MLOps

Automated Model Training

1. Data Engg.
2. MLOps Into
   - Cloud storage
   - Cloud Comput
3. CI/CD
   CM
   L Expo

“Introduction

Introduction to MLOps

Data Science & ML

- Capable of solving complex real-world problems, transforming businesses, and delivering values in all domains
- In forefront due to the access to
    - Large datasets
    - Inexpensive on-demand compute resources, cloud solutions
    - Advanced easy to use ML tools
    - Rapid advances in ML methods and applications
    - Many businesses investing in developing their own Data Science team

Introduction to MLOps

Recent advances in Data Science & ML

- Many businesses investing in developing their own Data Science team
- Demand for Data Scientists and ML Engineers increased
- Emergence of MLOps
  - ML Operations is the practice of managing and operationalizing ML models throughout their lifecycle.
  - Unifies ML system developments and ML system operations
  - Automates and monitors all steps in ML system designing, development, integration, testing, releasing, deployment, data collection, cleaning, feature engineering, feature selection, retraining and infrastructure management, etc.

Introduction to MLOps

Benefits of MLOps

- Brings together data science, Machine Learning, software engineering, and operations to ensure the efficient design, development, deployment, and maintenance of ML models.
- Essential because it addresses the challenges of scaling, monitoring, and governing ML systems in production environments.
- By applying MLOps principles, organizations can streamline their ML workflows, improve model performance, and enhance collaboration between data scientists and engineers.
- Successful implementation of MLOps can lead to faster model deployment, reduced downtime, and increased business value from ML initiatives.

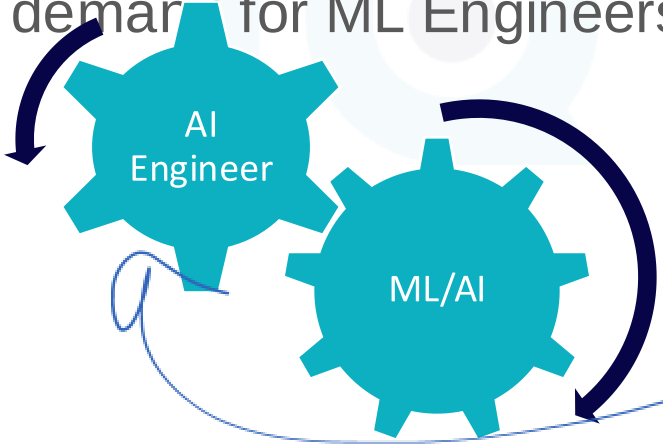
Rise of AI Engineers

Rise of AI Engineers

MLOps

- Need a systematic & efficient approach to be build ML models
- Needs to apply DevOps' best-practices to the emerging ML technologies
- Widespread adoption of ML models globally necessitates a sudden rise in demand for ML Engineers

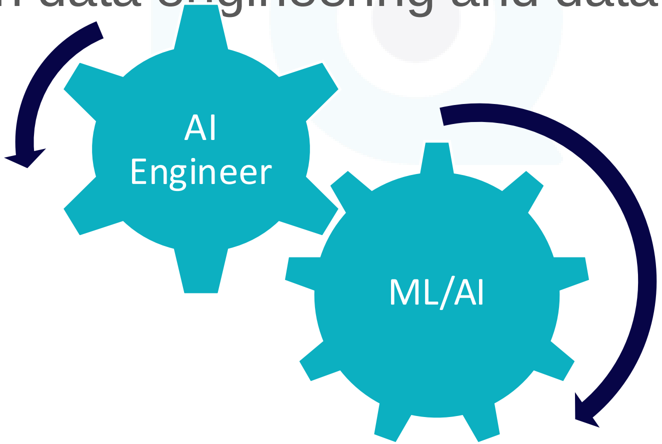
- DE/SME
→ Data Engineers
→ Data Scientist
- AI/ML Engg.



Rise of AI Engineers

MLOps

- Cross-trained enough to become proficient at both data engineering and data science
- Primarily come from data engineering backgrounds
- Sits at the crossroads of data science and data engineering, and has proficiency in both data engineering and data science

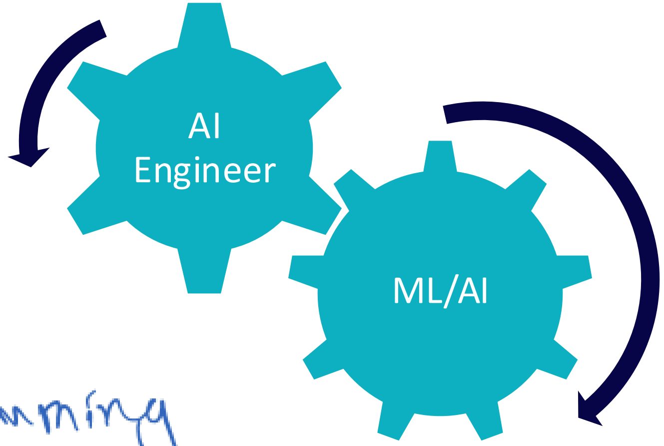


Rise of AI Engineers

MLOps

- **Cloud Native ML Platform :** AWS SageMaker, Azure ML Studio, no code GCP AI Platform
- **Containerized Workflow:** Docker, Kubernetes, public and private containers
- **Serverless technology:** AWS Lambda, AWS Athena, Google Cloud Foundation, Azure Functions
- **Special hardware for ML:** GPUs, TPUs, A14, AWS Inferentia Elastic inference
- **Big data platforms:** Databriks, Hadoop/Spark, snowflake, Amazon EMR Google Big Query, Google big Query

Programming



Rise of AI Engineers

Challenges

- ML deeply depends on Cloud Computing
- Raw ingredients of ML requires massive compute, extensive data and specialized hardware
- ML Systems
  - Mainly focus on data engineering, data processing, problem feasibility and business alignment
  - Primary focus is on business with ML rather than code (HIPPO effect)
  - Most ML systems are not code native, use academic software packages that do not scale for large-scale problems

“MLOps

MLOps

Need for MLOps

- ML models are, in general, not moving into production, and it is the impetus for the emergence of MLOps as a critical standard
- **Efficiency**
  - ML systems with MLOps automate and scale ML model deployment and monitoring in production, enhancing efficiency
- **Risk Mitigation**
  - ML systems address challenges in DS, ML, and AI by enabling rigorous testing, validation, and responsible practices, minimizing negative outcomes
- **Responsible AI Adoption:**
  - ML systems promote ethical guidelines and accountable AI practices through fairness, transparency, and interpretability features, ensuring trustworthy and accepted AI solutions

MLOps

What is it?

- The process of automating ML using DevOps methodologies
- Shares lineage with DevOps, which demands automation
  - Broken if not automated
  - Should not have humans as levers of the machine

History of Automation : humans are the least valuable doing repetitive jobs but are most valuable using technology as the architects

MLOps

How MLOps is different from DevOps?

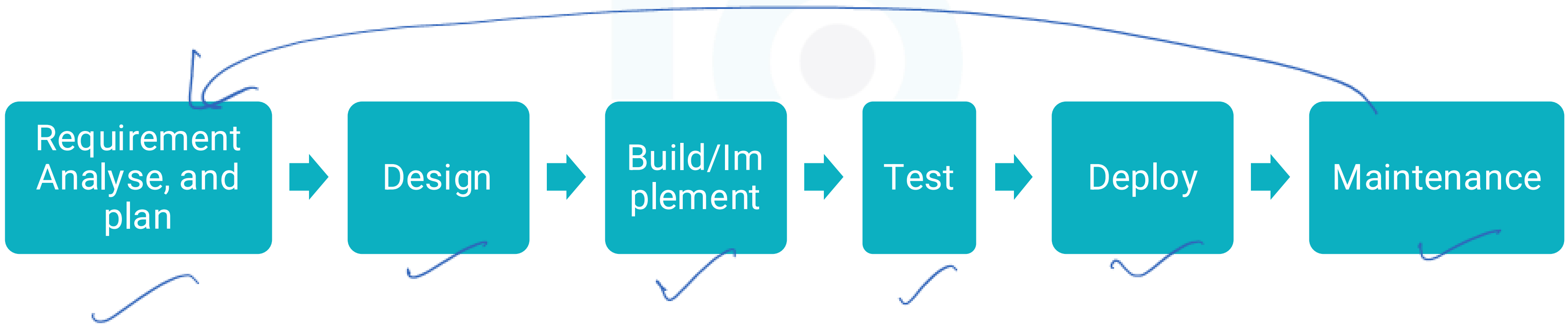
- With MLOps, not only the software engineering process needs full automation, but so do the data and modelling
- Model training and deployment is added to DevOps
- Monitor new things that break automation
- **Model drift**: degradation in the performance of a ML system in production
  - **Data Drift:** Distribution of unseen data differs from the distribution of the training data, include changes due to seasonality, consumer preferences, the addition of new products, etc.
  - **Concept Drift:** Change in the functional relationship between a model's input and output data.

Software System Development and Deployment

Software System Development and Deployment

Waterfall Methodology

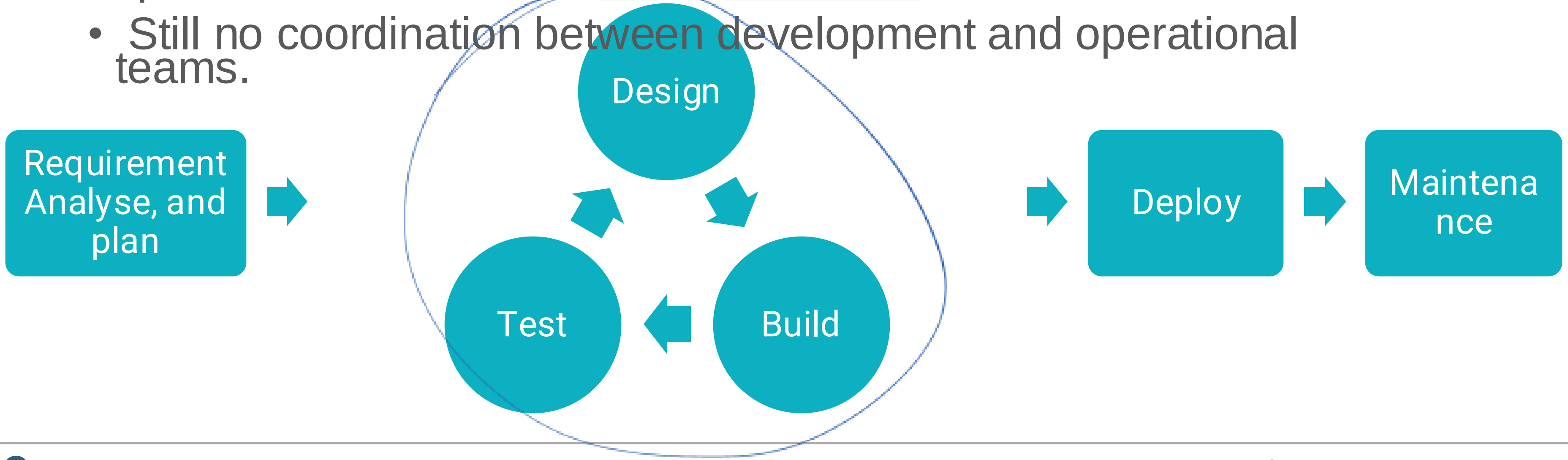
- Sequential and linear process, and cannot turn back
- No coordination between different teams
- Hard to maintain the software



Software System Development and Deployment

Agile Methodology

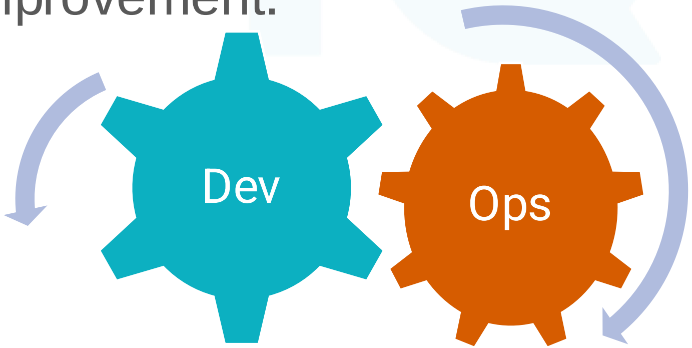
- Iterative and flexible approach to software development.
- Focuses on collaboration, adaptability, and responding to changes quickly.
- Divides the development process into short iterations called sprints.
- Still no coordination between development and operational teams.



Software System Development and Deployment

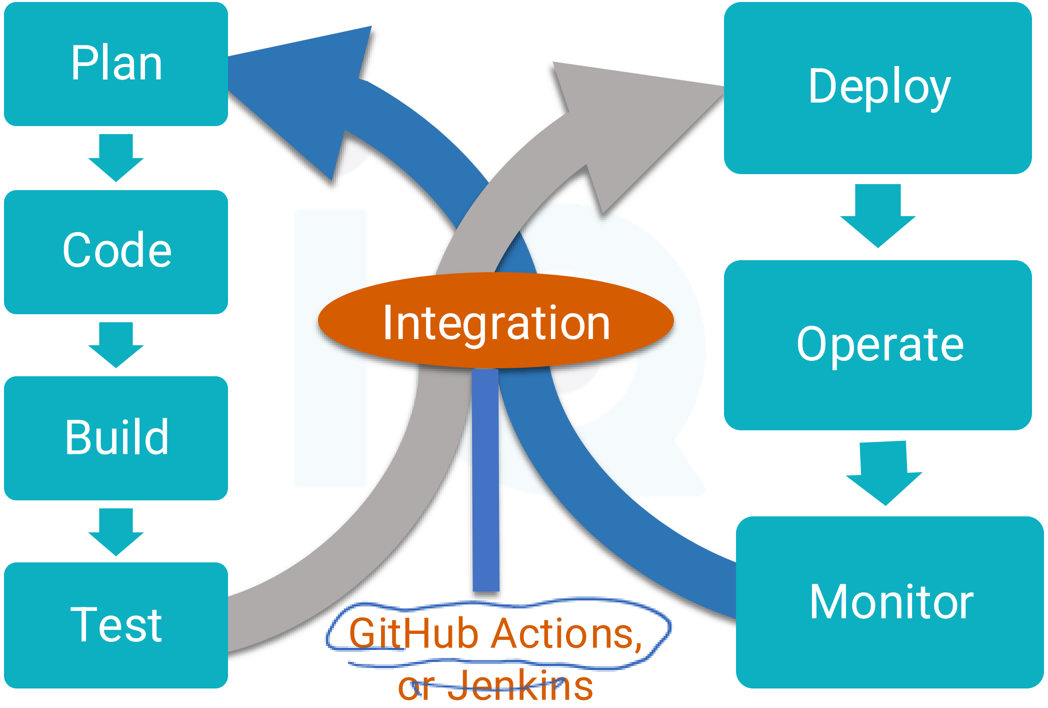
DevOps Methodology

- An extension of the Agile methodology that emphasizes collaboration and integration between development and operations teams.
- Collaboration and communication between development and operations teams.
- Continuous integration and deployment (CI/CD) for faster releases.
- Infrastructure as Code (IaC) for managing and replicating infrastructure
- Monitoring and feedback loops for proactive issue detection and continuous improvement.



Software System Development and Deployment

DevOps Methodology



Software System Development and Deployment

DevOps Methodology

Version Control
- source code management,
- github

Continuous
- compile, validate
- code review, integration

Continuous
- transform the application to the test servers
- performing unit test

Continuous
- Deploy the tested package on the production server for release

Plan

monitor

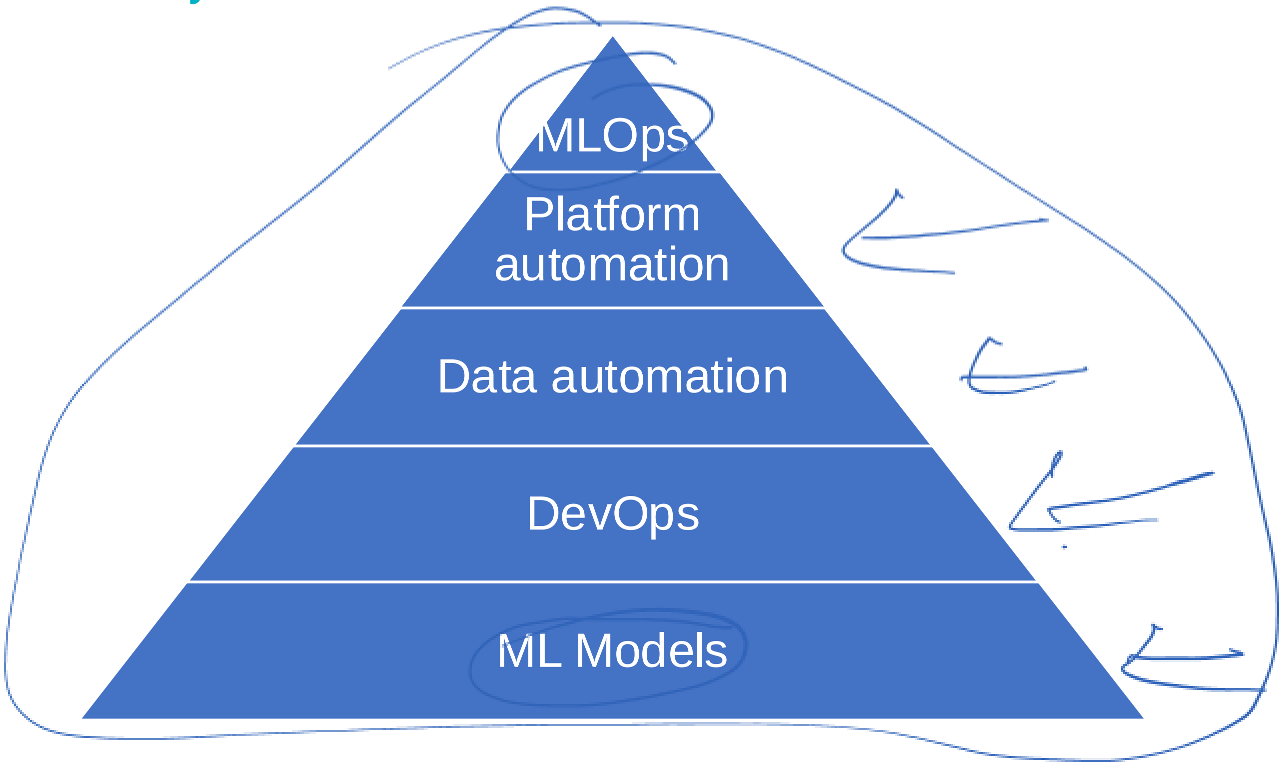
Software System Development and Deployment

Additional steps in MLOps (Recap)

- With MLOps, not only the software engineering process needs full automation, but so do the data and modelling
- Model training and deployment is added to DevOps
- Monitor new things that break automation
- **Model drift**: degradation in the performance of a ML system in production
  - (i) **Data Drift:** Distribution of unseen data differs from the distribution of the training data, include changes due to seasonality, consumer preferences, the addition of new products, etc.
  - (ii) **Concept Drift:** Change in the functional relationship between a model's input and output data.

Software System Development and Deployment

MLOps Hierarchy of Needs



Software System Development and Deployment

Additional steps in MLOps (Recap)

- Have you heard Platform Engineering?
- How it fits into MLOps?
- Why do we need Platform Engineering?
- What is IDP (Internal Developer Platform)?

A team maintains digital infrastructure.

Stage 1
Engg
Stage 2
AI Engg | DE
Data Scient | Data Engg
IDP
CEO / CAIO
Stage 3
Stage 4 (PE)

Aspect	Platform Engineering	Internal Developer Platform (IDP)
Definition	A discipline focused on designing, building, and managing platforms that enable efficient developer workflows.	A product or toolset created by platform engineers to provide developers with self-service capabilities.
Scope	Broader in scope, encompassing the design, implementation, and operation of platforms tailored to organizational needs.	Narrower in scope, acting as the operational output of platform engineering to support developers.
Role	Platform engineers are responsible for creating and maintaining the infrastructure and tools that form the platform.	Developers use the IDP to manage resources, deploy applications, and monitor performance autonomously.
Purpose	Focuses on improving developer experience by reducing cognitive load, standardizing processes, and enabling self-service.	Provides a centralized interface for developers to access tools, automate workflows, and manage environments.
Components	Includes infrastructure management, DevOps practices, governance frameworks, and tooling integration.	Comprises tools for CI/CD pipelines, monitoring, environment provisioning, and deployment management.
User Base	Platform engineers collaborate with DevOps teams, developers, and stakeholders to build the platform.	Developers are the primary users of the IDP as a self-service tool for their workflows.
Outcome	A robust platform that supports scalable, secure, and efficient software delivery across teams.	A streamlined interface that empowers developers to focus on coding rather than infrastructure tasks.

“Kubeflow, Apache Airflow & ClearML

Comprehensive MLOps Stack

Kubeflow: ML on Kubernetes

- Core Concept
  - Platform dedicated to making ML workflows on Kubernetes simple, portable, and scalable
  - Provides end-to-end orchestration of ML systems with standardized components
- Key Capabilities
  - Notebooks: Interactive development environments for data exploration
  - Pipelines: Composable, reproducible ML workflows
  - Training Operators: Distributed training for TensorFlow, PyTorch, MXNet, and more
  - Model Serving: Standardized deployment with monitoring capabilities
- Business Value
  - Reduces infrastructure complexity for ML workloads
  - Enables consistent environments from development to production
  - Accelerates time to market for ML initiatives

Apache Airflow: Workflow Orchestration

- Core Concept
  - Platform to programmatically author, schedule, and monitor workflows
  - Workflows defined as Directed Acyclic Graphs (DAGs) of tasks
- Key Capabilities
  - Rich scheduling capabilities with complex dependency management
  - Extensive operator library for interacting with various systems
  - Comprehensive UI for monitoring workflow execution and history Robust error handling and recovery mechanisms
- Business Value
  - Eliminates manual intervention in recurring workflows
  - Provides visibility into complex data pipelines
  - Ensures reliable execution of mission-critical processes
  - Enables workflow evolution with minimal disruption

ClearML: ML Operations & Experiment Management

- Core Concept
  - End-to-end platform for managing the entire ML lifecycle MLFlow
  - Automated tracking, orchestration, and deployment of ML workloads + Open Source
- Key Capabilities
  - ML Artifacts Experiment Manager: Automatic logging of code, data, parameters, and results
  - DVC Orchestration: Distributed execution of ML tasks across computing resources On-Prem
  - ML Archite Data Management: Version tracking and dataset management
  - Model Registry: Central repository for trained models with deployment capabilities
- Business Value
  - Increases data scientist productivity by reducing operational overhead
  - Enhances collaboration through centralized experiment tracking
  - Improves model governance with comprehensive audit trails

MLOps Maturity Enhancement

For Capstone Project

- From Ad-hoc Experiments to Production Systems
  - Level 0: Manual processes with limited reproducibility
  - Level 1: Pipeline automation and experiment tracking
  - Level 2: Continuous integration and deployment for ML
  - Level 3: Automated monitoring, testing, and retraining
- How These Tools Accelerate Maturity
  - Kubeflow: Standardizes infrastructure and deployment patterns
  - Apache Airflow: Automates recurring workflows and dependencies
  - ClearML: Provides visibility, tracking, and governance
- Organizational Impact
  - Reduced time from development to production
  - Increased model reliability and performance
  - Better utilization of computational resources Enhanced compliance and governance

Solving Common ML Production Challenges

- Challenge: Model Performance Degradation
  - Kubeflow: Canary deployments and A/B testing frameworks
  - Airflow: Scheduled evaluation jobs and alerts
  - ClearML: Automated drift detection and performance monitoring

+ MLflow
+ Weights & Biases.

- Challenge: Reproducibility and Governance
  - Kubeflow: Version-controlled pipeline definitions
  - Airflow: Auditable workflow execution history
  - ClearML: Complete lineage tracking from data to deployment
- Challenge: Resource Optimization
  - Kubeflow: Efficient resource allocation on Kubernetes
  - Airflow: Intelligent task scheduling based on priorities
  - ClearML: Dynamic resource allocation based on task requirement

Implementation Strategy and ROI

include a few lines in your code

- Phased Implementation Approach
  - Start with experiment tracking and basic workflow automation
  - Gradually introduce CI/CD for ML models
  - Implement advanced monitoring and automated retraining
- Expected ROI Metrics
  - 40-60% reduction in time spent on operational tasks 10
  - 30-50% faster model iteration cycles
  - 70-90% improvement in model deployment reliability
  - 20-30% better resource utilization 10%. ⇐ 50%.
- Success Factors
  - Executive sponsorship and clear ownership
  - Cross-functional team involvement
  - Focus on solving specific business problems & incremental adoption with measurable outcomes

Advanced MLOps Integration

Combining Kubeflow, Apache Airflow & ClearML

Data Management → **Model Development** → **Deployment** → **Monitoring** → **Governance**

Collection & Preparation | Training & Evaluation | CI/CD for ML Models | Performance & Drift | Compliance & Security

Kubeflow

Kubernetes-native · Notebooks · Training Operators · Containers · GPU Support · Pipelines

- Standardized ML components
- Distributed training at scale
- KFServing for model deployment
- Katib for hyperparameter tuning

Apache Airflow

DAG Workflows · Scheduling · Python-based · Connectors · Web UI · Sensors

- Complex workflow orchestration
- Rich dependency management
- Extensible operator ecosystem
- Error handling & recovery

S

ClearML

Experiment Tracking · Dataset Management · Agent Infrastructure · PyTorch Support · TensorFlow · Multi-cloud

- Automated experiment tracking
- Resource management & queuing
- Model registry & versioning
- Zero-code performance monitoring

Integration Benefits

Efficiency · Governance · Collaboration · Automation · Scalability · Flexibility

- End-to-end ML lifecycle management
- Seamless platform interoperability
- Standardized workflows & practices
- Reduced operational overhead

Meet @ 10-30 am

Integration Patterns

Airflow as Control Plane	Kubeflow Pipeline Core	ClearML as ML Orchestrator
Centralized workflow orchestration	Containerized ML workflow	Automated experiment management
Scheduled training on Kubeflow	Distributed training at scale	Centralized model registry
ClearML metrics tracking	Reusable pipeline components	Scheduled by Airflow
Temporal dependencies management	Experiment tracking with ClearML	Executed on Kubeflow infrastructure

Complete MLOps Technology Stack

Kubernetes · Kubeflow · Istio · Helm · Apache Airflow · Jenkins · Argo CD · TensorFlow · PyTorch · Scikit-learn

Apache Spark · Pandas · Hadoop · PostgreSQL · MongoDB · Redis · MinIO · Prometheus · Grafana · Elasticsearch

JupyterHub · MLflow · ClearML · AWS · GCP · Azure

SQL