MLDM

MLOPS

Claudio Sartori

DISI
Department of Computer Science and Engineering – University of Bologna, Italy
claudio.sartori@unibo.it

What is MLOps?

- MLOps stands for Machine Learning Operations.
- It focuses on optimizing the end-to-end machine learning lifecycle.

Goals of MLOps

- Improve collaboration among teams
  - data scientists, developers, and operations
- Streamline and automate tasks in the ML workflow

1 Components of MLOps 4

2 The MLOps pipeline 18

3 What is good data for ML? 39

4 MLOps tools 43

5 Summary 54

Components of MLOps I

- Collaboration and Communication
  - Facilitate effective communication between teams
- Version Control
  - Track changes in code, data, and models for reproducibility
- Automation
  - Automate tasks like model training, testing, and deployment
- CI/CD Practices
  - Implement continuous integration and continuous deployment for seamless workflows.
- Monitoring and Logging
  - Use monitoring tools to track real-time model performance.

Components of MLOps II

- Model Registry
  - Catalog and manage different versions of trained models.
- Infrastructure as Code (IaC)
  - Define model deployment infrastructure in code for consistency.
- Security and Governance
  - Ensure security and compliance with regulations.

The Role of Automation in MLOps

- Reduces manual errors.
- Increases efficiency.
- Ensures consistency in deployment processes.

CI/CD Practices

Continuous Integration and Continuous Deployment

- Automates testing and deployment of models.
- Enables seamless integration of changes into production.

Real-time Monitoring and Logging

- Identifies issues and drift in data distributions.
- Monitors model performance in production environments.

Managing Model Versions - Model Registry

- Catalogs and manages different versions of trained models.
- Facilitates easy retrieval and deployment of specific model versions.

Infrastructure as Code (IaC) – Ensuring Consistency

- Defines infrastructure in code for reproducibility.
- Maintains consistency across different environments.

Security and Governance

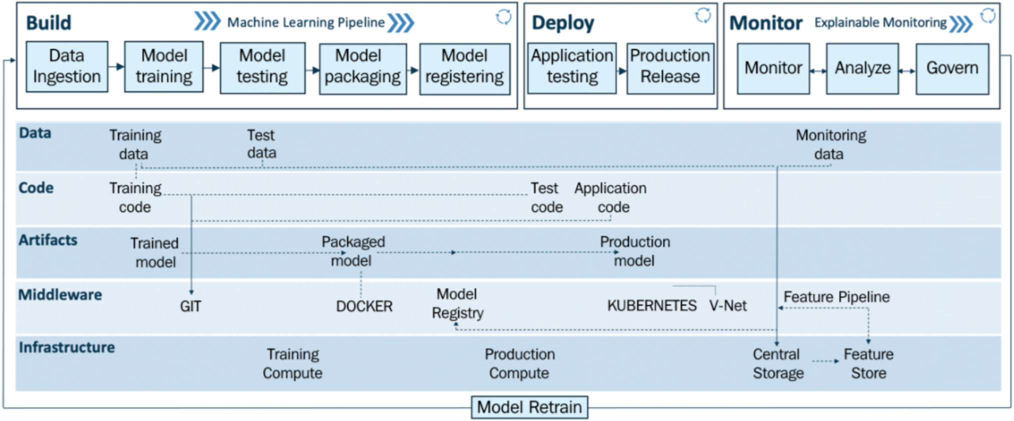
Protecting Data and Ensuring Compliance

- Considers security measures for sensitive data.
- Ensures compliance with regulations.

Benefits of MLOps

- Reliable and scalable deployment of machine learning models.
- Consistent and reproducible workflows.
- Enhanced collaboration and communication among teams.

MLOps Workflow - I



MLOps Workflow - II

- MLOps pipeline (build, deploy, and monitor)
    - the upper layer
- Drivers: Data, code, artifacts, middleware, and infrastructure
    - mid and lower layers

A use case

- operationalize[1] an image classification service to classify cats and dogs in a pet park in Barcelona, Spain.
- The service will identify cats and dogs in real time from the inference data coming from a CCTV camera installed in the pet park.
- The pet park provides you access to the data and infrastructure needed to operationalize the service.

[1] prototyping and deploying for production

Data and Infrastructure

- Data
  - The pet park has given you access to their data lake containing 100,000 labeled images of cats and dogs, which we will be used for training the model.
- Infrastructure
  - Public cloud (IaaS).

1 Components of MLOps 4

2 The MLOps pipeline 18
- Build 20
- Deploy 29
- Monitor 34

3 What is good data for ML? 39

4 MLOps tools 43

5 Summary 54

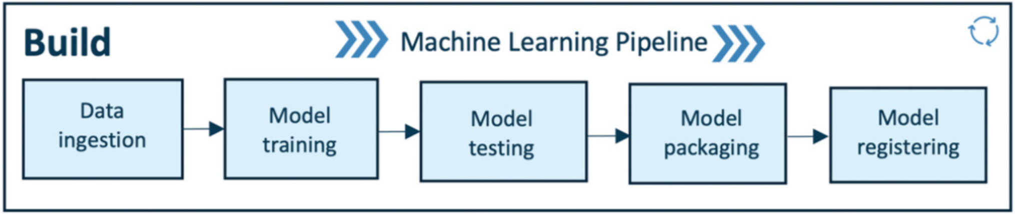
The MLOps pipeline

- operationalize[2] an image classification service to classify cats and dogs in a pet park in Barcelona, Spain.
- The service will identify cats and dogs in real time from the inference data coming from a CCTV camera installed in the pet park.
- The pet park provides you access to the data and infrastructure needed to operationalize the service.

[2] prototyping and deploying for production

Build

- core ML pipeline
  - training, packaging, and versioning the ML models
- includes computation facilities
  - for example, the CPU or GPU on the cloud or distributed computing



BUILD – Model training

- Modular scripts or code perform all the traditional steps in ML, before training or retraining any model
  - data preprocessing, feature engineering, and feature scaling
- The ML model is trained while performing hyperparameter tuning to fit the model to the dataset (training set).
  - efficient and automatic solutions such as Grid Search or Random Search exist.

Model training

Use case implementation

- We implement all the important steps to train the image classification model.
- For this case, we train a convolutional neural network (CNN[3]) for the image classification service.
- The following steps are implemented: data preprocessing, feature engineering, and feature scaling before training, followed by training the model with hyperparameter tuning.
- As a result, we have a CNN model to classify cats and dogs with 97% accuracy.

[3] https://towardsdatascience.com/wtf-is-image-classification-8e78a8235acb

BUILD – Model testing

- evaluate the trained model performance on a separated set of data points named test data
  - which was split and versioned in the data ingestion step.
- The inference of the trained model is evaluated according to selected metrics as per the use case.
- The output of this step is a report on the trained model's performance.

Model testing

Use case implementation

- In this case, we look for precision and the recall score
  - to validate the model's performance in classifying cats and dogs
  - to assess false positives and true positives to get a realistic understanding of the model's performance.
- If and when we are satisfied with the results, we can *proceed to the next step*
- or else *reiterate the previous steps* to get a decent performing model for the pet park image classification service.

BUILD – Model packaging

- the model can be serialized into a file
- or containerized (using Docker) to be exported to the production environment.

Model packaging

Use case implementation

- The model we trained and tested in the previous steps is serialized to an ONNX [4] file
- It is ready to be deployed in the production environment.

[4] https://onnx.ai

BUILD – Model registering

- The model that was serialized or containerized in the previous step is registered and stored in the model registry.
- A registered model is a logical collection or package of one or more files that assemble, represent, and execute your ML model.
- e.g. a classification model can be comprised of a vectorizer, model weights, and serialized model files.
- All these files can be registered as one single model.
- After registering, the model (all files or a single file) can be downloaded and deployed as needed.

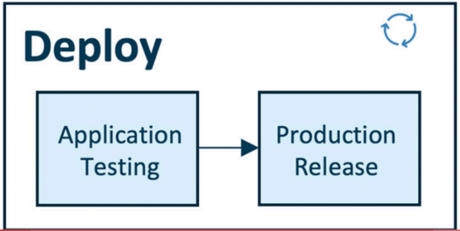
Model registering

Use case implementation

- The serialized model in the previous step is registered on the model registry and is available for quick deployment into the pet park production environment.
- By implementing the preceding steps, we successfully execute the ML pipeline designed for our use case.
- As a result, we have trained models on the model registry ready to be deployed in the production setup.
- Next, we will look into the workings of the deployment pipeline.

Deploy

- Make the ML models operational
- Test model performance and behavior in a production or production-like (test) environment to ensure the robustness and scalability of the ML model for production use.
- Streamlined CI/CD pipelines connecting the development to production environments.



Deploy - Application testing

- Rigorously test all the trained models for robustness and performance in a production-like environment called a test environment.
- Deploy the models in the test environment.
- Perform predictions using test data
    - not used for training the model; test data is sample data from a production environment
- Performance results are automatically or manually reviewed by a quality assurance expert.

Deploy - Application testing

Use case implementation

- We deploy the model as an API service on an on-premises computer in the pet park, which is set up for testing purposes.
- This computer is connected to a CCTV camera in the park to fetch real-time inference data to predict cats or dogs in the video frames.
- The model deployment is enabled by the CI/CD pipeline.
- Test the robustness of the model in a production-like environment, that is, whether the model is performing inference consistently,
- Testaccuracy, fairness, and error analysis.
- At the end of this step, a quality assurance expert certifies the model if it meets the standards.

Deploy - Production release

- Previously tested and approved models are deployed in the production environment for model inference to generate business or operational value.
- This production release is deployed to the production environment enabled by CI/CD pipelines.

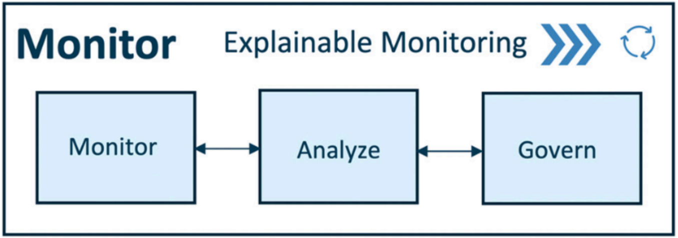
Deploy - Production release

Use case implementation

- We deploy a previously tested and approved model (by a quality assurance expert) as an API service on a computer connected to CCTV in the pet park (production setup).
- This deployed model performs ML inference on the incoming video data from the CCTV camera in the pet park to classify cats or dogs in real time.

Monitor

- Works in sync with the deploy module
- Pre-defined metrics
- Telemetry data
- Explainability methods



Monitor

- Capture critical information to monitor data integrity, model drift, and application performance

Analyze I

- Statistical properties of the target variable
- Explainability techniques
- Over time, the statistical properties of the target variable can change in unforeseen ways.
- This change is called model drift
  - For example, in a case where we have deployed a recommender system model to suggest suitable items for users, user behavior may change due to unforeseeable trends that could not be observed in historical data that was used for training the model.
- It is essential to consider unforeseen factors to ensure deployed models provide the best and most relevant business value.

Analyze II

- In case of model drift
  - alert the stakeholders or the quality assurance officier
  - switch or update the models
  - re-train the pipeline

Govern

- Generate alerts to trigger govern actions
- Compliance with the laws
    - explainability
    - transparency
    - end-to-end traceability

What is good data for ML?

- Good ML models are a result of training on good-quality data.
- A pre-requisite is to have good-quality data.
- We need to process the data to increase its quality.
- Determining the quality of data is essential.
- Five characteristics

Five characteristics of good data for ML I

- Accuracy
  - Inaccurate data lead to poor ML models
  - Confirm wether the data represent adequately the real world
- Completeness
  - Check the comprehensiveness of the data.
  - e.g. consider the coverage of time and space dimensions, population, product families . . .

Five characteristics of good data for ML II

- Reliability
  - Contradictions and duplications
  - Bias and distributions
- Relevance
  - with respect to the ML task at hand
- Timeliness

TensorFlow Extended (TFX)

- Developed by Google for TensorFlow users.
- End-to-end platform for deploying production-ready machine learning models.
- Supports component-based architecture for flexibility.
- Integrates with Apache Airflow for orchestration.

MLflow

- Open-source platform developed by Databricks.
- Designed for managing the end-to-end machine learning lifecycle.
- Components include Tracking, Projects, Models, and Registry.
- Language-agnostic and supports multiple ML libraries.

Kubeflow

- Kubernetes-native platform for deploying and managing ML models.
- Supports end-to-end orchestration, from data preprocessing to model deployment.
- Integrates various tools like TensorFlow, PyTorch, and Jupyter notebooks.
- Extensible and customizable for different workflows.

Apache Airflow

- Open-source platform for orchestrating complex workflows.
- Widely used in MLOps for scheduling and monitoring data workflows.
- DAG (Directed Acyclic Graph) structure for defining workflows.
- Integrates with various databases, storage systems, and ML platforms.

DataRobot

- Automated machine learning platform.
- Provides end-to-end automation from data preparation to model deployment.
- Focus on democratizing machine learning for users with varying skill levels.
- Supports collaboration and model explainability.

Domino Data Lab

- Collaboration platform for data mining and MLOps.
- Enables reproducibility and collaboration in a centralized environment.
- Supports version control, experiment tracking, and model deployment.
- Integrates with popular data mining tools.

Azure Machine Learning

- Microsoft's cloud-based MLOps platform.
- Integrates with popular Azure services for data storage, compute, and deployment.
- Provides tools for model training, deployment, and monitoring.
- Supports a wide range of frameworks and languages.

DVC (Data Version Control)

- Open-source version control system for machine learning projects.
- Focuses on managing datasets and models in a reproducible way.
- Works well with existing version control systems like Git.
- Supports experimentation tracking and collaboration.

Seldon Core

- Open-source platform for deploying, scaling, and managing machine learning models on Kubernetes.
- Supports a wide range of ML frameworks.
- Provides features for A/B testing, canary deployments, and scaling.
- Enables production-ready model serving.

Algorithmia

- MLOps platform for deploying, managing, and scaling machine learning models.
- Supports model deployment via REST APIs.
- Provides model versioning, monitoring, and collaboration features.
- Focuses on making models available as microservices.

Differences between CRISP-DM and MLOps I

- Purpose and Focus
  - CRISP-DM Structured methodology for data mining projects, covering the entire project lifecycle.
  - MLOps Focuses on the operationalization and management of machine learning models in production.

Differences between CRISP-DM and MLOps II

- Activities
  - CRISP-DM Involves business understanding, data preparation, modeling, evaluation, and deployment.
  - MLOps Involves model deployment, continuous integration, continuous delivery, monitoring, and managing the machine learning model lifecycle.
- Methodological vs. Operational
  - CRISP-DM Methodological framework for structuring the data mining process.
  - MLOps Operational practices and tools for managing machine learning models in production.

Differences between CRISP-DM and MLOps III

- Timing
  - CRISP-DM Applied throughout the entire data mining project.
  - MLOps Becomes prominent once the machine learning model is ready for deployment and continues throughout its operational lifecycle.
- Collaboration
  - CRISP-DM Emphasizes collaboration between business stakeholders, data scientists, and domain experts.
  - MLOps Encourages collaboration between data scientists, developers, and operations teams for deployment and maintenance.

CRISP-DM and MLOps – Summary

- Distinct concepts addressing different aspects of the data mining and machine learning lifecycle
- CRISP-DM guides the entire data mining project
- MLOps specifically addresses the operational aspects of managing machine learning models in production
- They are complementary in an end-to-end workflow

Bibliography

- Raj, E. (2021).
  Engineering MLOps : Rapidly Build, Test, and Manage Production-ready Machine Learning Life Cycles at Scale.
  Packt Publishers.