Machine Learning Operations

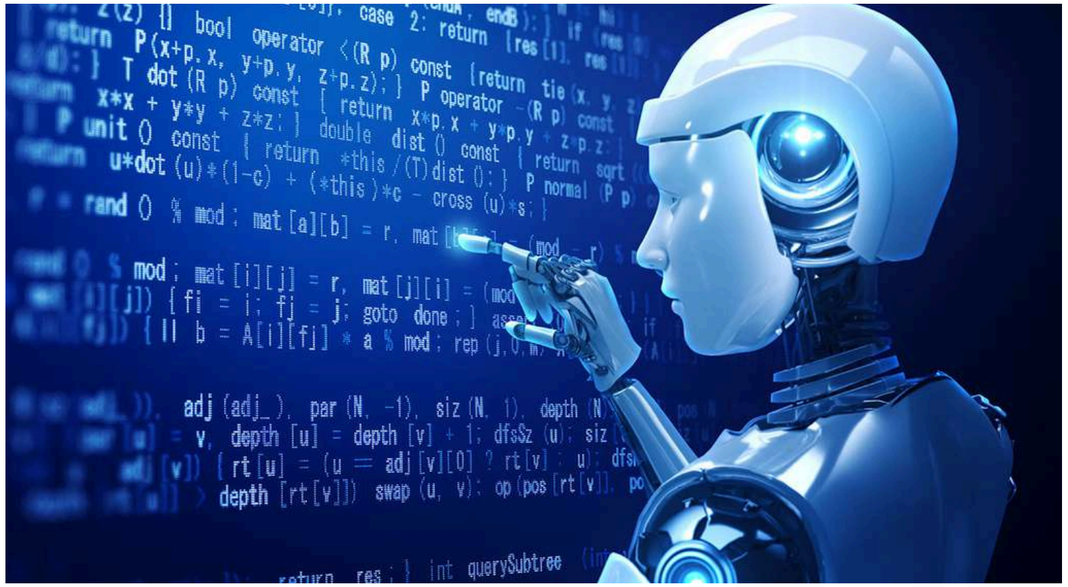
MLOps: From Model to Production

Presented By:
Ahmed Layouni

OVERWIEW

- Introduction to MLOps
- ML Lifecycle vs MLOps Lifecycle
- MLOps Architecture Components
- Real-World Examples
- Evolution and Importance of MLOps
- MLOps vs. DevOps
- Challenges in MLops
- Limitations of MLOps

What is MLOps and Why is it Important?

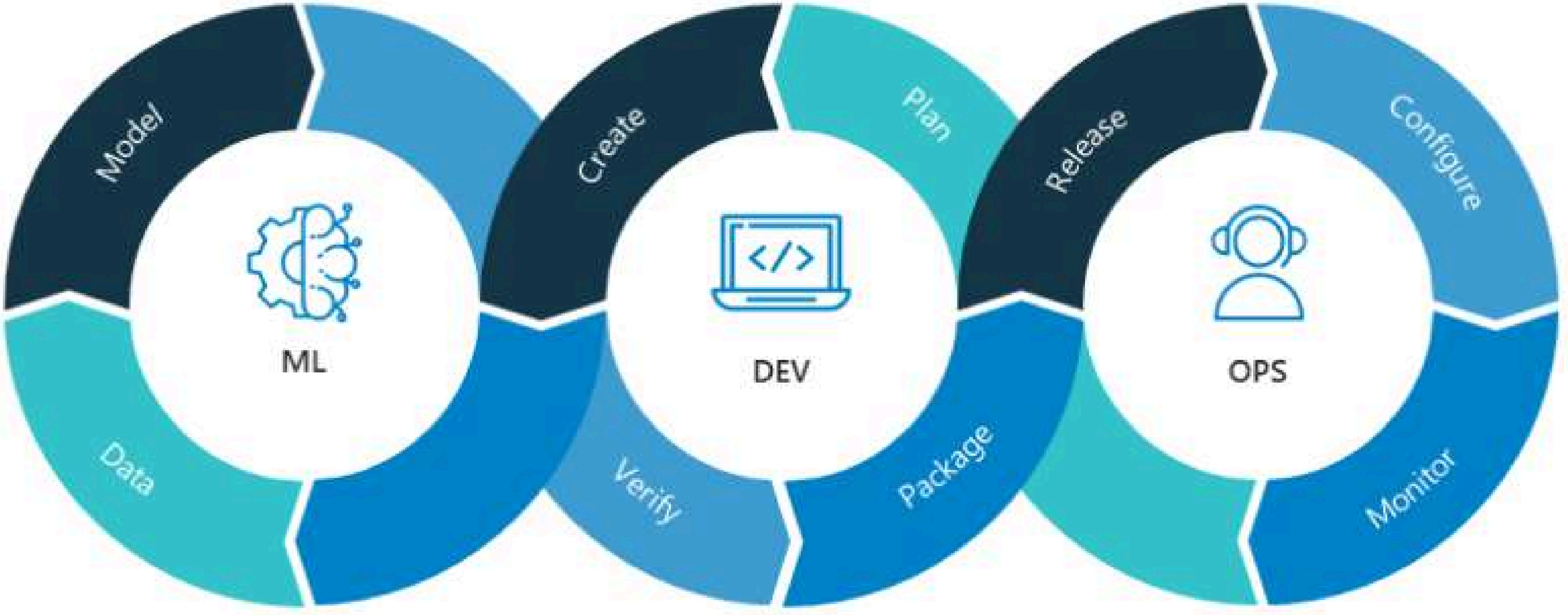


What is MLOps?

MLOps (Machine Learning Operations) is a set of practices designed to reliably and efficiently deploy and maintain machine learning models in production environments. It focuses on automating and streamlining the entire process, from building to deployment and ongoing management.

Bridging the Gap

MLOps is crucial because it bridges the gap between data science and IT operations. It ensures continuous integration and delivery (CI/CD) of ML models, leading to consistent performance, scalability, and faster time-to-market for AI-driven solutions.


Model
Data
ML
Create
Plan
Verify
Package
DEV
Release
Configure
Monitor
OPS

How Did We Get to MLOps?

1 2 3

Early Machine Learning

Early Machine Learning: Initial ML efforts were manual, with little integration between data science and IT operations.

DevOps Influence

DevOps Influence: Adoption of DevOps principles for software development highlighted the need for similar practices in ML.

Emergence of MLOps

Emergence of MLOps: Integration of CI/CD, automated testing, and monitoring into the ML lifecycle led to the development of MLOps.

Why MLOps is Important ?

Risk Mitigation

Ensures models are monitored and maintained to mitigate risks such as model drift and degradation.

Scalability

Facilitates the deployment of multiple models efficiently, enabling organizations to handle increased model complexity.

Performance Monitoring

Continuously tracks model performance to ensure they meet business objectives.

Why MLOps is Important?

Compliance

Helps meet regulatory and governance requirements, ensuring models are transparent and explainable.

Iteration Speed

Improves iteration speed and quality when developing ML models, allowing for faster deployment of updates.

Automation

Enables safe incremental updates and automated testing to maintain reliability.

A little comparisation: With and Without Mlops

Without MLOps:

- Difficult to move from experimentation to production
- Manual model deployment and updating
- Lack of reproducibility and monitoring

With MLOps:

- Models can be deployed faster and more reliably
- Automated pipelines reduce human error
- Continuous monitoring and feedback improve model performance over time

MLOps lifecycle vs DevOps lifecycle

Traditional lifecycle in ML :

- Data Collection
- Data Preparation
- Model Training
- Evaluation
- Deployment

MLOps Lifecycle adds:

- Continuous Integration (CI) and Continuous Deployment (CD)
- Model and Data Versioning
- Automated Testing
- Real-time Monitoring
- Model Retraining and Governance

MLOps vs DevOps

Similarities:

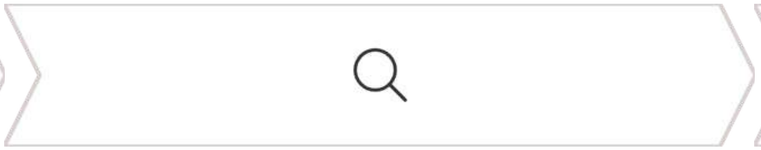
- Robust automation and trust between teams.
- Collaboration and increased communication.
- End-to-end service life cycle (build, test, release).
- Continuous delivery and high quality.

Differences:

- Software code is relatively static but ML models are dynamic and constantly learning.
- MLOps requires continuous monitoring and adaptation to new data.

MLOps Architecture Components

Data Collection and Storage



Data Preparation and Exploration

Model Training and Validation

Model Storage and Versioning

Model Evaluation and Explainability

Model Deployment

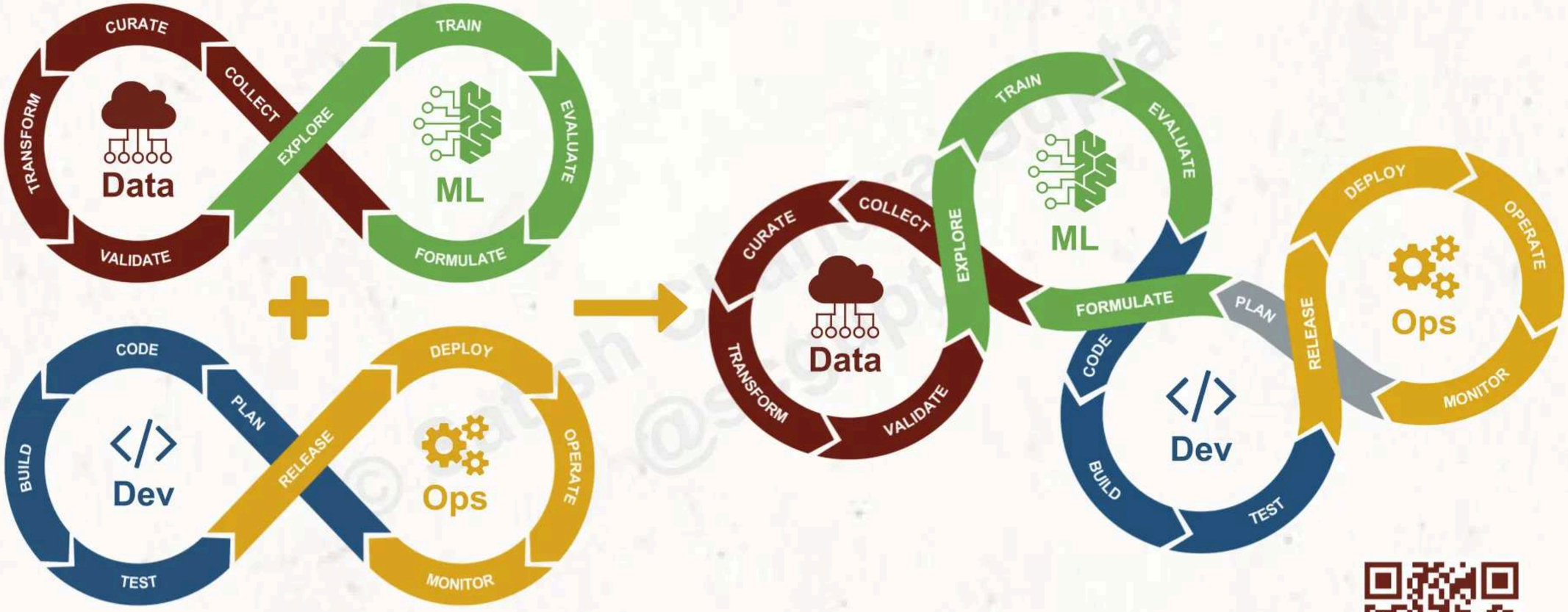
Monitoring and Maintenance

CI/CD & Automation

Feedback Loop and Improvement

MLOps = DataML + DevOps

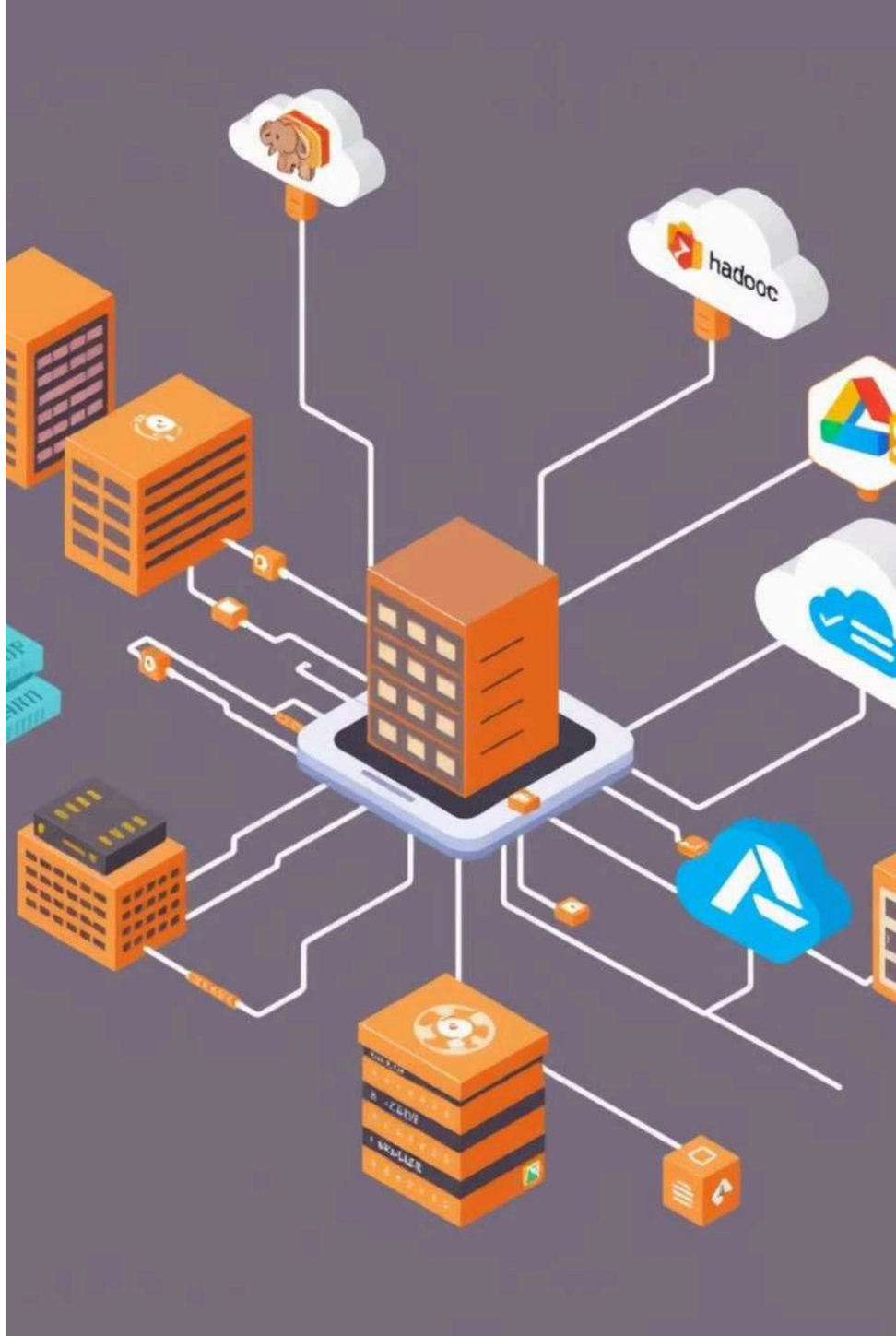
ml4devs.com/mlops-lifecycle



scgupta.me
twitter.com/**scgupta**
linkedin.com/in/**scgupta**

Data Collection and Storage

- **Sources:** Databases, Data Warehouses, APIs, Flat Files
- **Storage Solutions:** Hadoop, Amazon S3, Google Cloud Storage, Azure Blob Storage
- **Objective:** Centralize data for easy access and management



Data Preparation and Exploration

Data Cleaning

Remove inconsistencies and errors to ensure data quality and reliability for model training.

Exploratory Data Analysis (EDA)

Understand data patterns, distributions, and relationships to uncover insights and guide feature selection.

Feature Engineering

Transform raw data into useful features that improve the performance and accuracy of machine learning models.

Tools: Jupyter Notebooks, Pandas, Spark, Databricks are commonly used platforms and libraries for these critical data tasks.

Model Training and Validation

Model Training

An iterative process of testing various algorithms and meticulously tuning hyperparameters to optimize model

Model Validation

A crucial step to ensure the trained model generalizes effectively to unseen data, confirming its reliability and preventing

Experiment Tracking

Systematically recording all configurations, parameters, and results for each training run, enabling reproducibility and comparative analysis.

Key Tools

Leveraging powerful frameworks and libraries like TensorFlow, PyTorch, Scikit-learn, and MLflow to build, validate, and manage models efficiently.

Model Storage and Versioning

Repository

Centralized storage with version control, ensuring all models are safely kept and easily accessible.

Versioning

Tracking different iterations of models and their associated metadata is crucial for reproducibility and auditing.

Tools

Leveraging specialized tools like Git, MLflow, and ModelDB for efficient model management and collaboration.

Model Evaluation and Explainability

Model Performance

Utilizing metrics such as Accuracy, Precision, Recall, and F1 Score to assess model performance and reliability.

Prediction Transparency

Employing advanced techniques to interpret how models make predictions, ensuring transparency and understanding.

Tool Utilization

Leveraging specialized tools like SHAP, LIME, and ELI5 to facilitate comprehensive model evaluation and explanation.

Model Deployement

Process:

Model Export: Export models in a portable format like PMML, ONNX, or Docker containers.

Batch Inference: Use batch processing for model inference on scheduled intervals.

Testing Environment: Create a mirrored production environment for thorough testing.

Tools

Apache Airflow for scheduling, Docker for containerization.

Challenges

Ensuring consistency between testing and production environments.

Monitoring and Maintenance

Continuous Monitoring

Track model performance and detect data or concept drifts to ensure ongoing accuracy.

Periodic Retraining

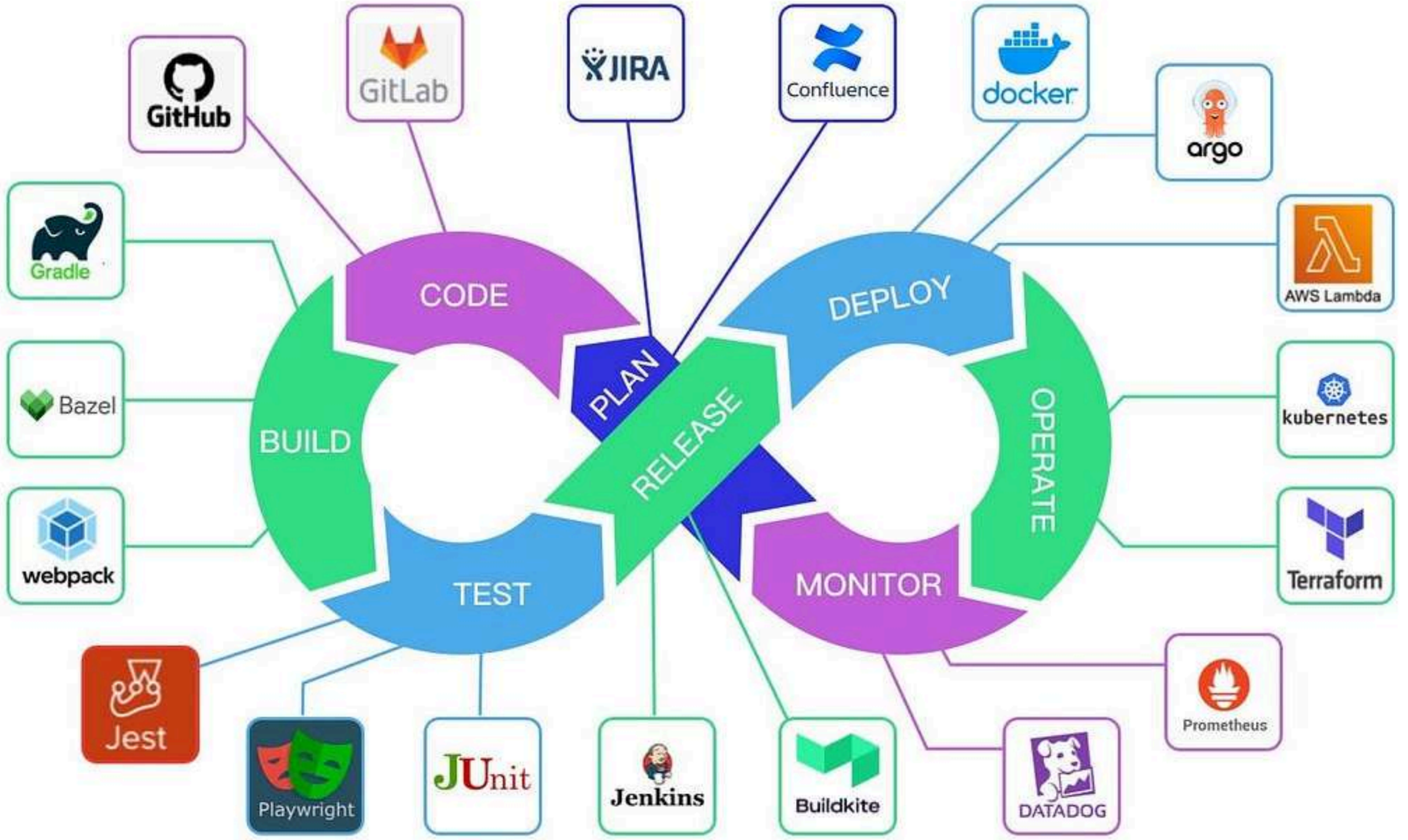
Regularly update models with new data to maintain relevance and adapt to changing patterns.

Comprehensive Logging

Record all model predictions and errors for audit trails, debugging, and performance analysis.

Tools: Prometheus, Grafana, ELK Stack

CI/CD Pipeline Explained in 5 Minutes



CI/CD & Automation

CI (Continuous Integration)

Automates testing and integration of code changes.

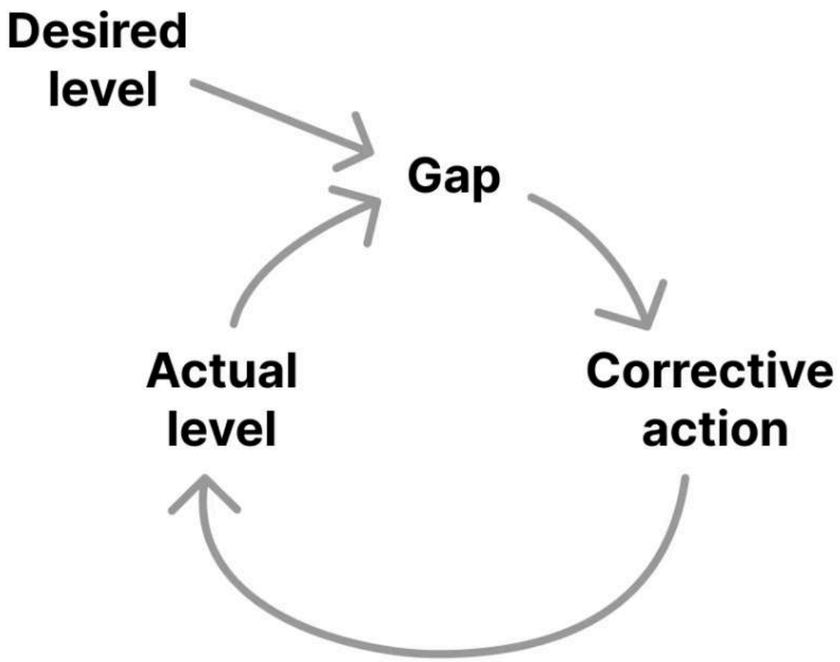
CD (Continuous Deployment)

Automates retraining and redeploying models.

Tools

GitHub Actions, Jenkins, or Kubeflow Pipelines .

Feedback Loop and Improvement



Feedback Collection

From end-users and stakeholders

Collaboration Tools

Jira, Trello, Slack

Iterative Improvement

Use feedback to enhance models

Problems and Solutions in MLOps

Data Drift

Monitor feature and label drift, retrain models regularly.

Model Performance Degradation

Use continuous monitoring and automated retraining.

Scalability Issues

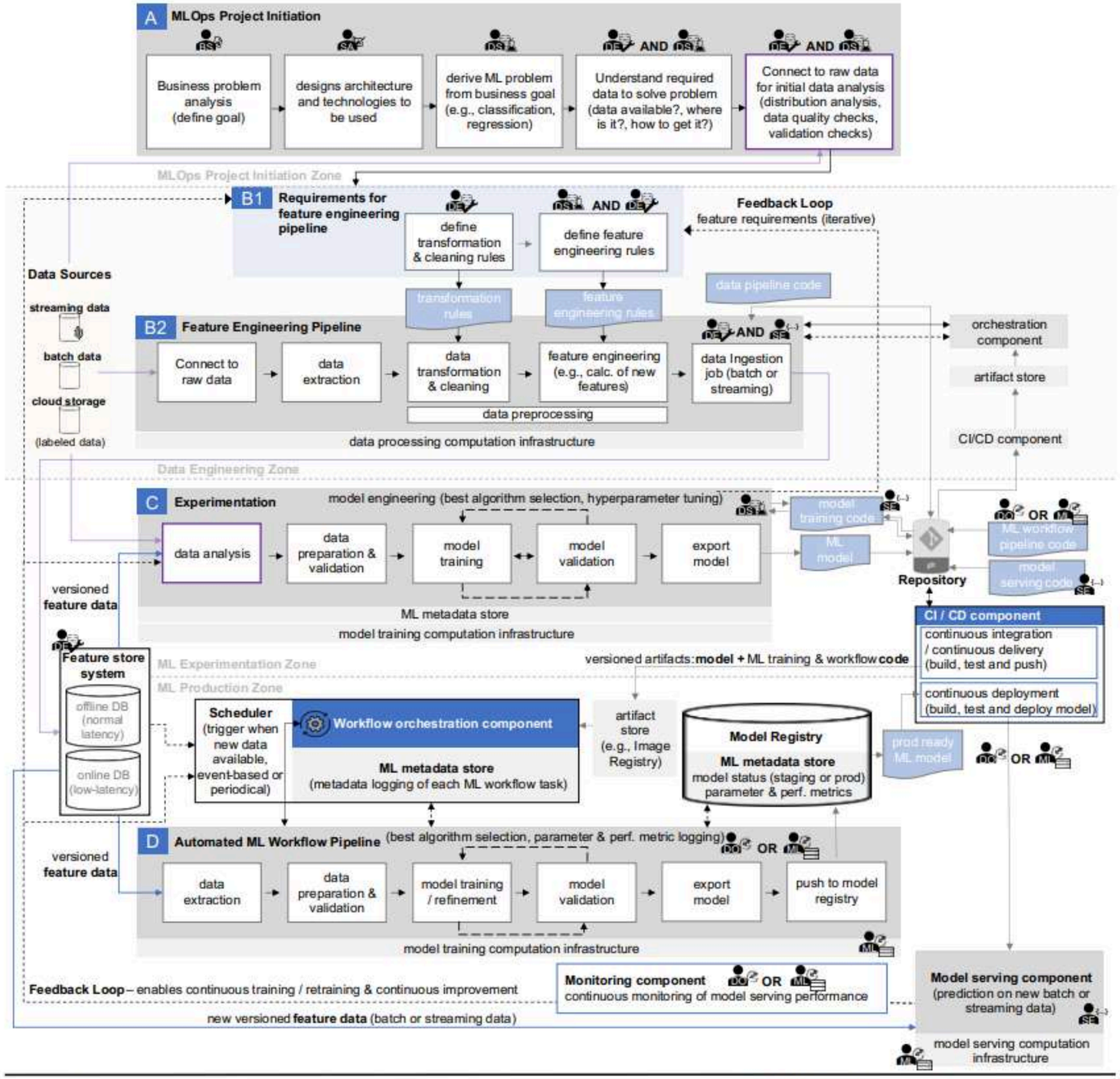
Implement containerization and orchestration with Kubernetes.

Compliance and Governance

Ensure thorough documentation and use explainability tools.

Integration Challenges

Use CI/CD pipelines and collaboration tools to streamline integration.


A MLOps Project Initiation
Business problem analysis (define goal)
designs architecture and technologies to be used
derive ML problem from business goal (e.g., classification, regression)
Understand required data to solve problem (data available?, where is it?, how to get it?)
Connect to raw data for initial data analysis (distribution analysis, data quality checks, validation checks)
MLOps Project Initiation Zone
B1 Requirements for feature engineering pipeline
define transformation & cleaning rules
define feature engineering rules
Feedback Loop
feature requirements (iterative)
Data Sources
streaming data
batch data
cloud storage
(labeled data)
transformation rules
feature engineering rules
data pipeline code
B2 Feature Engineering Pipeline
Connect to raw data
data extraction
data transformation & cleaning
feature engineering (e.g., calc. of new features)
data Ingestion job (batch or streaming)
data preprocessing
data processing computation infrastructure
orchestration component
artifact store
CI/CD component
Data Engineering Zone
C Experimentation
model engineering (best algorithm selection, hyperparameter tuning)
data analysis
data preparation & validation
model training
model validation
export model
ML metadata store
model training computation infrastructure
model training code
ML model
ML workflow pipeline code
model serving code
Repository
CI / CD component
continuous integration / continuous delivery (build, test and push)
continuous deployment (build, test and deploy model)
versioned feature data
Feature store system
offline DB (normal latency)
online DB (low-latency)
ML Experimentation Zone
ML Production Zone
versioned artifacts: model + ML training & workflow code
Scheduler (trigger when new data available, event-based or periodical)
Workflow orchestration component
ML metadata store (metadata logging of each ML workflow task)
artifact store (e.g., Image Registry)
Model Registry
ML metadata store
model status (staging or prod)
parameter & perf. metrics
prod ready ML model
D Automated ML Workflow Pipeline
(best algorithm selection, parameter & perf. metric logging)
data extraction
data preparation & validation
model training / refinement
model validation
export model
push to model registry
model training computation infrastructure
Monitoring component
continuous monitoring of model serving performance
Feedback Loop – enables continuous training / retraining & continuous improvement
new versioned feature data (batch or streaming data)
Model serving component
(prediction on new batch or streaming data)
model serving computation infrastructure
LEGEND
Business Stakeholder
IT Solution Architect
Data Scientist
Data Engineer
Software Engineer
DevOps Engineer
ML Engineer
General process flow
Data Engineering flow
Model / Code flow
Feedback loop flow
Versioned Feature Flow

What are the challenges in MachineLearning ?

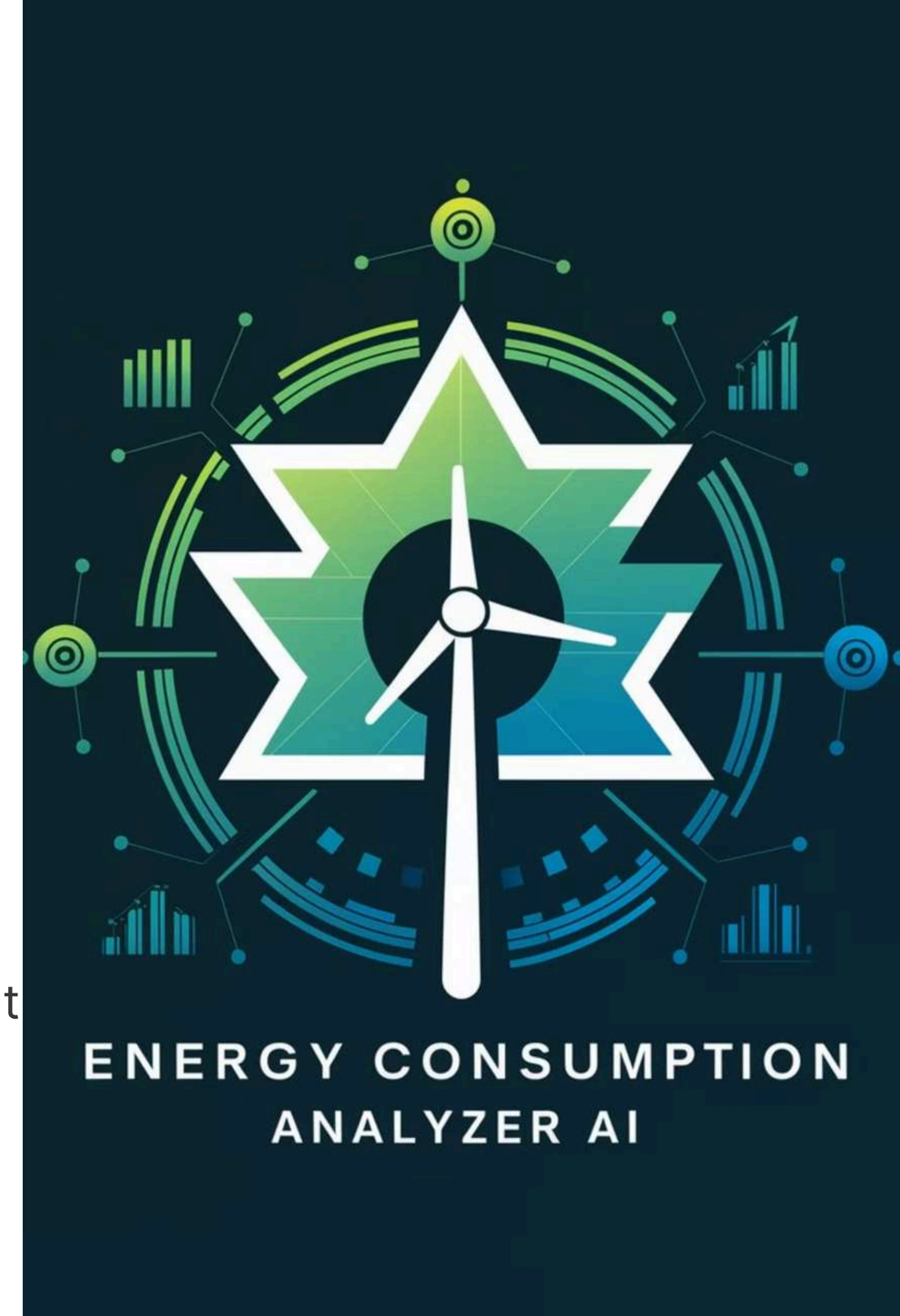
- **Data Quality:** Noisy or incomplete data leads to bad models
- **Model Drift:** Models may get outdated as real-world data changes
- **Tool Overload:** Many tools to learn and integrate
- **Training at Scale**:Training deep models on massive datasets
- **Dependencies**: Constantly changing data and shifting business needs.
- **Team Collaboration**:Aligning data scientists, ML engineers, DevOps, and business teams
- **Data Privacy and Governance:**Legal and ethical responsibility for sensitive data (e.g., health, finance)

Real-World Examples

- **Consumer Credit Risk Management:**
  - **Background:** Use case for credit scoring.
  - **Model Development:** Addressing bias and regulatory requirements.
  - **Deployment:** Strategies for ensuring compliance and robustness.

Real-World Examples

- **Consumption Forecasting:**
    - **Power Systems:** Use in predicting energy consumption.
        - **Implementation:** Challenges and deployment strategies.



Real-World Examples

- ***Marketing Recommendation Engines:***
    - **Rise of Recommendation Engines:** Importance in e-commerce.
    - **Model Deployment:** Handling scalability and real-time scoring.

Limitations of MLOps

Complexity: Implementing MLOps can be complex and require significant resources.

Cost: High initial setup costs for infrastructure and tools.

Skill Requirements: Requires a skilled team with expertise in data science, DevOps, and software engineering.

Maintenance: Continuous monitoring and maintenance are necessary to ensure optimal performance.