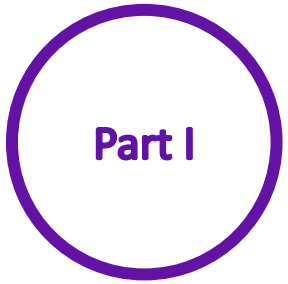
PROPHECY LABS

AI4Business MLOps and Monitoring

Roadmap AI4Business



Introduction to AI

Developing AI tools

Data and Value

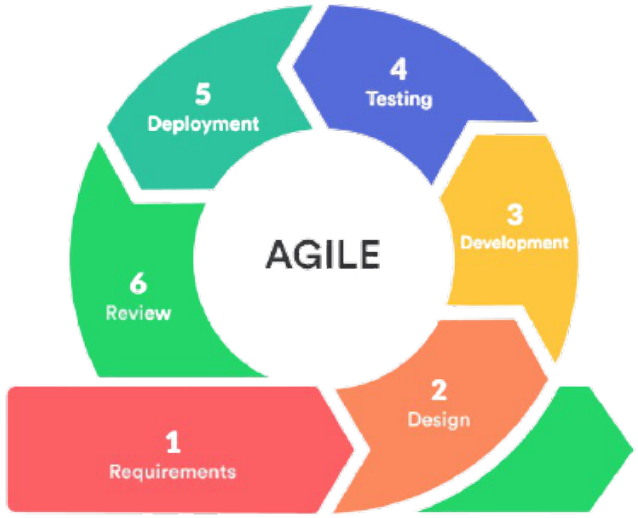
Deploying AI

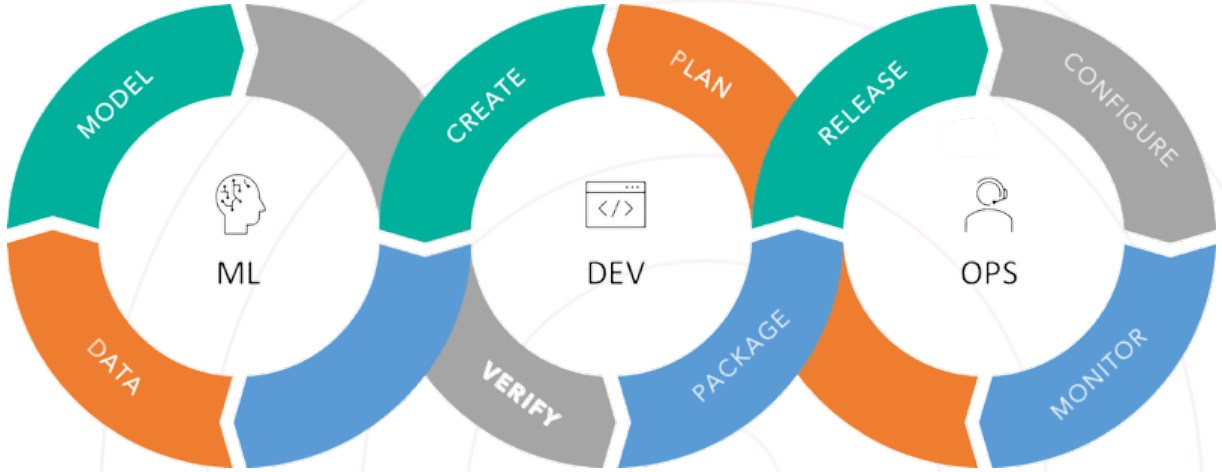
MLOps and Monitoring

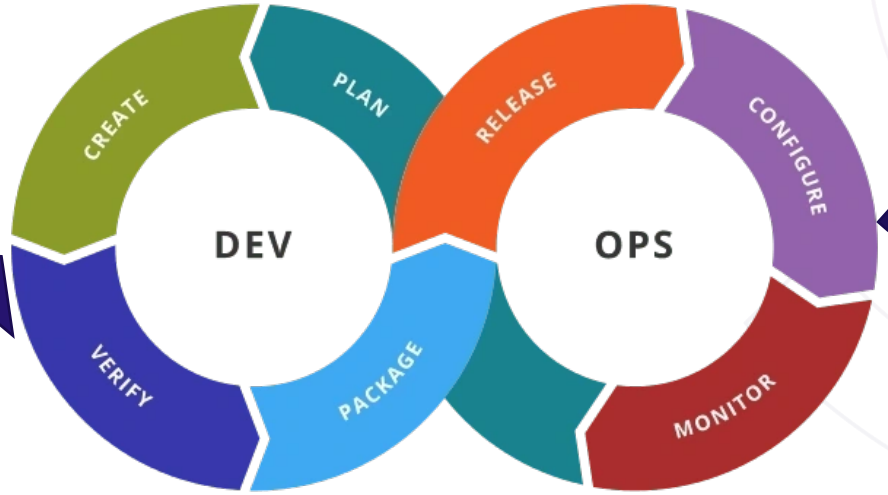
Table of contents

1 Introducing MLOps

The road so far







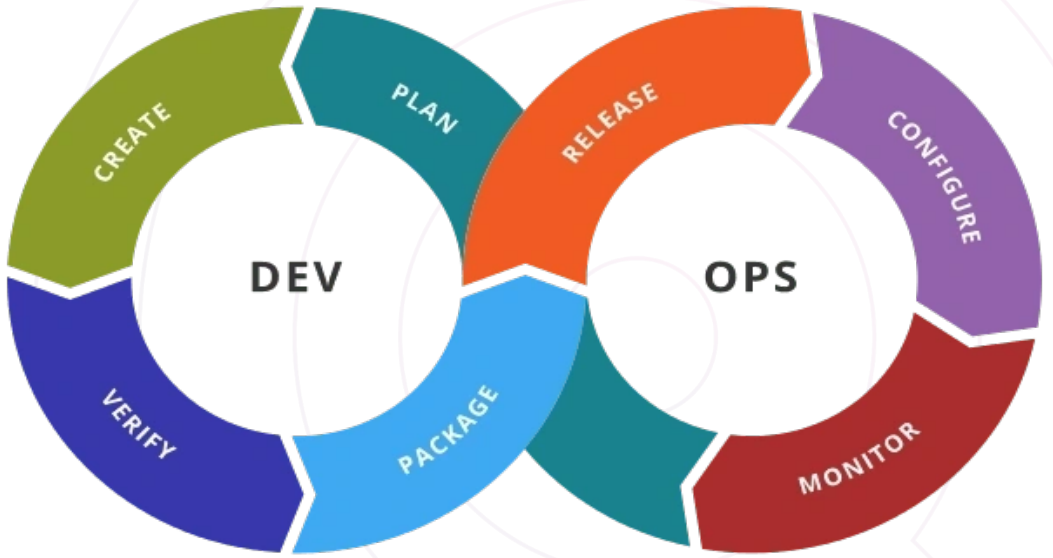
DevOps

What is DevOps?

Tools and best practices to improve software development process and operations

Advantages:

- Speed
- Rapid Delivery
- Reliability
- Improved collaboration
- Security



Continuous improvement

CI/CD Deliver faster and better

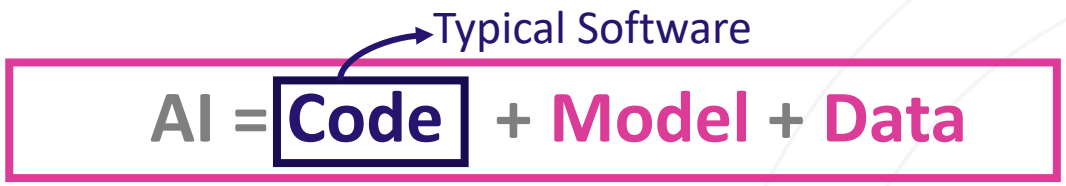
Continuous Integration

- Allows small incremental improvements
- Automates the build, test, and packaging of applications in a reliable and repeatable way.
- Streamlines code changes
- Set of practices performed as developers are writing code

Continuous Delivery

- Automated delivery of code
- Allows for continuous deployment
- Set of practices performed after the code is completed

Is DevOps enough?



Typical software developer's flow

Save code changes → Refresh → Changed

Typical data scientist's flow

Save code change → Spin a cluster → Deploy code
→ Transfer Data → Model Training

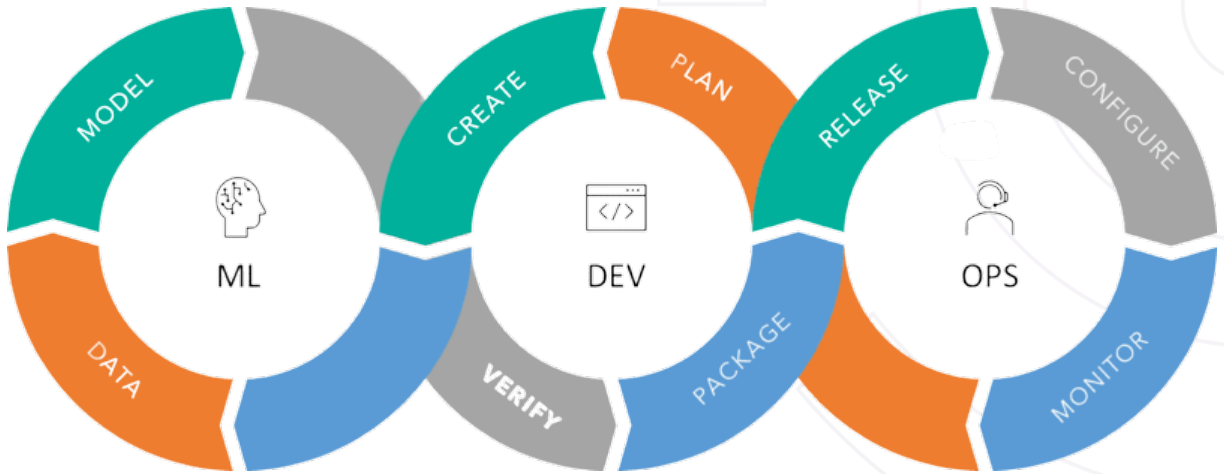
Why is DevOps not enough?

Traditional software	Machine learning systems
Only Code	Code + Data
Easy to get feedback	Code change → retrain model
Code is relatively static	Models learn constantly

Enters MLOps

Machine Learning Operations (MLOps)

Practice that aims to make developing and maintenance of machine learning systems in production smooth and efficient, augmenting their long-term value while reducing the risk associated it with it.



MLOps 4 pillars

Collaborative

Reproducible

Scalable

Continuous

Collaborative

Collaborative ML

- MLOps ensures that all steps in the ML system are transparent
- MLOPs eases collaboration through all the project life cycle

Visible ✓ ***Auditable*** ✓

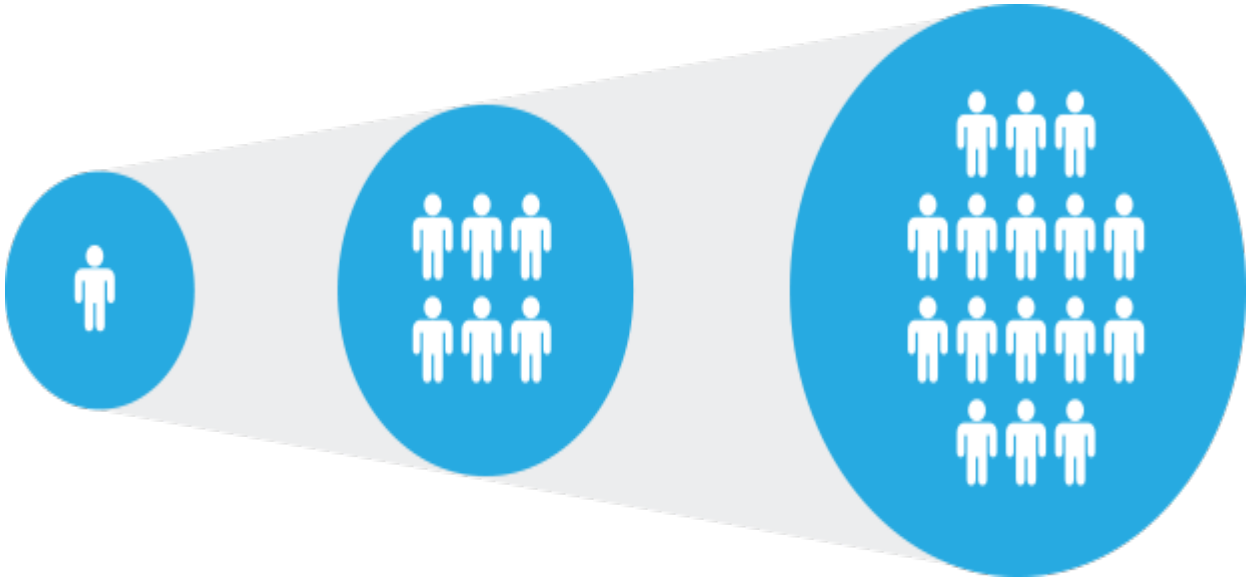
Reproducible

Reproducible ML

- MLOps enforces storing of all artifacts
- Versioning more than code: data, models, meta data, etc.

Process ≠ Experiment

Scalable

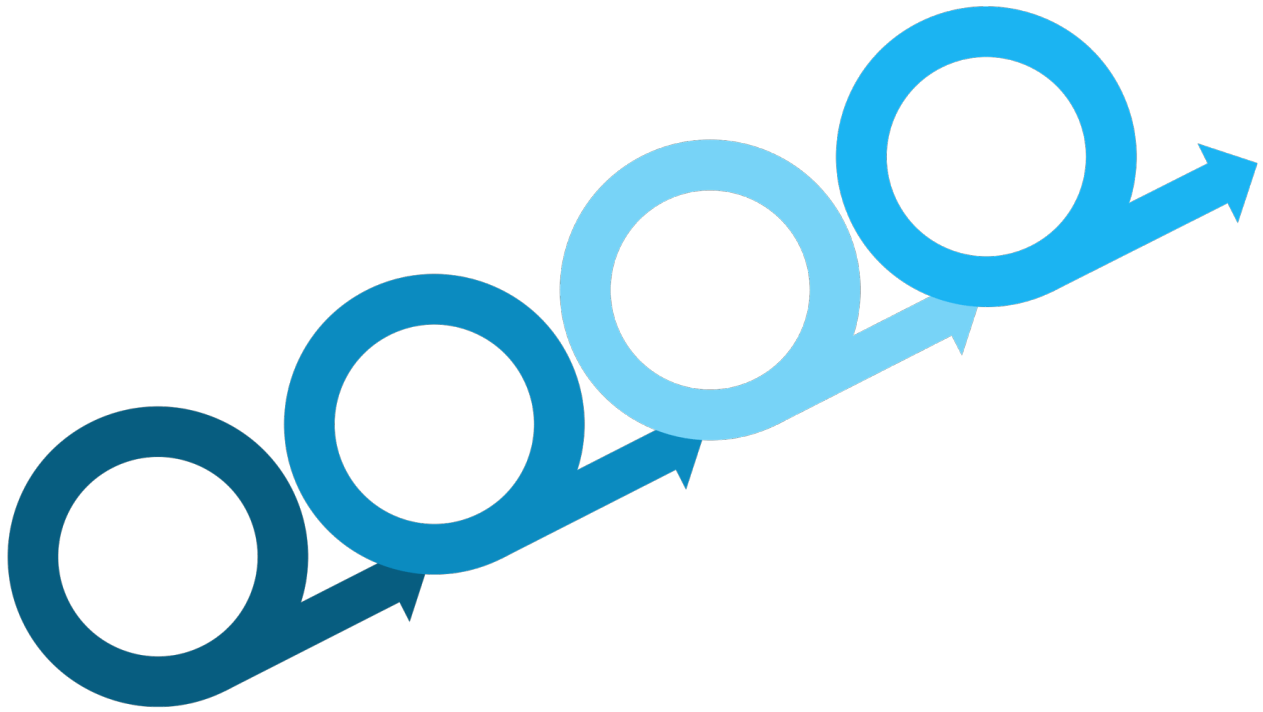


Scalable ML

- MLOps eases expansion of infrastructure to scale projects
- Volumes of data can grow quickly, so the set up should grow naturally

Natural growth

Continuous



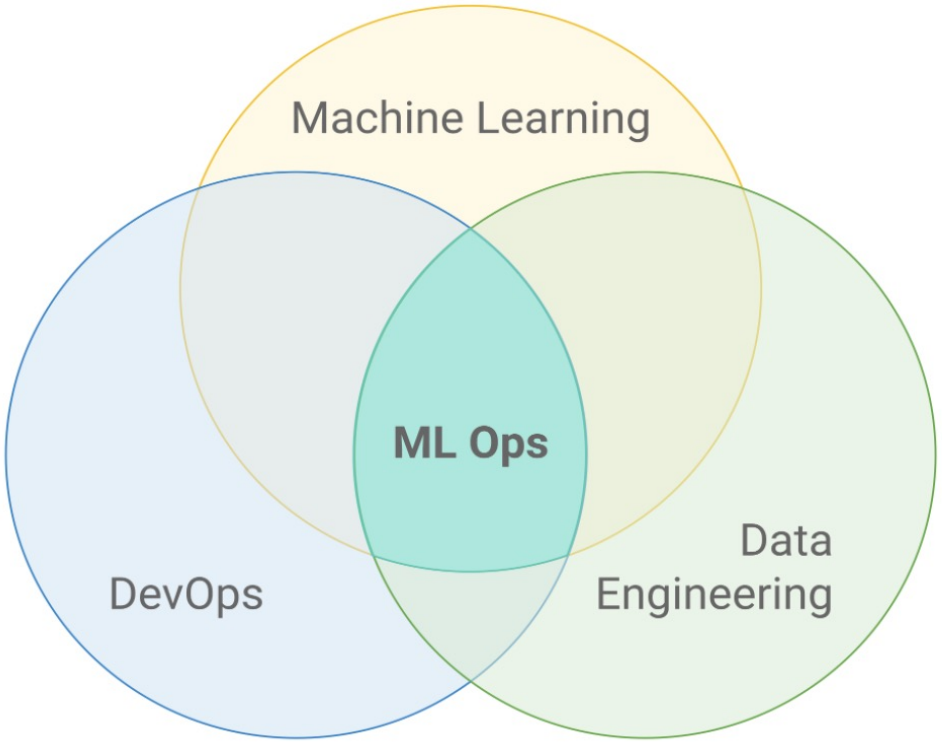
Continuous ML

- Ensure a CI/CD process is central to MLOps
- ML should be thought as a continuous process
- Retraining models should be effortless

Ad-hoc < Automated

2 Value of MLOps

We need MLOps



ML systems are complex

- Models don’t last forever
- Many teams need to collaborate
- ML systems change with data
- Need to monitor many aspects

MLOps is growing

Forbes: by 2025 the MLOps business will grow to $4 billion

Not a software solution

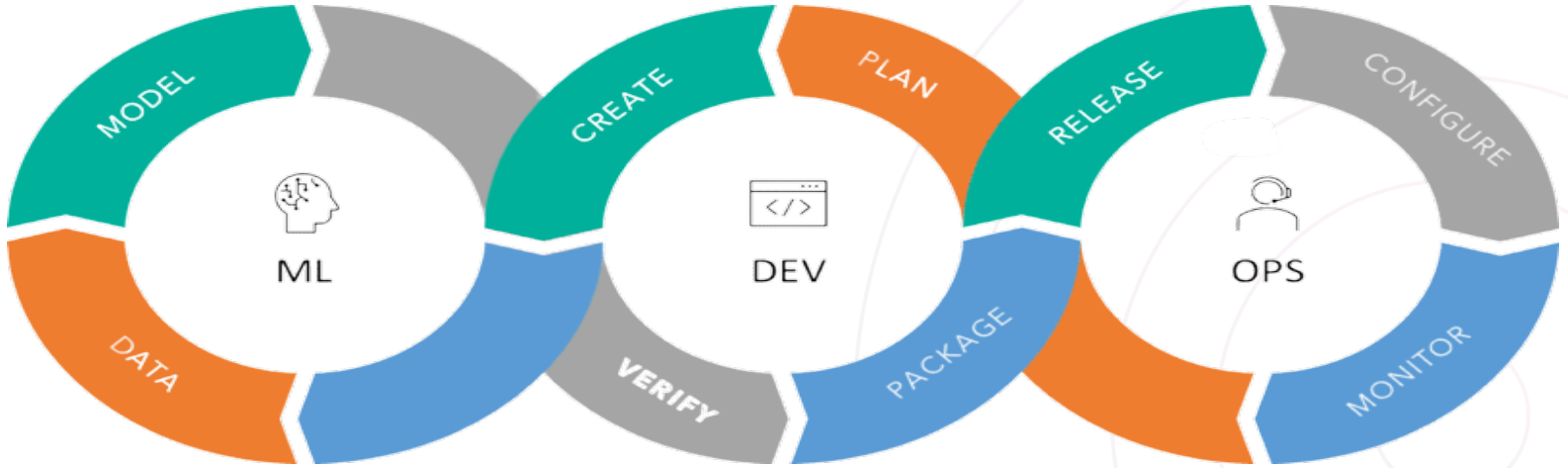
MLOps is not any particular software solution but:

Tools + Best Practices

Focus on:

- Speed up innovation through experimentation
- Create reproductible workflows
- Streamline the deployment process
- Manage the complete ML project lifecycle

Speeding up innovation



How:

- Brings all development and operational teams together
- Focuses on incremental improvements in development
- Uses monitoring validation and management systems to check progress

Reproductible workflows

Why do we need it?

- Reduce variability across iterations
- Resiliency to failure
- Smart copy

How do we achieve it?

- Focus on traceability
- Enforce the use of abstractions (e.g. pipelines)
- Track resources (data and models)

Streamline deployment

CI/CD is your friend

- MLOps enforces the good use of CI/CD principles
- Continuous and traceable improvement/changes

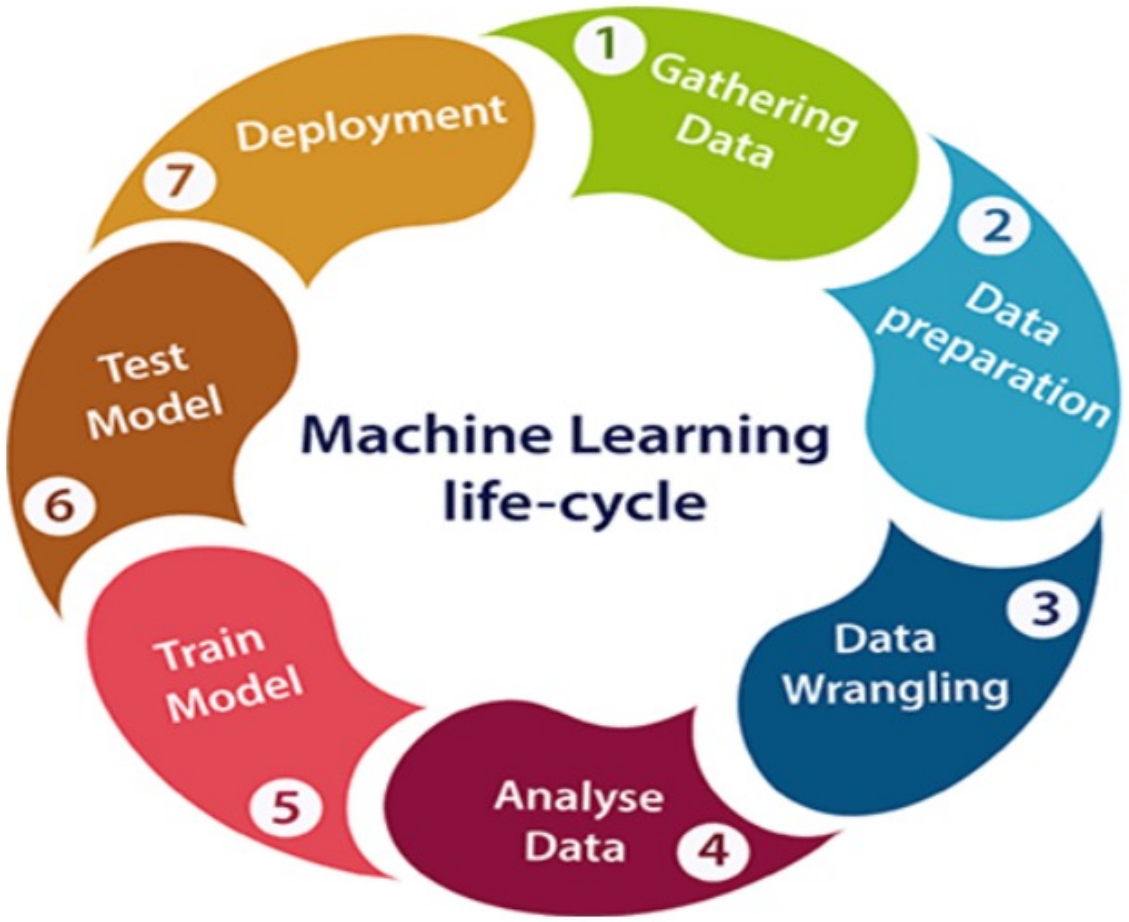
Manage infrastructure

- MLOps also aims to manage the infrastructure
- Enforces scalability and portability

Manage Configurations

- Abstracts configurations
- Making changes in a small configuration does not imply refactoring

Manage model lifecycle



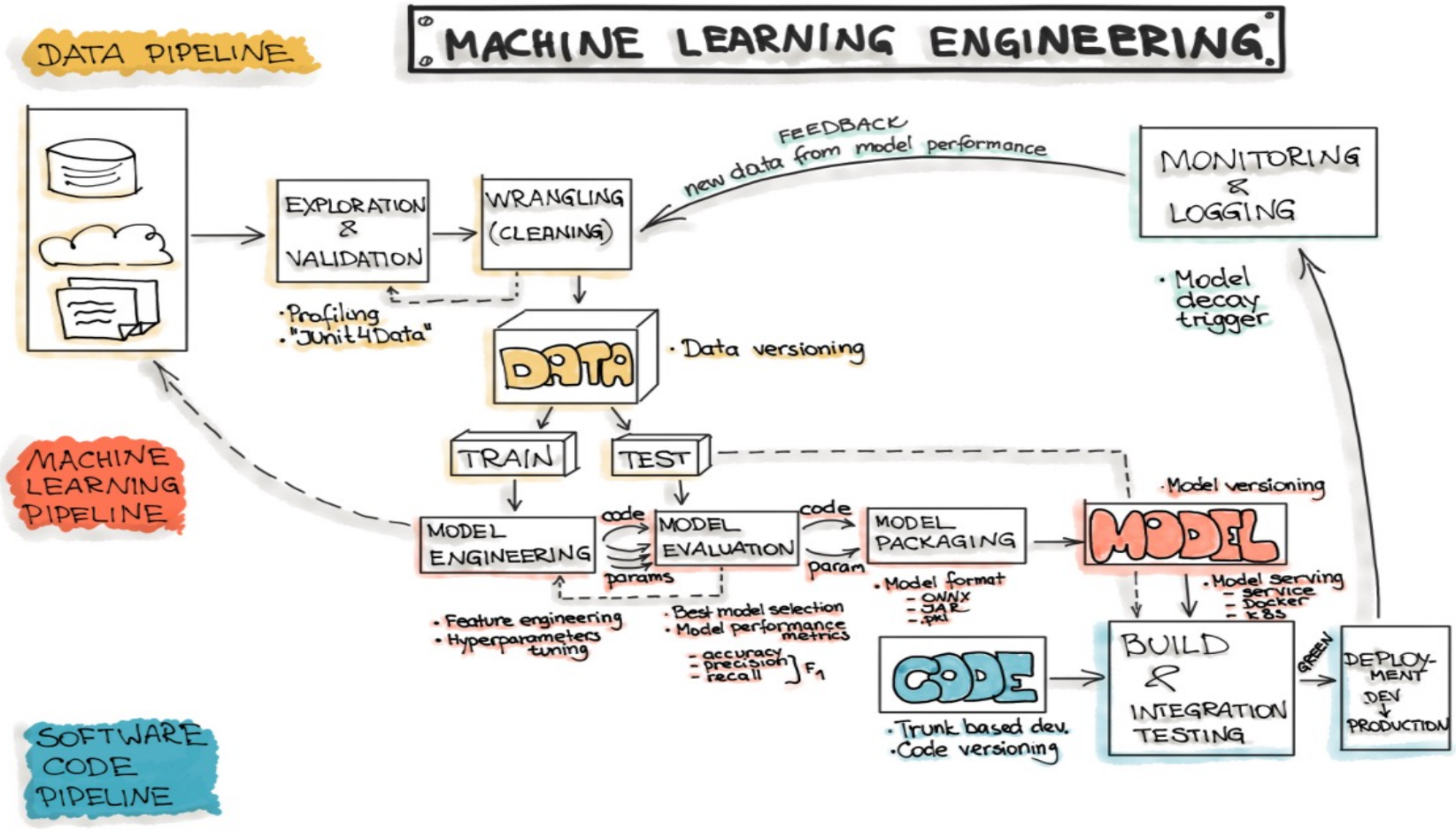
The picture is pretty, however

- In reality this order is often broken
- Each step can be complex

MLOps to the rescue

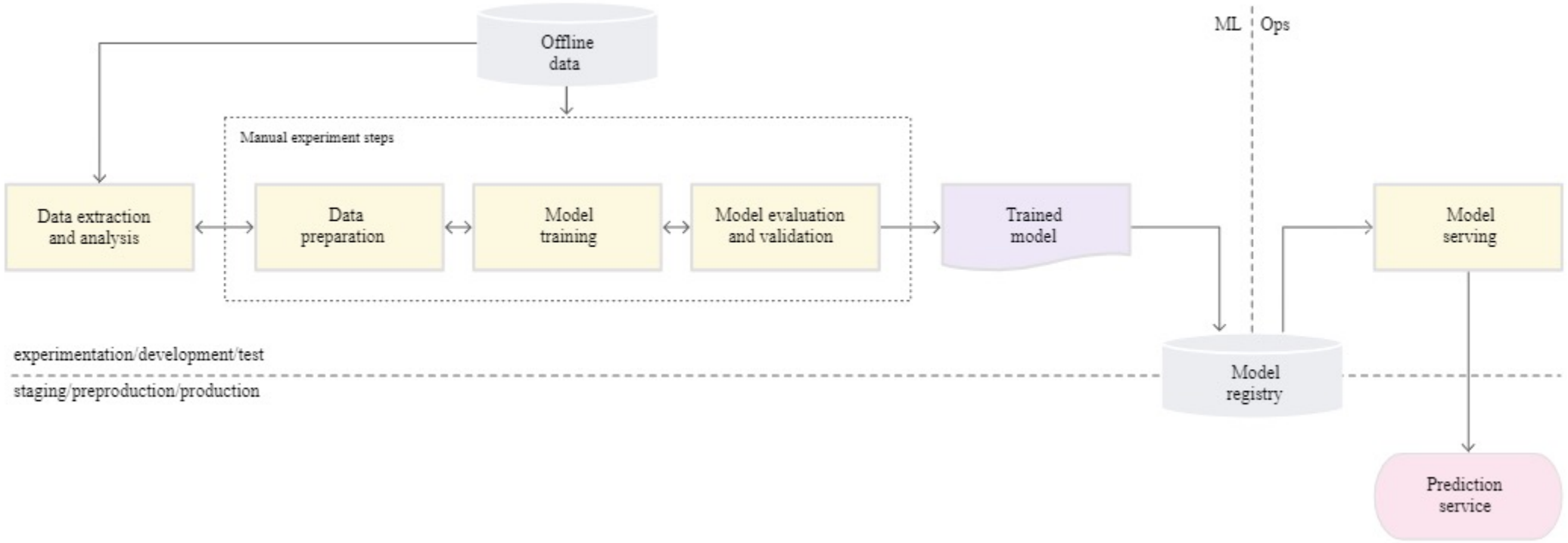
- Automate and manage the workflow
- Experiment tracking
- Guidelines for cross-functional teams
- Integrate experimentation with deployment

Manage the project lifecycle



But is it really necessary

I have a manual process that looks like this



Google - CD and automation pipelines in ML

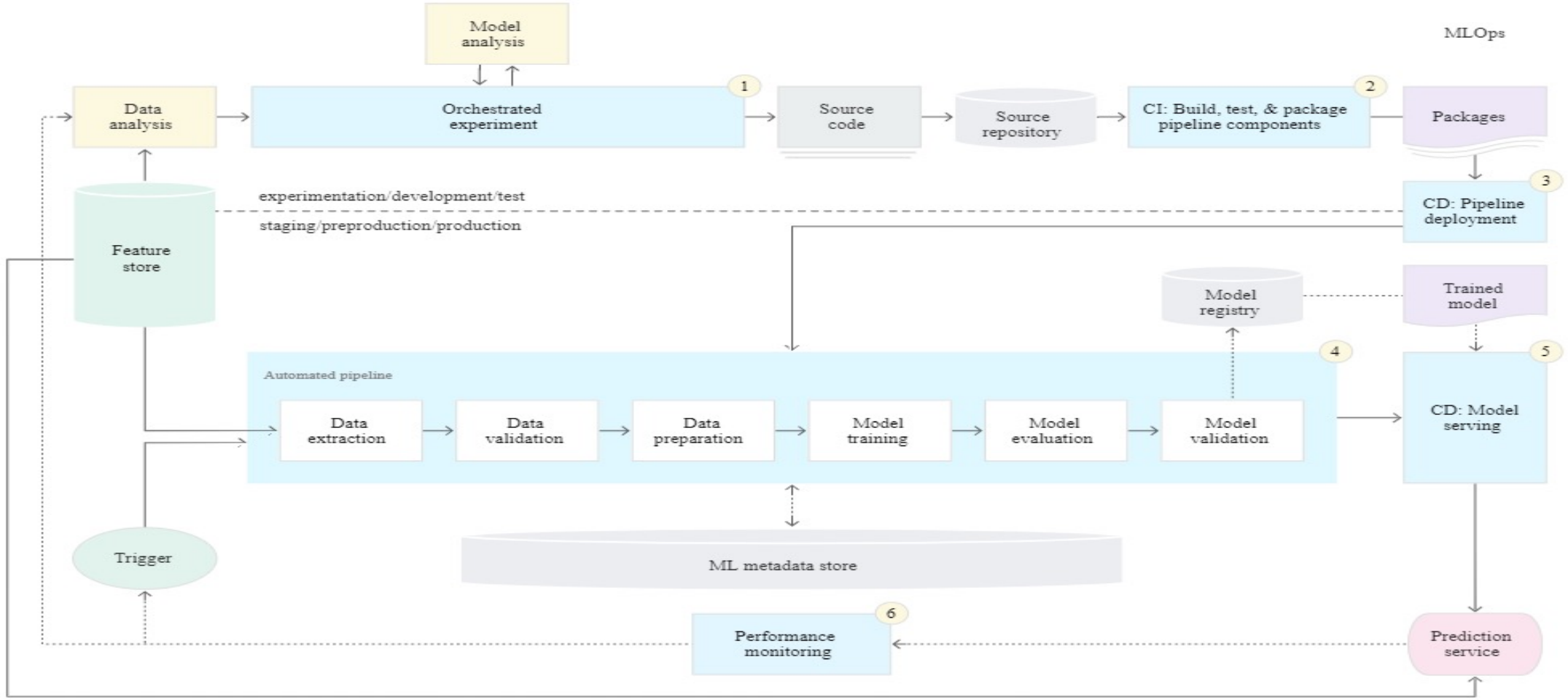
Even if you do

Even if:

- You have very competent data scientists
- You have a very solid IT department
- They know how to work well together
- They are able to automate model predictions
- And …..

You will still need a permanent reproducible record of what happened at every single step + processes to maintain and monitor your ML system

Mature framework



Google - CD and automation pipelines in ML

Reduce technical debt

MLOps main goal is to reduce technical debt

Some of the question posed by **MLOps**?

- How easily can a new algorithmic approach be tested at full scale?
- What is the transitive closure of all data dependencies?
- How precisely can the impact of a new change to the system be measured?
- Does improving one model or signal degrade others?
- How quickly can new members of the team be brought up to speed?

Address regulatory challenges

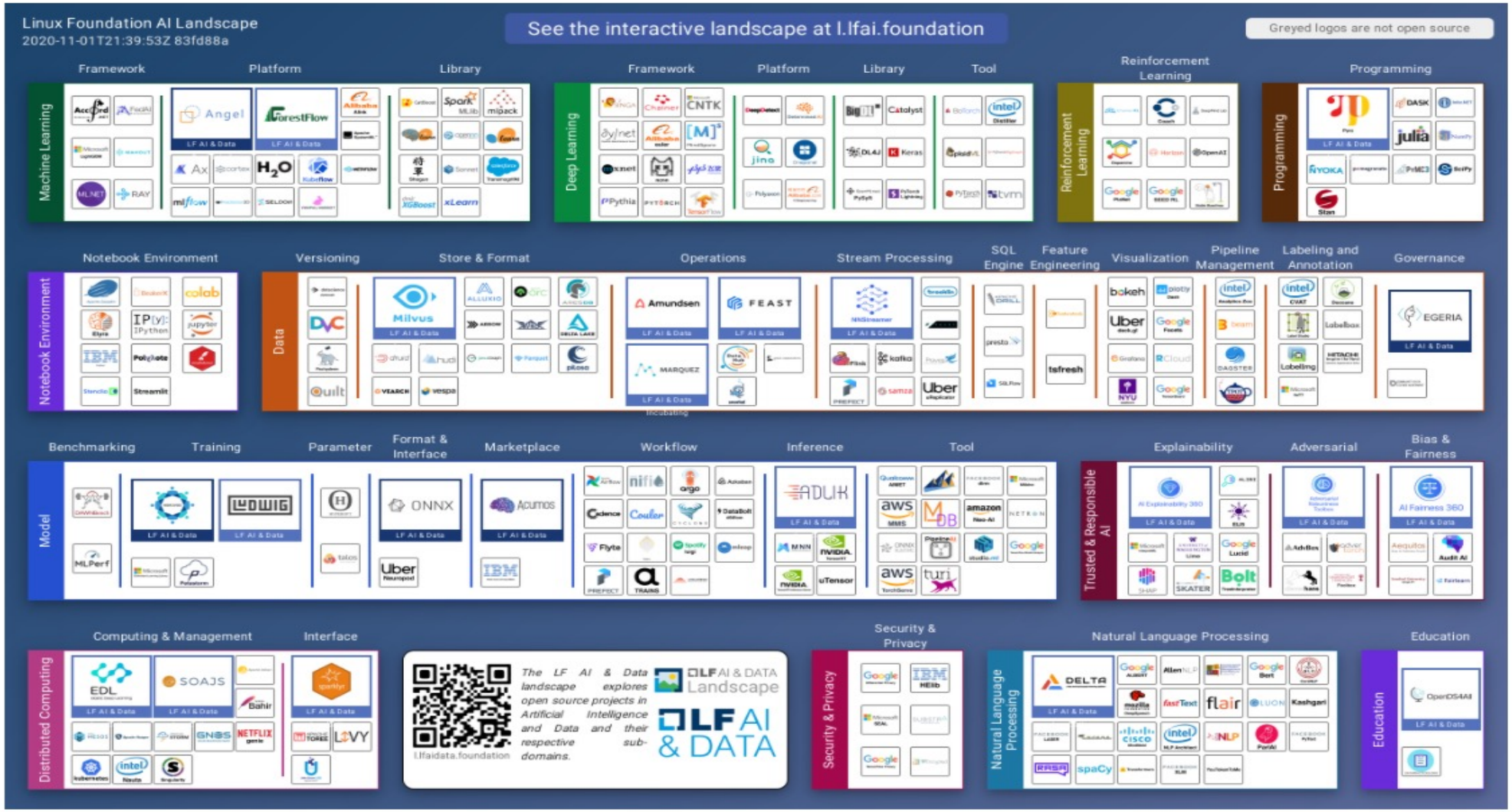
Regulation is increasing

- GDPR
- European legal framework for AI

MLOps related to regulation?

MLOps puts the operations team at the forefront of regulation and best practices

MLOps has a price



Institue for ethical AI


3 MLOps Myths

MLOps myths

5 common MLOps myths

MLOps is …

1. Easy
2. Only useful at scale
3. Too expensive
4. The same as DevOps
5. Not requiring governance

MLOps is easy

There is more to MLOps than deployment

- Many factors dictating the success of ML systems
- Hidden technical debt accumulates over time

There are many factors at play

- Maintaining a model entails maintaining all its components
- Models don’t live in an island
- Governance and DevOps integration

DIY culture is detrimental for MLOps

- Is expensive to have all expertise in-house
- MLOps technology is advancing extremely fast

MLOps is only useful at scale

Same issues apply regardless of amount of models

- Good practices should always be present
- Resources might differ but the need for processes is the same

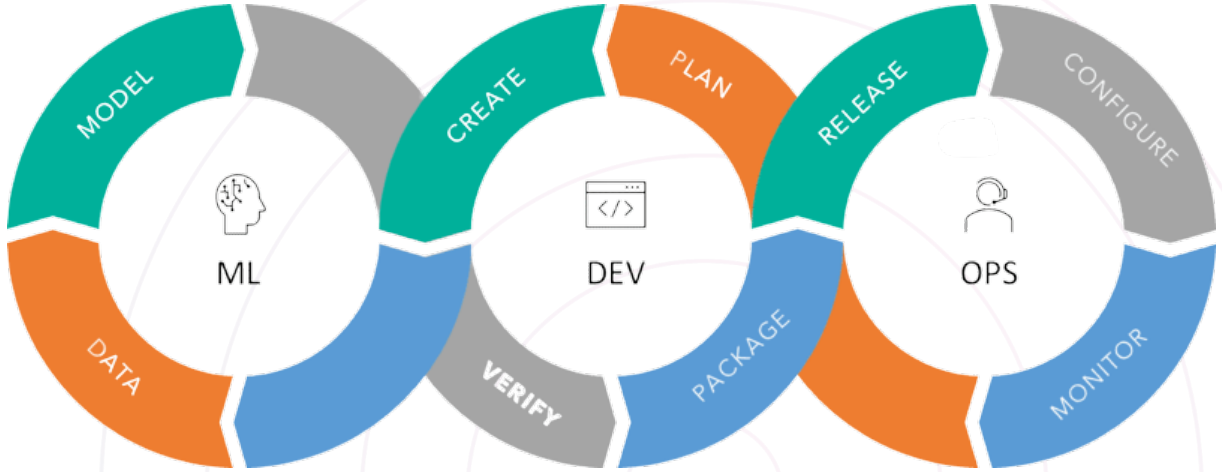
Develop with the future in mind

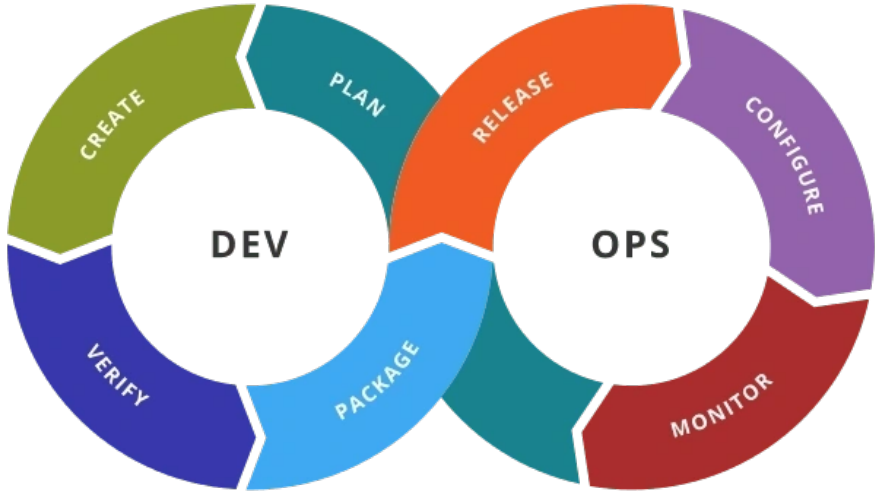
- Having robust pipelines will lower maintenance cost
- Monitoring and auditing is almost impossible
- Processes are easy to smart copy

Maintenance cost grows exponentially

- If deployment and experimentation is not streamlined finding errors is difficult
- Everything becomes an ad-hoc task

MLOps is the same as DevOps





Conceptually similar but …

MLOps is much more **experimental**

Performance **degradation** of the system

Model **monitoring**

Team composition:

Testing involves model validation

MLOps is expensive

Better to have and not need than need and not have

You can't afford not to have it

- Adapt rapidly to environmental changes
- Maintainability and auditability is cumbersome without it
- Not only about the tools but also the processes
- Other risks (e.g., operational, reputational)

It becomes more expensive over time

- The more you wait the more difficult is to implement it
- MLOps is not an expense but an investment

MLOps doesn't require governance

ML Governance refers to the policies that organizations set around their models and overall machine learning platforms.

ML systems are complex

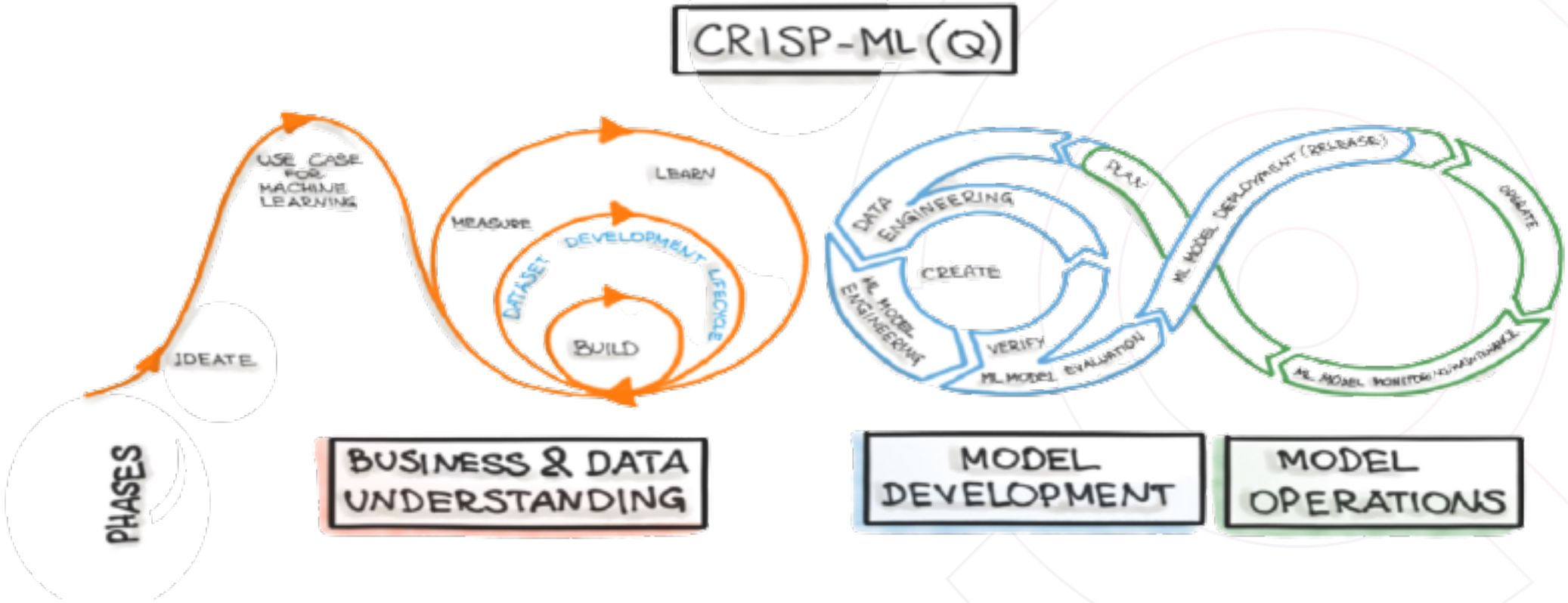
- Require data, maybe subject to regulation
- Tend to degenerate over time
- Security issues around them
- Clients might interact or be affected by them

With MLOps, governance isn't manual

- Tools to support this task

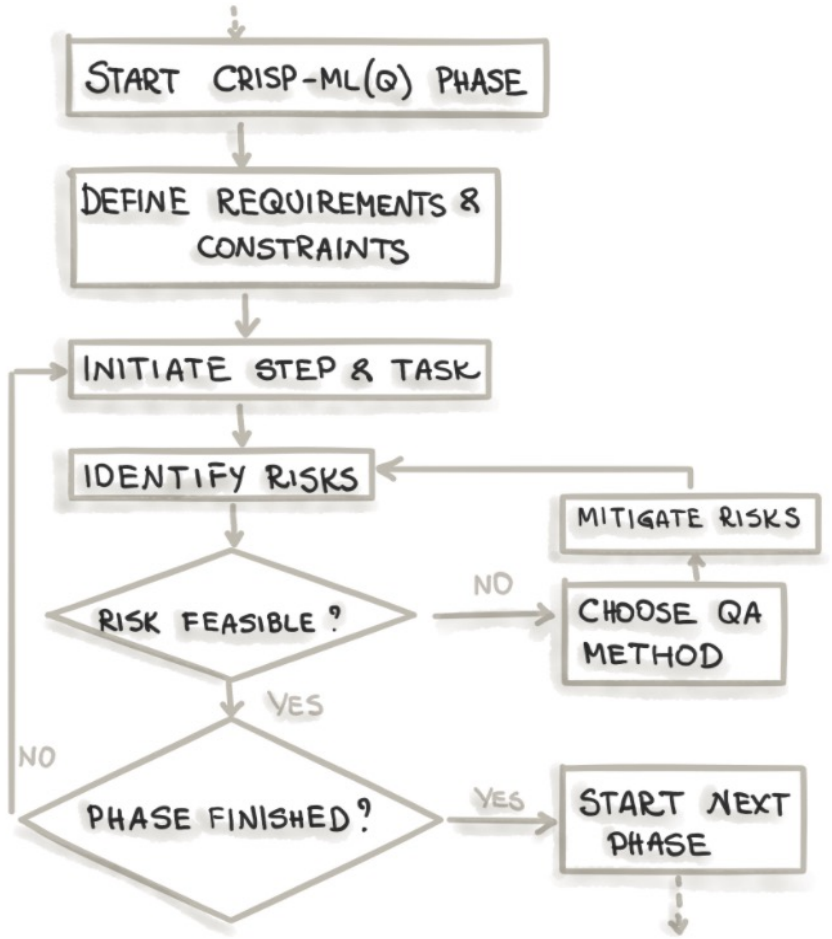
4 CRISP – ML(Q)

Introducing CRISP - ML(Q)



Google - CD and automation pipelines in ML

CRISP-ML(Q) Phases



Six phases:

- Business and Data Understanding
- Data Engineering (Data Preparation)
- Machine Learning Model Engineering
- Quality Assurance for Machine Learning Applications
- Deployment
- Monitoring and Maintenance

Business and Data Understanding

From **Business to ML** objectives

Defining the scope

- Data scientist and Business

Establishing a success criteria

- Business, ML and Economic

Feasibility

- Applicability of ML technology, legal constraints and requirements of the application

Data Collection

- Having the data is required before starting as well as data version control

Data Quality Verification

- Data description + requirements + verification

Data Engineering

Data preparation serves the purpose of producing a data set for the subsequent modelling phase.

Data Selection

- Feature selection, data selection, unbalanced classes

Clean Data

- Noise reduction, data imputation

Construction of Data

- Feature engineering, data augmentation

Standardization of data

- Data format, normalization

Machine Learning Engineering

The goal of the modelling phase is to craft one or multiple models that satisfy the given constraints and requirements

Literature research

- Screen the literature, search for baselines, don't re-invent the wheel

Quality measures and Model selection

- Use the right measures and models for the problem at hand

Domain knowledge

- Incorporate the domain knowledge available, only if improves performance

Reproducibility

- Method and results reproducibility is non-negotiable

Experimental documentation

- Keep track of the changed model performance

Quality Assurance

Validate performance

- Generalization performance on a test set

Determine Robustness

- Real life data can be noisy

Increase explainability for ML practitioner and end user

- Explainable models are easier to improve and more likely to be accepted

Compare result with success criteria

- If success criteria not met, one should backtrack to previous phases

Deployment

The deployment phase of a ML model is characterized by its practical use in the designated field of application

Select right architecture

- Select right architecture for your models, scalability is paramount

Model evaluation under production conditions

- Production data and environment might widely differ from development

User acceptance and usability

- Model might still be unusable, incomprehensible or susceptible to outliers

Minimize risks of unforeseen errors

- Have a fall back plan in case model fails

Deployment strategy

- Deployment should be incremental

Monitoring and Maintenance

The risk of not maintaining the model is the degradation of the performance over time which leads to false predictions

Data drift over time

- Input data is not always similar to training data

ML systems require monitoring

- ML systems are complex, with many possible points of failure

Models need to be updated

- Performance of model deteriorates over time, so they need to be updated

Is not always about the data

- Technical monitoring is also a must

5 Observability

Motivation

What happens if your AI starts making **bad decisions**?

How would you find out?

When would you find out?

Would you know **why**?

Can you trust AI?

Why ML fails in production

Feedback Loops between ML and Environment

Drift in population

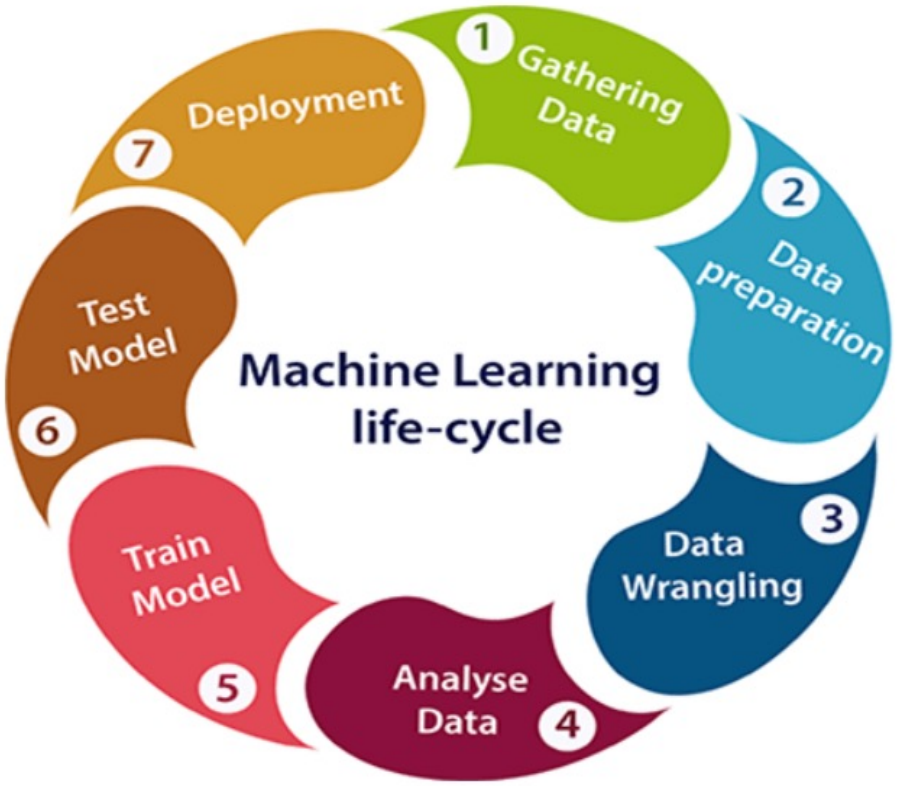
Noise

Random Shocks

Long cycles

AI Observability

Observability is the practice of obtaining a deep understanding into your model's performance across all stages of the model development cycle



Timely Detection

- Only solve after identification
- Reducing identification cost
- Production and development

Timely resolution

- Diminish time of resolution
- Identify root cause
- Provide a course of resolution

Observability components

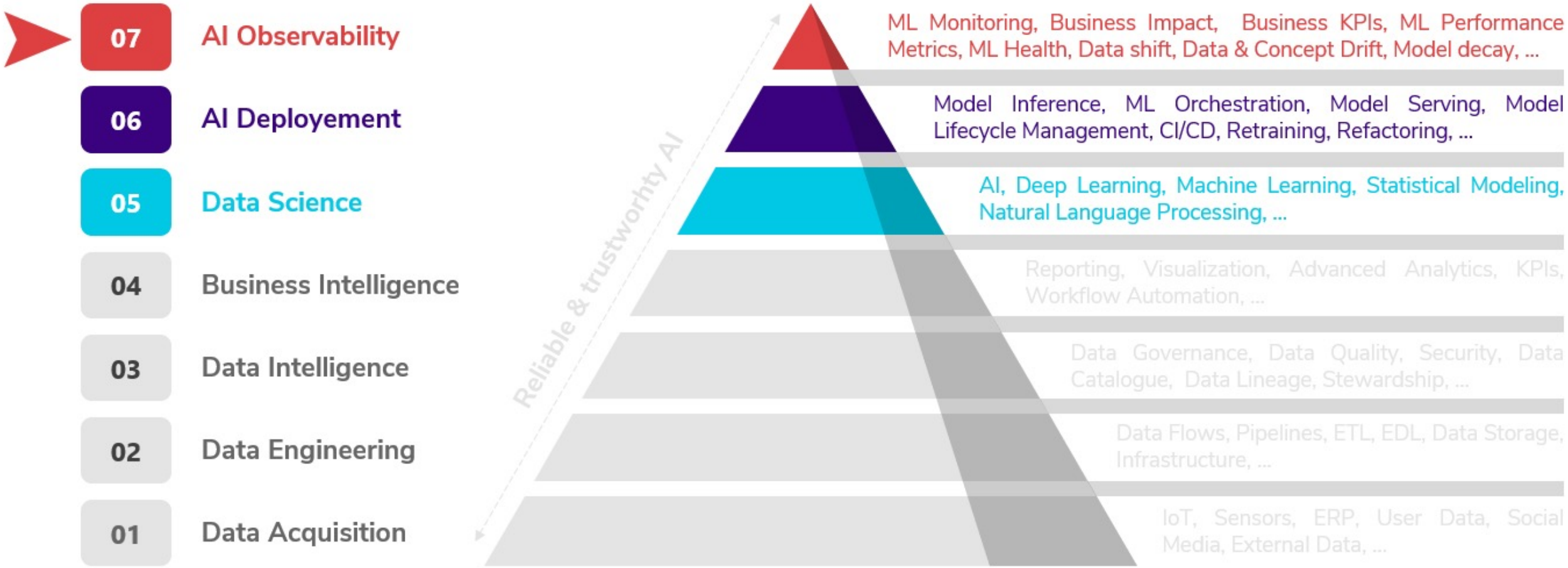
Monitor drift

Root cause analysis

Performance metrics

Feedback loops

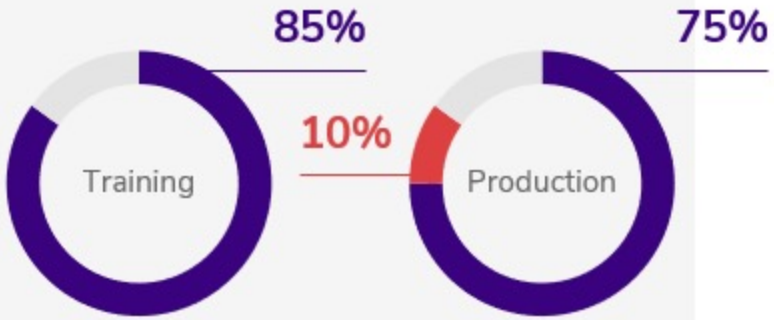
ML hierarchy of needs



6 Monitoring

AI algorithms deteriorate over time

AI performance drop



The first results after having deployed an AI model, might not live up to the expected training performance. Besides modeling mistakes like overfitting etc., **under specification** or **data shift** might have caused the AI models to fail instantly when put in production.

AI performance decay

It is also possible that AI models only start failing after being deployed for while. This might be caused by **data drift**, where the statistical properties of the model input change. Or **concept drift** where the statistical properties of the outcome AI models are trying to predict fundamentally change over time.

Business impact decline

Often business decisions are taken based on AI generated output. Hence, AI misjudgements, might **cascade** and lead to orders of magnitude more business value lost. It might even lead to AI destroying more value than it was anticipated to bring. On top, making sure business KPIs and technical metrics stay aligned is a challenge. As they can start **diverging** over time.

Monitoring Performance

Track general model performance metrics

- Make sure a ground truth is established, then track model validation metrics

Use granular behavioural metrics

- Go beyond typical performance metrics and also track the behaviour of your model

Track feature behaviour

- Changes in the input data will affect performance of the model

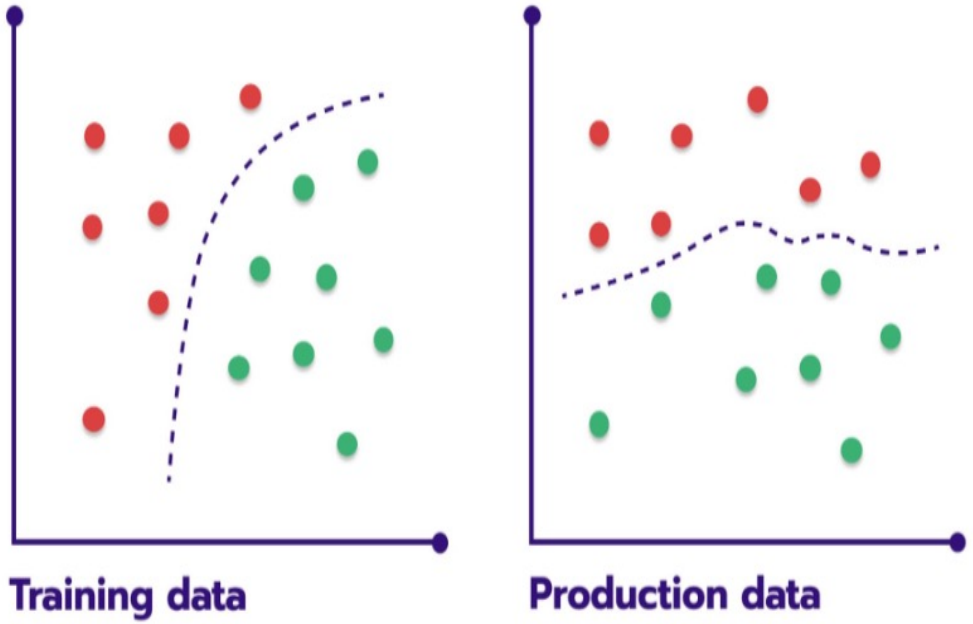
Collect metadata

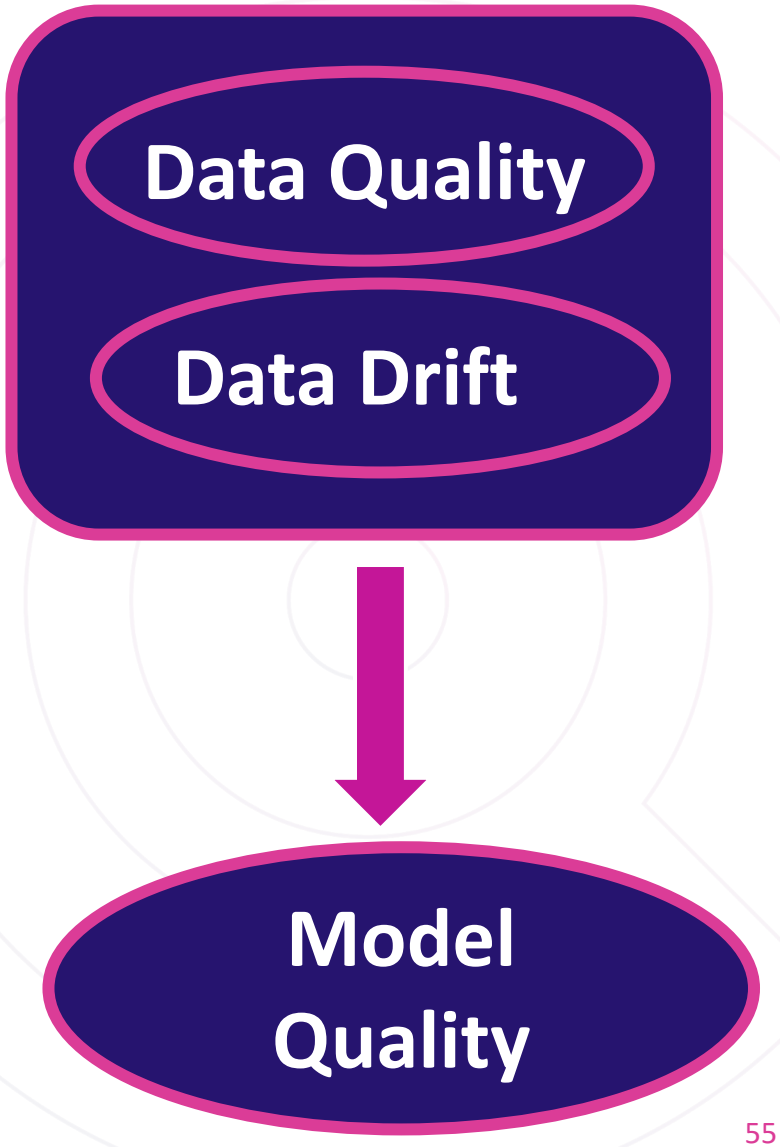
- Use metadata for segmentation of metric's behaviour

Track data every step of the way

- Understand your data at train, test and prediction time

Data changes





Monitoring data quality

Data processing issues

Model receives incorrect data

- Models receive wrong data or no data at all in production

What can go wrong?

- Wrong source
- Lost access
- Bad queries
- Infrastructure update
- Bad feature code

Data schema change

Model can't deal with severe schema changes

- Models expect the data in certain format

Data catalogue should be part of the designs

- If data changes often, factor it into your design

Changes might be well intended

- Domain experts might see the changes as positive

There must be clear ownership of data

- Especially in large complex organizations

Data loss at source

Data can be lost

- If not replicated, be lost forever

If not tracked it can be hard to identify early

- If no process uses the data, be lost until too late

Data can be corrupted

- Worst case: data can be damaged and still provided

Effects can be local

- Data can partially damaged and harder to identify

Broken upstream models

Model data dependency

- One model's output can be another one's input

Cascade of failing

- If one model fails, all dependant models will fail

Special care is needed

- Linked systems bear an obvious risk.
- Already difficult to monitor one model.

Ensuring data integrity

Errors happen, but what to do?

Data schema

- Perform a feature level automated check

Missing data

- Set a combination of monitoring policies to detect and correct

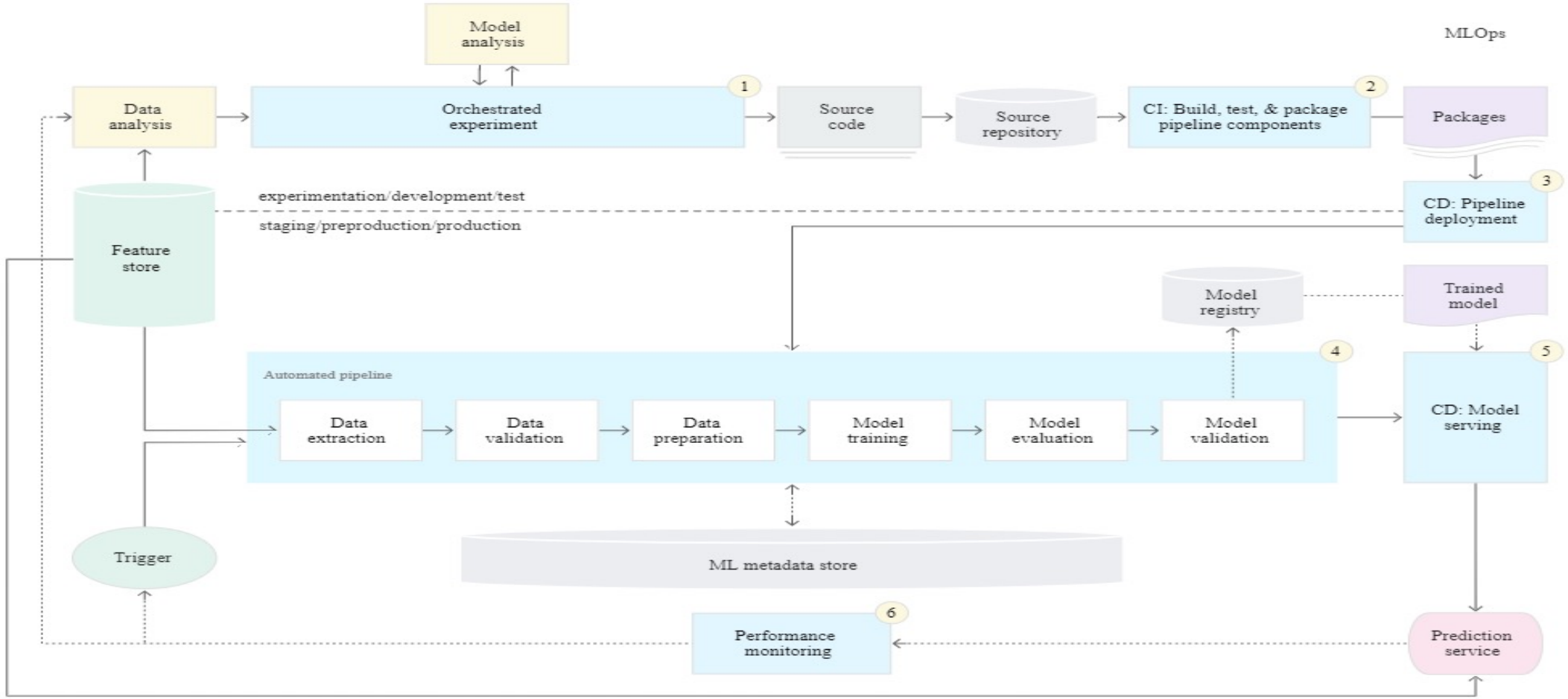
Feature values

- Check if values are abnormal or data shift

Feature processing

- Validate each step of your data's prepossessing journey

MLOps and Monitoring

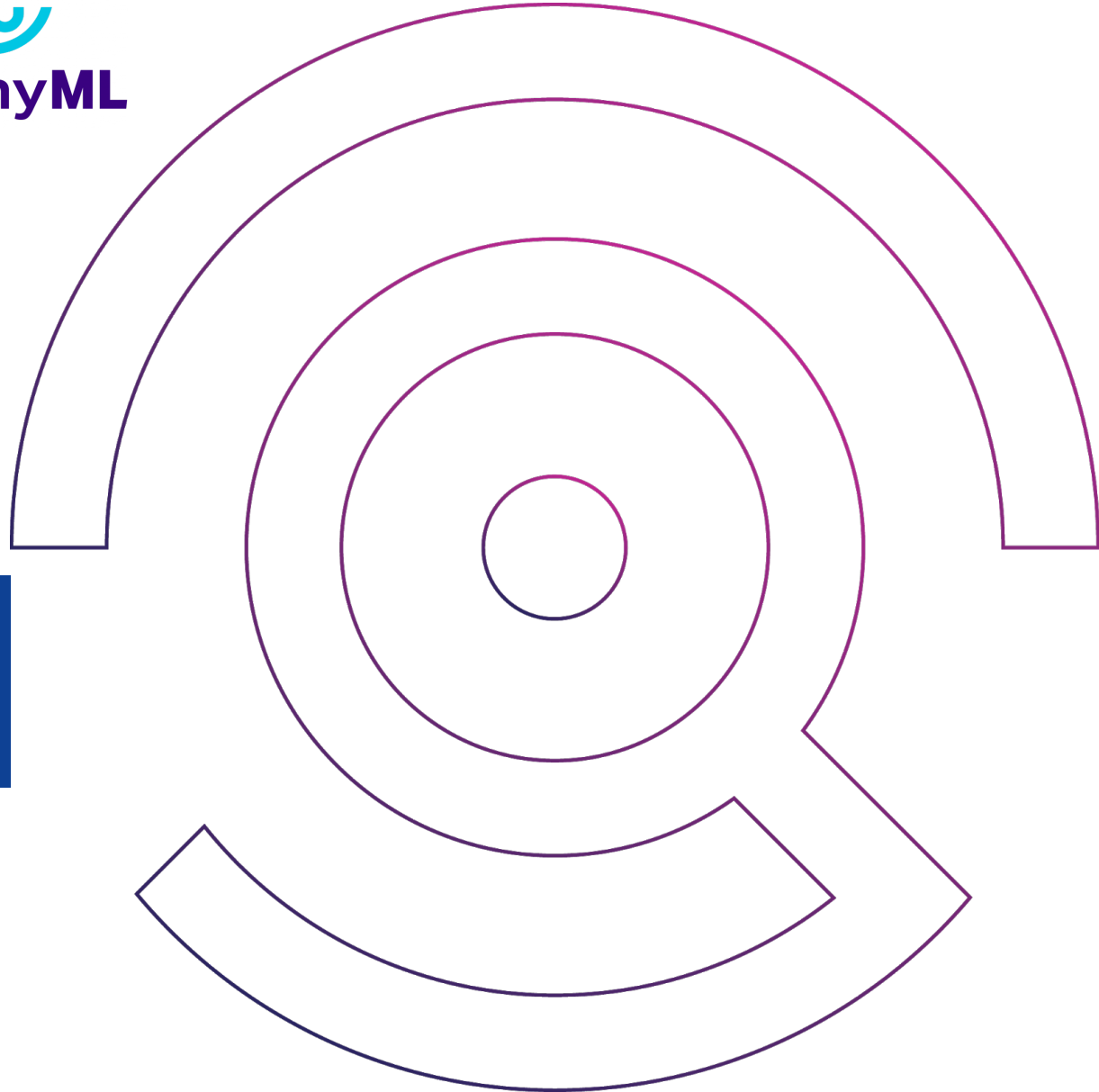


PROPHECY LABS

ctg

nannyML

AI4Business



ESF
INVESTEERT IN JOUW TOEKOMST

Europese Unie

Vlaanderen
is werk