guide](#)
- [The ML Test Score: A Rubric for ML Production Readiness and Technical Debt Reduction](#)
- [MLOps: Continuous delivery and automation pipelines in machine learning](#)
- [MLOps: What It Is, Why It Matters, and How to Implement It](#)

NON BRAND DATA



Non-Brand Data

Non-Brand Data aims to make forever your data career stand out by providing various tips of Machine Learning, Technology News, and Python Packages.

By Cornelius Yudha Wijaya · Over 2,000 subscribers

cornelliusyudhawijaya@gmail.com

Subscribe

No thanks >