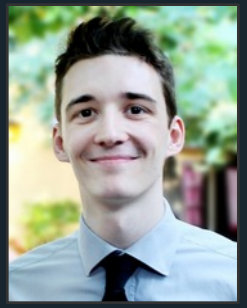
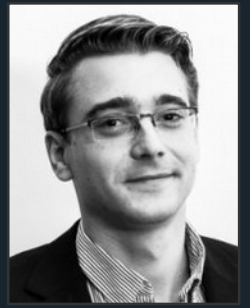
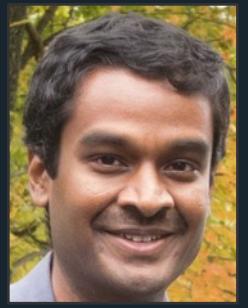
An Experimentation and Analytics Framework for Large-Scale AI Operations Platforms

Thomas Rausch

Waldemar Hummer

Vinod Muthusamy

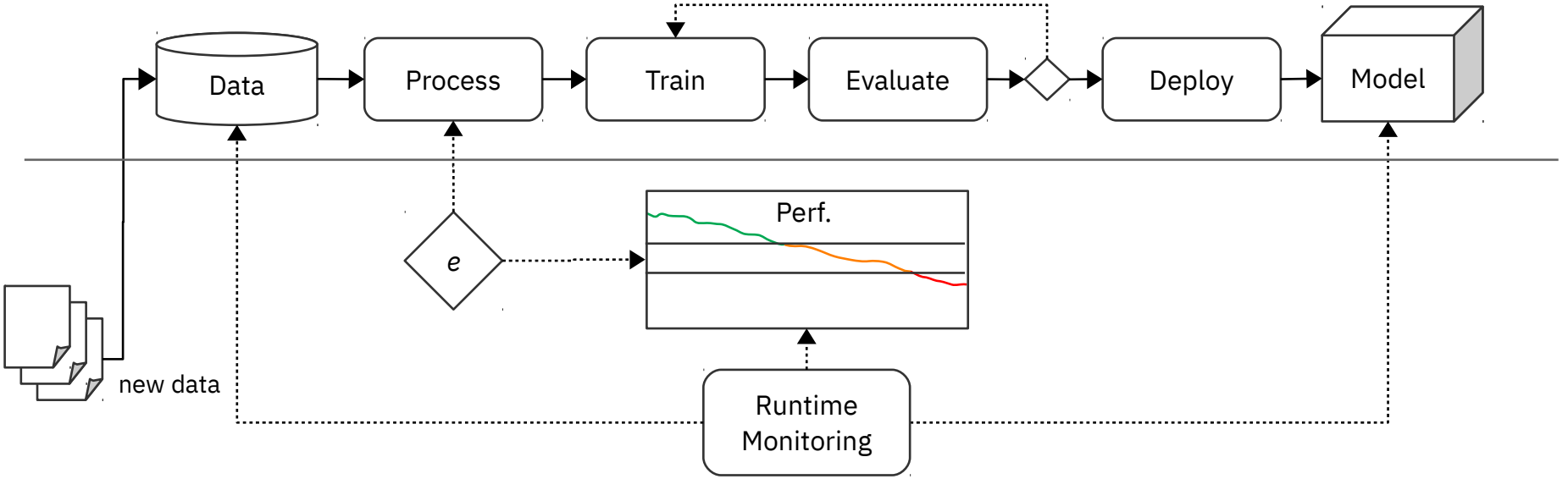




usenix ASSOCIATION OpML'20, July 27 – August 3, 2020

IBM **Research AI**

TU WIEN TECHNISCHE UNIVERSITÄT WIEN

Operationalizing AI



Watson AI OpenScale

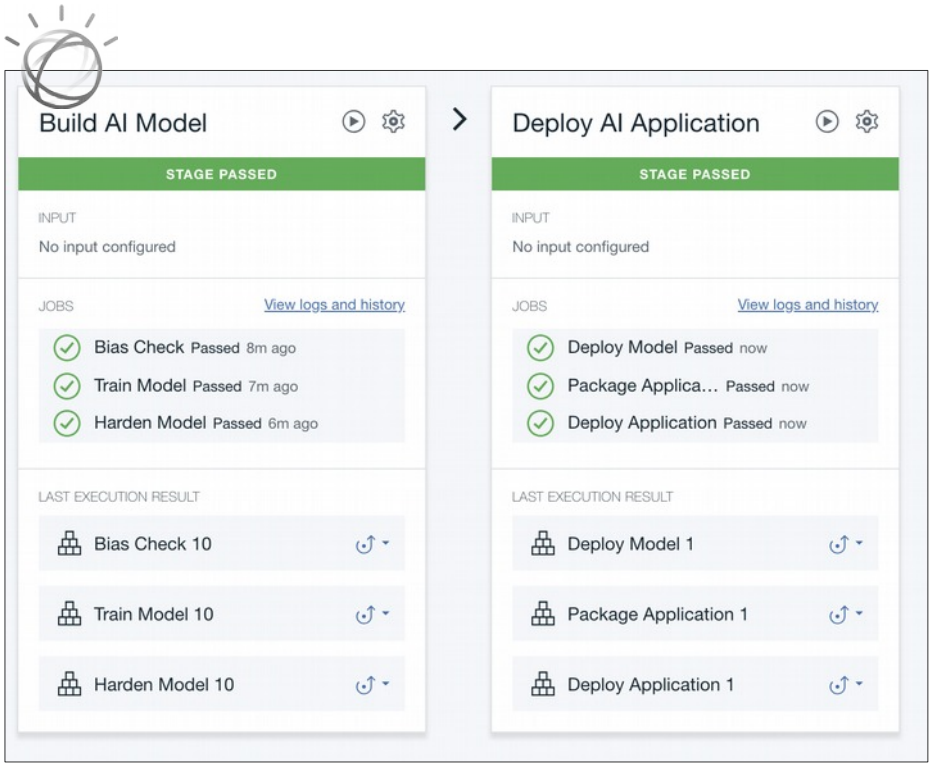
TensorFlow Extended

Azure ML Pipelines ...

ModelOps: Cloud-based Lifecycle Management for Reliable and Trusted AI

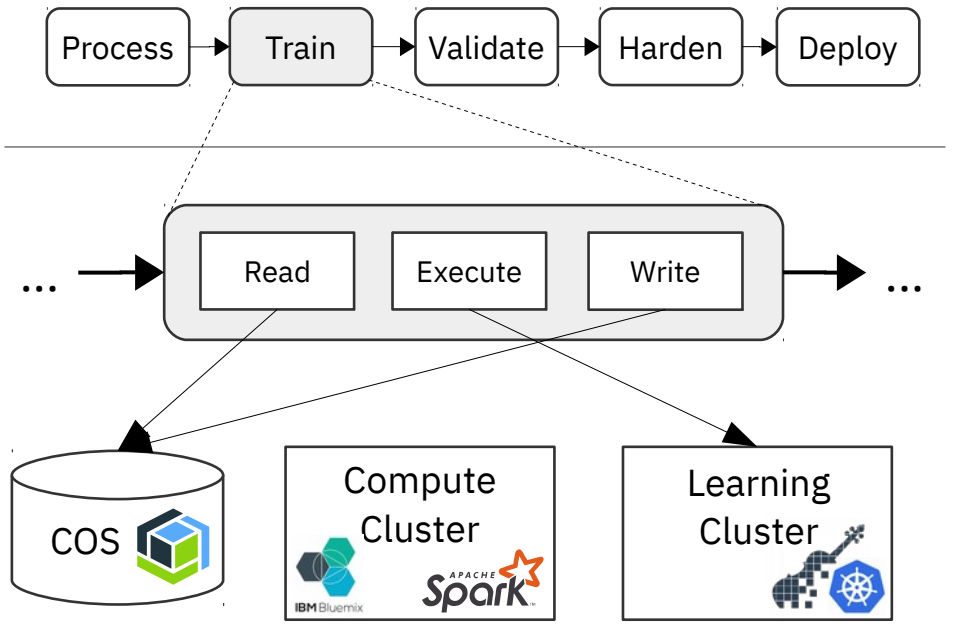
Waldemar Hummer, Vinod Muthusamy, Thomas Rausch, Parijat Dube, Kaoutar El Maghraoui
IBM Research AI
1101 Kitchawan Rd, Yorktown Heights, NY 10598, United States
{whummer,thomas.rausch}@ibm.com, {vmuthus,pdube,kelmaghr}@us.ibm.com

Operationalizing AI: Three Views



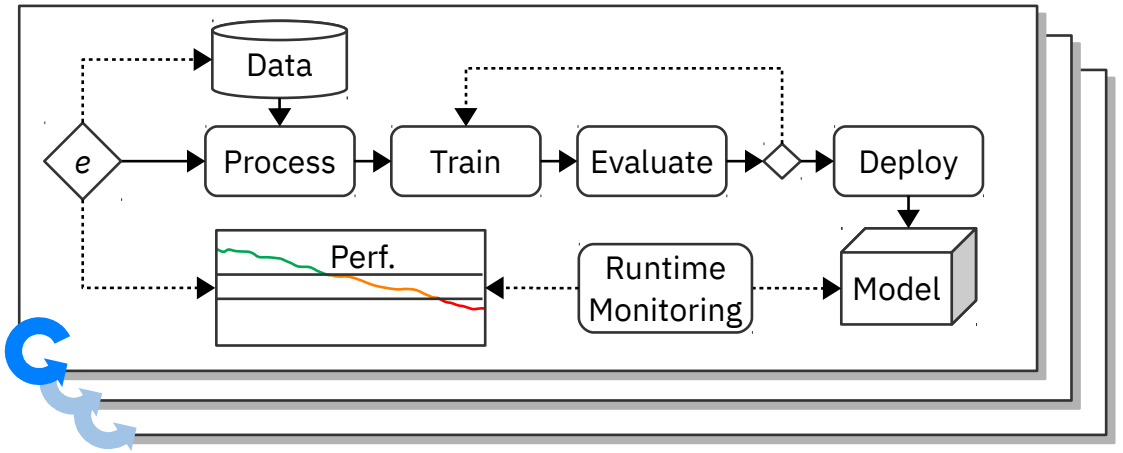
User View

Abstract **Pipeline** View



Platform View

Operating Automated AI Pipelines



e Retraining rule/trigger

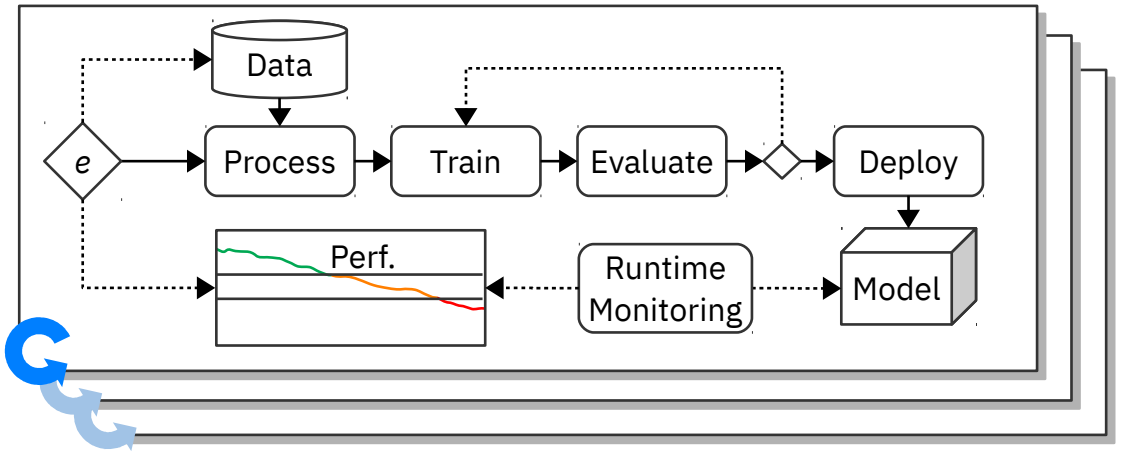
```
if performance < 0.8 and
   cnt(new_data) > n
```

```
every 4 weeks
```

?

- **Models become *stale***, automatic retraining will become common
- Operators will be challenged to maintain many automated continuous training loops
- **Cost-benefit trade-off between retraining and model performance improvement**

Operating Automated AI Pipelines



e Retraining rule/trigger

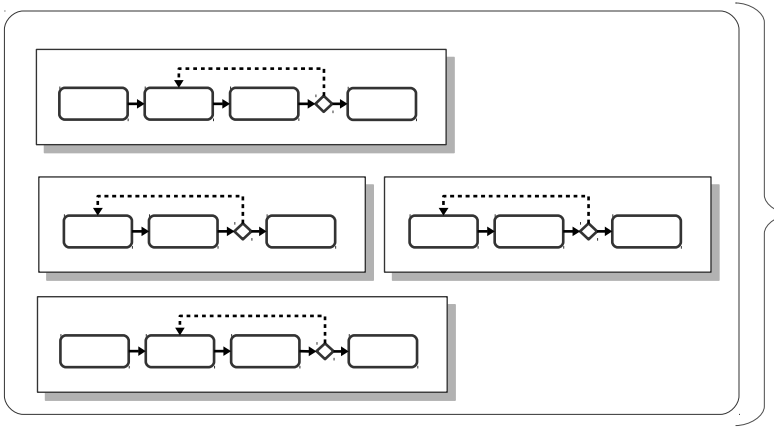
```
if performance < 0.8 and
   cnt(new_data) > n
```

```
every 4 weeks
```

Scheduler / Optimizer

- **Probabilistic parameters**
  - Expected improvement (**EI**)
  - **Risk** (e.g., human intervention)
  - **Priority** (how important is this model/tenant)?
- **Resource availability**

Infrastructure / Budget



Fitness?

The AIOps Scheduling Problem

DRAFT

Given a set of **continuous training loops** that are **scheduled to run automatically to meet SLAs**

where SLAs are soft-constraints imposed on the scheduling policy, e.g.,
- ACU > 0.8
- at least every 4 weeks
- at X new data

and given the current state of the execution environment, and historical information on pipeline executions

assign to each pipeline a timestamp at which to execute the pipeline next, such that the following goals are met or optimized:
- Maximize estimated improvement (EI)
- Minimize SLA violations
- Maintain fairness across users
- Balance resource use of infrastructure (given regular operations and serving of jobs)

Maximize estimated improvement (or minimize 'stalenes
- Estimated improvement of a pipeline could be calculate
  - the number of newly available labeled data for retra
  - time since last retraining
  - current model performance
- Hard facts: model drift

Minimizing SLA violations:
- Prioritize critical pipelines, i.e. where SLA violations ar
- Requires the calculation of a delta between the current state and the SLA goal (e.g. last execution time - min schedule frequency; or current observed ACU vs. SLA specified ACU)

Maintain fairness across users:
- Maximizing the overall EI may result in bias towards users that own many pipelines that degrade quickly
- Pipelines of users should not suffer from starvation

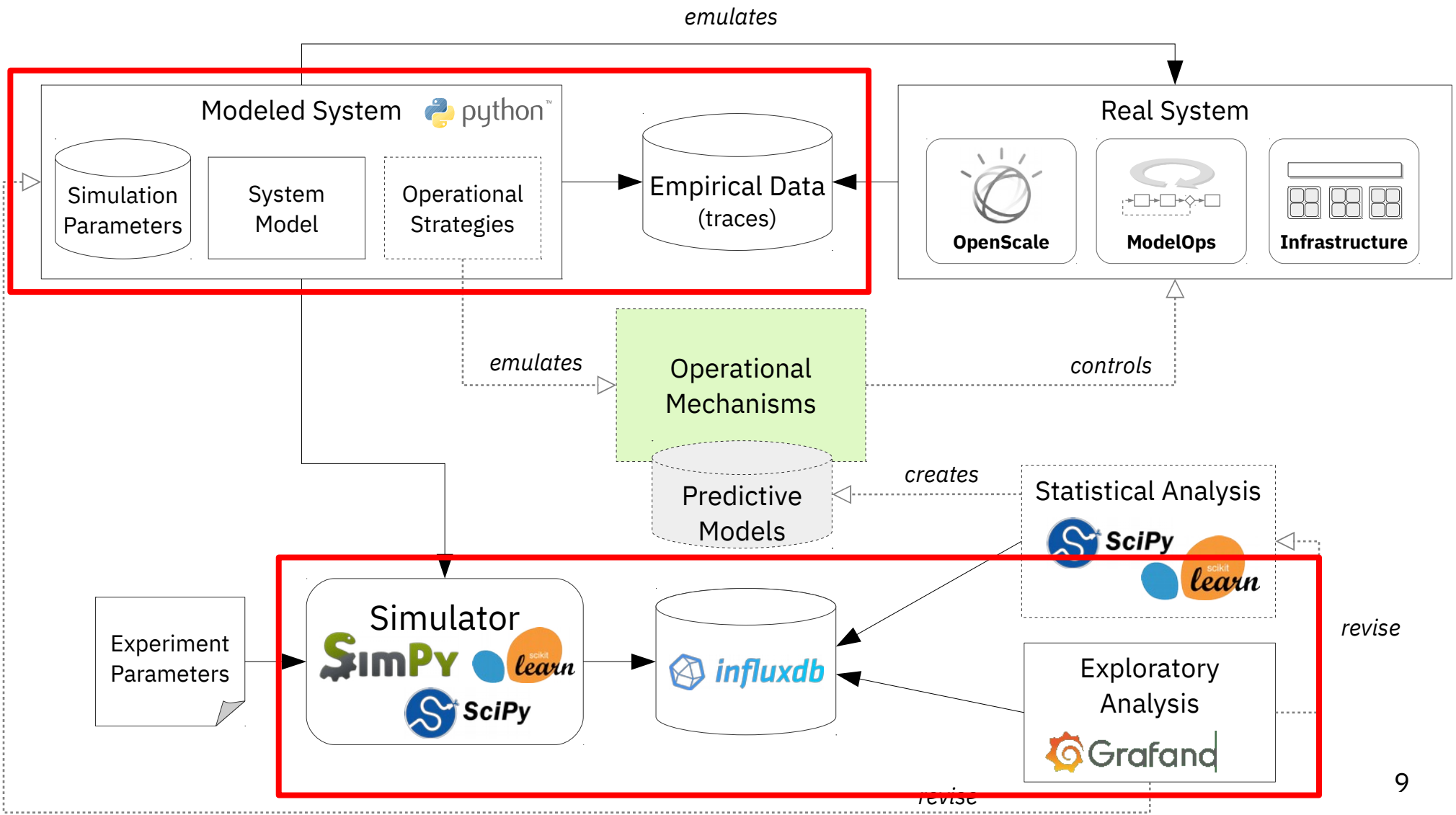
Balance resource use of infrastructure:

System model?
How to validate?

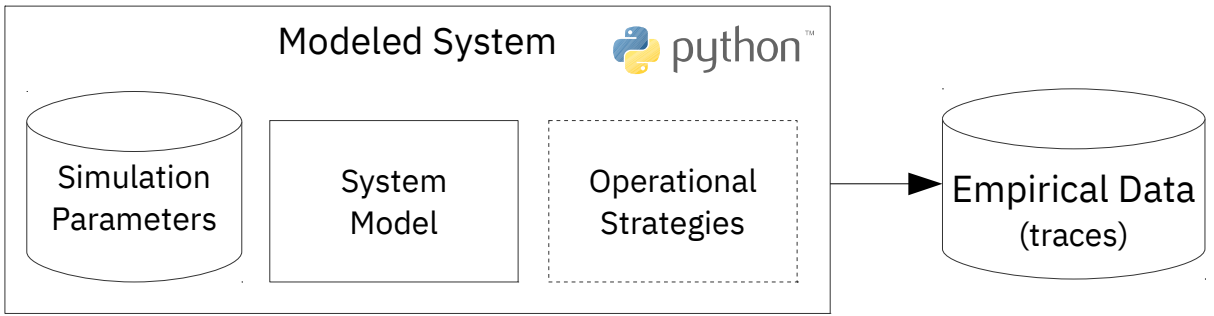
AI Ops Experimentation and Analytics Framework

AI Ops Experimentation & Analytics Framew.

- Develop and test operational mechanisms
- Use current understanding of platforms to build an AI Ops experimentation environment
  - Model a pipeline execution and AI operations platform
  - Generate synthetic pipelines and data
  - Simulate execution of pipelines
  - Study and analyze system behavior (answer "*what if*" questions)
- Requires good fidelity to be useful, i.e., exceed simple theoretical models, grounded using empirical data


emulates
Modeled System
python
Simulation Parameters
System Model
Operational Strategies
Empirical Data (traces)
Real System
OpenScale
ModelOps
Infrastructure
emulates
Operational Mechanisms
controls
Predictive Models
creates
Statistical Analysis
SciPy
scikit learn
Experiment Parameters
Simulator
SimPy
scikit learn
SciPy
influxdb
Exploratory Analysis
Grafana
revise
revise

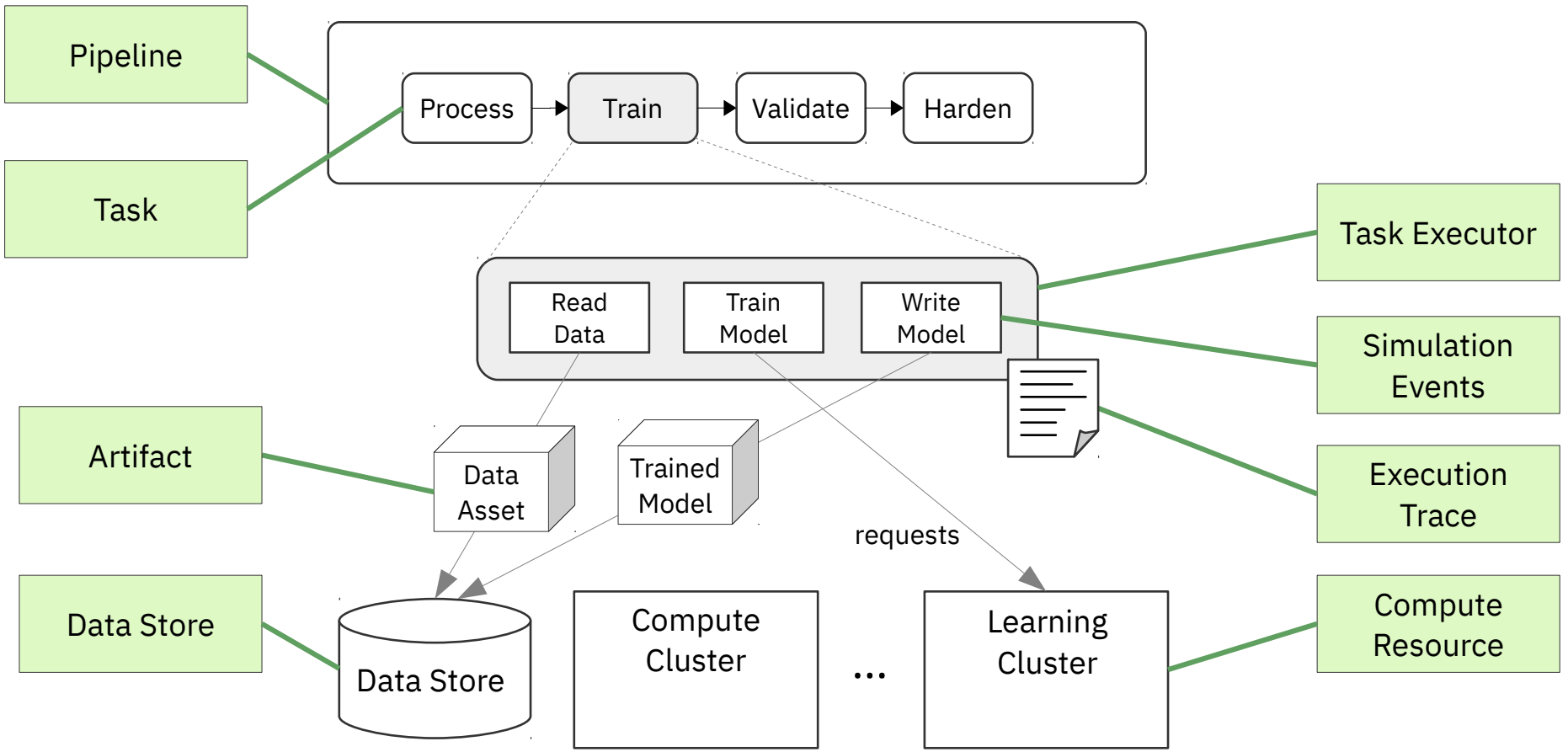
Developing the System Model



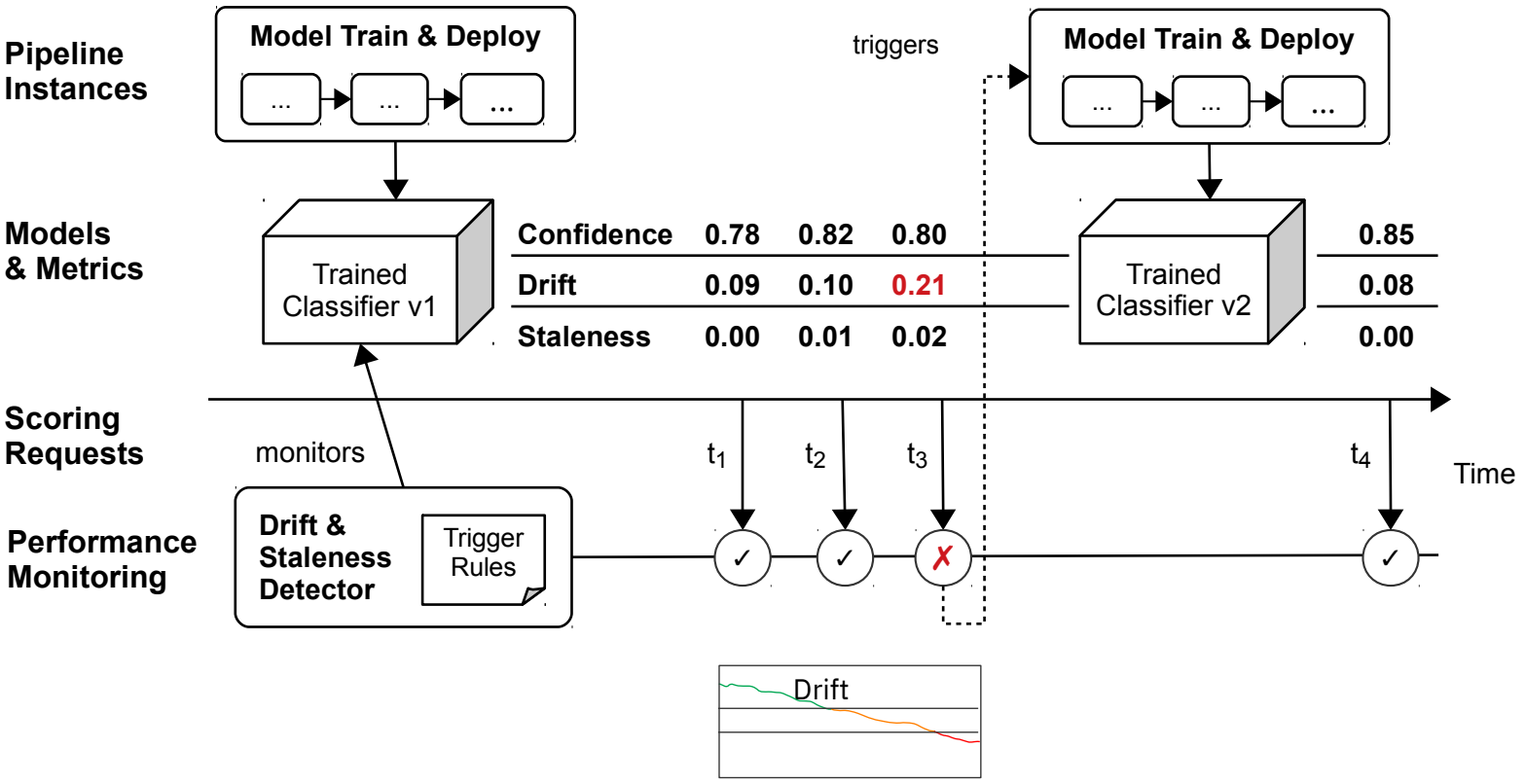
Key Ingredients

- System model for AI ops platforms
- Synthetic data and pipelines
- Process model

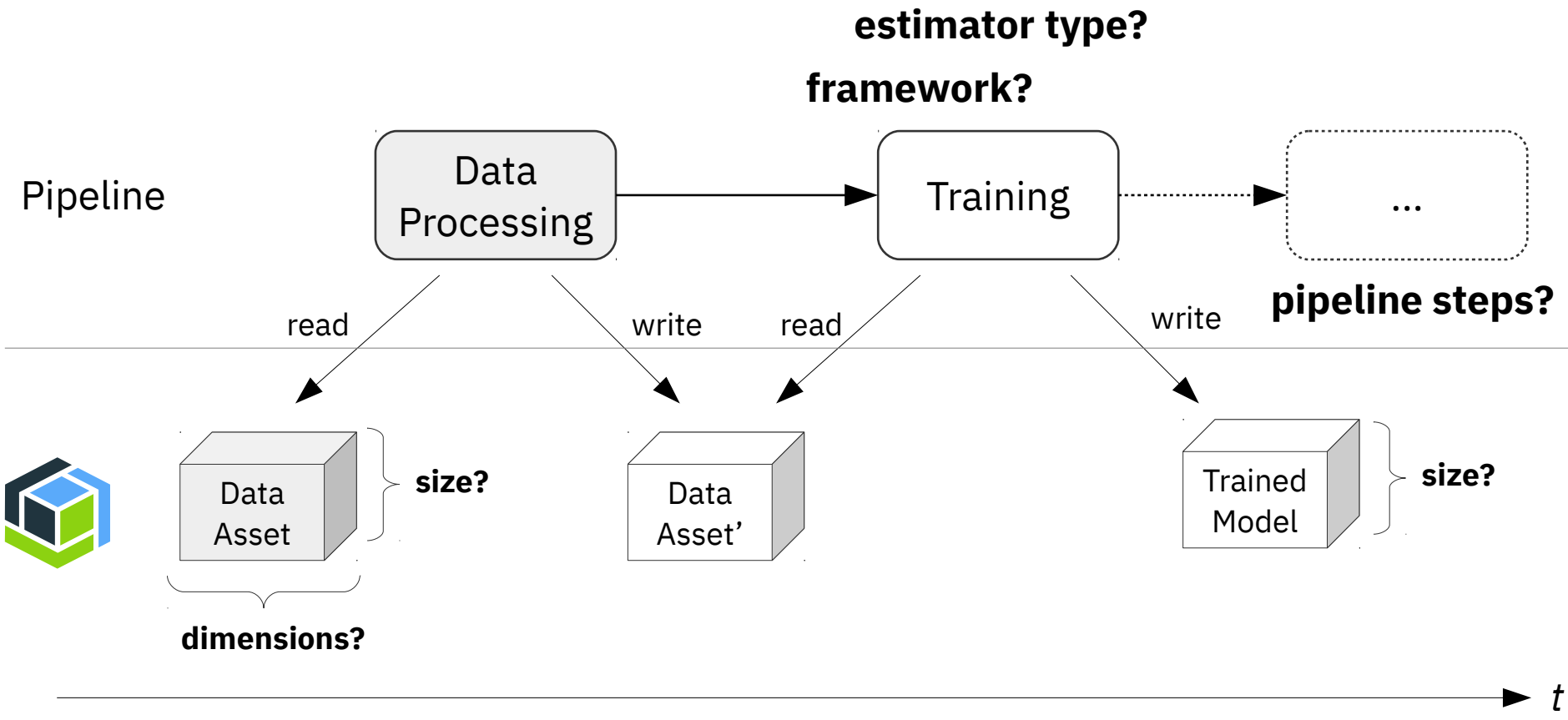
System model: build time



System model: run time

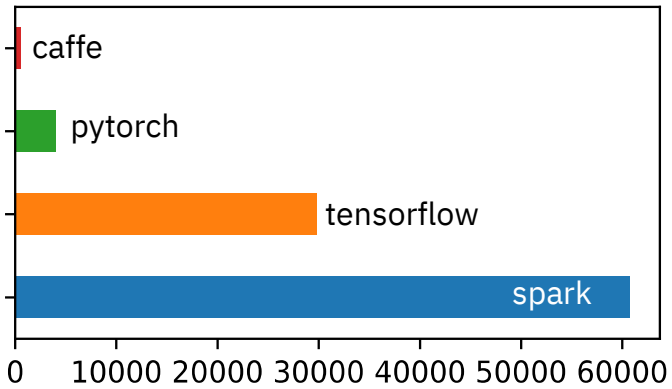


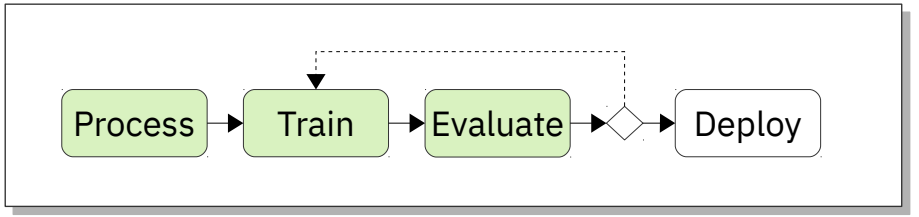
Synthetic data

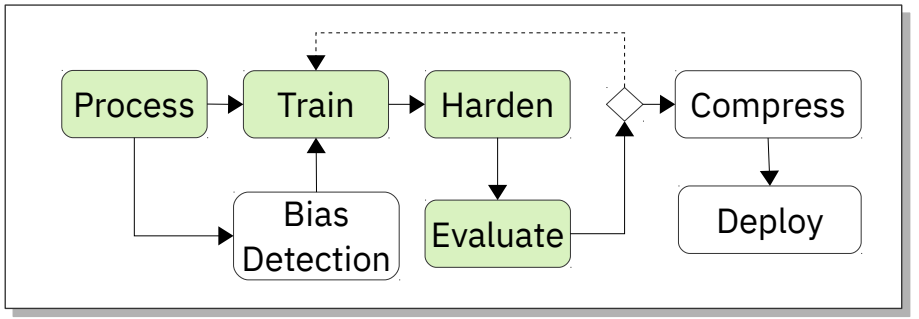


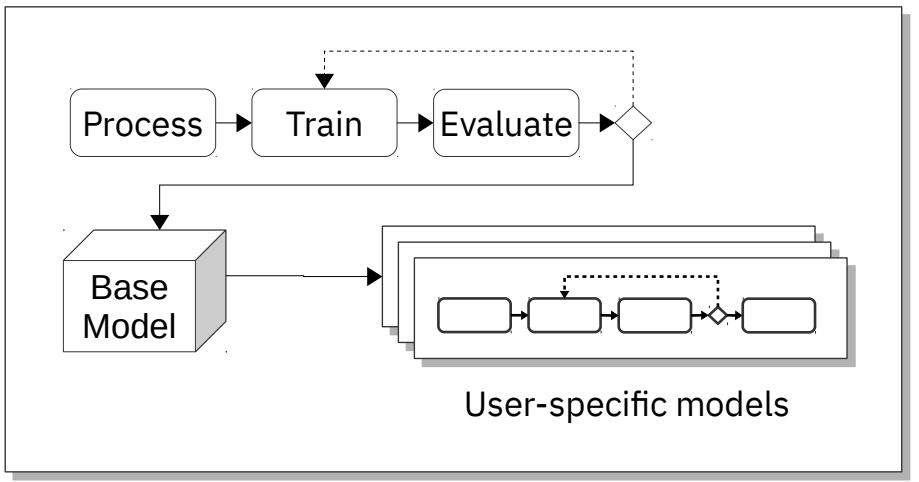
Synthetic data: pipelines

- Some pipelines we know about
  1. Simple (typical AI web platforms)
  2. Extended (custom workflows)
  3. Hierarchical (e.g., transfer learning or user-specific models)
- Frameworks used

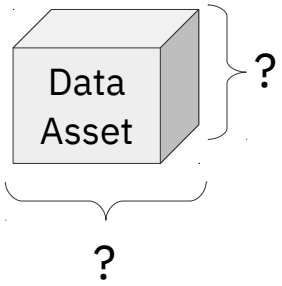


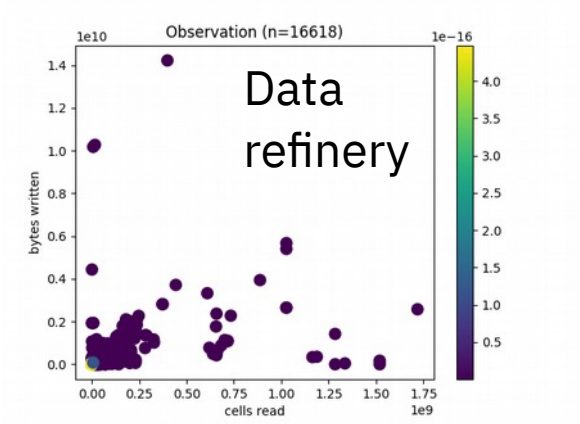






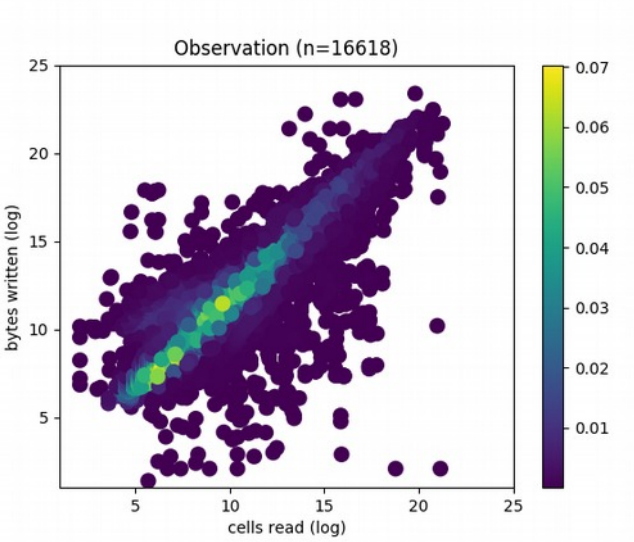
Synthetic data: assets



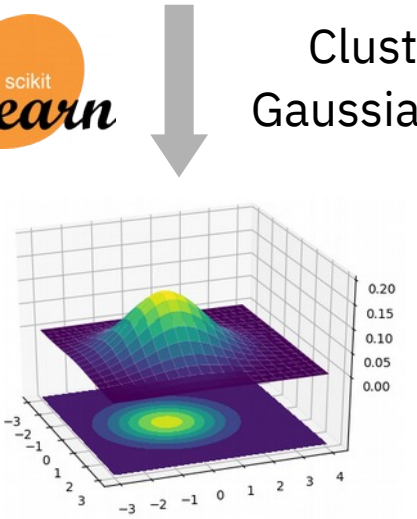


cells = columns * rows

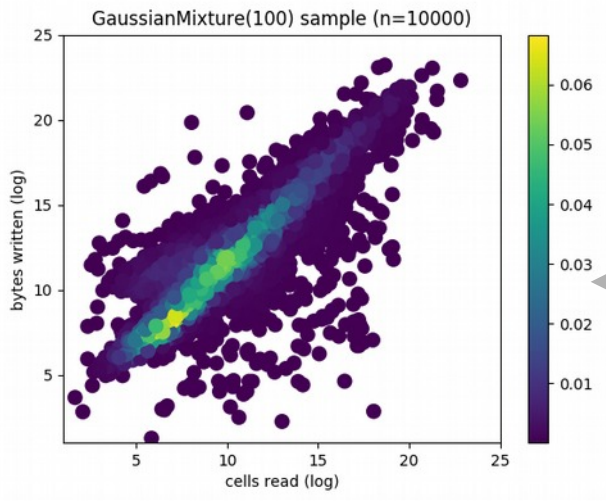
log_e trans-formation



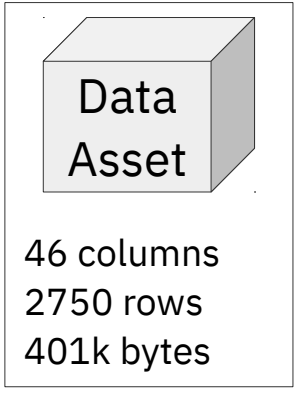
Cluster with Gaussian Mixture



Sample



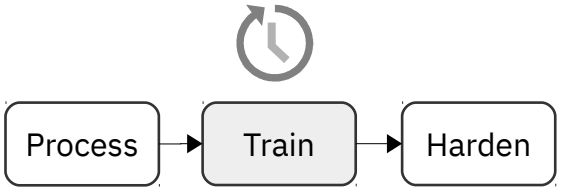
e^x trans-formation



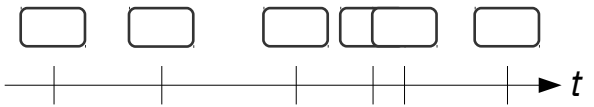
Example

Process model

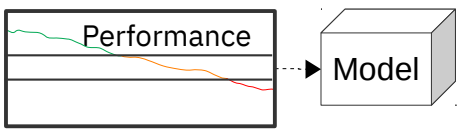
- How long do pipeline task execute?



- How frequently are pipelines executed?

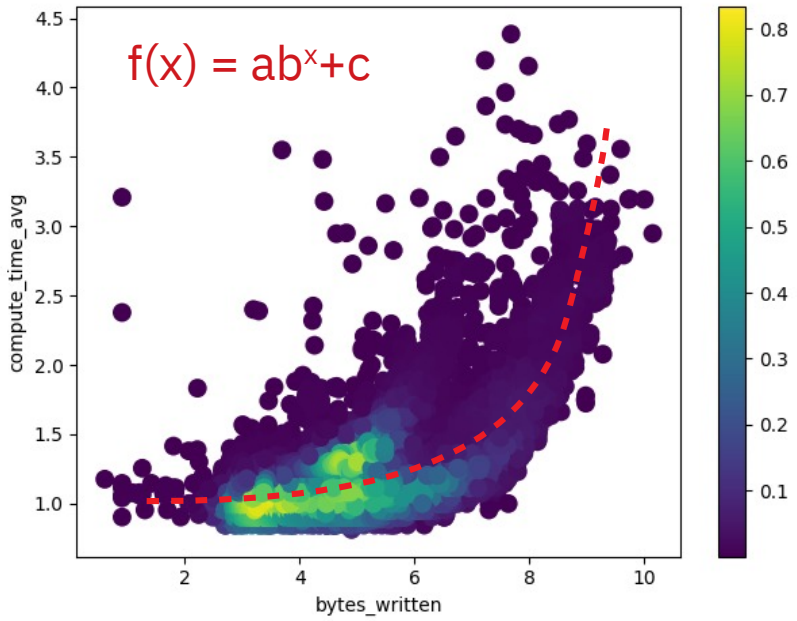


- How do models metrics change over time?

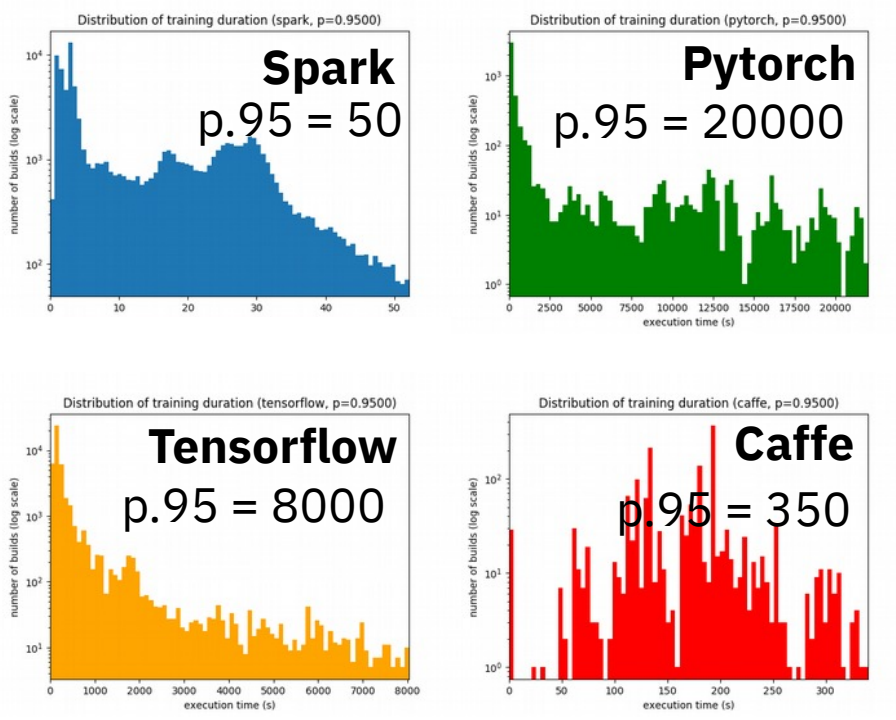


Process model: task computation time

Data Processing
(size of data asset vs processing time)

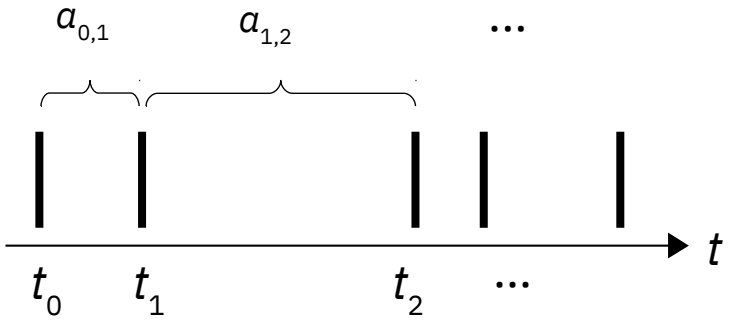


Training
(stratified per framework)

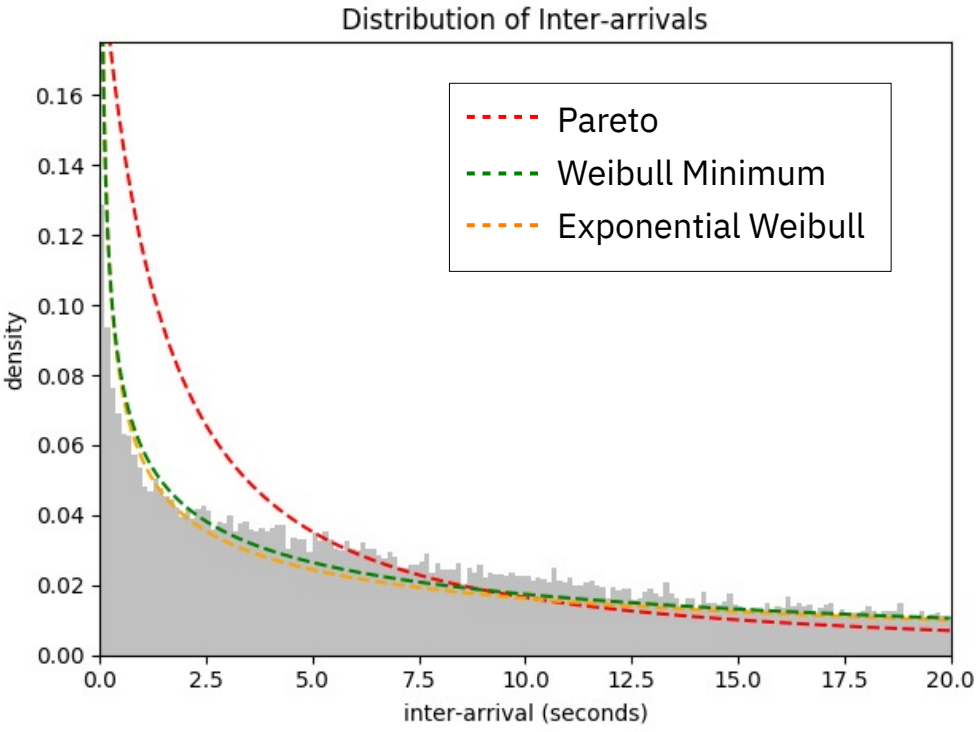


Process model: generating load

Pipeline **arrivals**
(sampled from training jobs)



We model the **interarrivals** *A*
as a random variable

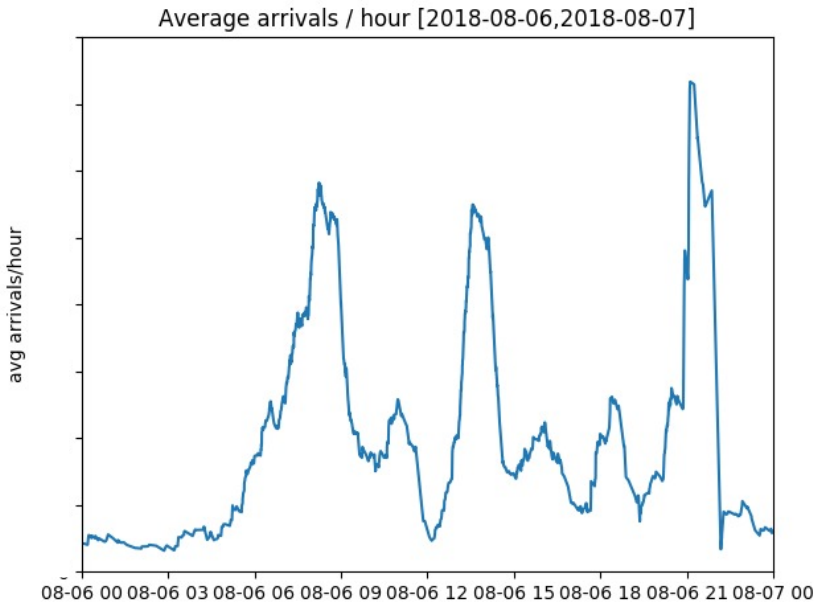


...

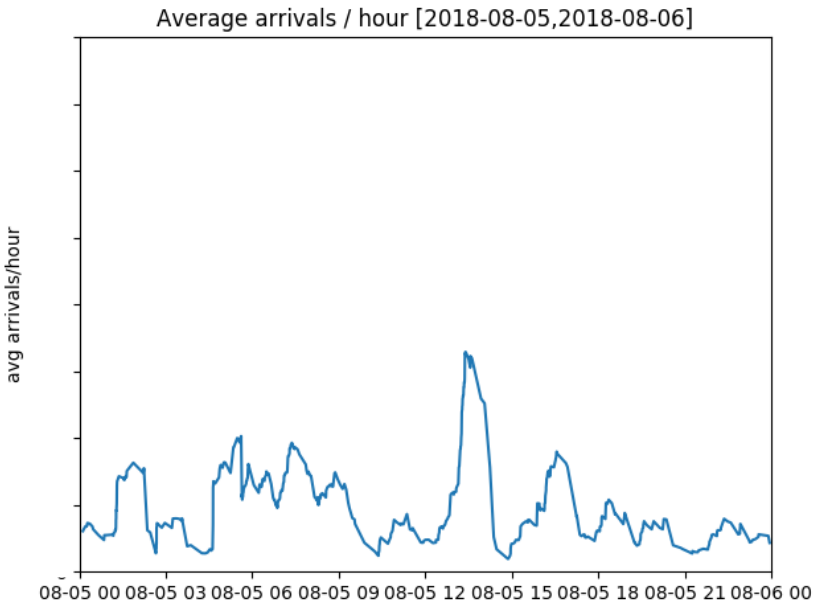
Long tail

Process model: arrival profiles

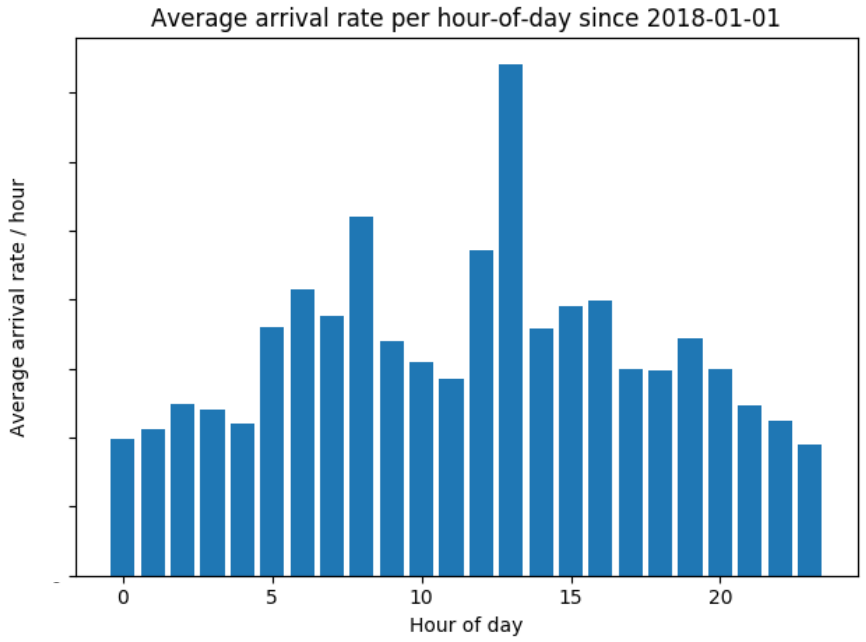
Just another manic **Monday** ...
(of training jobs)

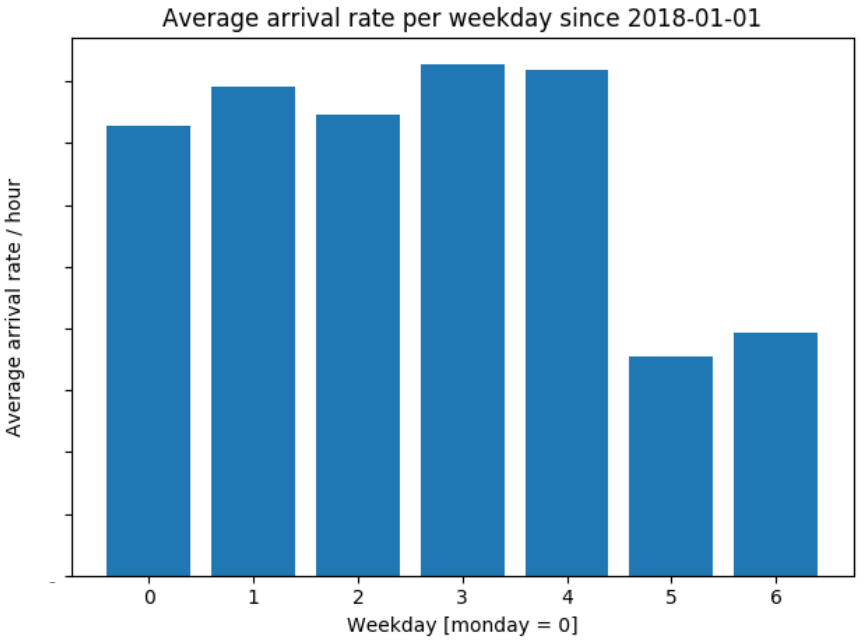


... wish it was **Sunday**.



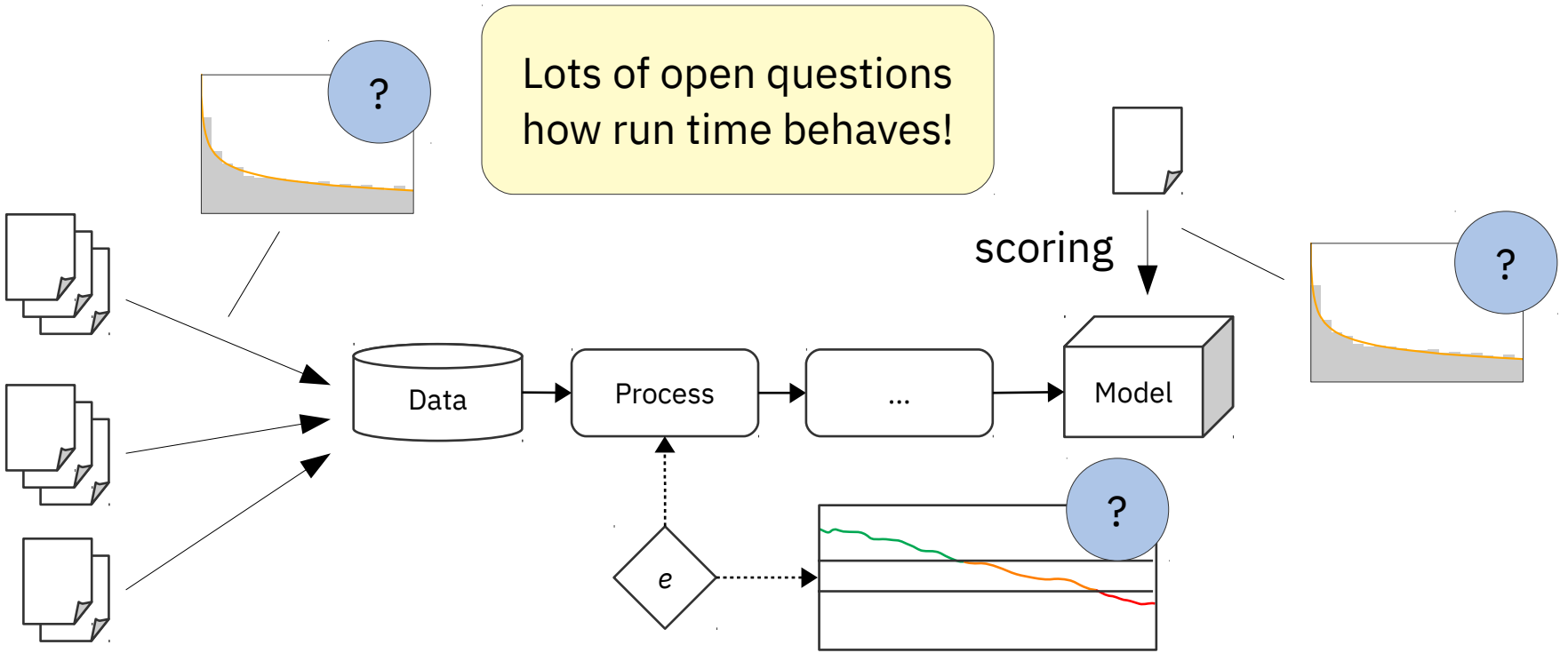
Process model: arrival profiles





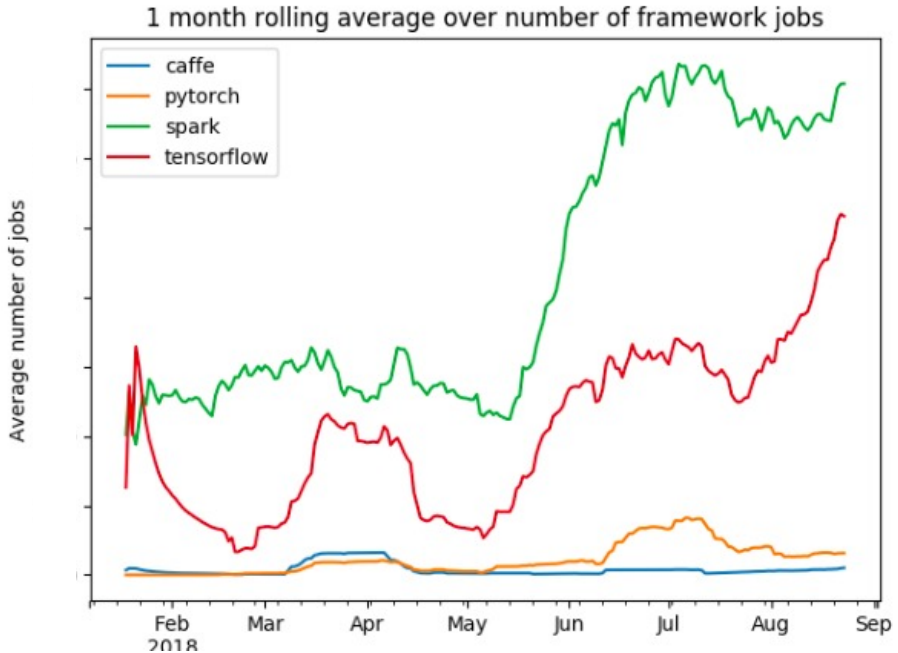
24 * 7 = 168 clusters

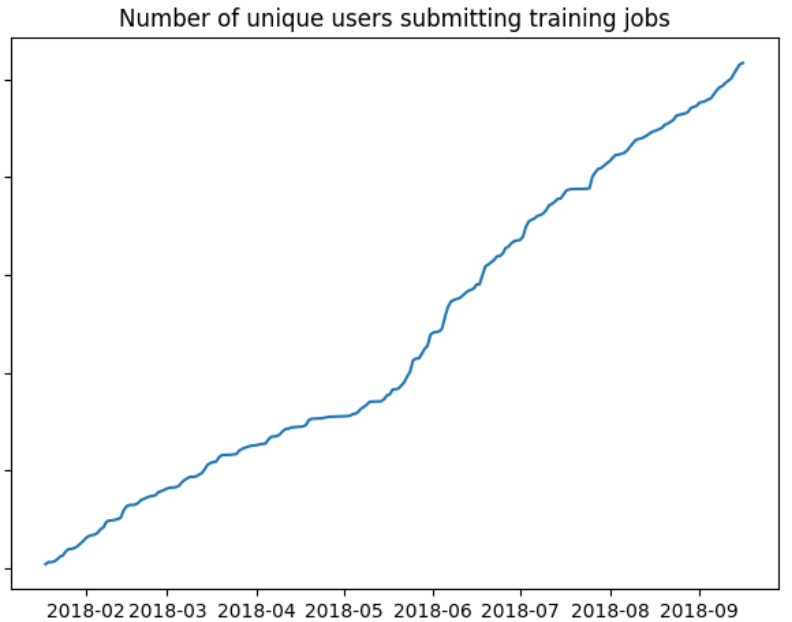
Process model: run time



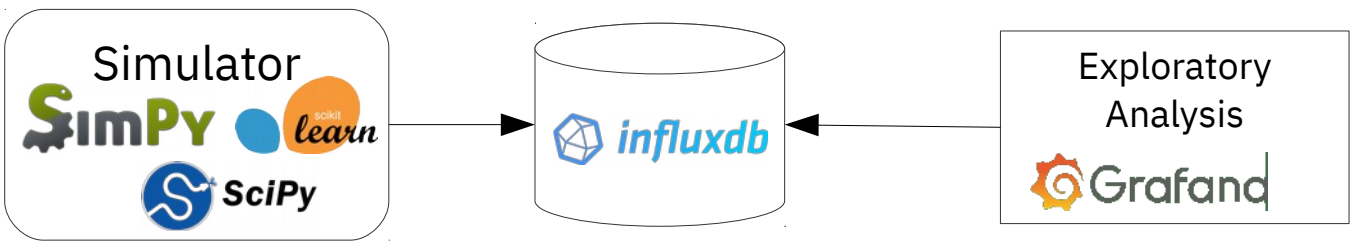
More open problems

- Conditional modeling between pipeline tasks
- Modeling of system dynamics and growth

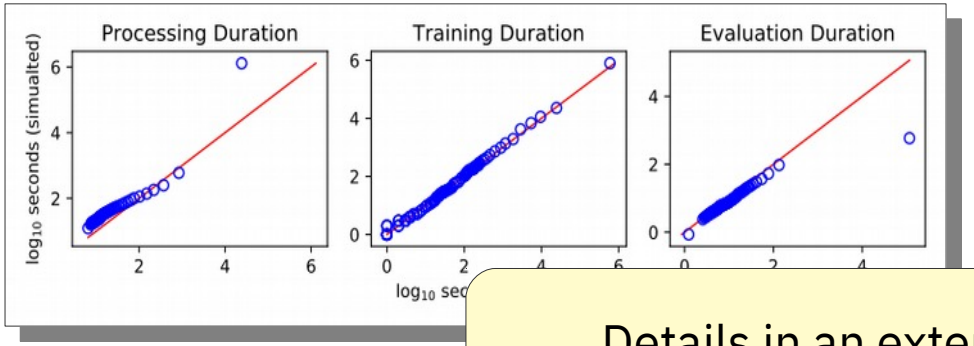


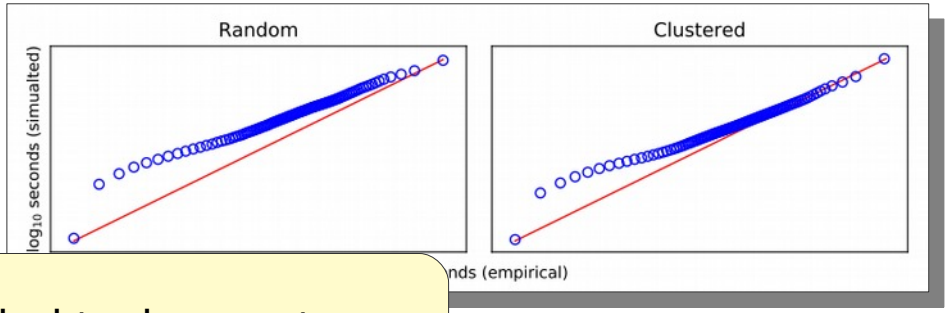


Demo


Simulator
SimPy
scikit learn
SciPy
influxdb
Exploratory Analysis
Grafana

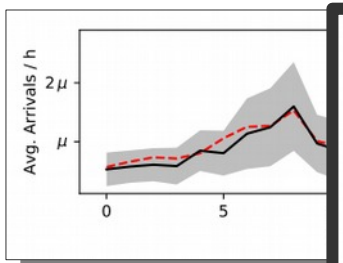
Simulation accuracy & performance

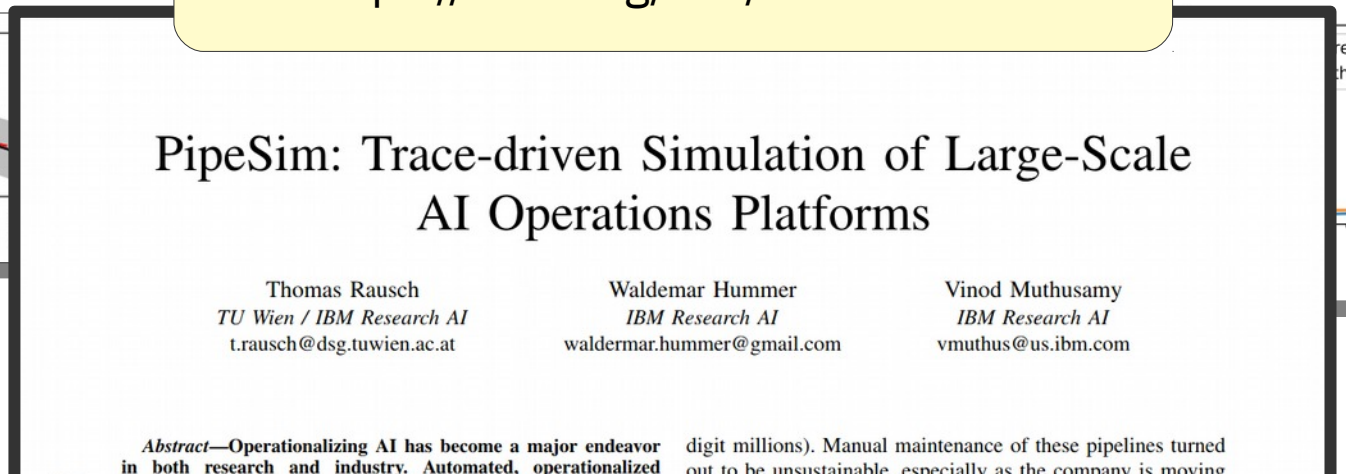




Details in an extended tech report

https://arxiv.org/abs/2006.12587





PipeSim: Trace-driven Simulation of Large-Scale AI Operations Platforms

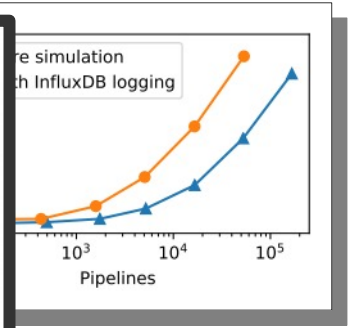
Thomas Rausch
TU Wien / IBM Research AI
t.rausch@dsg.tuwien.ac.at

Waldemar Hummer
IBM Research AI
waldermar.hummer@gmail.com

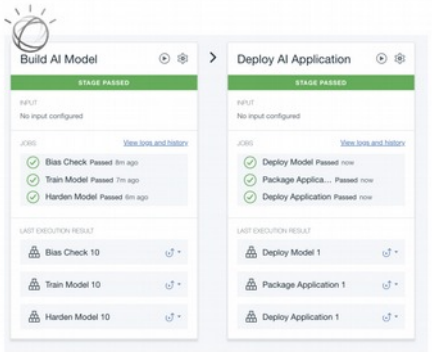
Vinod Muthusamy
IBM Research AI
vmuthus@us.ibm.com

***Abstract*—Operationalizing AI has become a major endeavor in both research and industry. Automated, operationalized**

digit millions). Manual maintenance of these pipelines turned

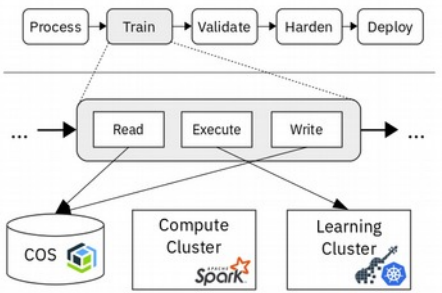


Operationalizing AI: Three Views



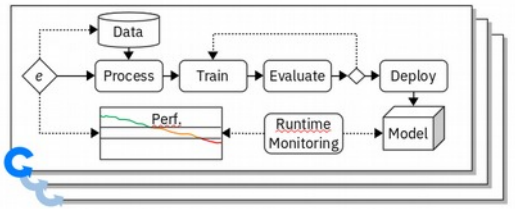
User View

Abstract **Pipeline** View



Platform View

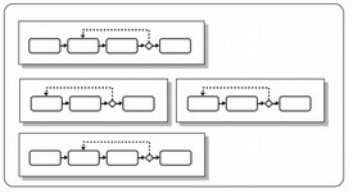
Operating Automated AI Pipelines



Retraining rule/trigger

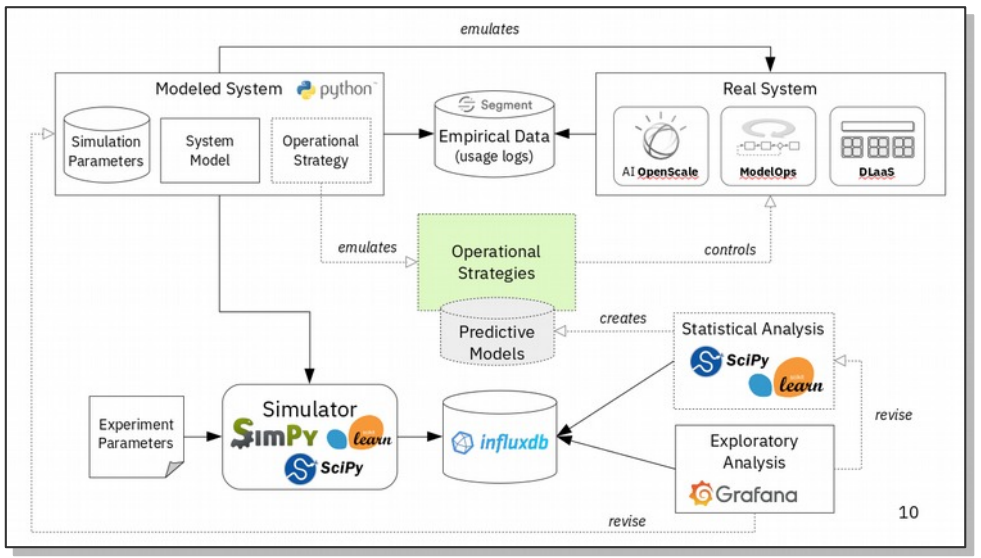
```
if performance < 0.8 and
   cnt(new_data) > n

every 4 weeks
```

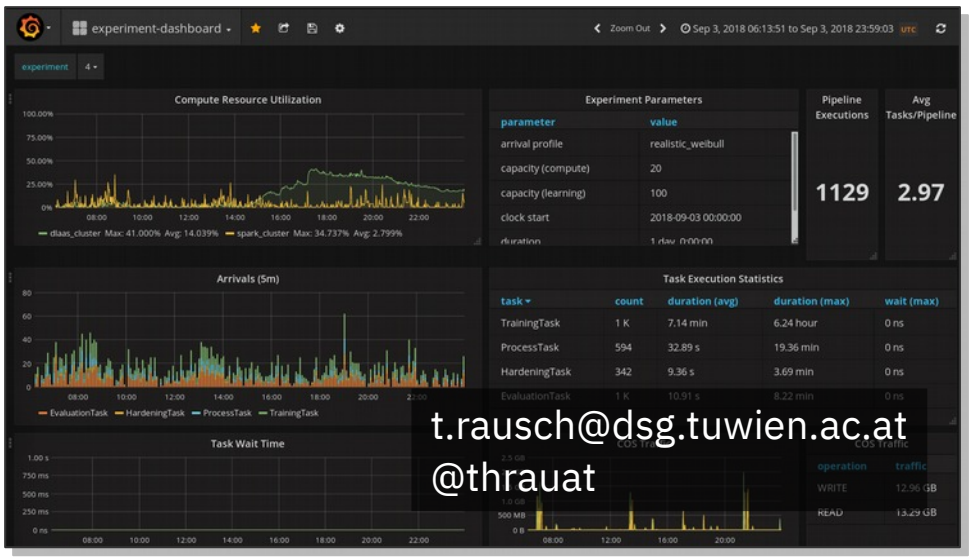
Infrastructure / Budget



Scheduler / Optimizer

- **Probabilistic parameters**
  - Expected improvement
  - Risk (e.g., human intervention)
  - Priority (how important is this model/tenant)?
- **Resource availability**





t.rausch@dsg.tuwien.ac.at
@thrauat