

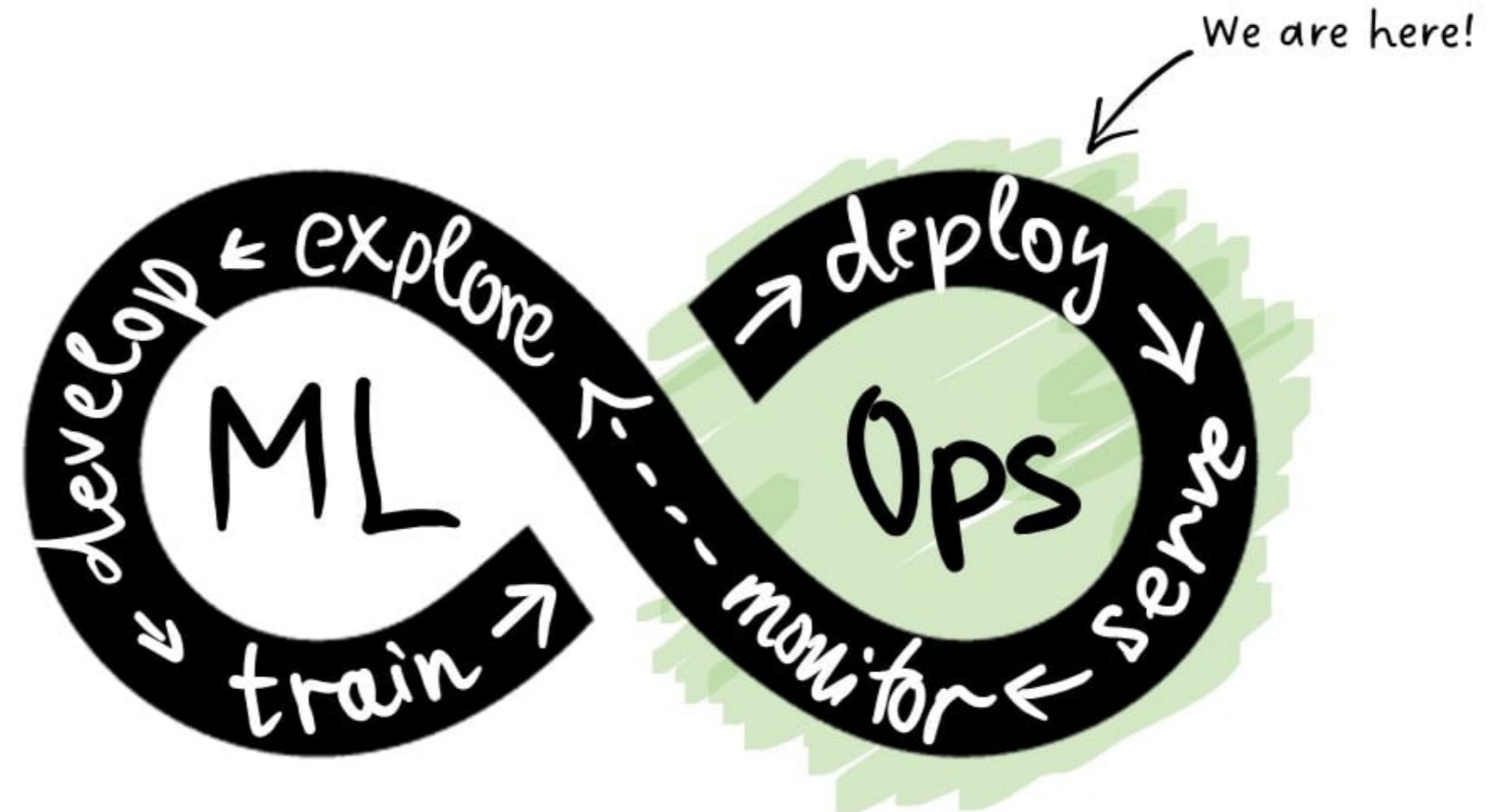
# Deployment-driven development

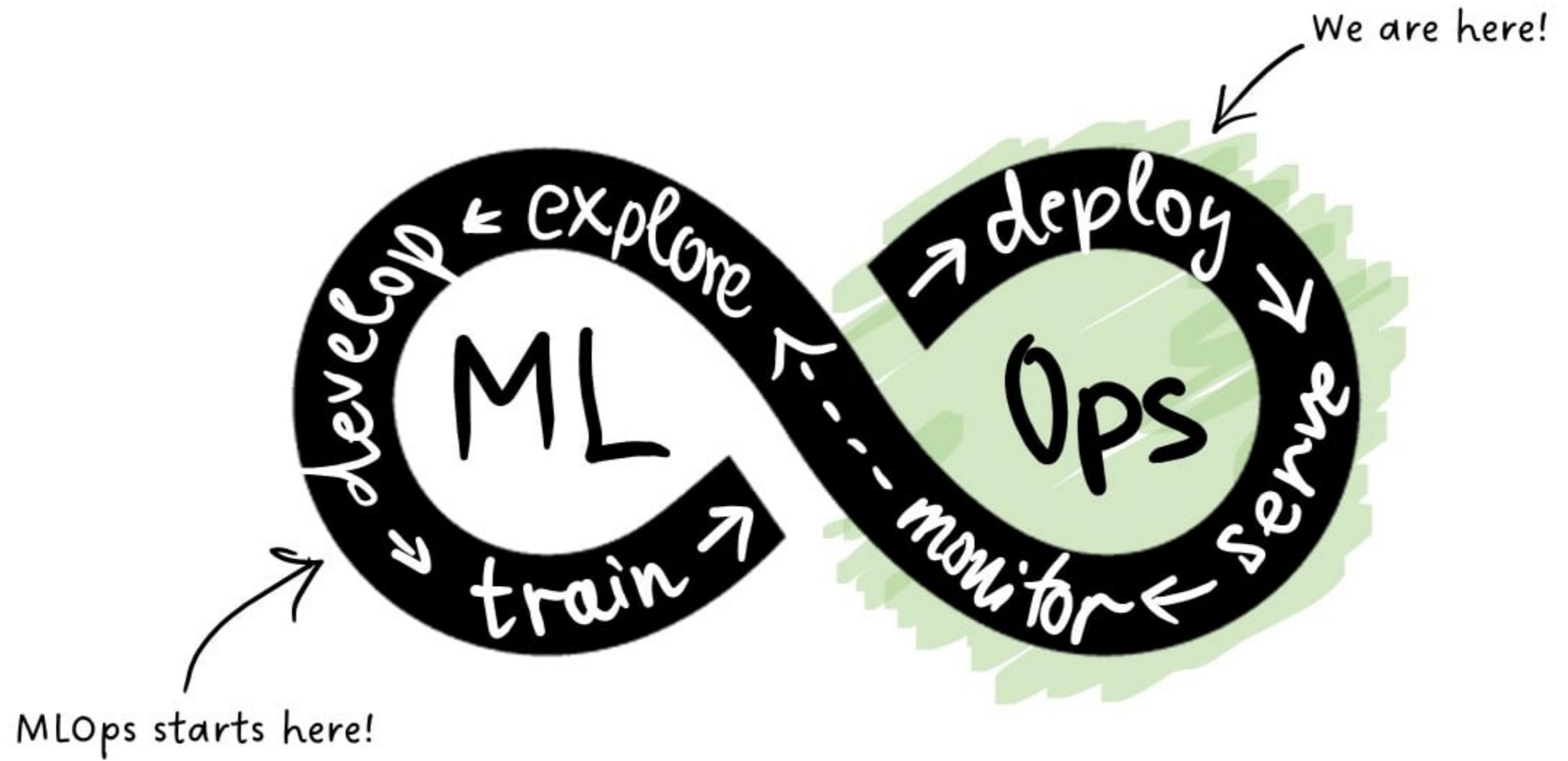
MLOPS DEPLOYMENT AND LIFE CYCLING

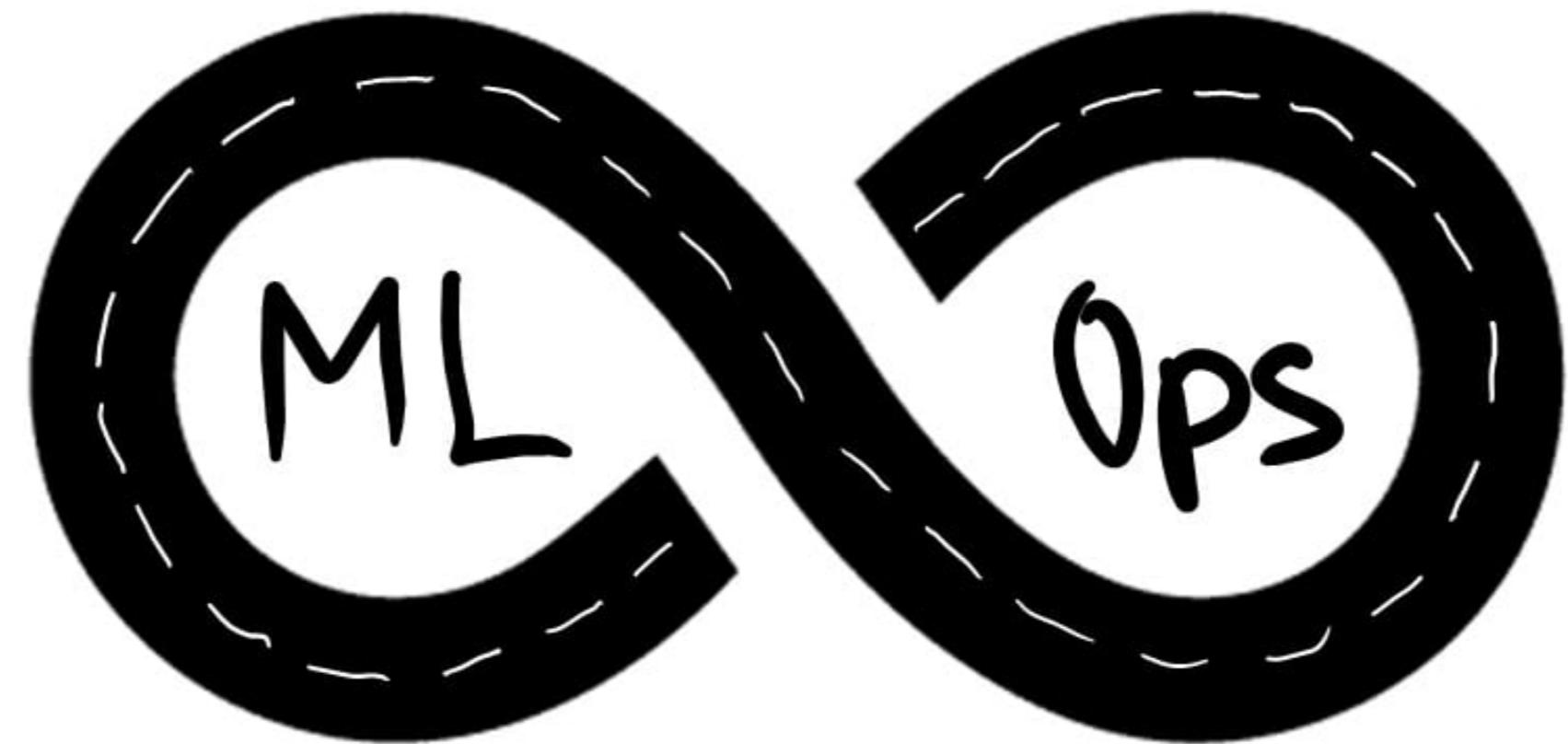


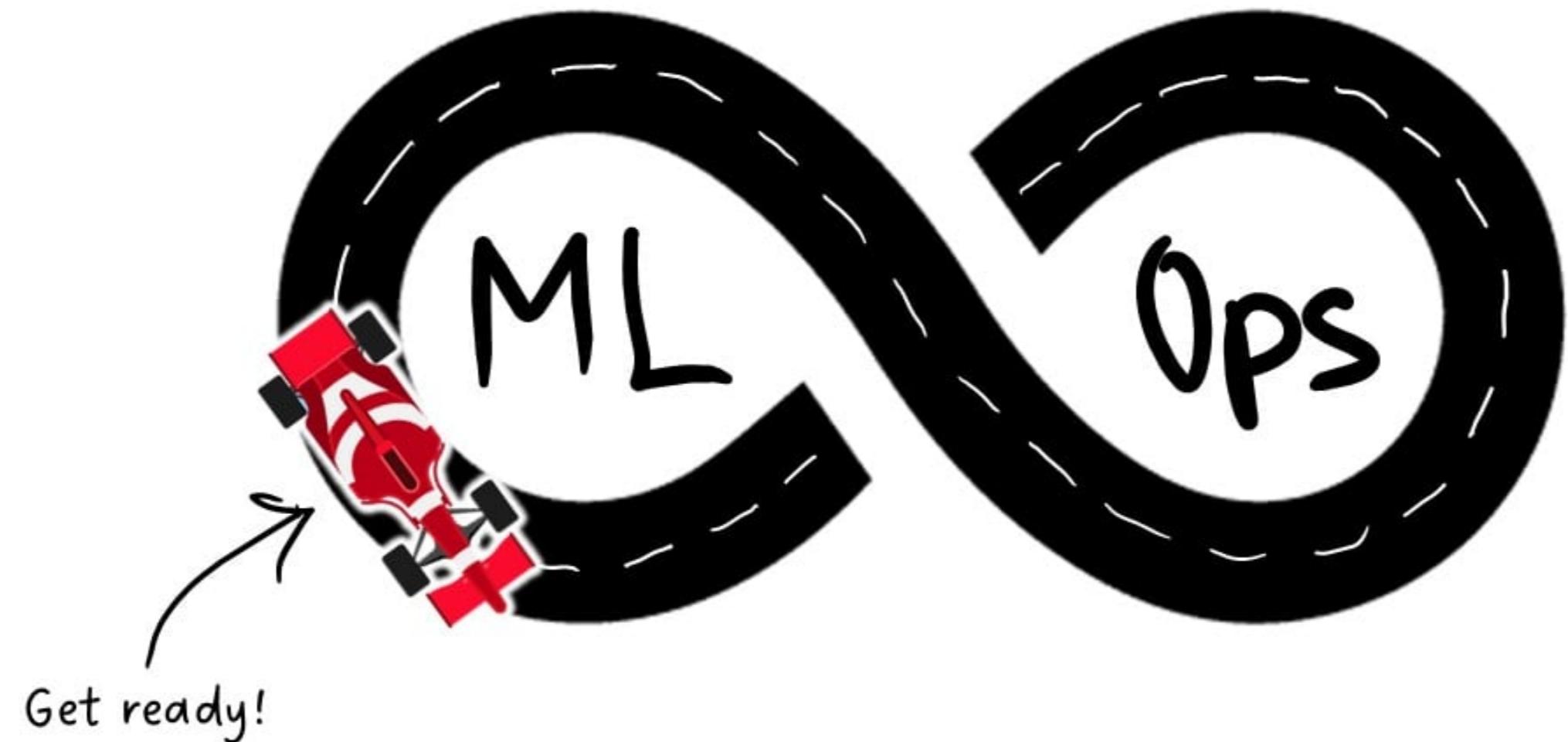
Nemanja Radojkovic

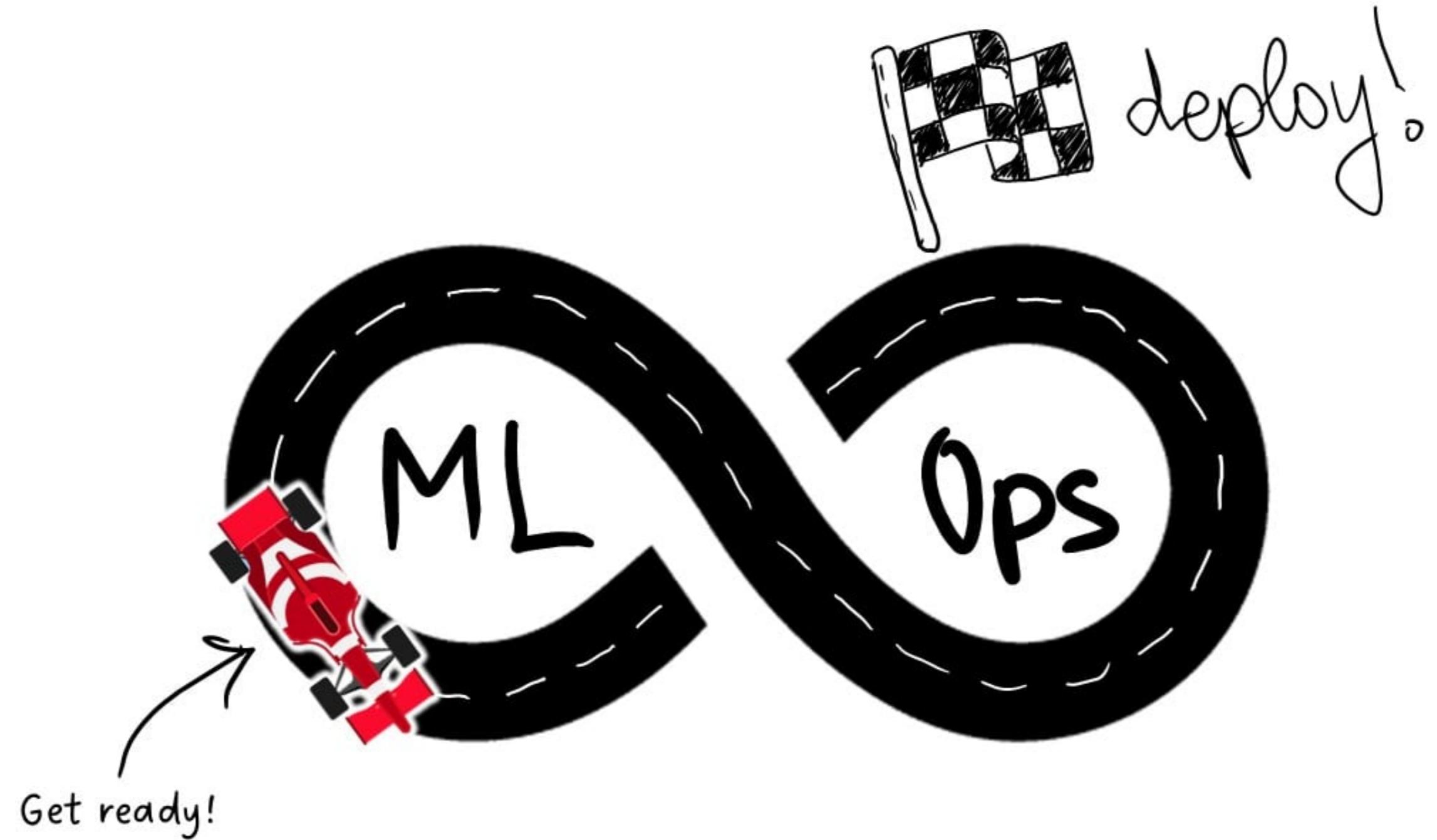
Senior Machine Learning Engineer

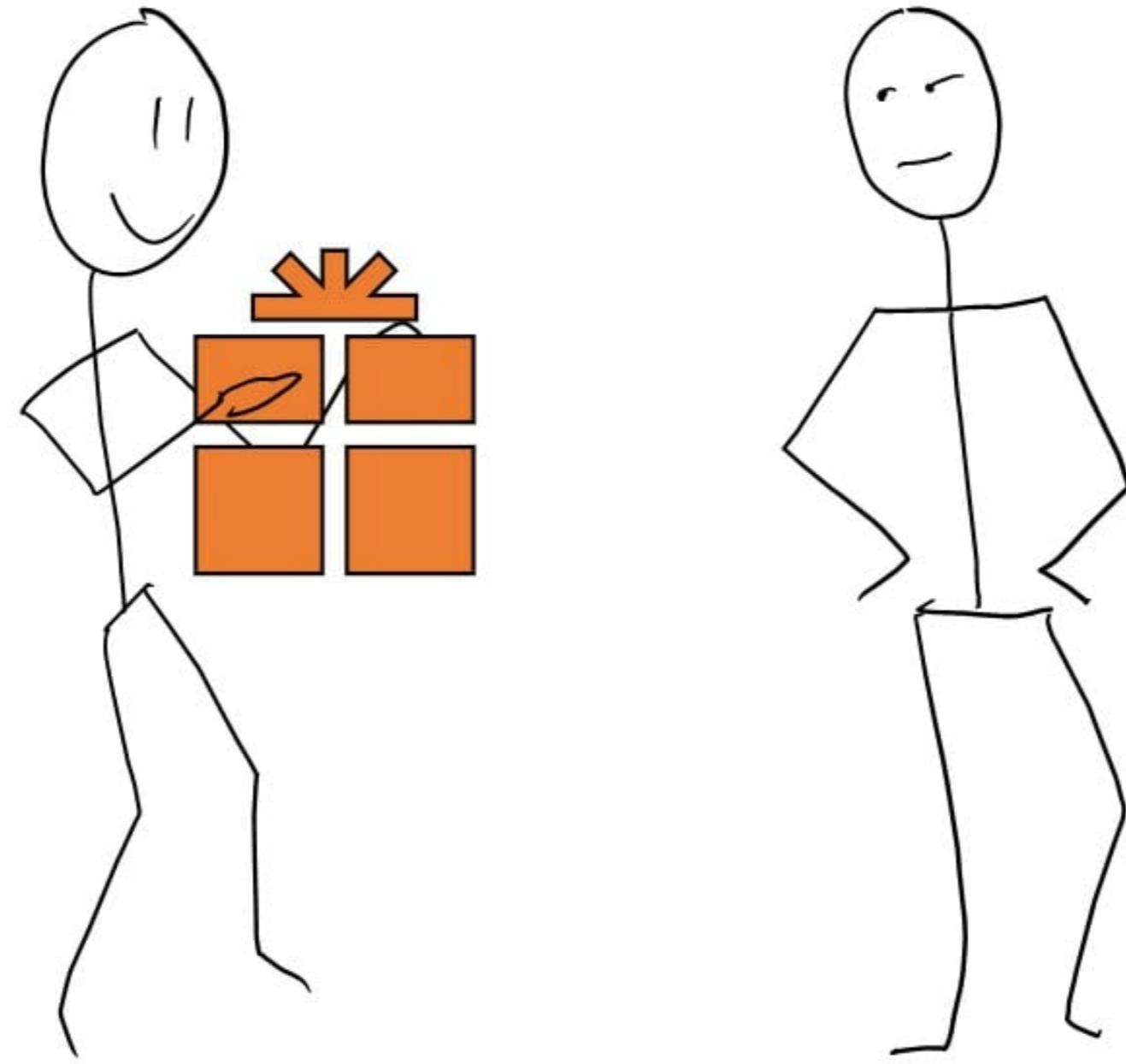






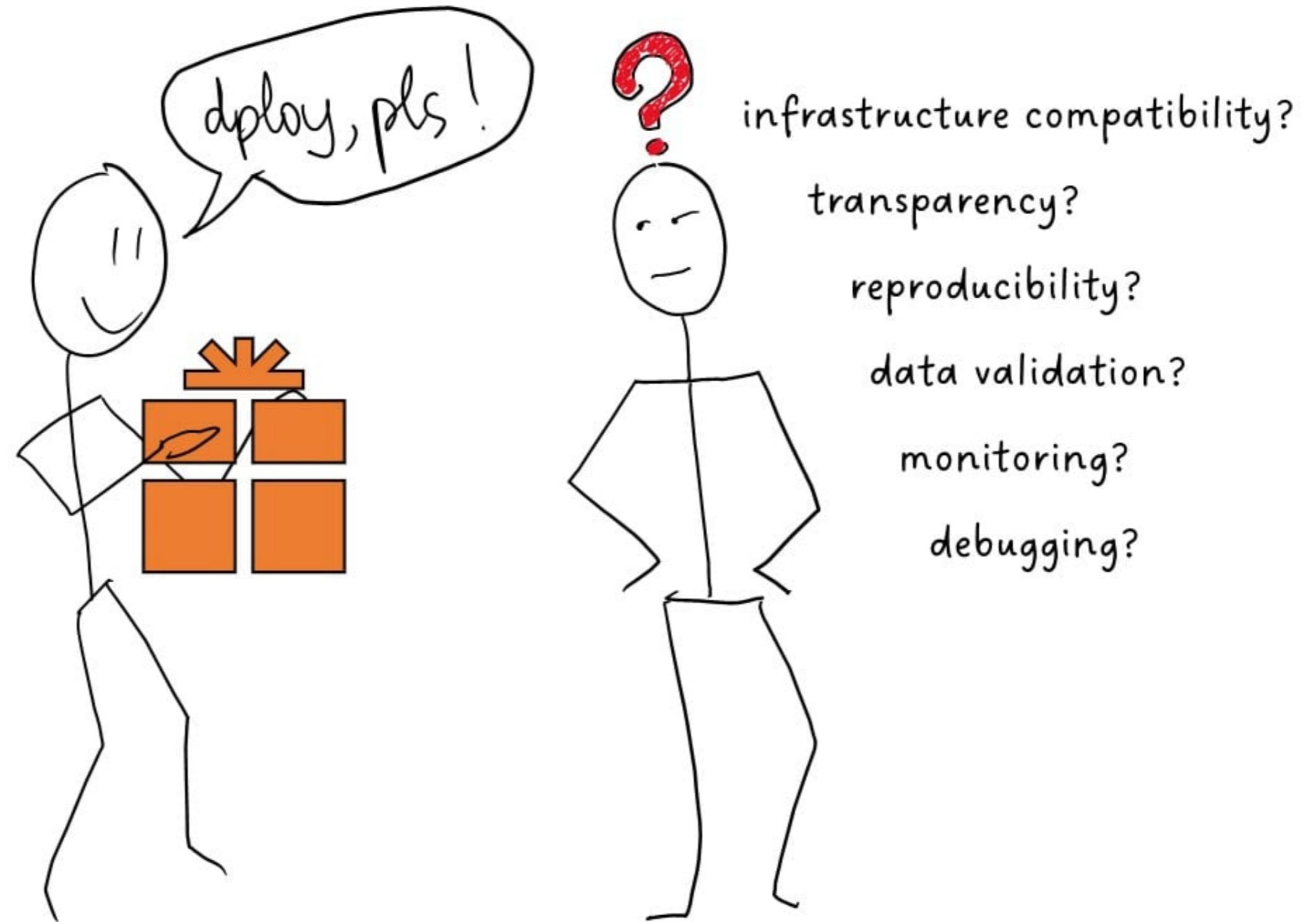






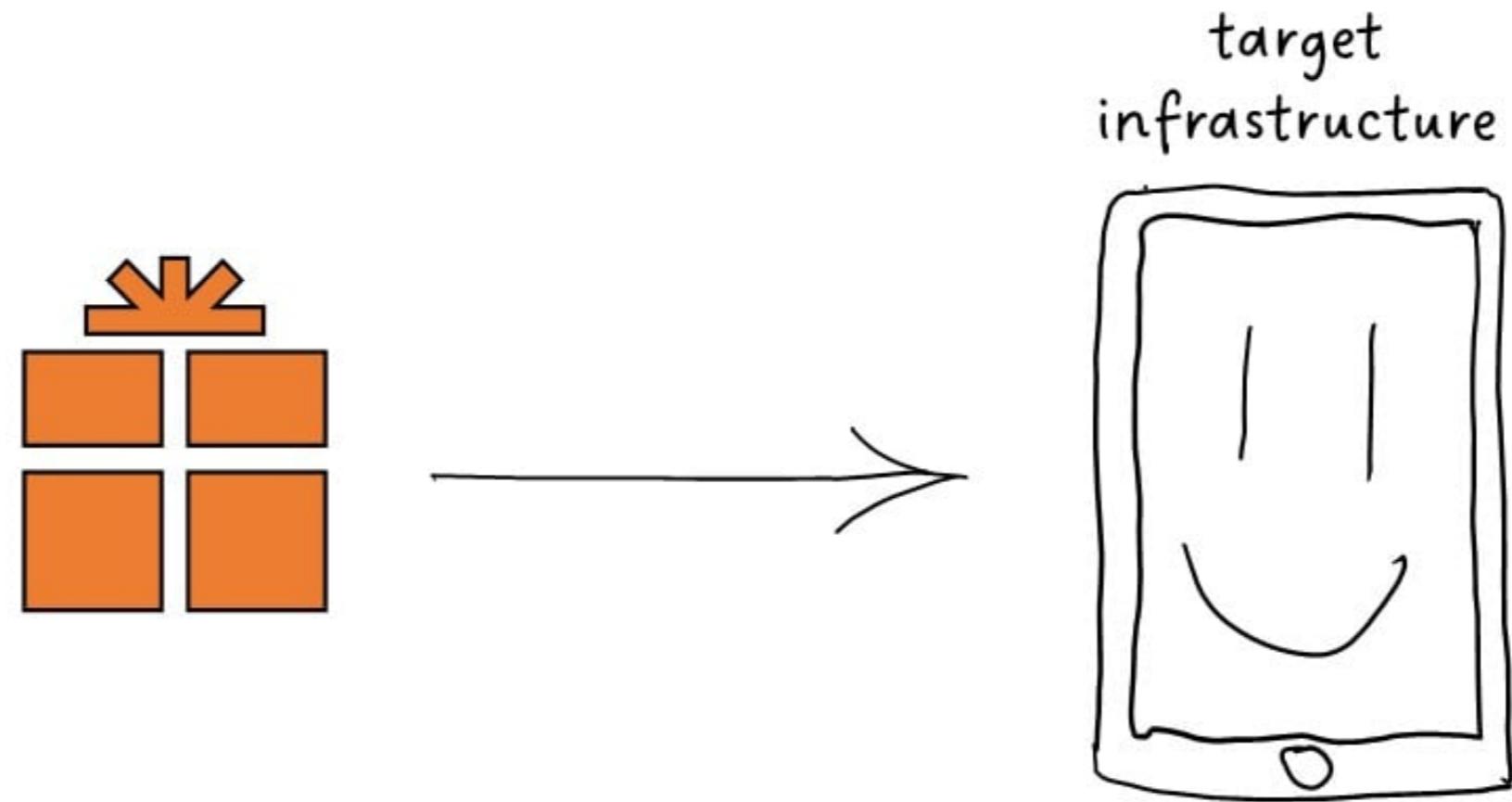


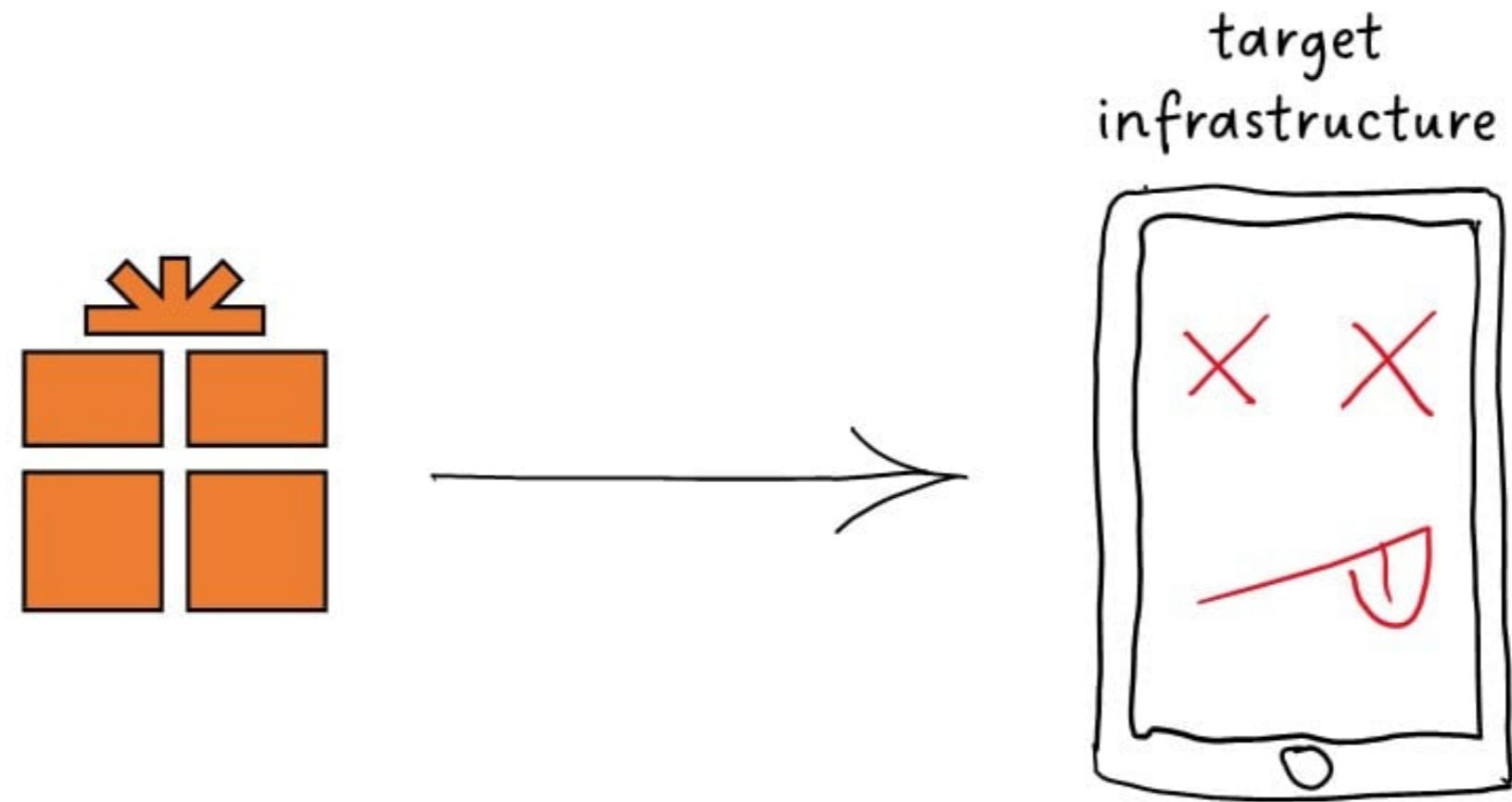




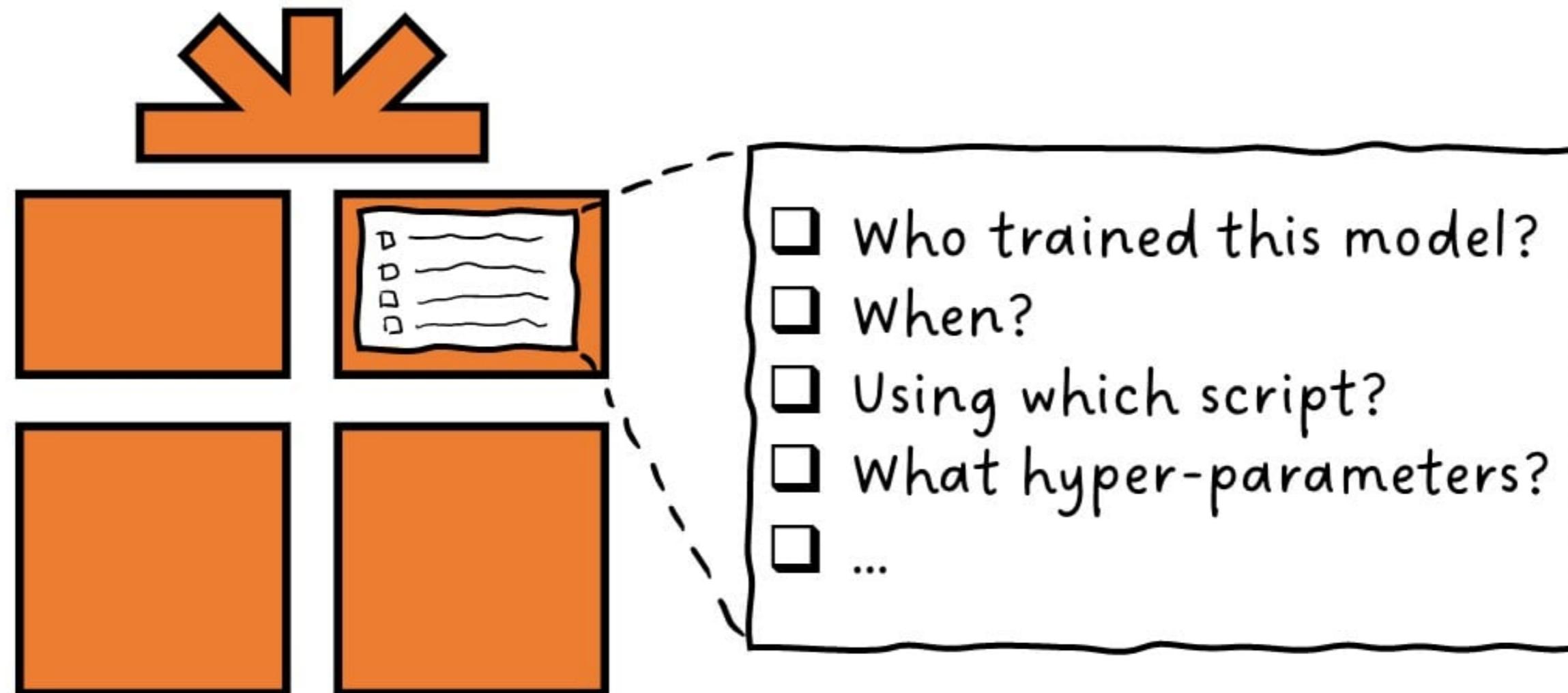
target  
infrastructure



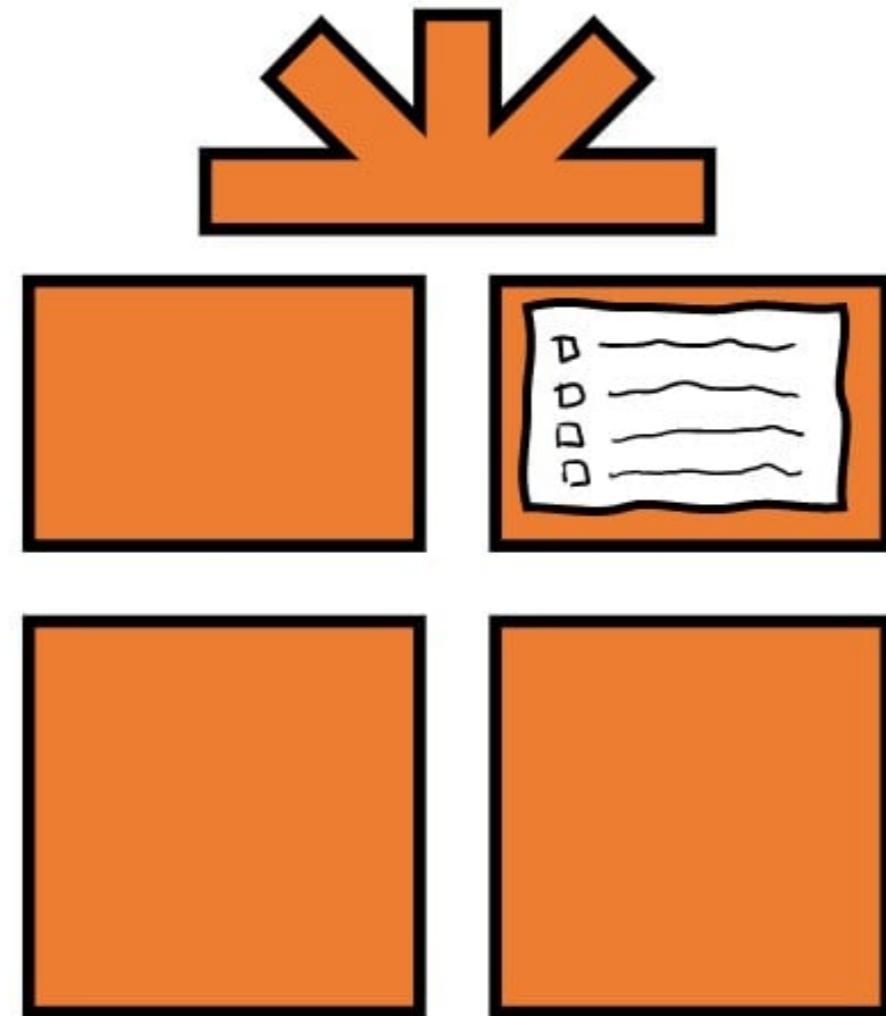




# Transparency & Reproducibility

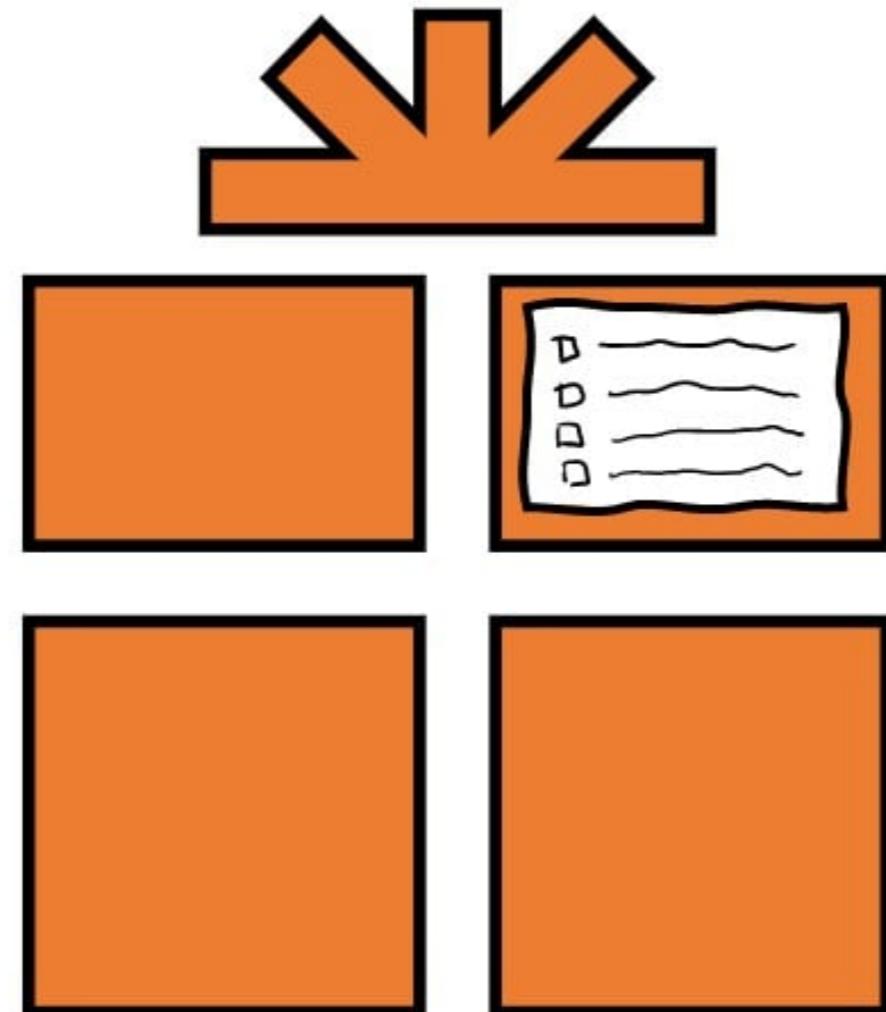


# Transparency & Reproducibility



No transparency &  
reproducibility?  
⇒ Return to sender!

# Transparency & Reproducibility



No transparency &  
reproducibility?  
⇒ Return to sender!

Versioned datasets + fully  
transparent pipelines?  
⇒ Good to go!

# Transparency & Reproducibility



No dilemma  
how we go from  
code + raw  
data to the  
final model

# Transparency & Reproducibility

No dilemma  
how we go from  
code + raw  
data to the  
final model

Recreating the  
exact model  
simple and  
straightforward

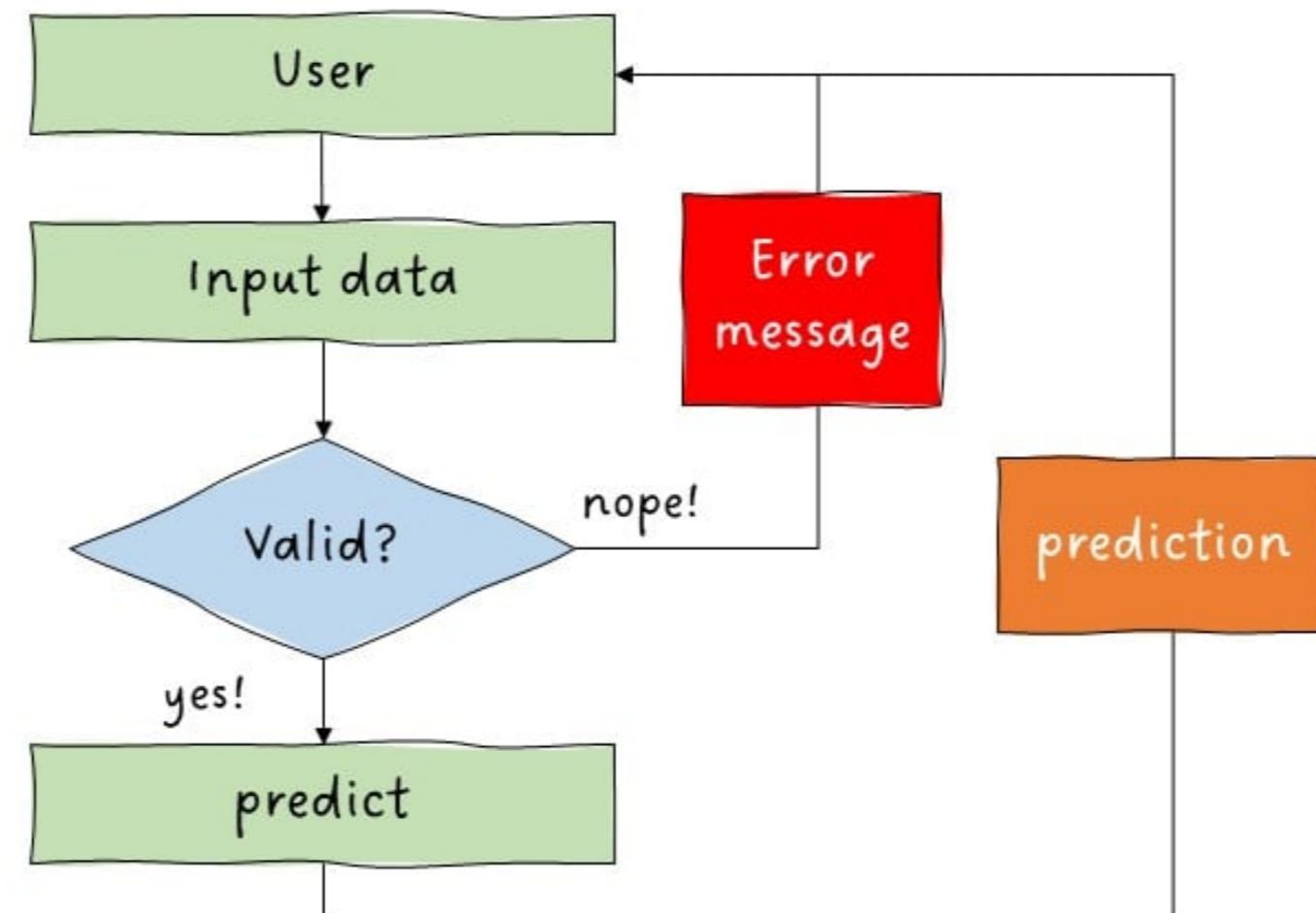
# Transparency & Reproducibility

No dilemma  
how we go from  
code + raw  
data to the  
final model

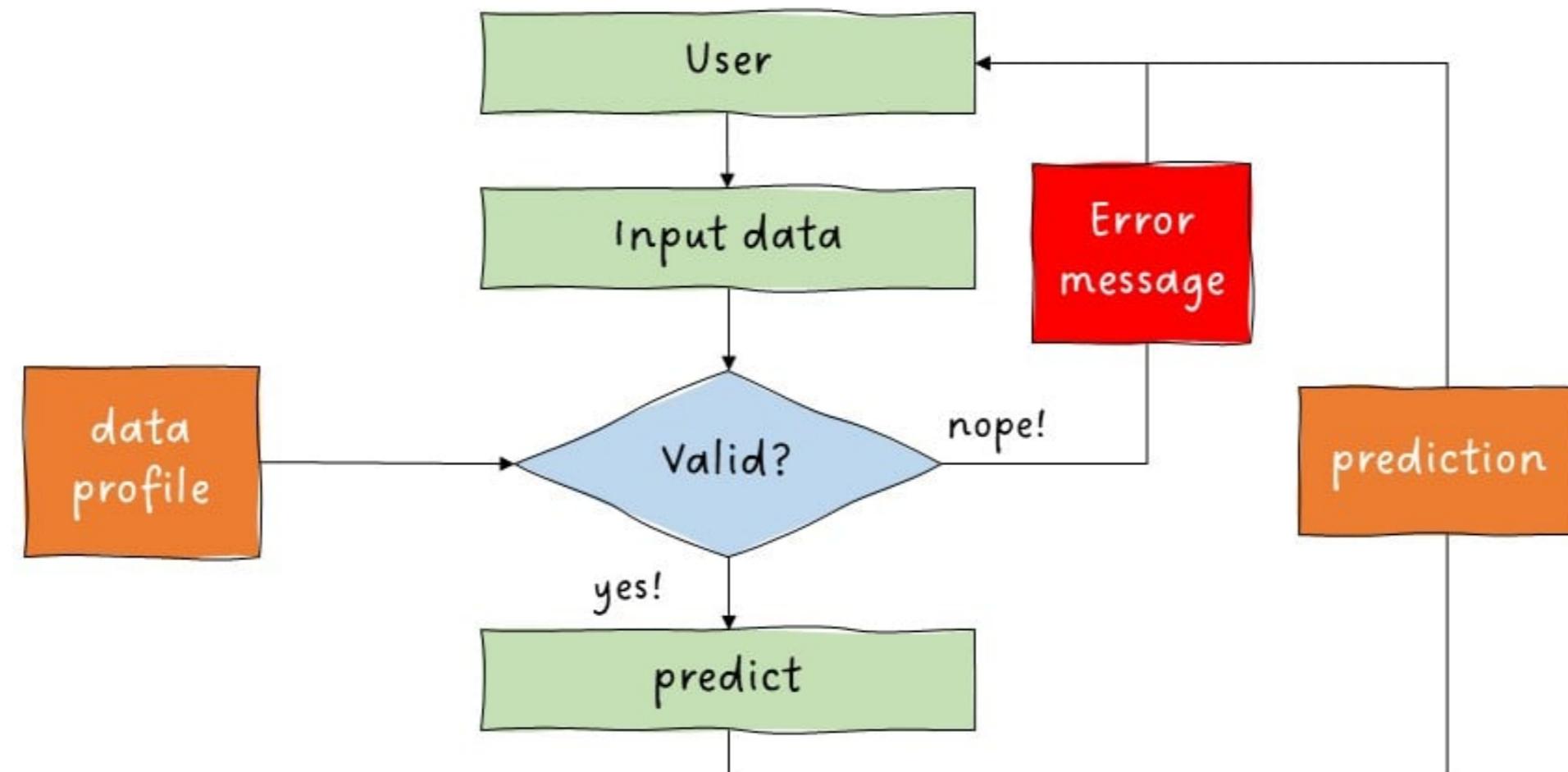
Recreating the  
exact model  
simple and  
straightforward

**bonus points:** log experiments in metadata store

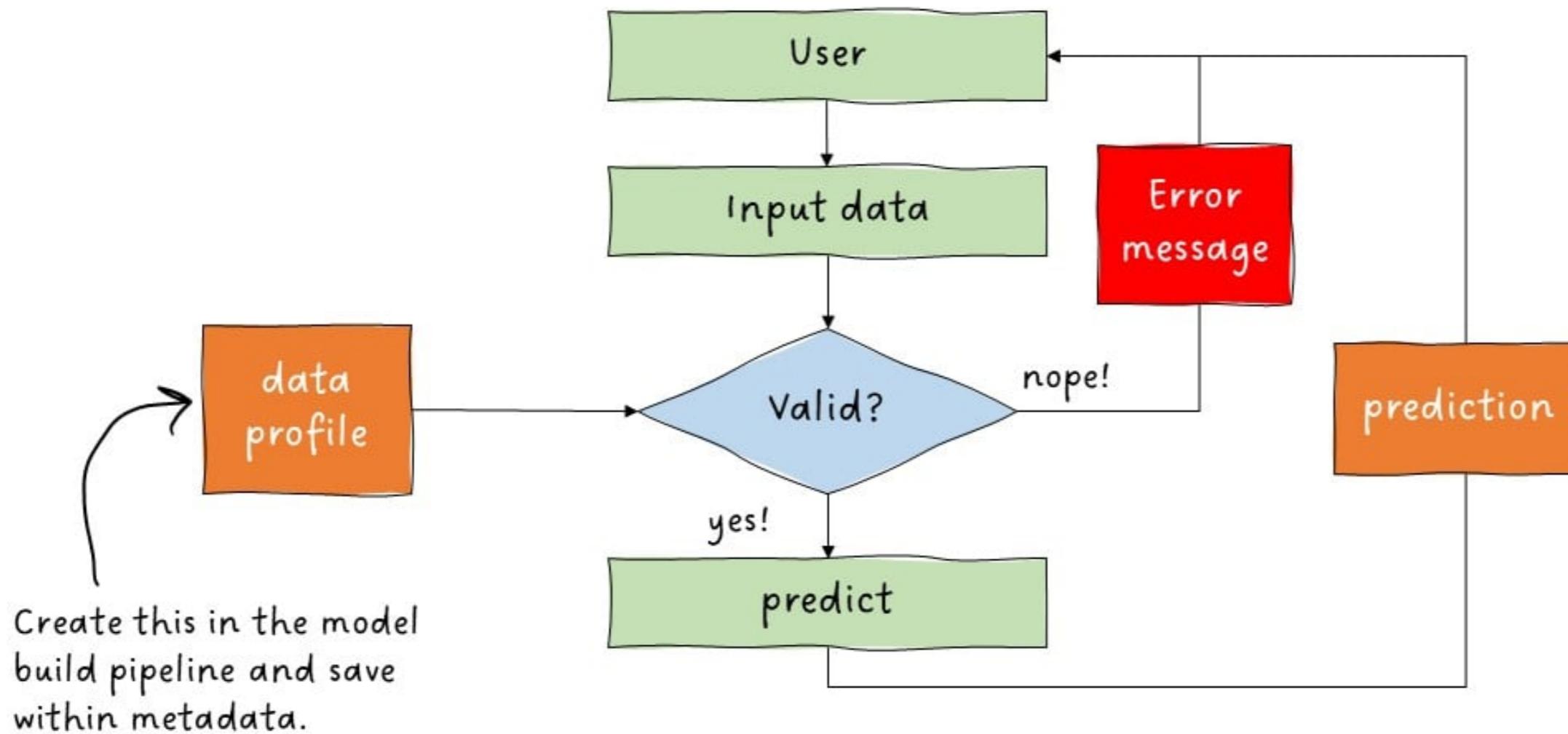
# Concern: Input data validation



# Concern: Input data validation



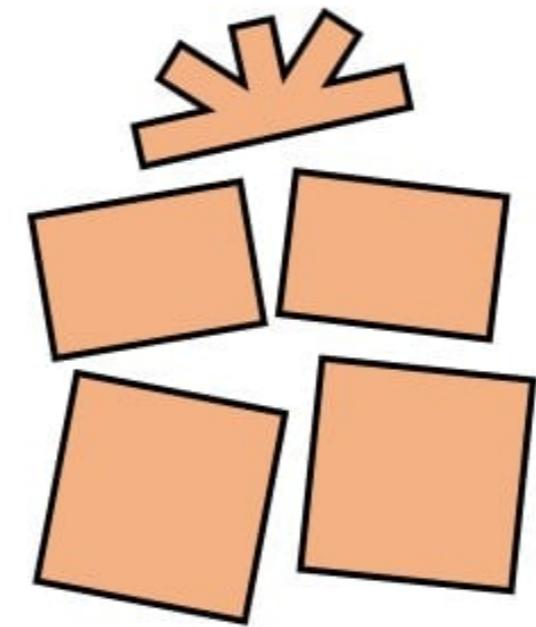
# Concern: Input data validation



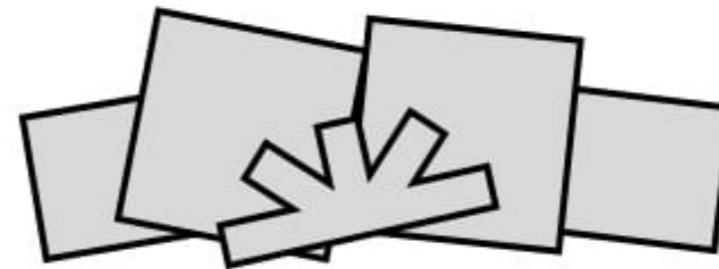
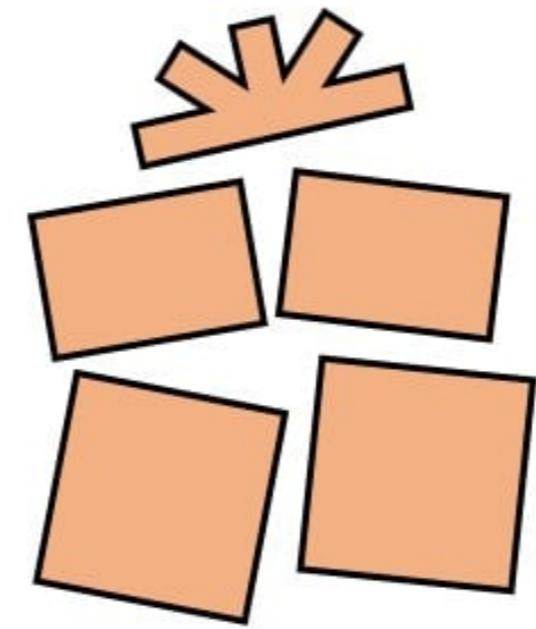
# Concern: Performance deterioration



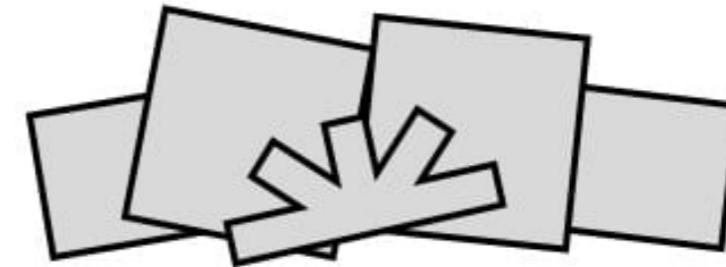
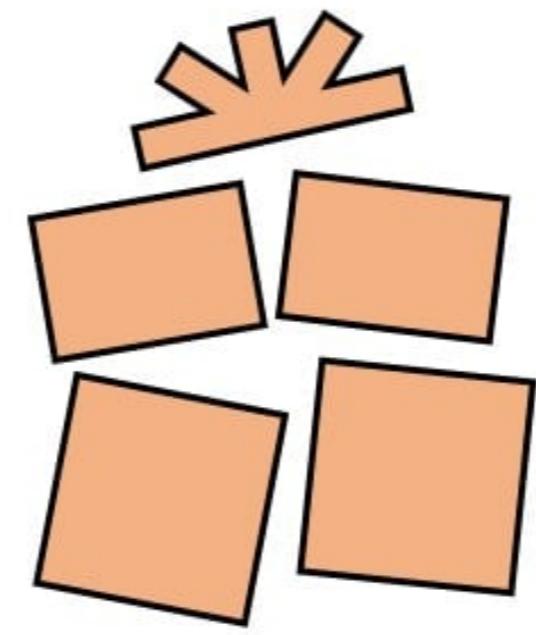
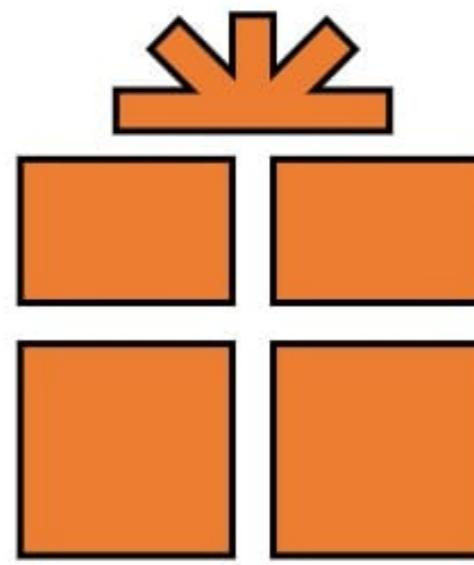
# Concern: Performance deterioration



# Concern: Performance deterioration

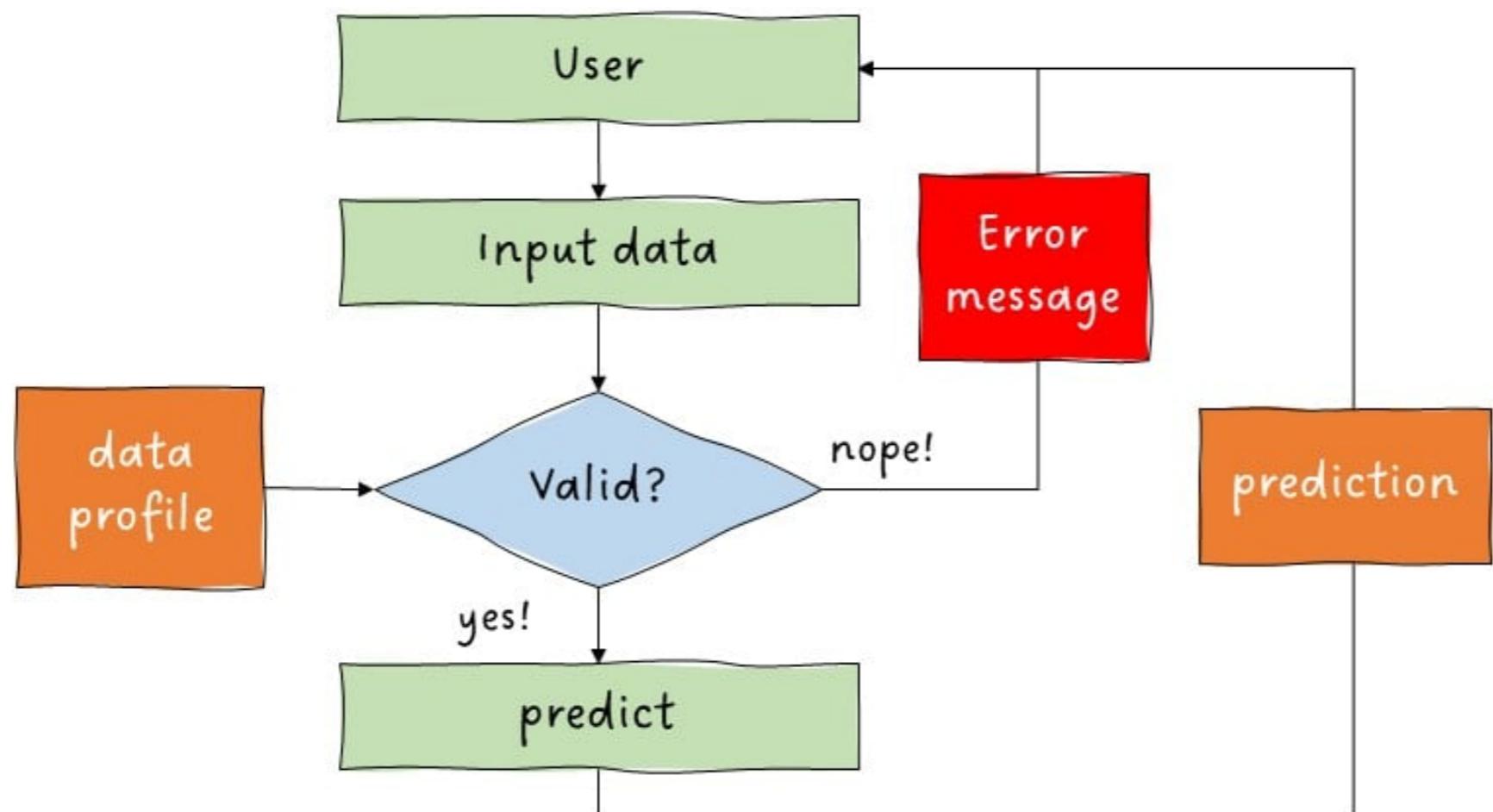


# Concern: Performance deterioration

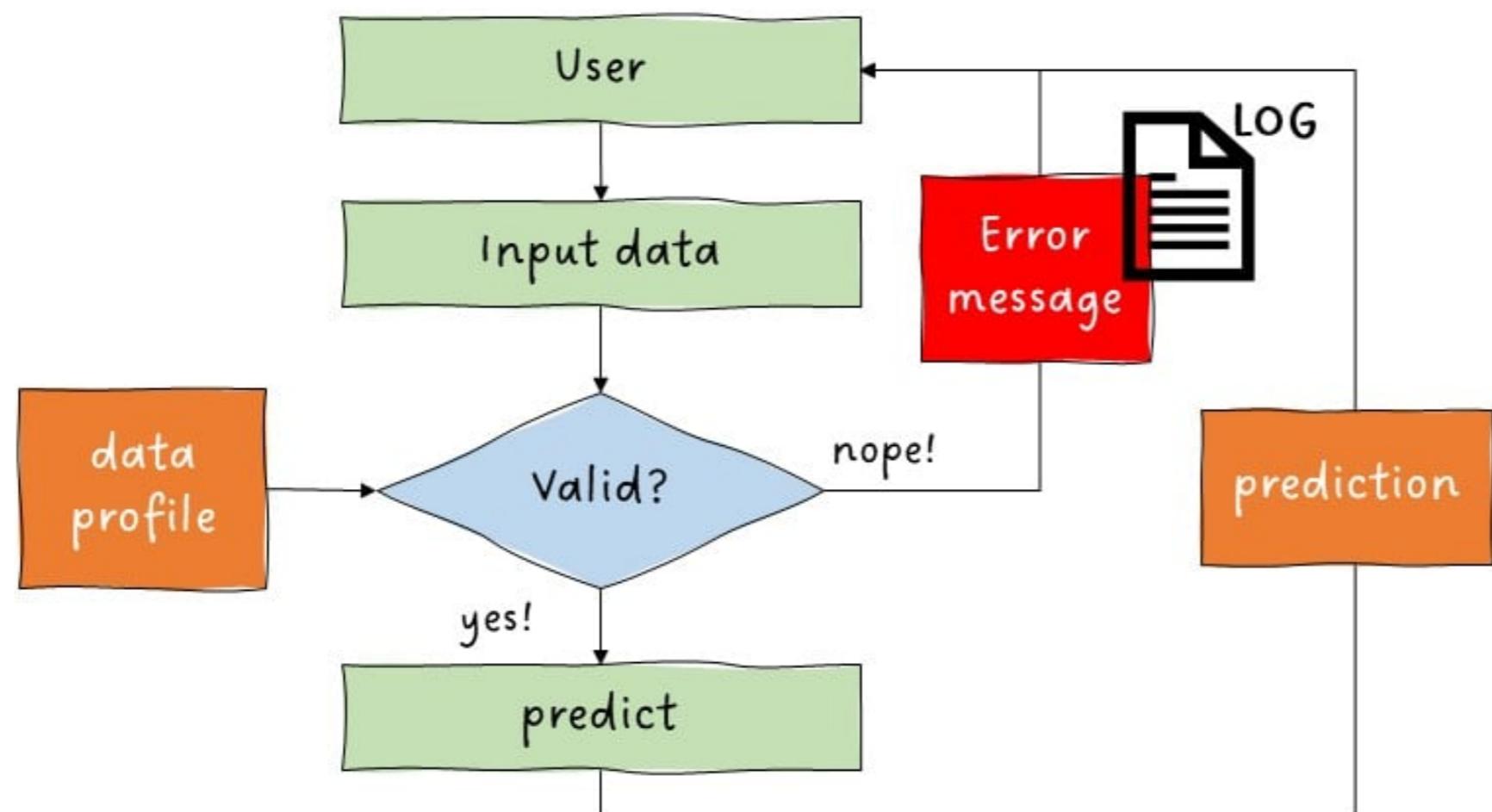


Minimum requirement: log inputs and outputs

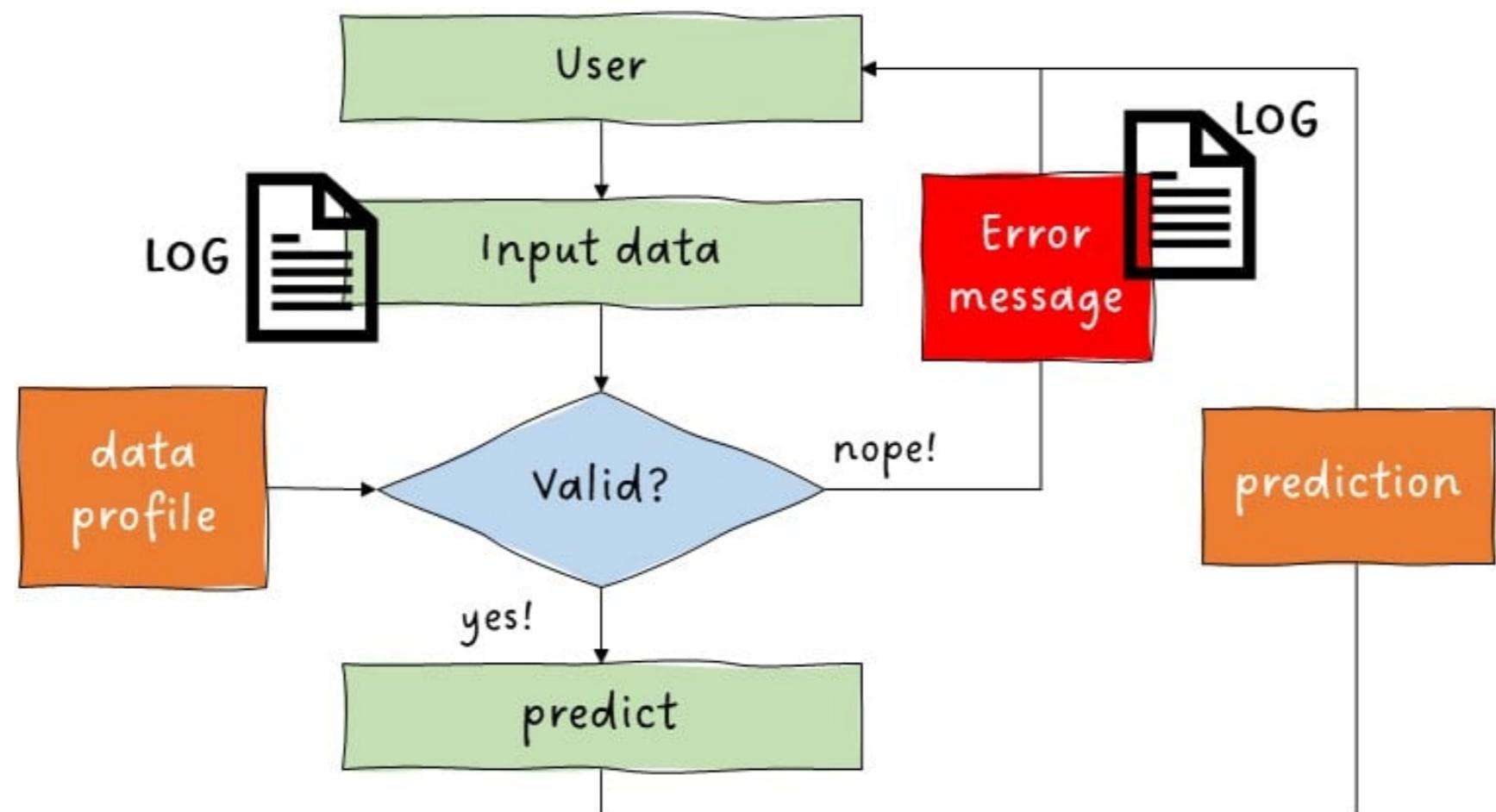
# Concern: Debugging



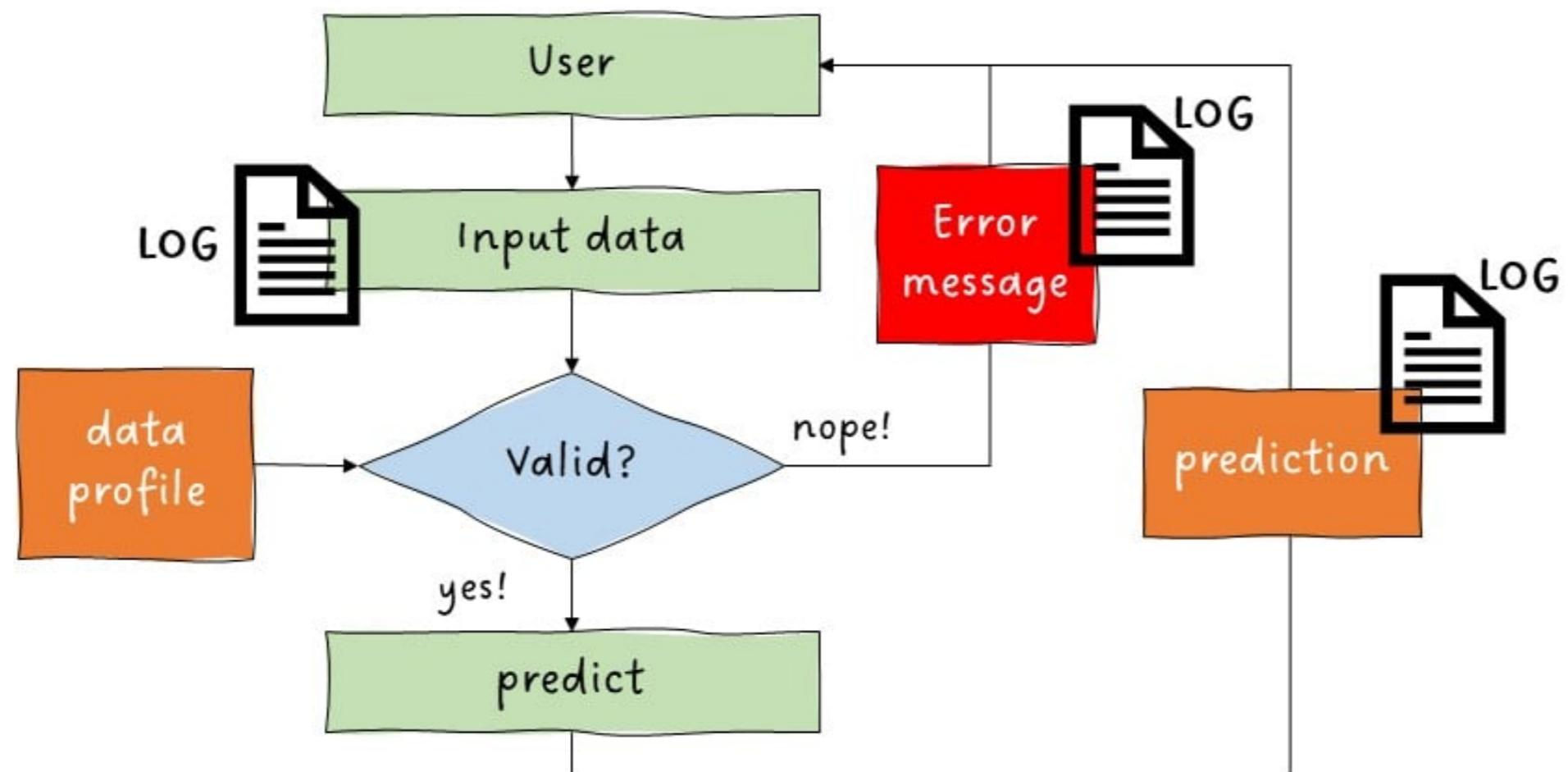
# Concern: Debugging



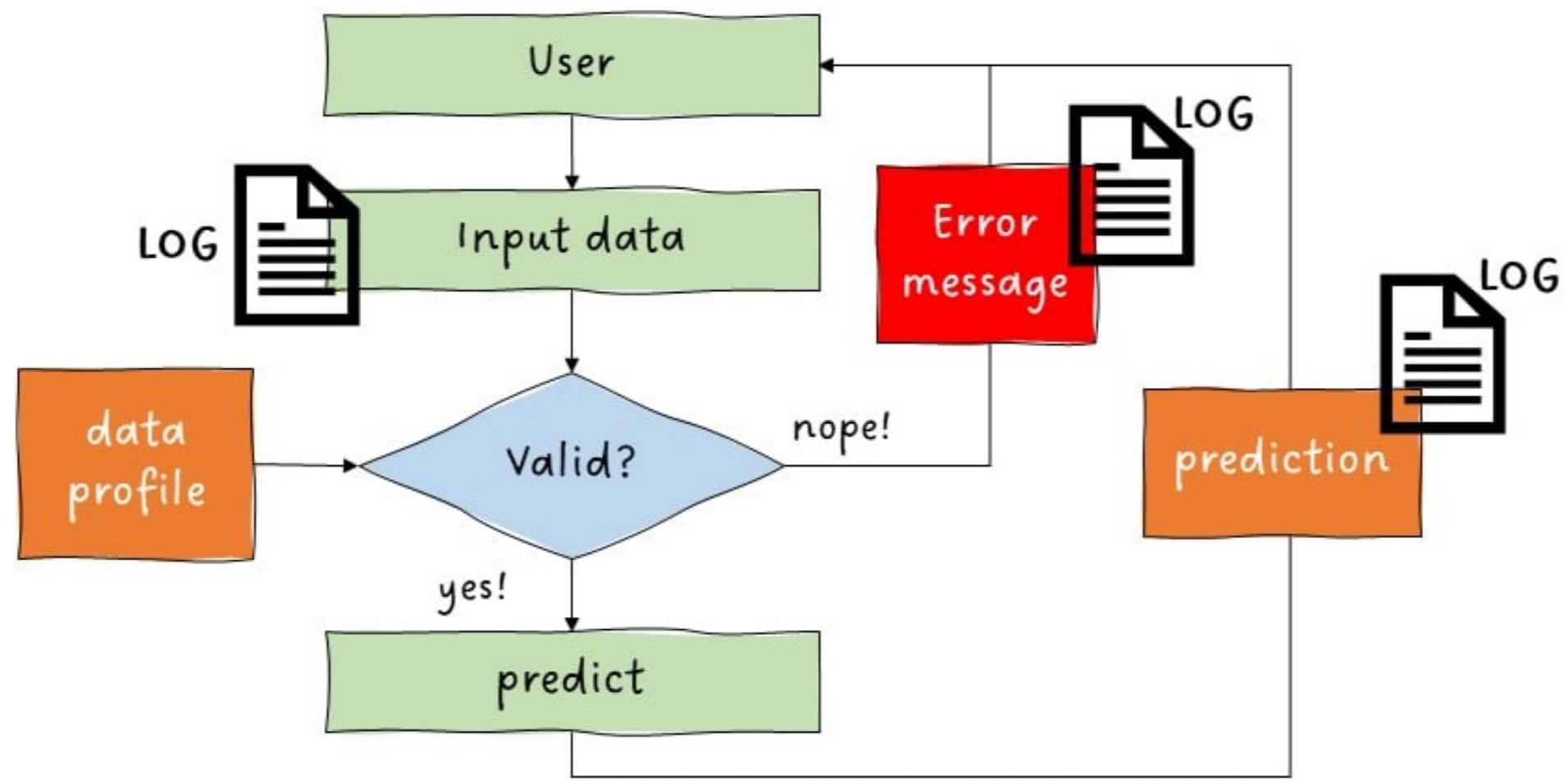
# Concern: Debugging



# Concern: Debugging



# Concern: Debugging



Don't  
improvise, use  
purpose-built  
tools!

Concern: Am I comfortable making changes to this code?



Concern: Am I comfortable making changes to this code?

- unit tests
- integration tests
- load tests
- stress tests
- deployment tests



KEEP  
CALM  
AND  
ML  
OPS

# **Let's practice!**

**MLOPS DEPLOYMENT AND LIFE CYCLING**

# Profiling, versioning, and feature stores

MLOPS DEPLOYMENT AND LIFE CYCLING



Nemanja Radojkovic

Senior Machine Learning Engineer

# Data profiling

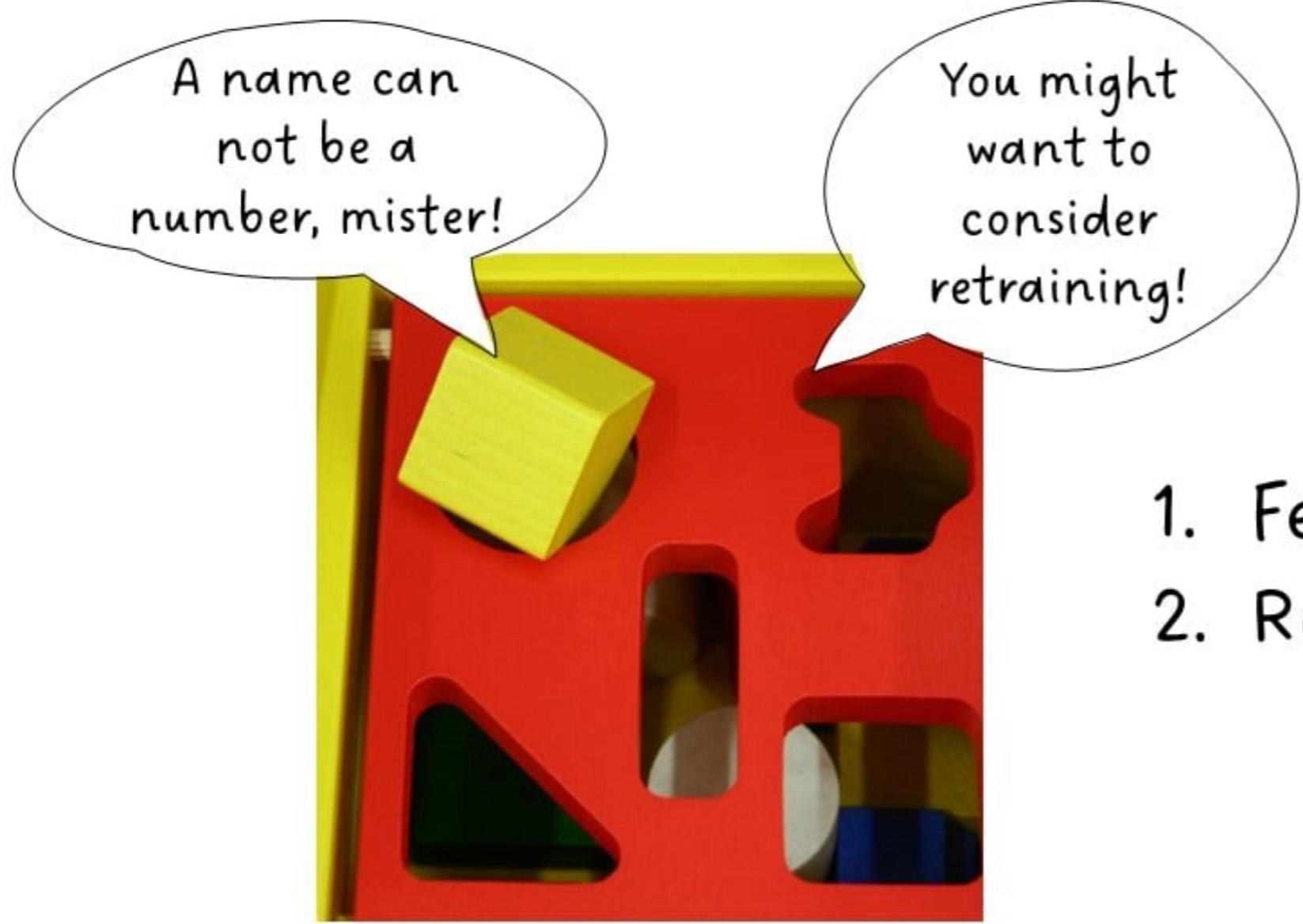
Automated data analysis and creation of high-level summaries (a.k.a. data profiles, expectations), used for validating and monitoring data in production







## 1. Feedback to the user



1. Feedback to the user
2. Retraining decisions

# Risk of NOT using data profiles

- Clients complaining, although *they* submitted erroneous inputs to the model
- No way to identify that data has drifted and our model is no longer valid

# ML pipeline checklist



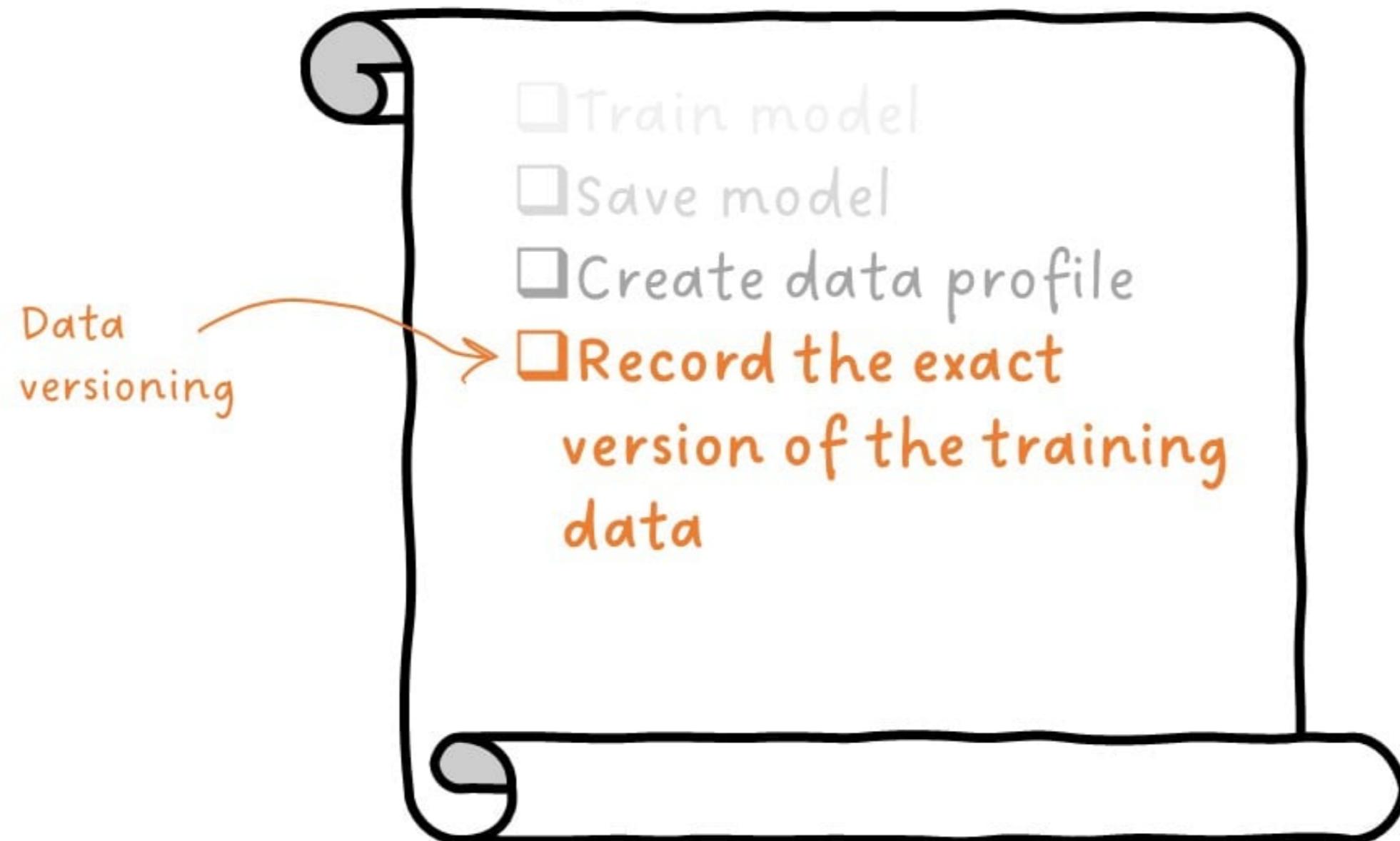


# great\_expectations

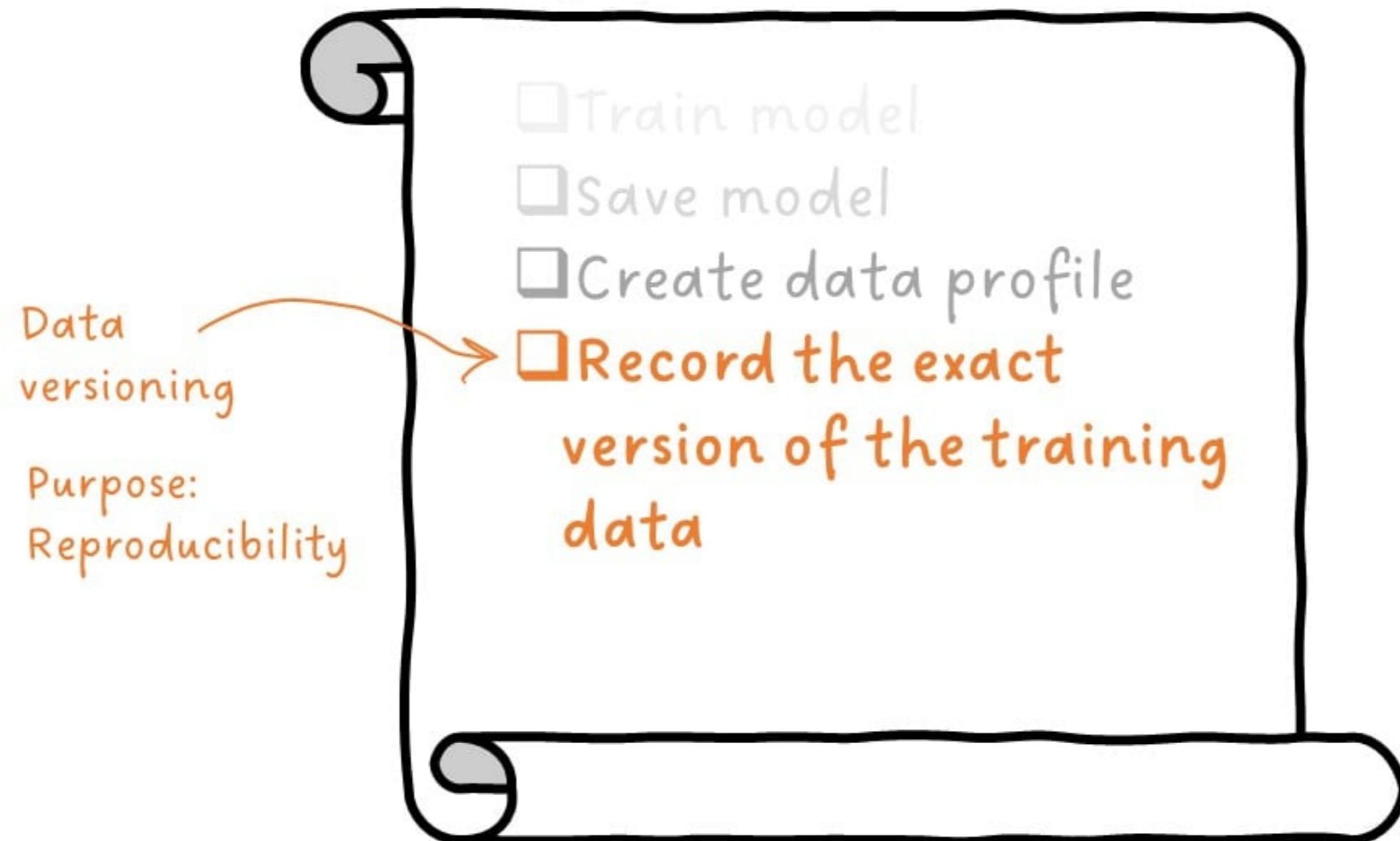
# ML pipeline checklist

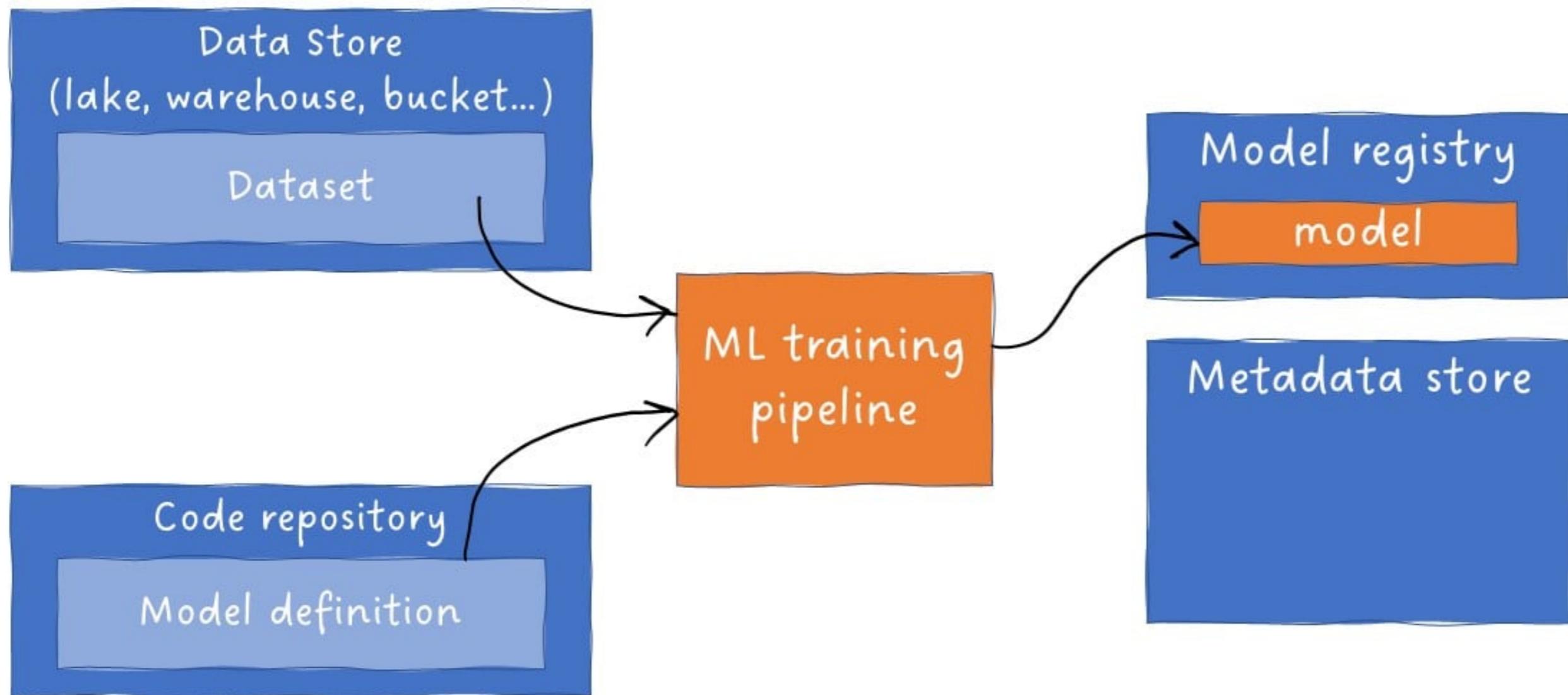
- 
- Train model
  - Save model
  - Create data profile
  - Record the exact version of the training data

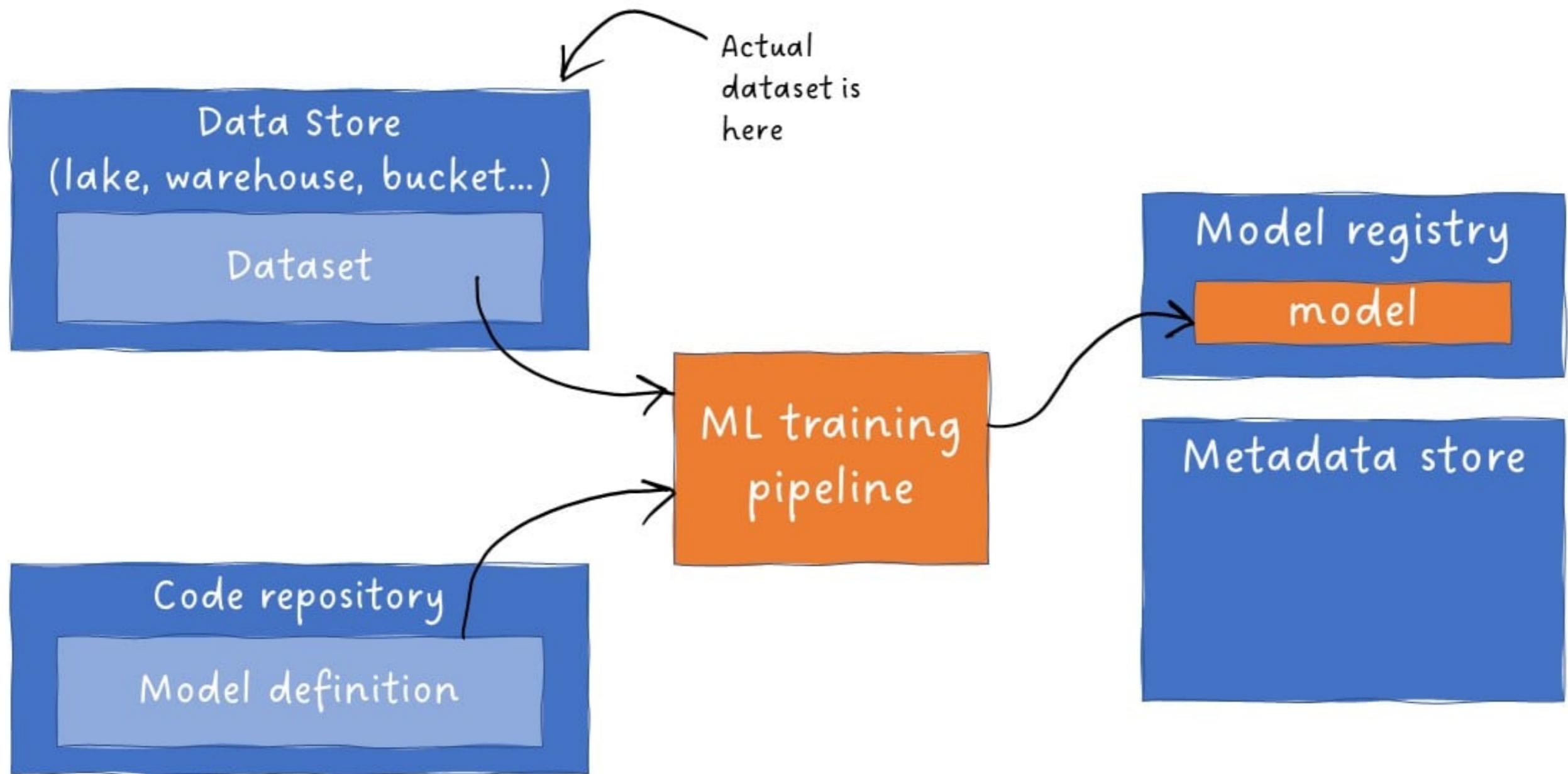
# ML pipeline checklist

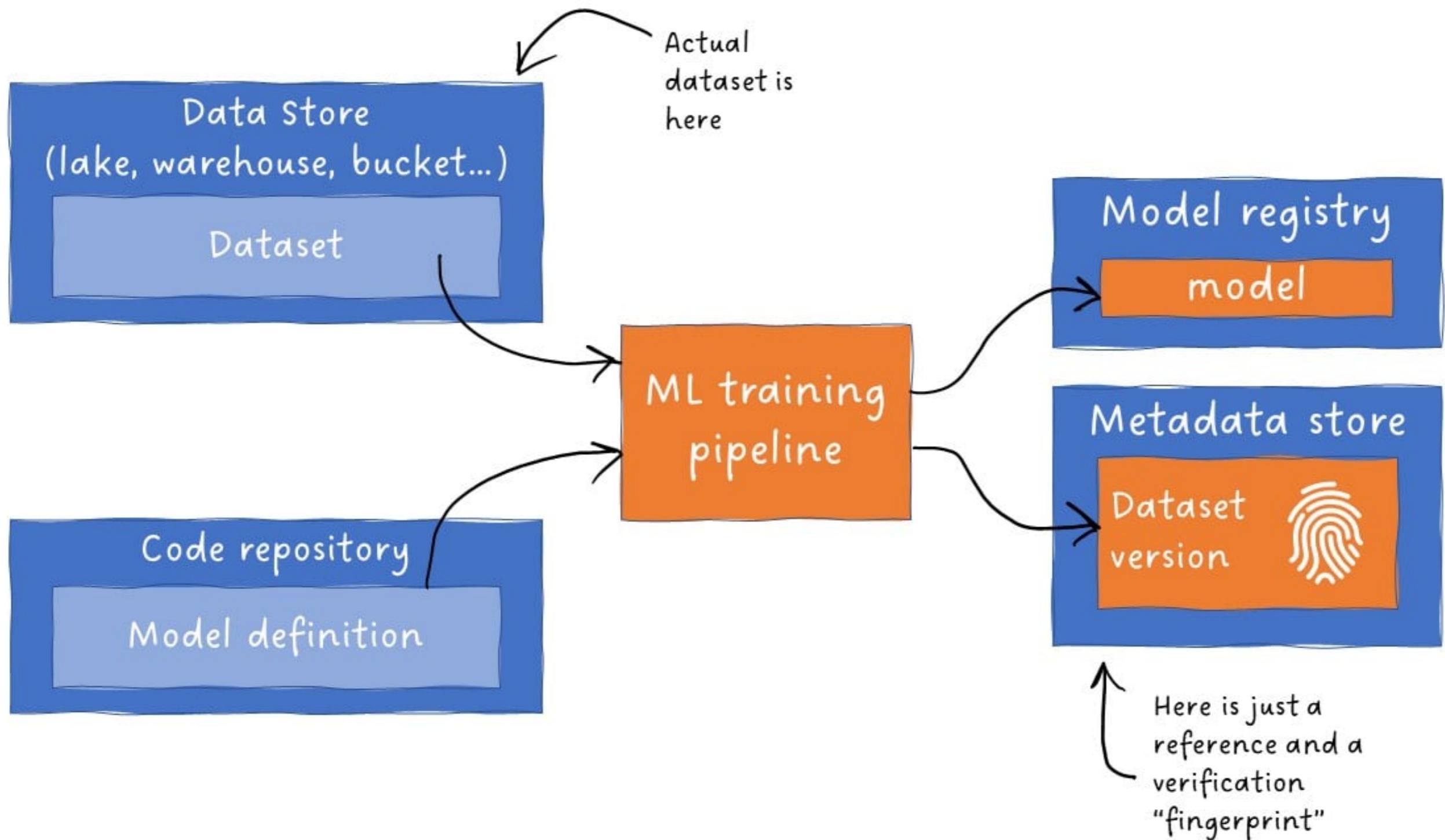


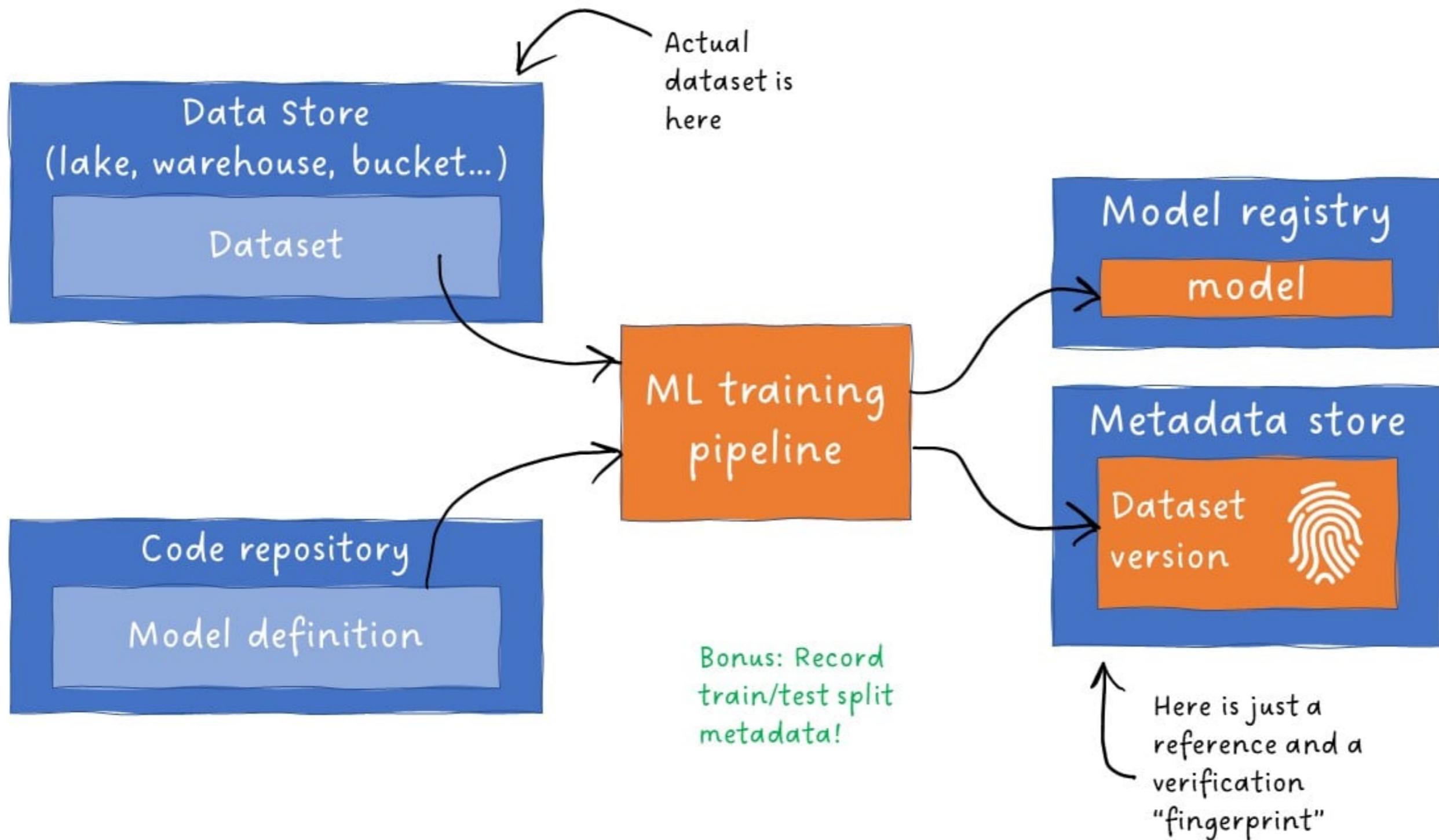
# ML pipeline checklist





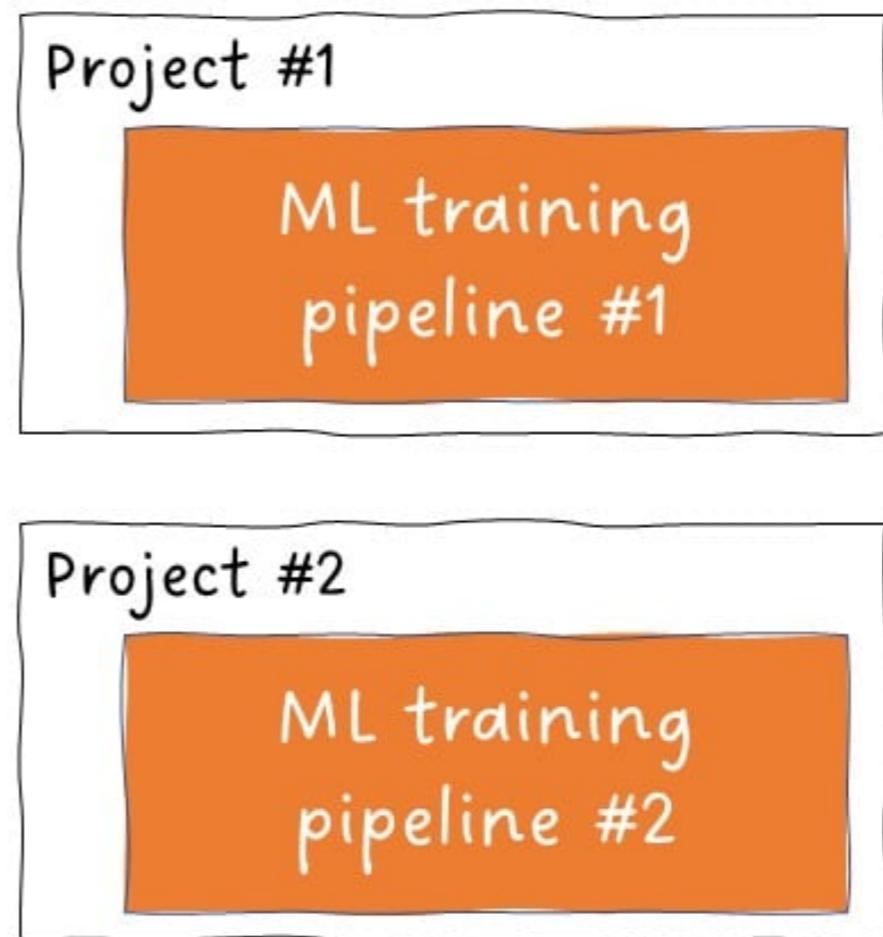
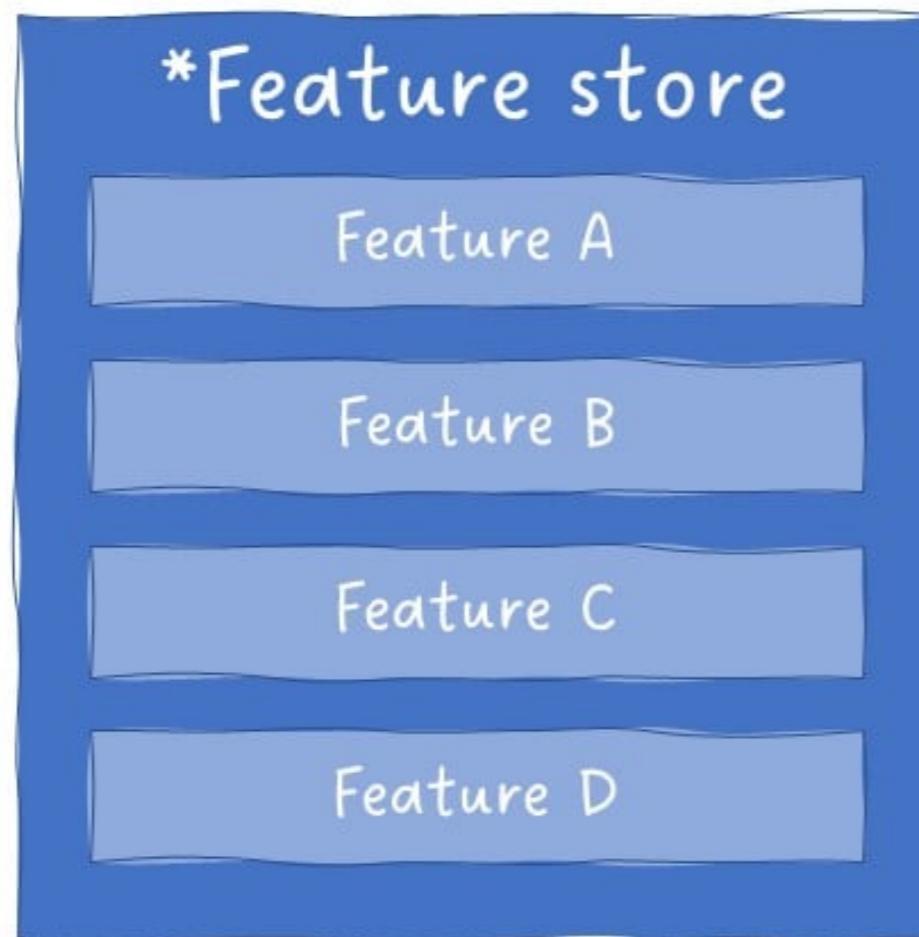


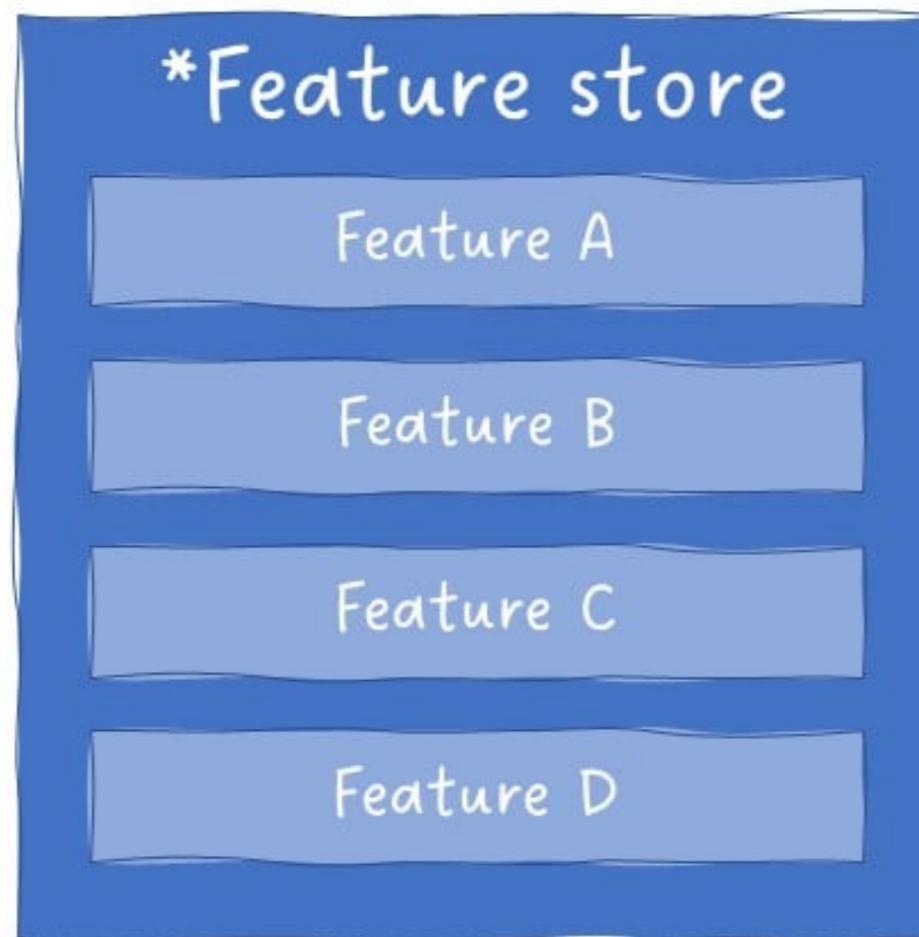




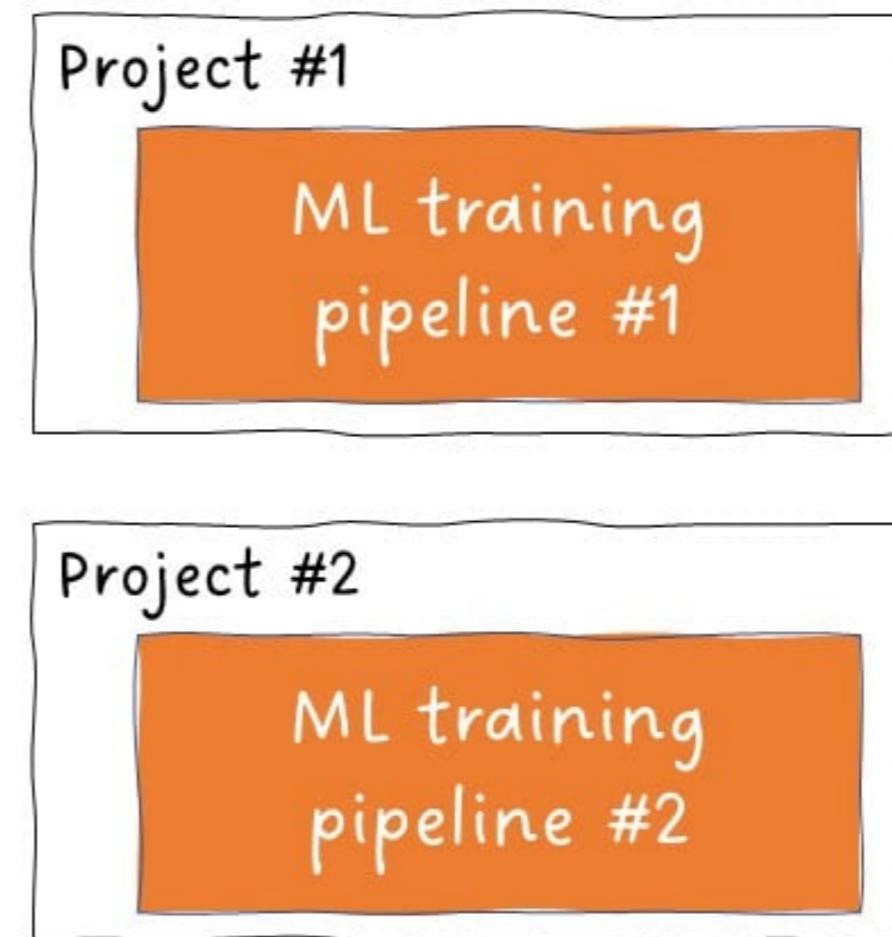


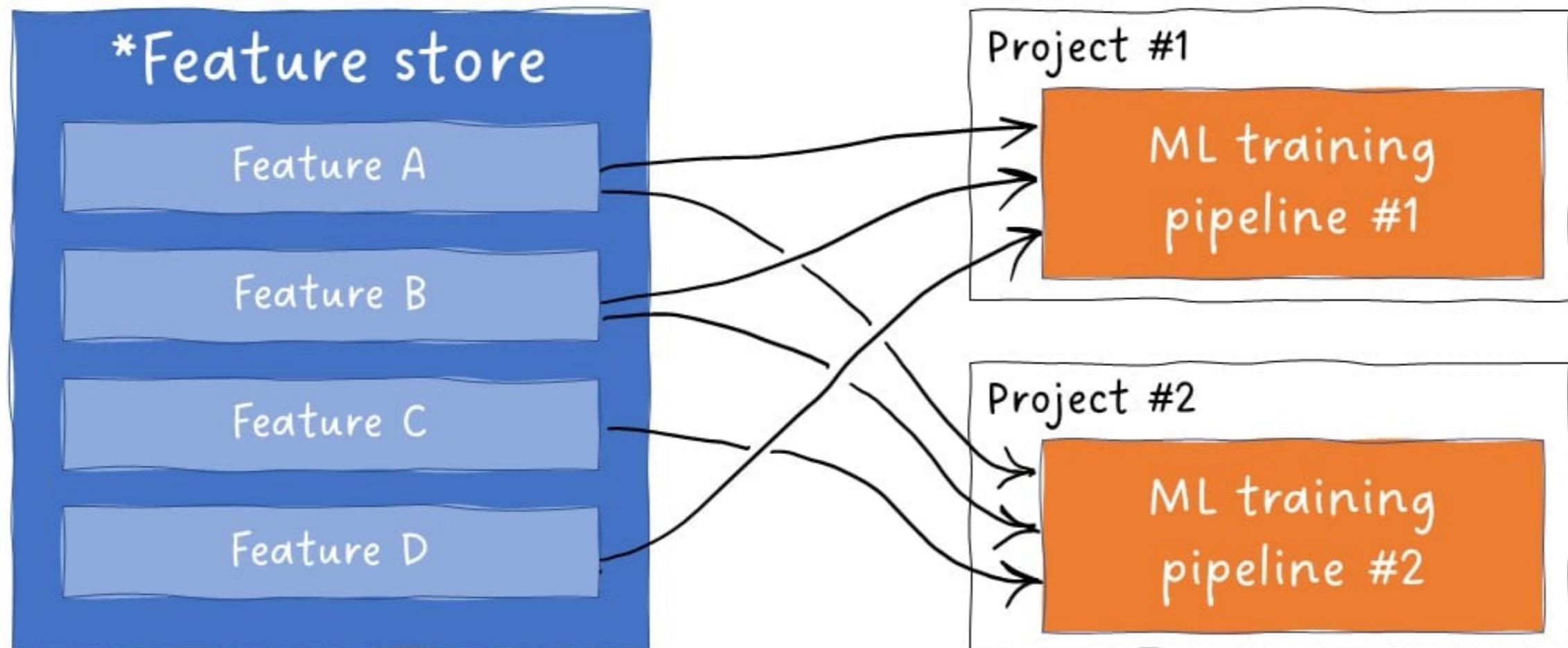
- Data
- Version
- Control



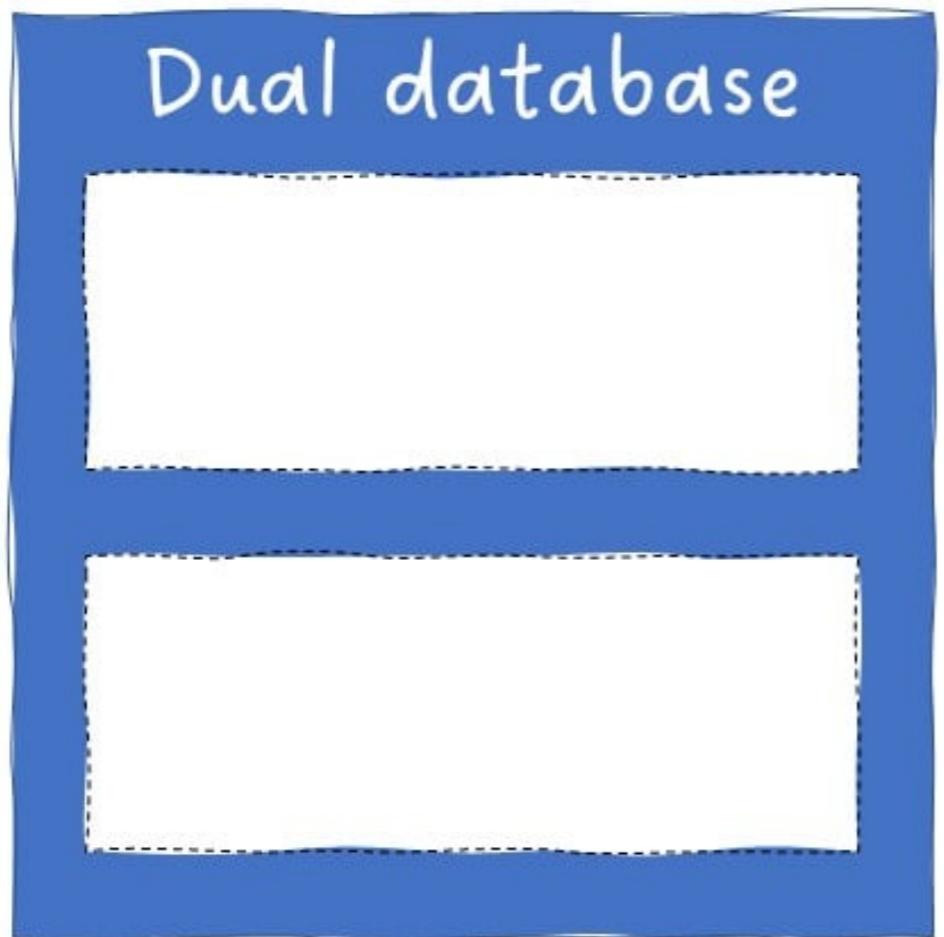


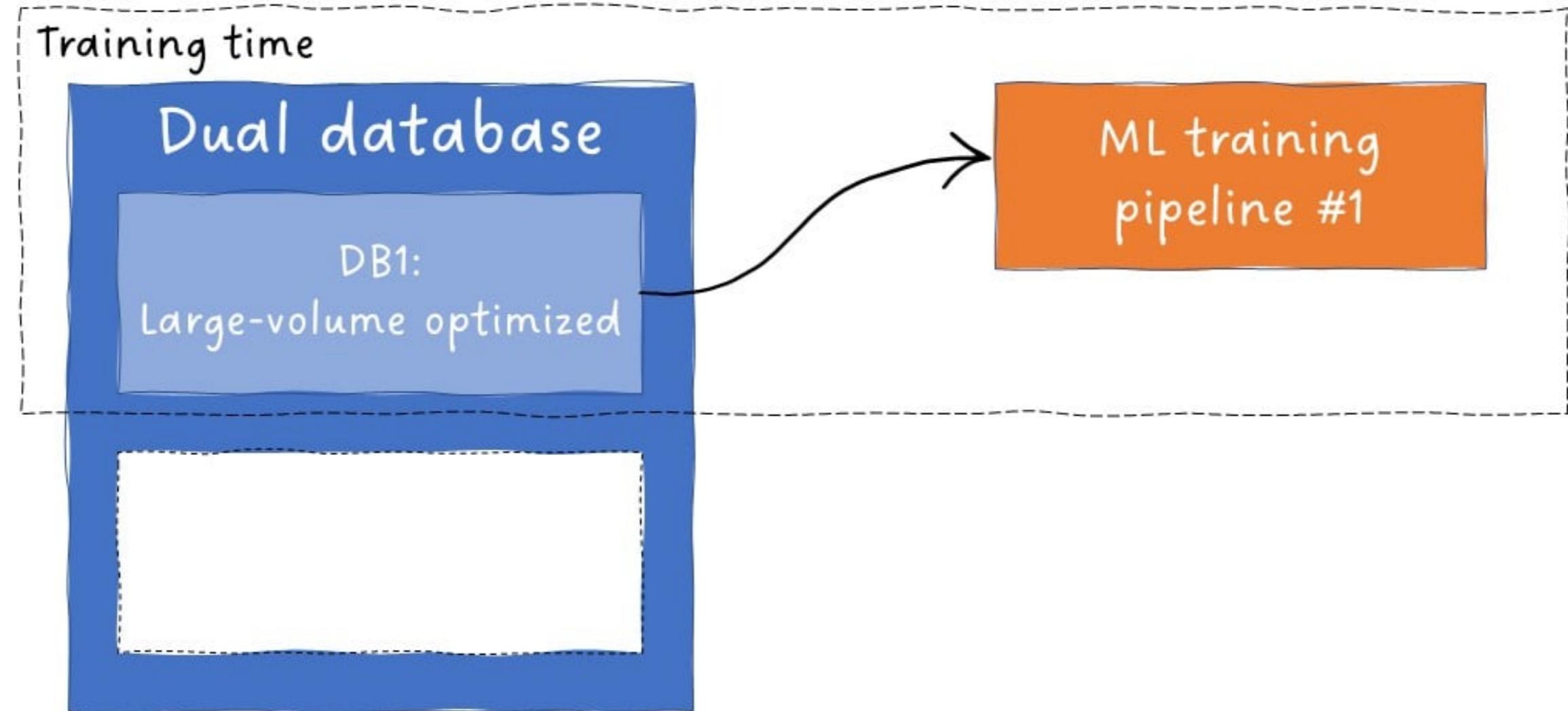
\* Essentially a database

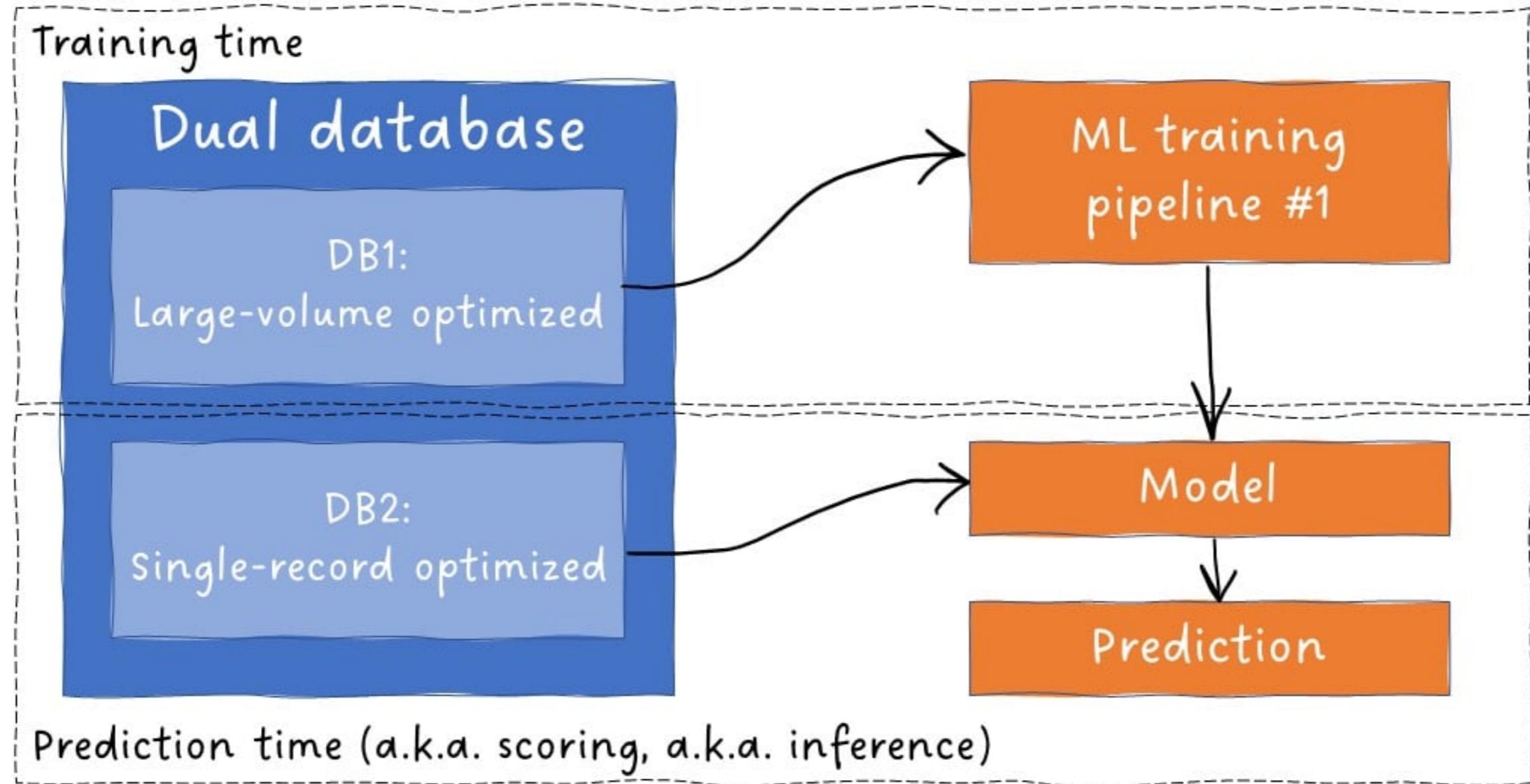




\* Essentially a database







# Feature store benefits

Reusability



# Feature store benefits

Reusability



Consistency  
(lower train/serve skew)

# Feature store benefits

Reusability



Consistency  
(lower train/serve skew)

Email in training

  Lorem ipsum dolor sit amet, consectetur adipisci  
  eiusmod tempor incididunt ut labore et dolore  
  magna aliqua. Ut enim ad minim veniam, quis nostrud  
  exercitationem ullamco laboris nisi ut aliquip ex ea  
  commodo consequat. Duis aute irure dolor in reprehenderit  
  in voluptate velit esse cillum dolore eu fugiat nulla  
  pariatur. Excepteur sint occaecat cupidatat non prosi  
  sunt in culpa qui officia deserunt mollit anim  
  Sed ut perspiciatis unde omnis iste natus e  
  accusantium doloremque laudantium, totam re  
  quae ab illo inventore veritatis et quasi a

Email for prediction

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <title>Title goes here</title>
  </head>
  <body>

  </body>
</html>
```

# **Let's practice!**

**MLOPS DEPLOYMENT AND LIFE CYCLING**

# Model build pipelines in CI/CD

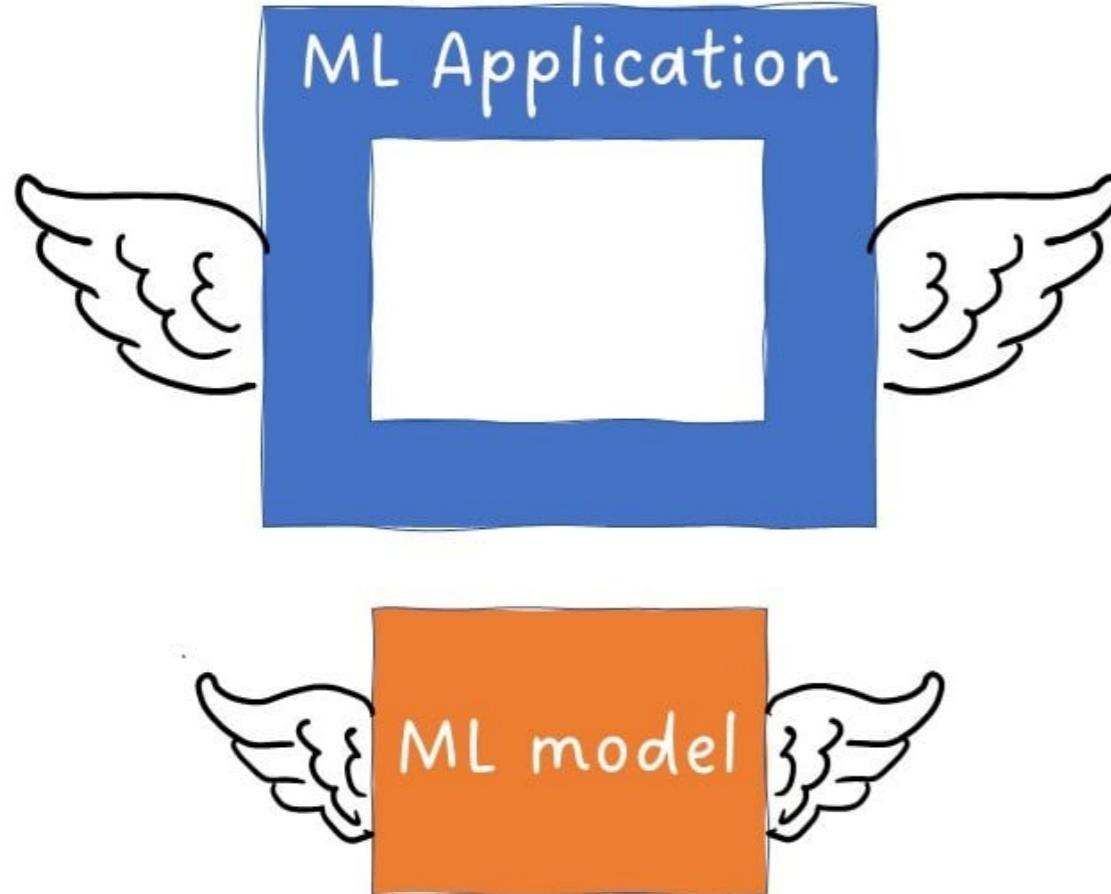
MLOPS DEPLOYMENT AND LIFE CYCLING



Nemanja Radojkovic

Senior Machine Learning Engineer

# Two build pipelines in ML

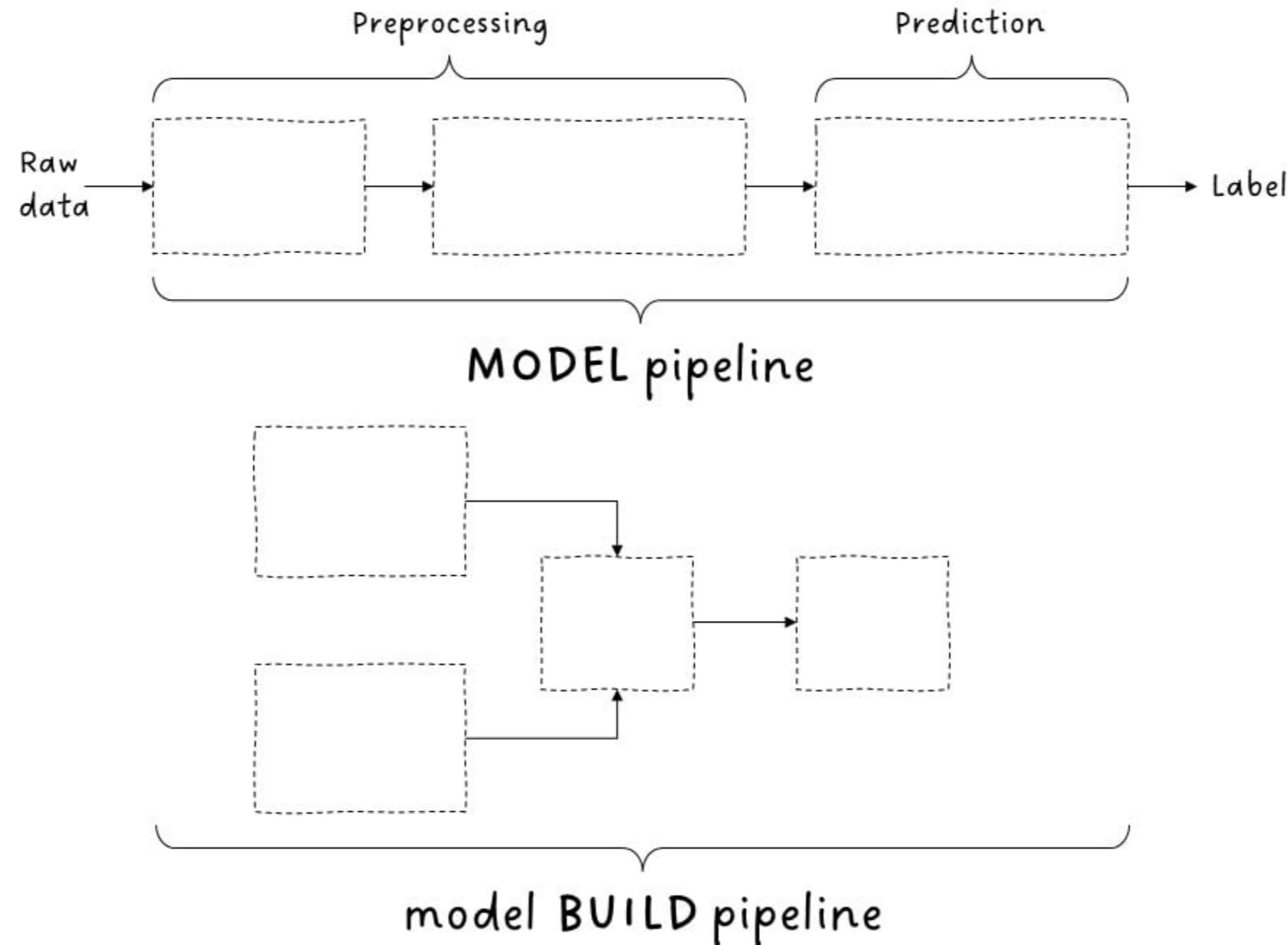


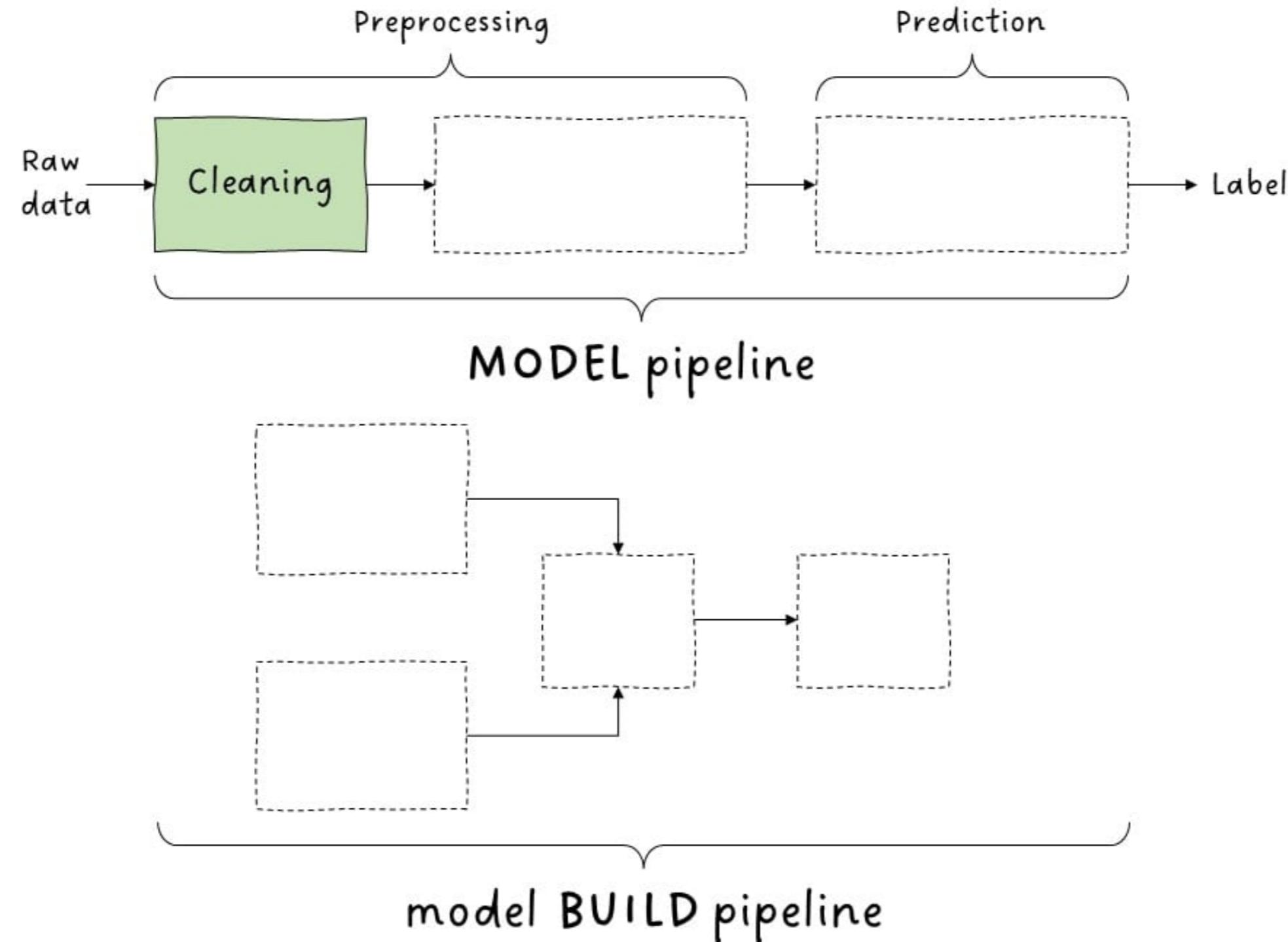
- One that builds the app that serves our model.
  - Standard software build pipeline
- One that builds (trains) the model itself
  - Central to the MLOps framework

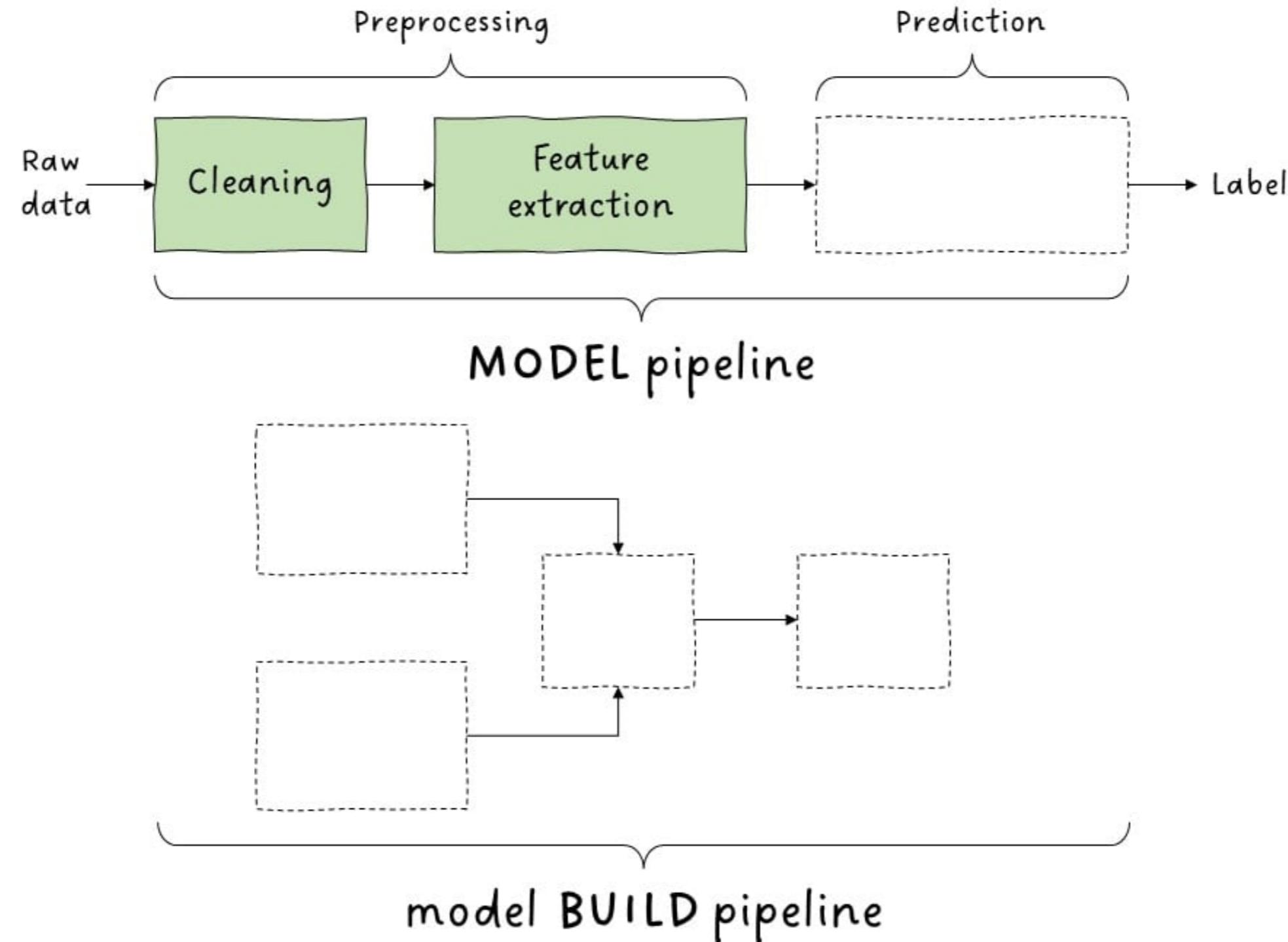
MODEL  
pipeline

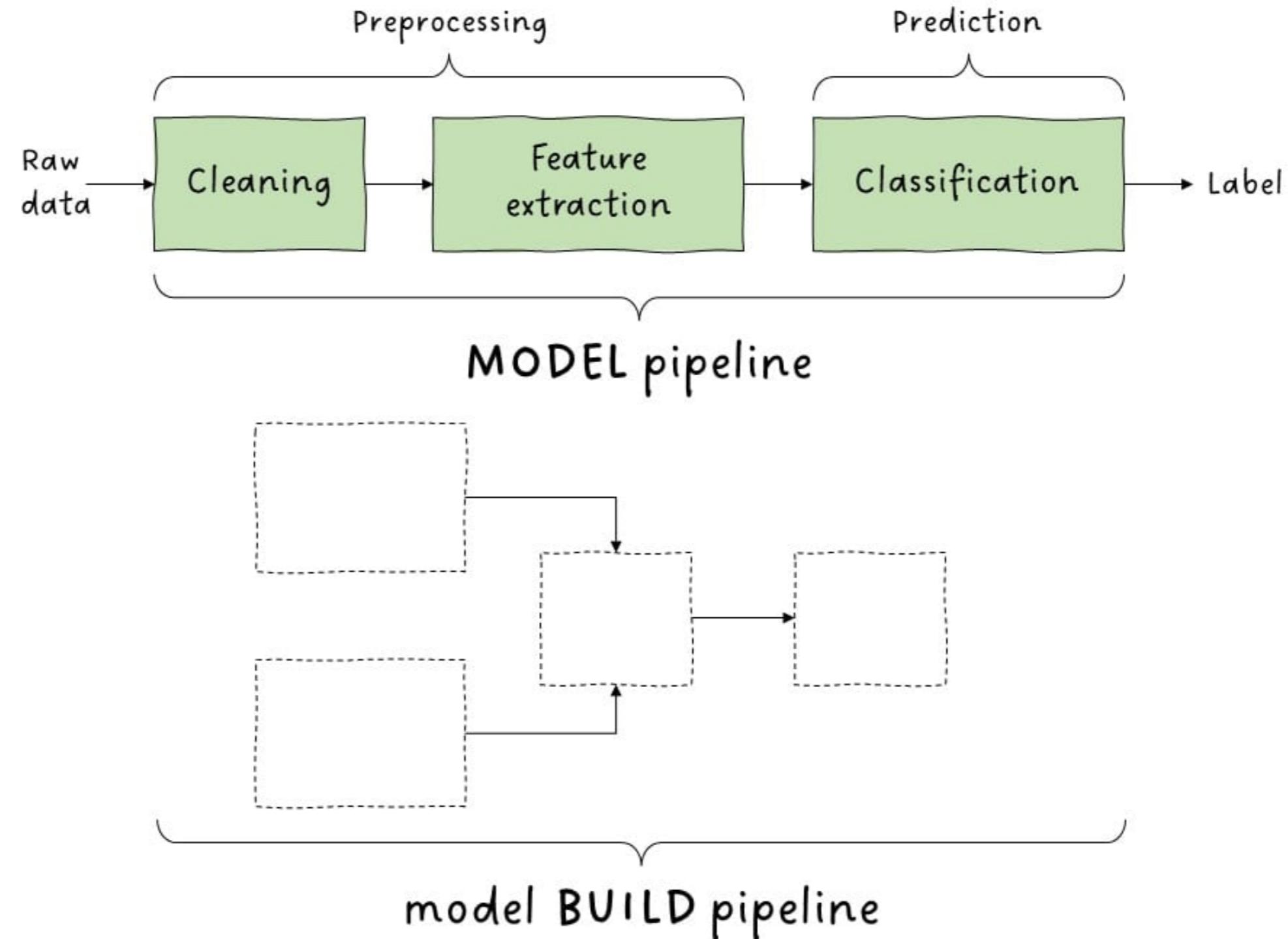
=/=

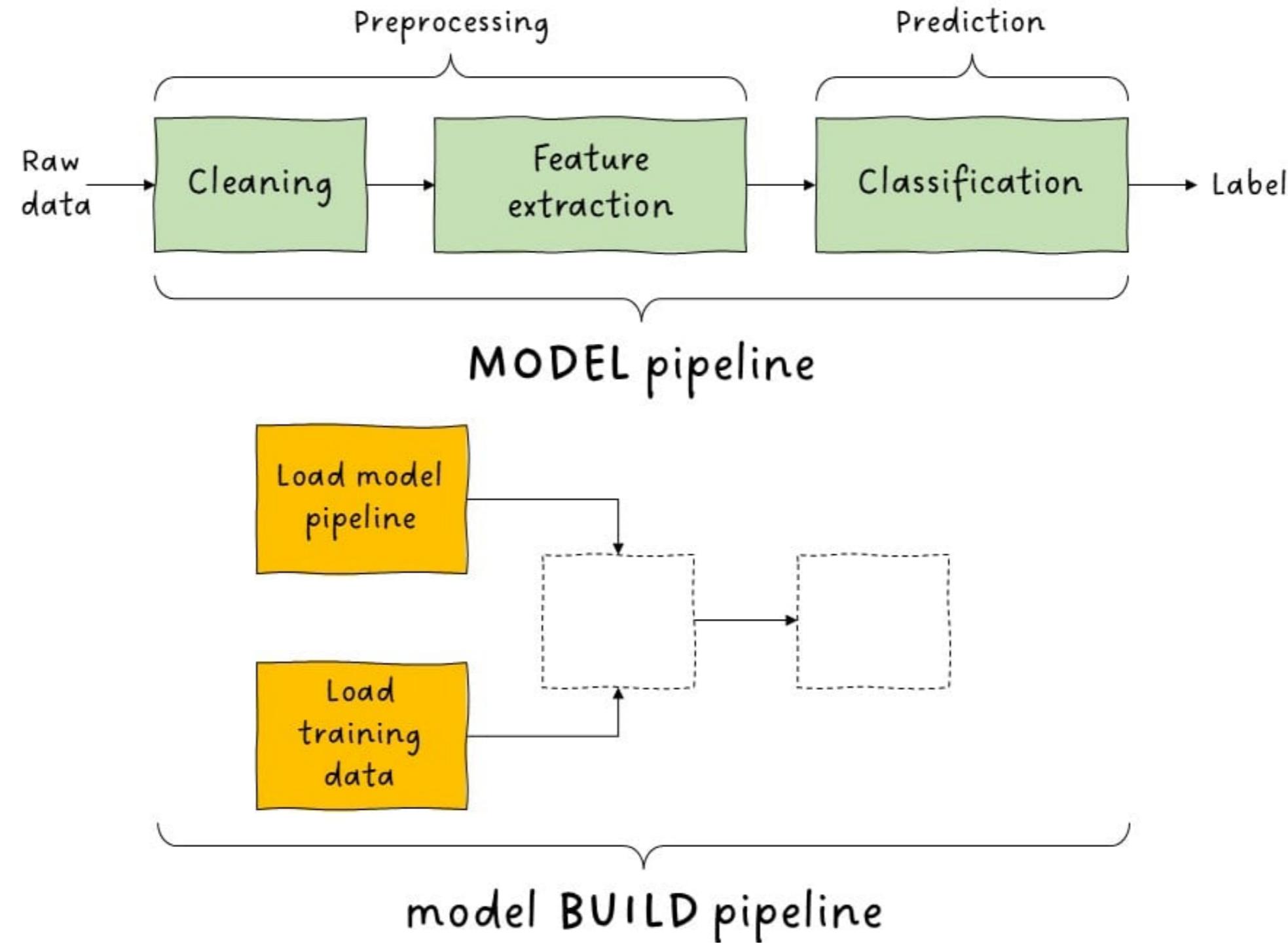
model  
BUILD  
pipeline

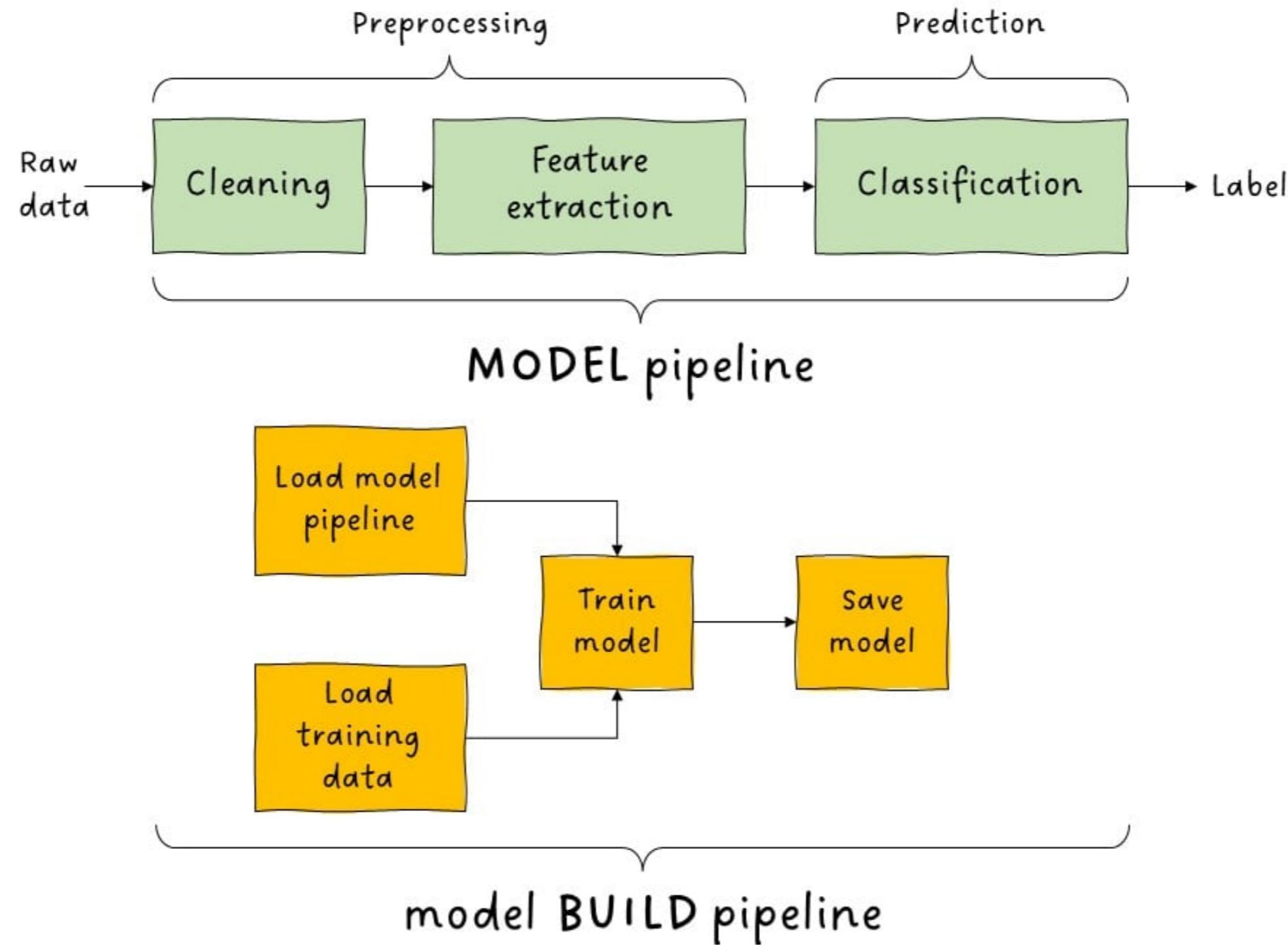


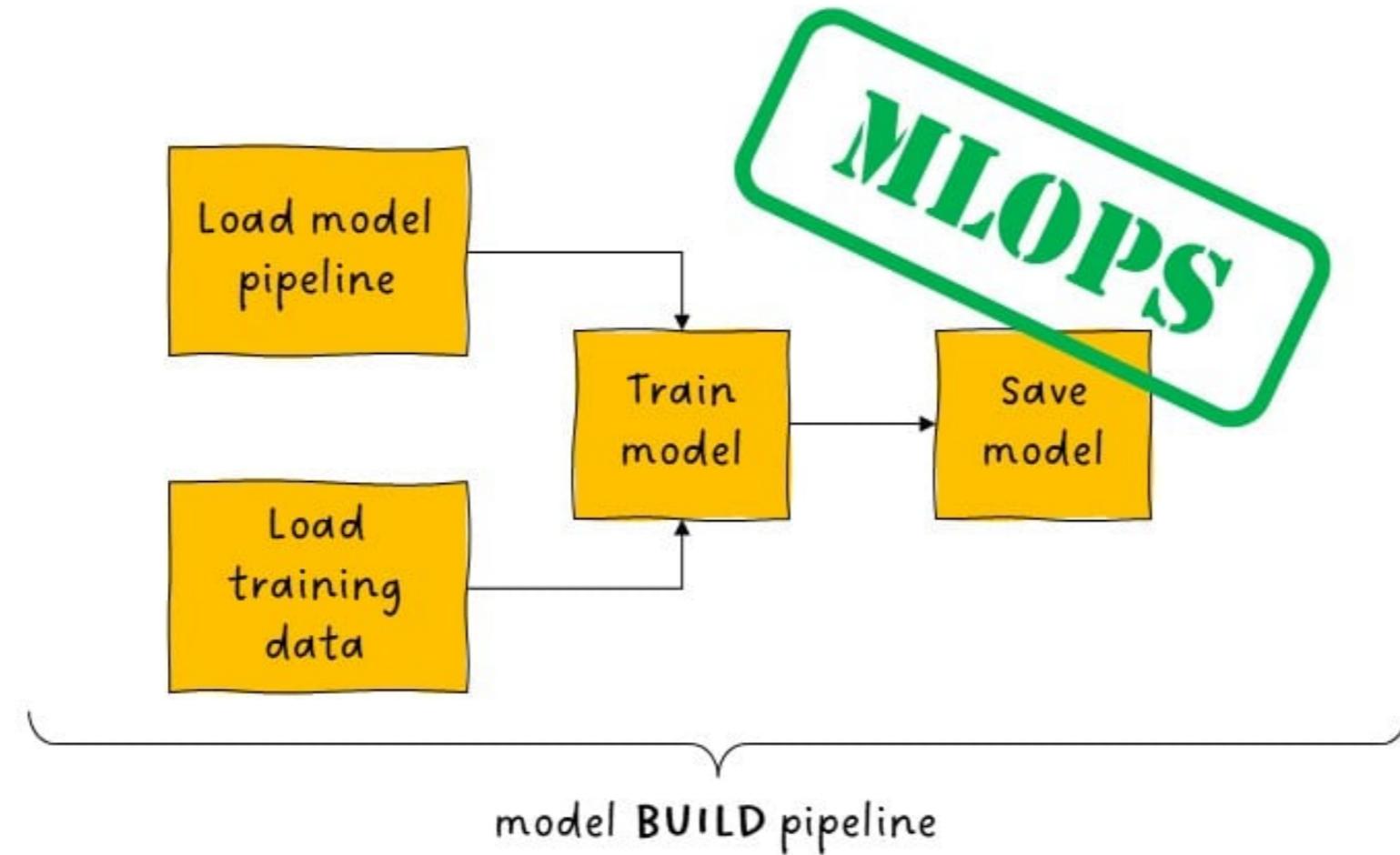




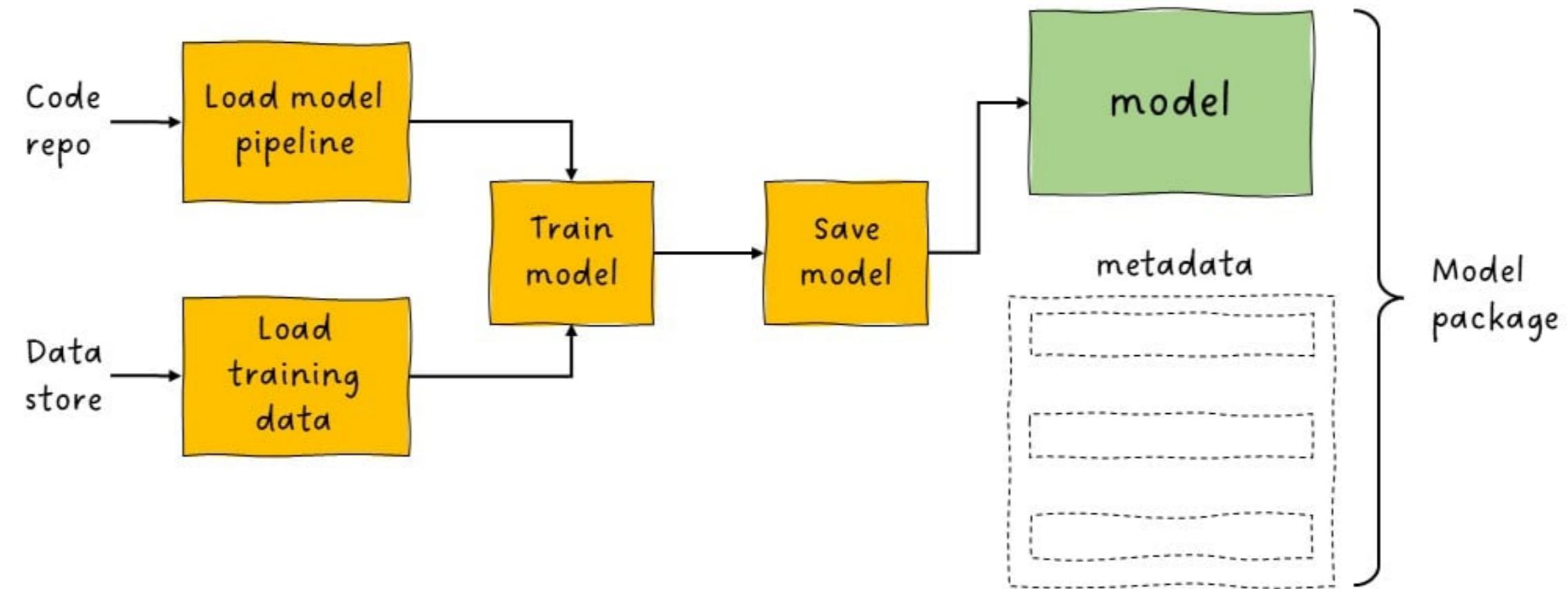


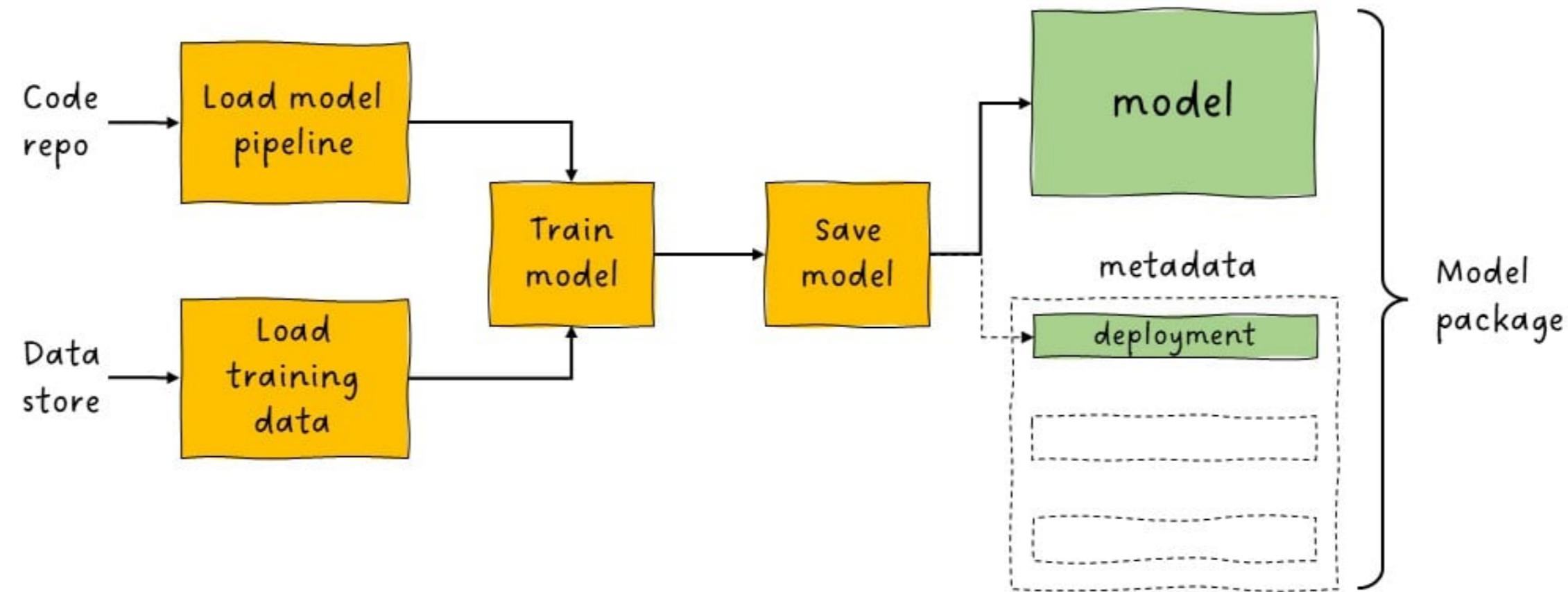




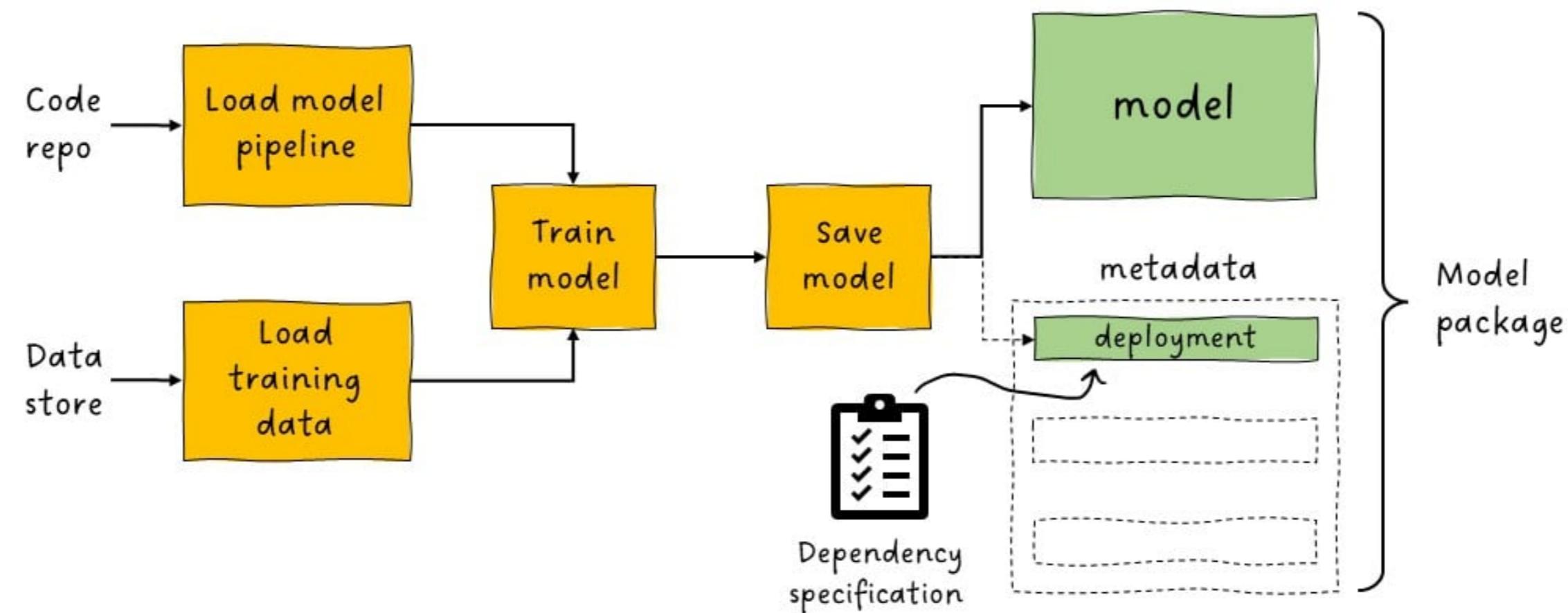


- ✓ Deployment
- ✓ Reproducibility
- ✓ Monitoring
- ✓ CI/CD integration



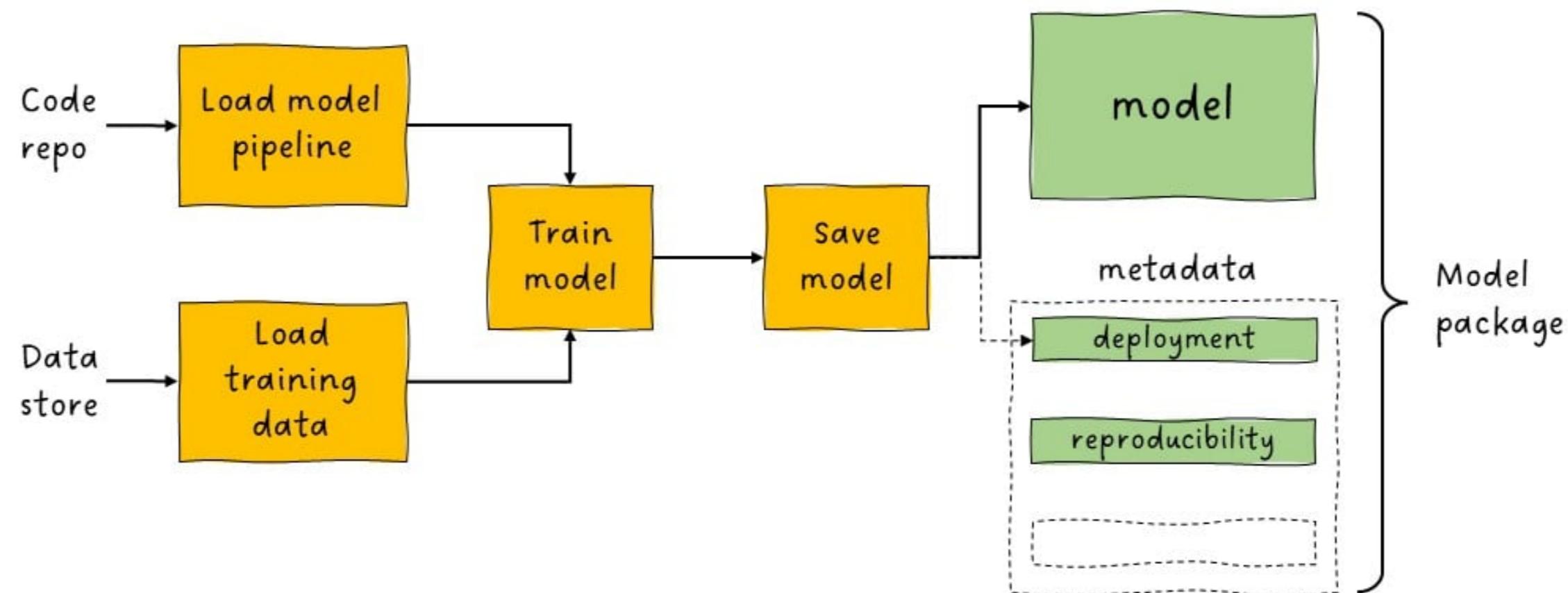


\* refresher: artifacts == pipeline outputs

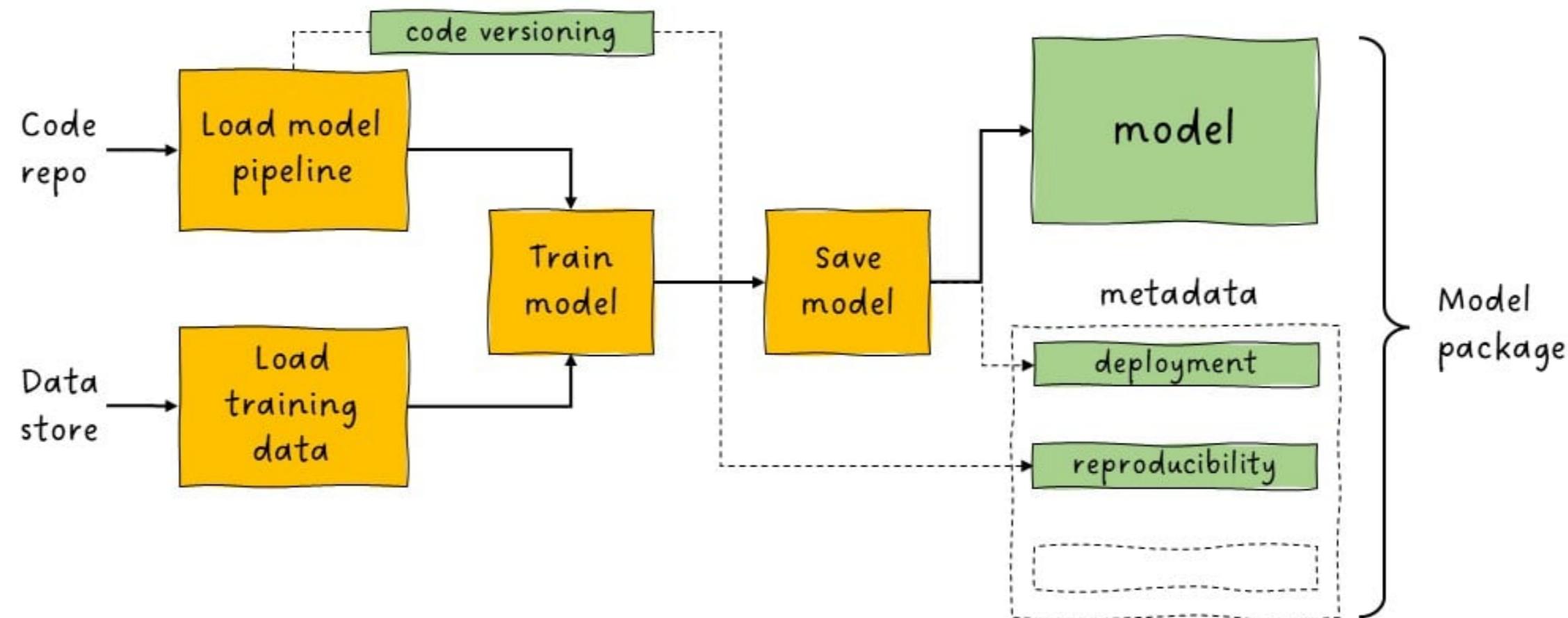


\* refresher: artifacts == pipeline outputs

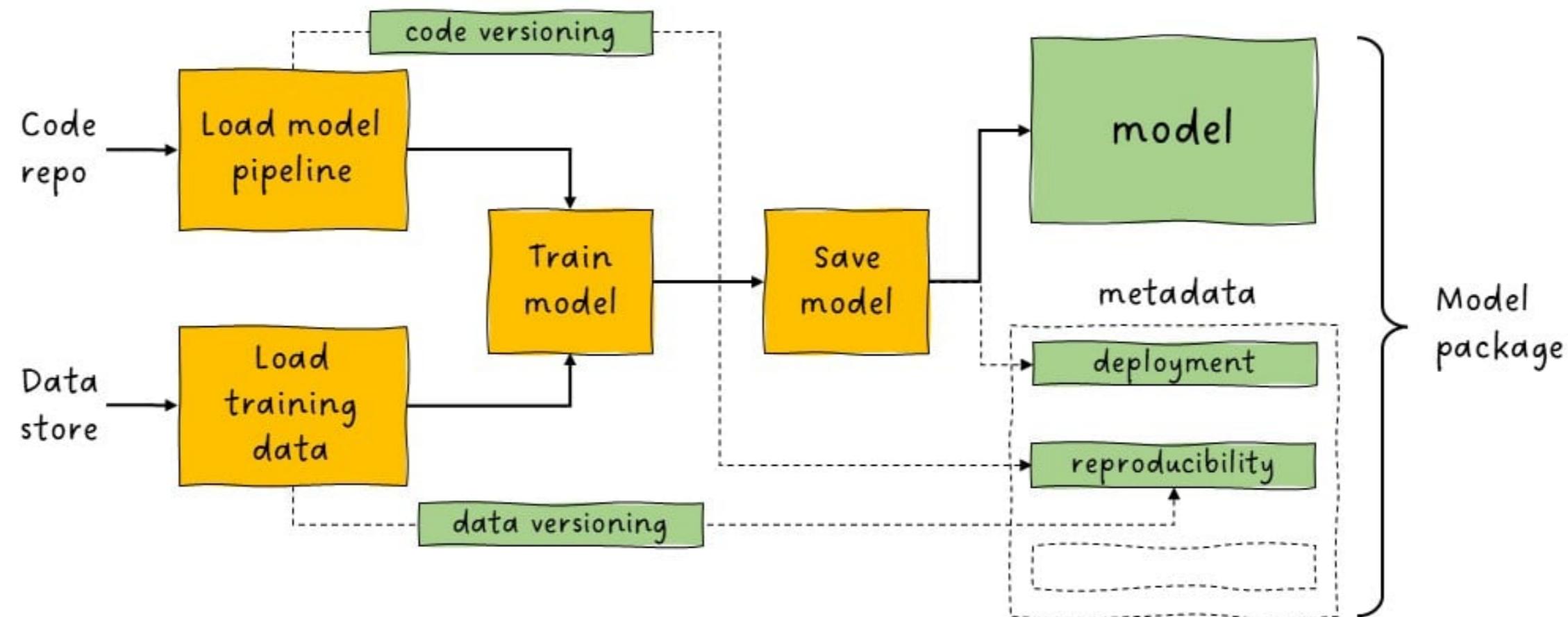
# Reproducibility = Control = Trust

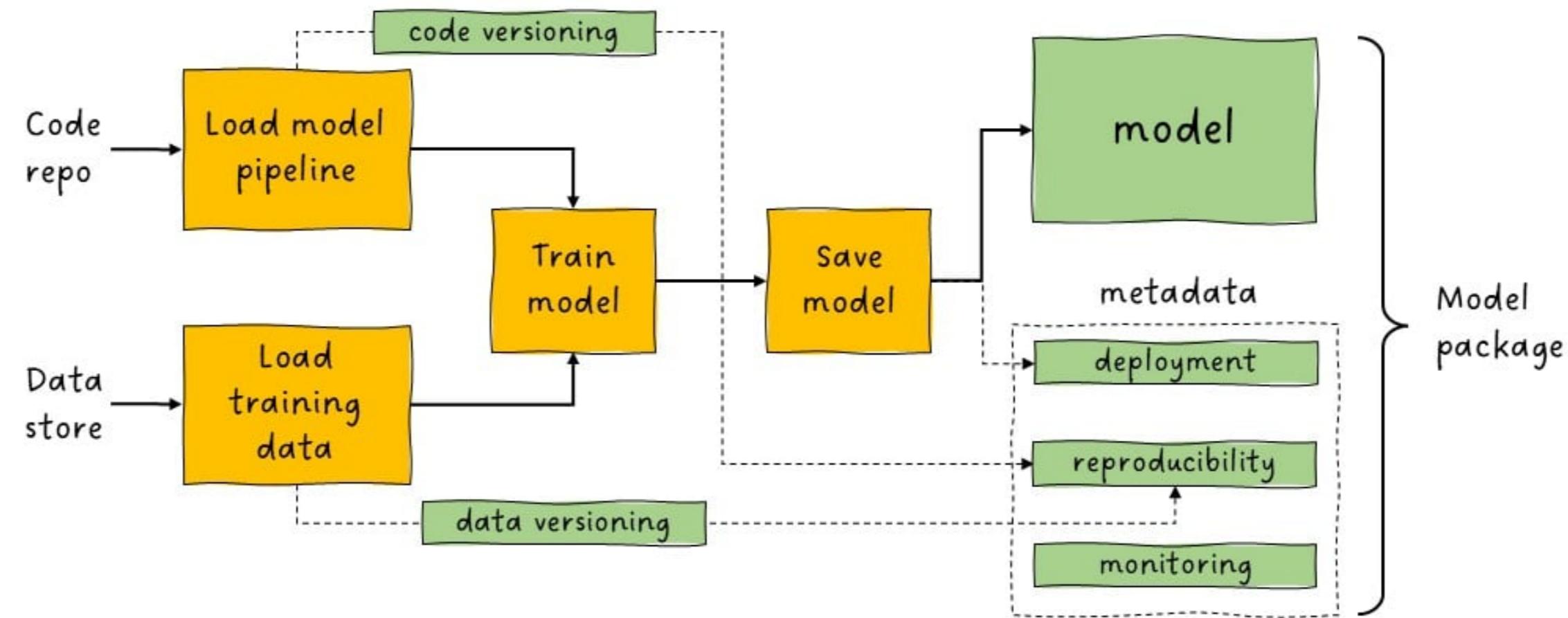


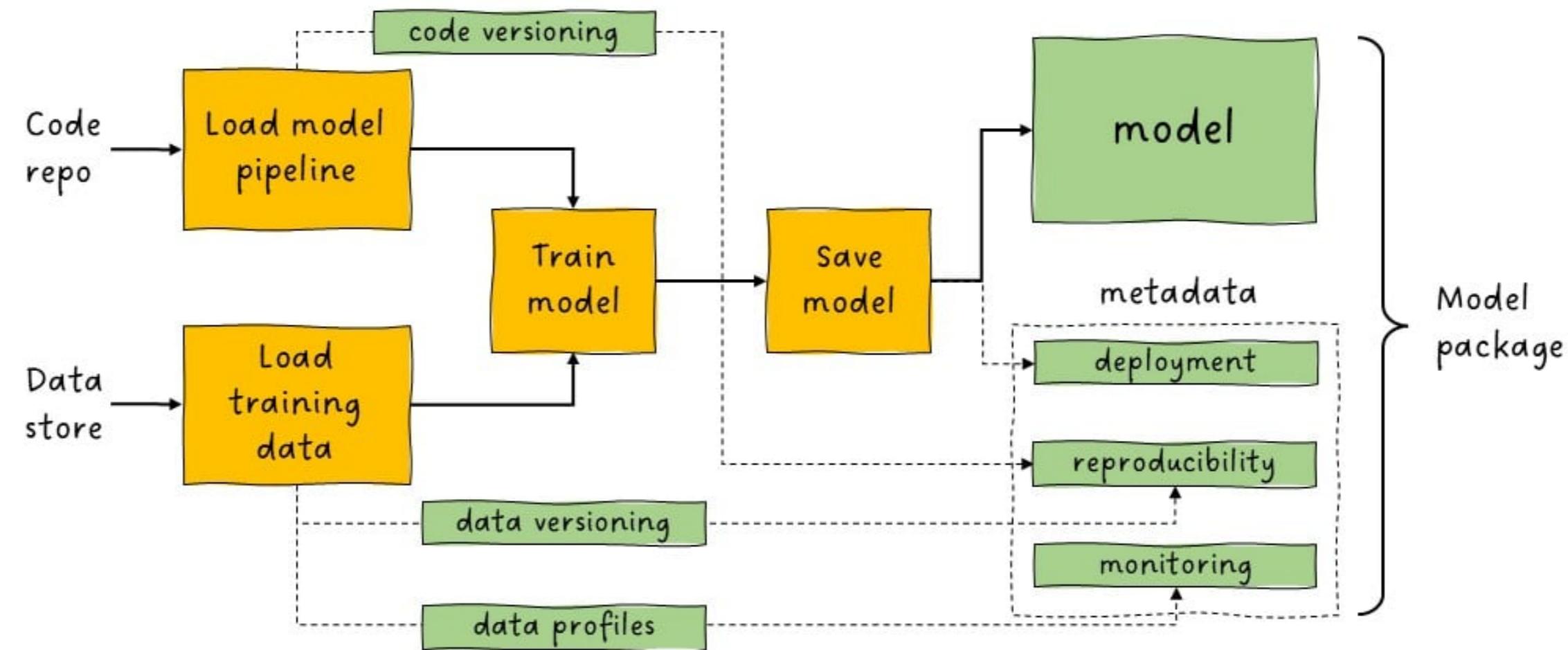
# Reproducibility = Control = Trust

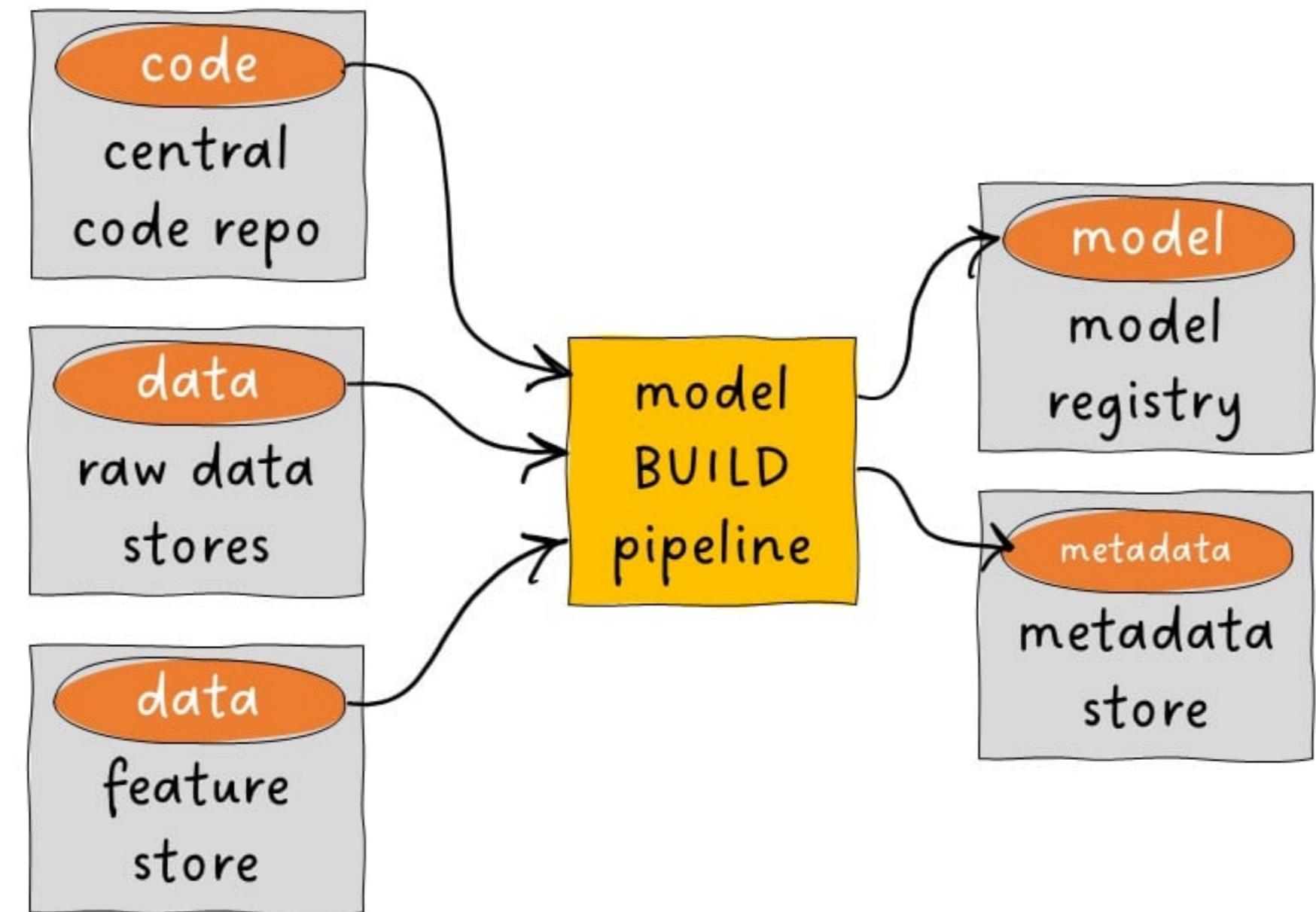


# Reproducibility = Control = Trust









# **Let's practice!**

**MLOPS DEPLOYMENT AND LIFE CYCLING**

# Model packaging

MLOPS DEPLOYMENT AND LIFE CYCLING



**Nemanja Radojkovic**

Senior Machine Learning Engineer





deployment





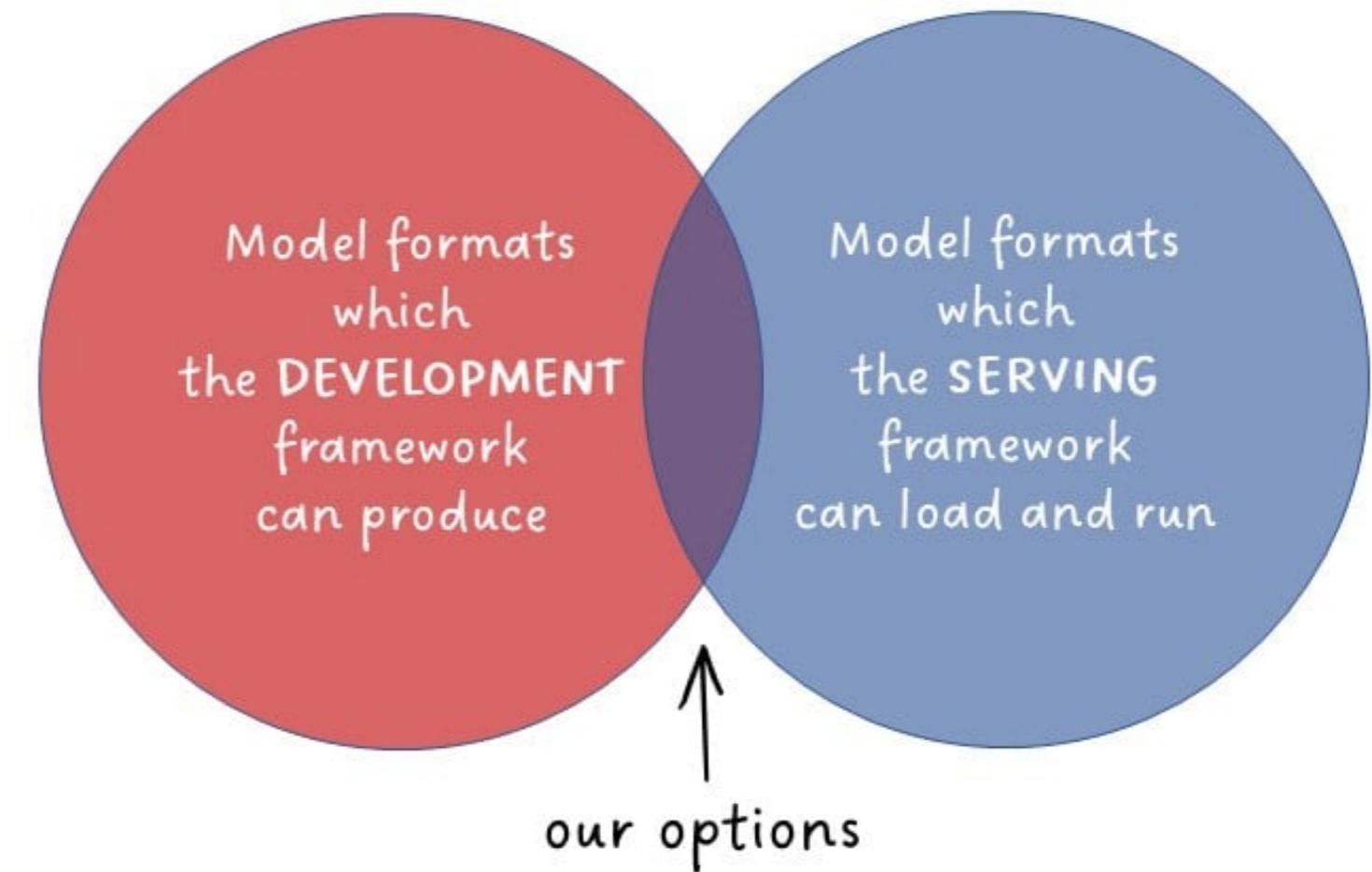
reproducibility

deployment

monitoring



## Model storage options



# Examples of model storage formats



 python  
pickle

The Python logo icon consists of two interlocking snakes, one blue and one yellow. To its right, the word 'python' is written in a lowercase, sans-serif font. Below it, the word 'pickle' is also written in a lowercase, sans-serif font.

# Examples of model storage formats



 python  
pickle

## PROs

- Universal
- Train using one programming language, serve using a completely different one.

## CONs

- Difficult to customize

# Examples of model storage formats



## PROs

- Universal
- Train using one programming language, serve using a completely different one.

## CONs

- Difficult to customize



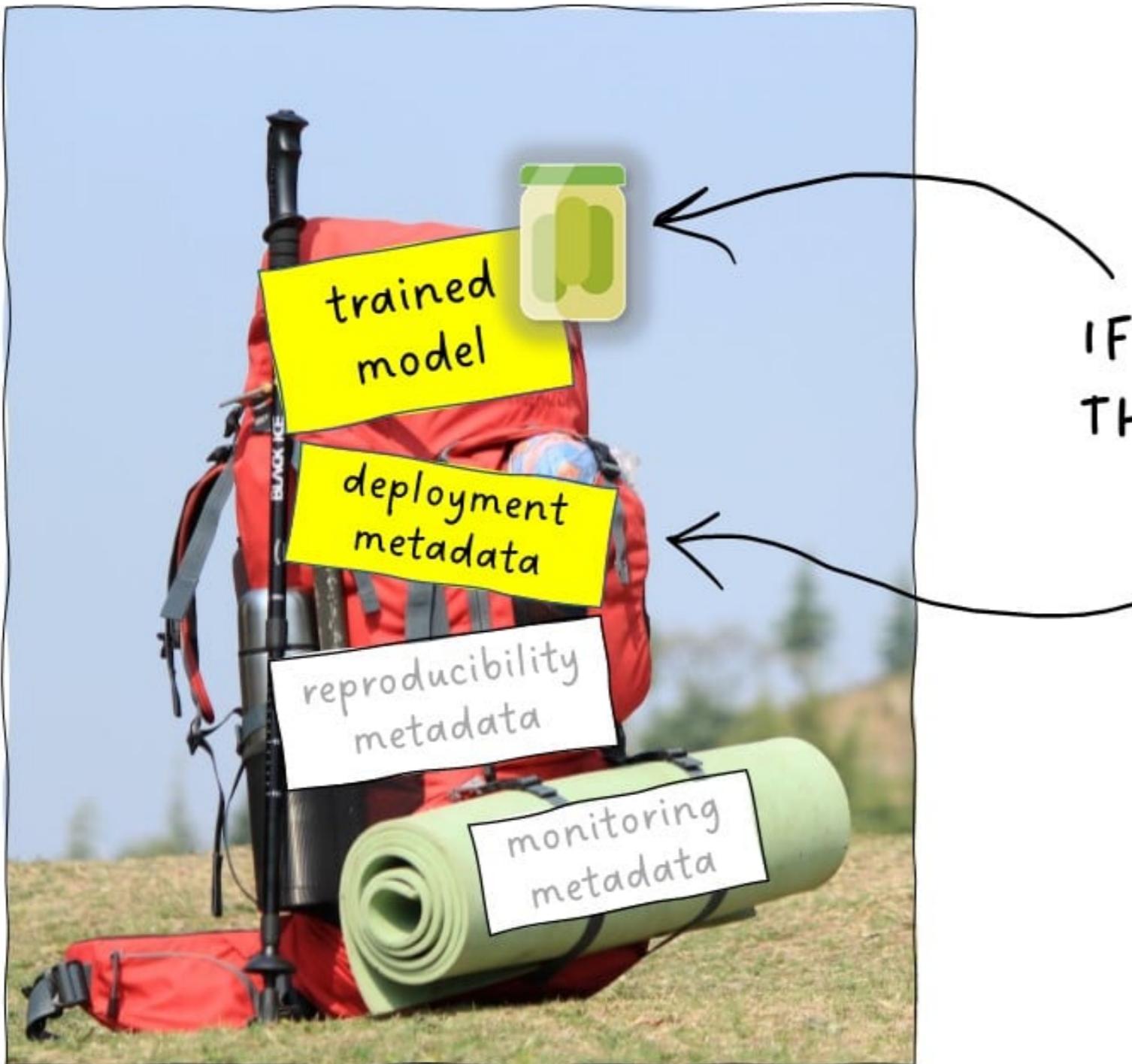
 python  
pickle

## PROs

- Python ecosystem
- Not ML specific, can store anything

## CONs

- No cross-platform compatibility. Training and serving environments must be identical.



IF `model_format == .pickle`

THEN:



store list of model's  
dependencies within model  
package and verify server  
compatibility when loading



## Reproducibility

Automated model re-creation

Reproducibility == Control == Trust



## Reproducibility checklist

- Pointer to the exact version of the model build pipeline code.
- Pointer to the exact versions of the datasets used during the training
  - including the train/splits during performance evaluation.
- Record of the performance achieved on the test set.



## Monitoring checklist

- Input data profile
- Output data profile



Lock 'n' load!

# **Let's practice!**

**MLOPS DEPLOYMENT AND LIFE CYCLING**