Machine Learning Model Management and Inference (MLMMI)
03 ML Platforms and Deployment

Dr. Arnab Phani
BIFOLD, TU Berlin
DEEM Lab

29.04.2026

PODS 2026 Keynote

Speaker: Jan Van den Bussche (Hasselt University)

Title: Models are data too: Towards expressive querying of machine-learning models

Abstract: Machine learning generates large amounts of models that should be managed by sound data management practices. In particular, we should be able to query these models using expressive, declarative query languages. In this talk we will survey some existing ideas, point at research challenges, and try to enthuse the research community into looking at a new kind of database queries. Since models are code, albeit learned, not written, we will also foray into the querying of program code.

Bio: Jan Van den Bussche is a professor of computer science in the Data Science Institute at Hasselt University (Belgium). He has been active in database theory research since 1989 and received his PhD in 1993 from the University of Antwerp. Together with his advisor Jan Paredaens he helped establish a "Belgian School" in database theory. He served as program chair for ICDT in 2001 and for PODS in 2006. He chaired the ICDT Council from 2001 until 2011, and chaired the PODS Executive Committee in 2017 and 2018. Broadly, his research interests revolve around the design and analysis of data models and query languages for a wide variety of data and computations.

Announcements

- **No Lecture on June 03**
  - I will be in India to attend SIGMOD
- **Programming Assignment 1 deadline : May 04**
  - MOSES registration before submission
- **Invited Talk: Nils Strassenburg**
  - May 13
  - Submit questions by May 11

Nils Straßenburg

Ph.D. Student
Room: F-1.04
Phone: +49-(0)331 5509-3429
E-Mail: Nils.Strassenburg(at)hpi.de

Full Profile

About Me

I am a PhD student in the Database Group at the Hasso Plattner Institute (HPI) in Potsdam, under the supervision of Tilmann Rabl.

My research focuses on ML systems, particularly ML model management and search. My work has been published in SIGMOD and EDBT. In addition to my research, I contribute to our lecture on big data systems, lead seminars on ML systems, and supervise master's theses.

Before starting my PhD, I earned a master's degree in IT-Systems Engineering from HPI and a bachelor's degree in Computer Science from the University of Hamburg. As part of my studies, I completed a six-month internship at SAP Labs France in Sophia Antipolis and spent a semester at ETH Zurich.

Nils's Papers:

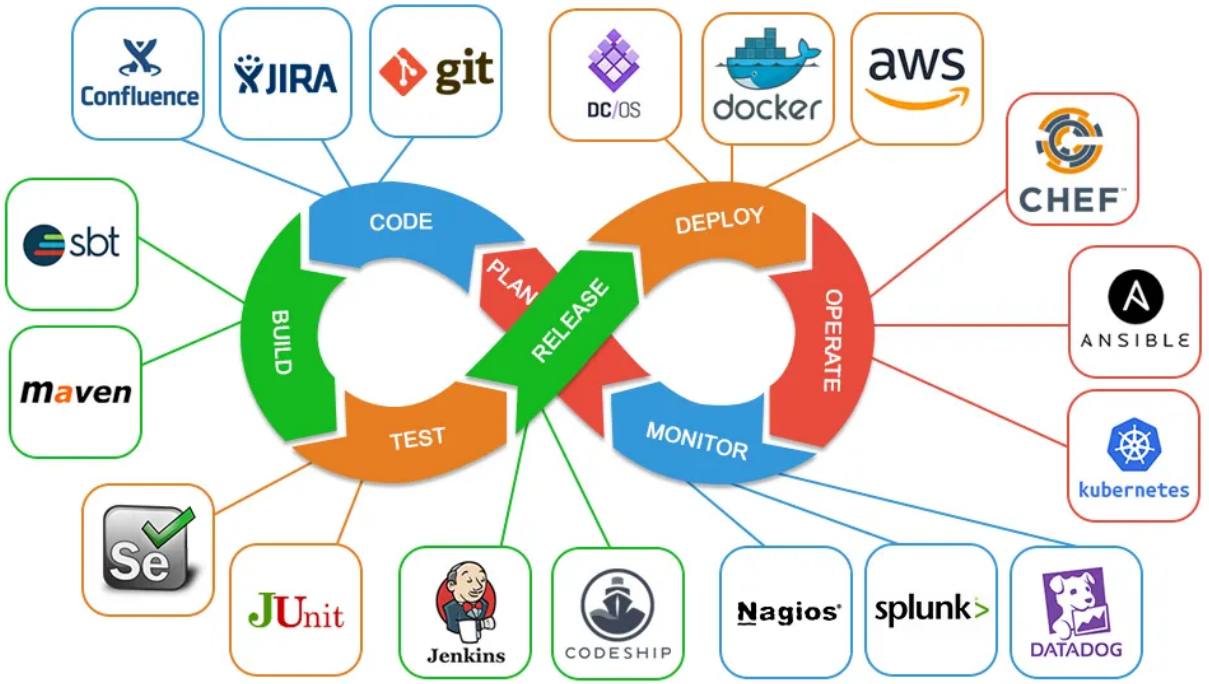
Efficiently Managing Deep Learning Models in a Distributed Environment. EDBT 2022.
Efficient Multi-Model Management. EDBT 2023.
Alsatian: Optimizing Model Search for Deep Transfer Learning. SIGMOD 2025.

Agenda

- **MLOps**
- **ML Platforms and Lifecycle Management**
- **Monitoring**
- **Data Validation**
- **LLMOps**

MLOps

Background: DevOps



- **DevOps**
  - Subarea of software engineering: software development + IT operations
  - Administering software in production; Cloud and agile
  - Fuses many historically separate job roles

CI / CD

Version Control

Building & Testing

Microservices

Infrastructure as code

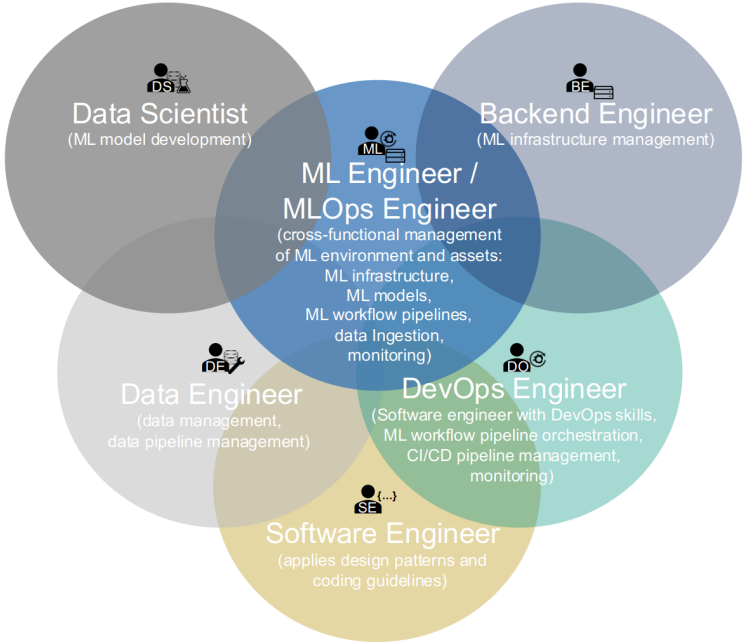
Monitoring & Logging

* https://medium.com/swlh/how-to-become-an-devops-engineer-in-2020-80b8740d5a52

MLOps Overview

- **MLOps = DevOps for ML-infused Software**

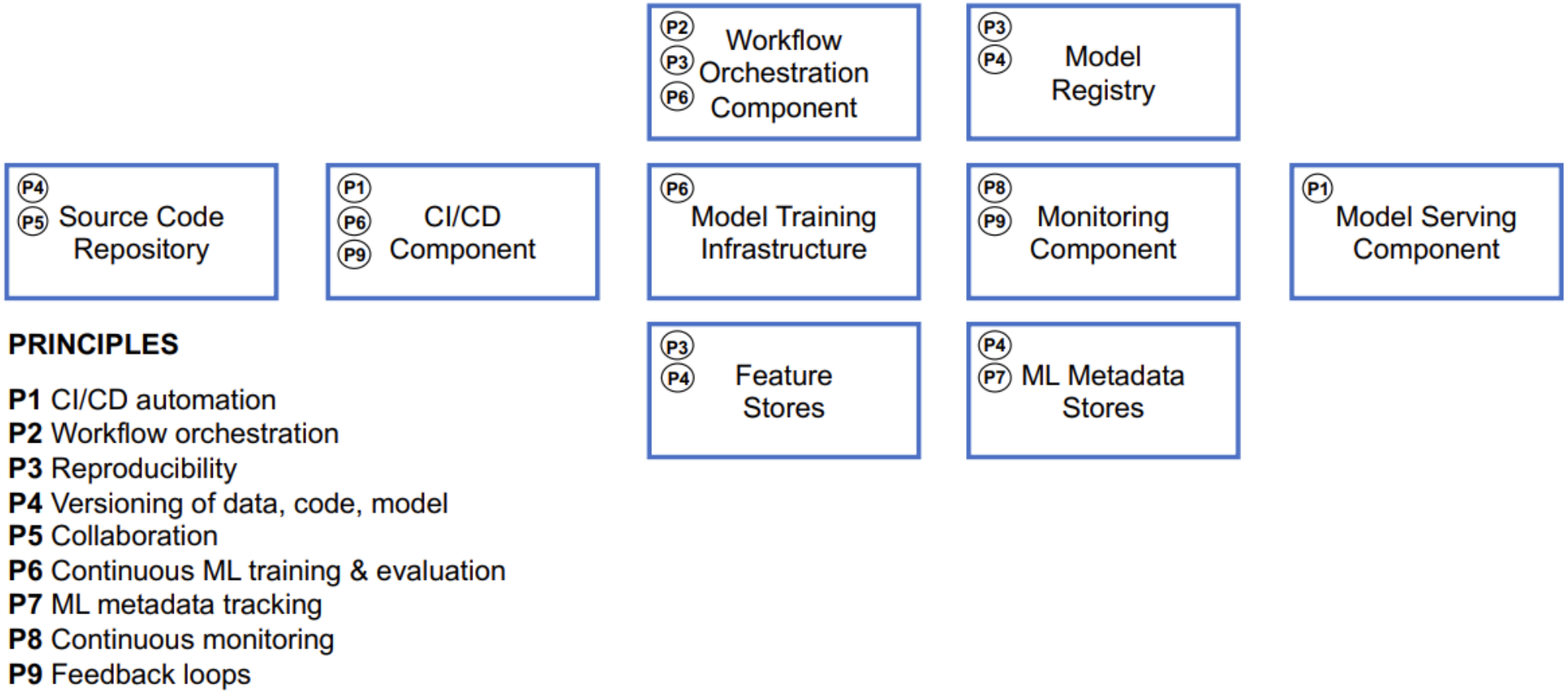
- **Beyond ML Model Codes**
  - Training and validation datasets
  - Data cleaning/prep/featurization codes/scripts
  - Hyperparameters and other configs
  - Ensembling
  - Hardware config



- **"3 Vs of MLOps"**
  - **Velocity**: Rapid experimentation, prototyping, and deployment with minimal friction
  - **Validation**: Checks on quality and integrity of data features, models, predictions
  - **Versioning**: Track deployed models and features to ensure provenance and fallback options

* Operationalizing Machine Learning: An Interview Study. arXiv. 2022.

Birds-eye View of MLOps

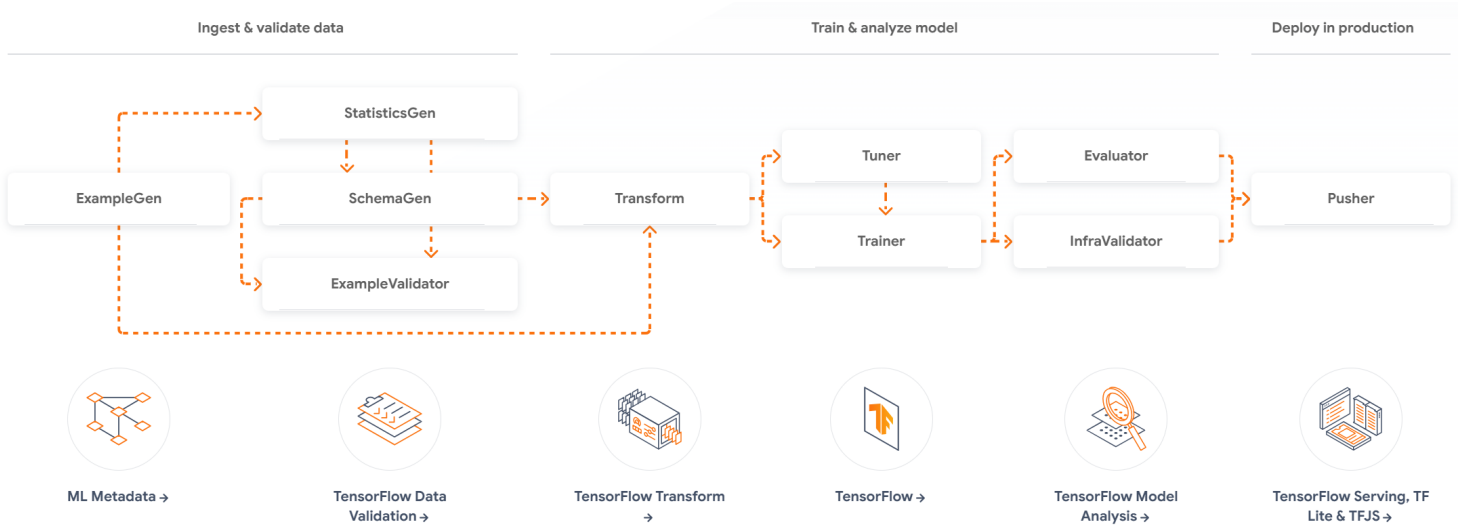


MLOps Principles and Associated Components

* Machine Learning Operations (MLOps): Overview, Definition, and Architecture. arXiv.

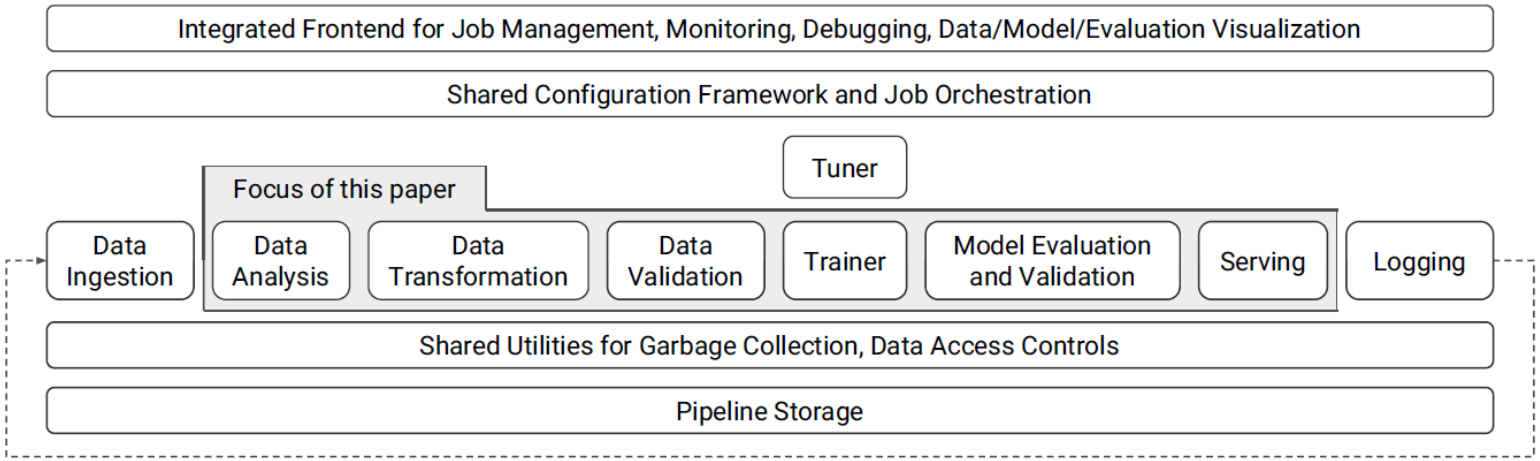
ML Platforms and Lifecycle Management

Production ML Challenges



- **Motivation**
  - **Unified platform** for diverse ML architectures (DNN, linear, tree-based, ..)
  - Continuous training (e.g., a moving window over the last `n` days) and serving
  - Human-in-the-loop; expose simple UI for deployment and monitoring; support various levels of expertise
  - Reliability and scalability; resilient from inconsistent data, software, config, and failures; scale gracefully to data volume and serving traffic

TFX: A TensorFlow-Based End-to-End ML Platform



- **TFX**
  - Full flexibility for implementing any type of (TensorFlow-based) models
  - Support for **continuous training**, warm-starting (transfer learning)
  - Model and data validation; slicing; monitoring
  - Automatic feature statistics generation including cross-feature statistics
  - Ensuring consistency of feature transformation during training and serving
  - **High-level model specification API** to hide implementation details

```
# Declare a numeric feature:
num_rooms = numeric_column('number-of-rooms')
# Declare a categorical feature:
country = categorical_column_with_vocabulary_list(
    'country', ['US', 'CA'])
# Declare a categorical feature and use hashing:
zip_code = categorical_column_with_hash_bucket(
    'zip_code', hash_bucket_size=1K)
# Define the model and declare the inputs
estimator = DNNRegressor(
    hidden_units=[256, 128, 64],
    feature_columns=[
        num_rooms, country,
        embedding_column(zip_code, 8)],
    activation_fn=relu,
    dropout=0.1)
# Prepare the training data
def my_training_data():
  # Read, parse training data and convert it into
  # tensors. Returns a mini-batch of data every
  # time returned tensors are fetched.
  return features, labels
# Prepare the validation data
def my_eval_data():
  # Read, parse validation data and convert it into
  # tensors. Returns a mini-batch of data every
  # time returned tensors are fetched.
  return features, labels
estimator.train(input_fn=my_training_data)
estimator.evaluate(input_fn=my_eval_data)
```

* TFX: A TensorFlow-Based Production-Scale Machine Learning Platform. KDD 2017.

Ease.ML: A Lifecycle Management System

- **Ease.ML**
  - **Automated toolchains** and well-defined processes that stitch together **existing**/upcoming tools to improve end-to-end development experience. Target users: domain experts w/ basic ML skills
  - Data ingestion templates (pdf to tables)
  - Estimate best possible accuracy from input datasets
  - Downstream task-aware automatic data cleaning
  - Automatic data discovery
  - Relies on AutoML for training models
  - CI of new and better models
  - Quality debugging and recommend improvements

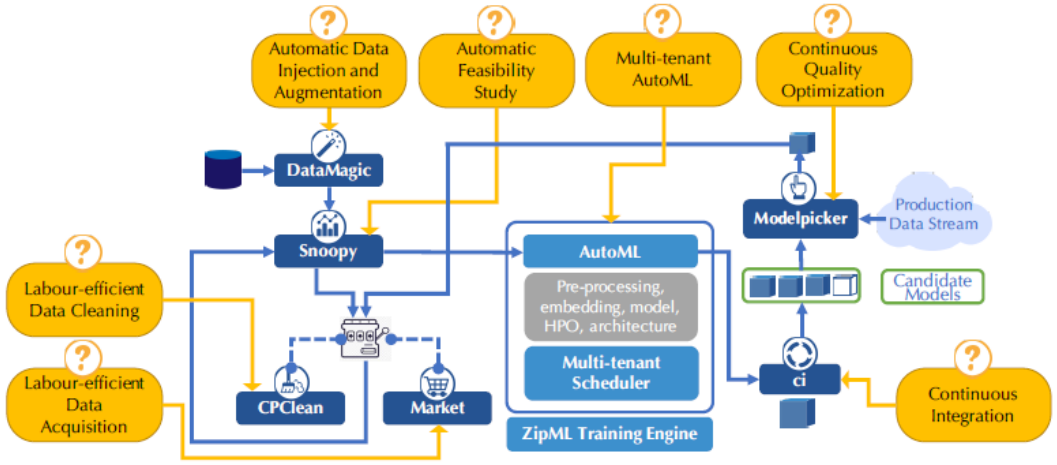


Figure 1: Ease.ML Process

- **System Design**
  - Probabilistic data model
  - Multi-armed bandits

* Ease.ML: A Lifecycle Management System for MLDev and MLOps. CIDR 2021.
* https://ease.ml/

MLflow

```
# Log parameters, which are arbitrary key-value pairs
mlflow.log_param("num_dimensions", 8)
mlflow.log_param("regularization", 0.1)

# Log metrics; each metric can also be updated throughout the run
mlflow.log_metric("accuracy", 0.8)
mlflow.log_metric("r2", 0.4)

# Log artifacts (arbitrary output files)
mlflow.log_artifact("precision_recall.png")
```

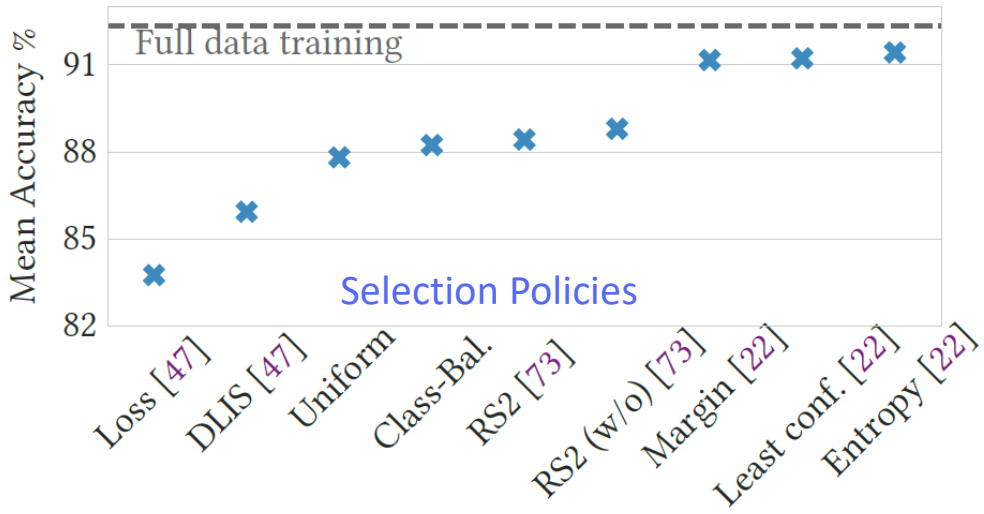
- **MLflow Tracking**
  - API for recording experiment runs including code, parameters, data, and metrics
  - These runs can be queried through an API/UI
- **MLflow Projects**
  - Format for packaging code into reusable projects (code, environment, parameters)
- **MLflow Models**
  - Generic format for packaging models that can work with diverse deployment tools

* Accelerating the Machine Learning Lifecycle with MLflow. IEEE Data Eng. Bull. 41(4). 2018

Monitoring

Post-Deployment Model Retraining

- **Continuous Training**
  - Retraining due to distribution drift
  - Retraining every day on petabytes of data
  - Finding the **optimal frequency** of retraining
- **Data Selection Policies**
  - Training on a selected sample w/ comparable accuracy
  - Finding the **right data selection** policy is difficult
  - Random access degrades throughput
- **Triggering Policies**
  - Domain-specific triggering policies
- **ML Platforms**
  - Do not support or require manual plumbing



MODYN

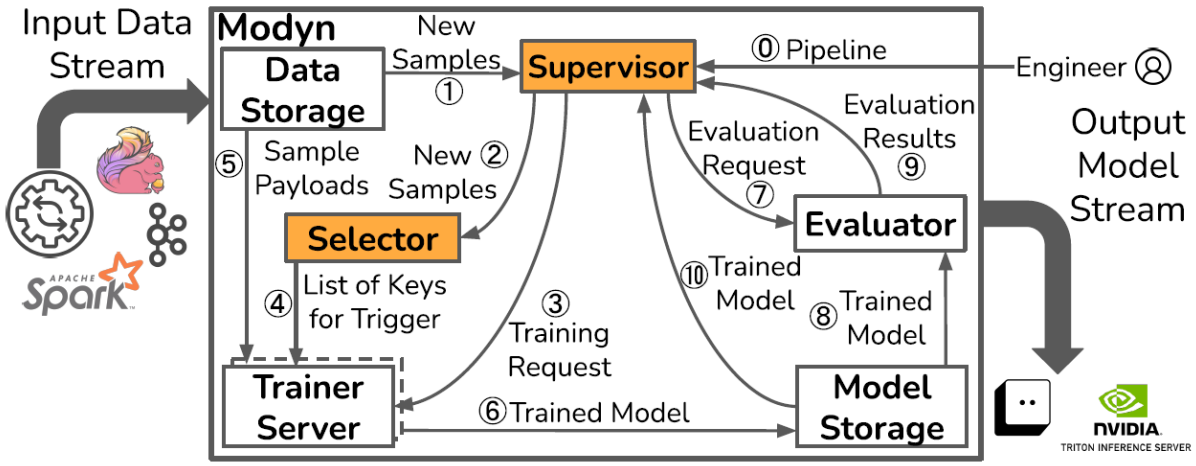


Fig. 3. MODYN's system design.

modyn

```yaml
model:
  id: ResNet18
  config:
    num_classes: 42
data:
  dataset_id: mnist
  transformations: ["transforms.Normalize(...)"]
  bytes_parser_function: |
    def bytes_parser_function(data: memoryview) -> Image:
      return Image.open(io.BytesIO(data)).convert("RGB")
trigger:
  id: DataAmountTrigger
  num_samples: 100
training:
  use_previous_model: True
  batch_size: 1234
  optimizers: ...
  optimization_criterion:
    name: "CrossEntropyLoss"
  selection_strategy:
    name: "CoresetStrategy"
    storage_backend: "database"
    tail_triggers: 0
    presampling_config: ...
    downsampling_config: ...
model_storage:
  full_model_strategy:
    name: "PyTorchFullModel"
  incremental_model_strategy:
    name: "WeightsDifference"
evaluation: ...
```

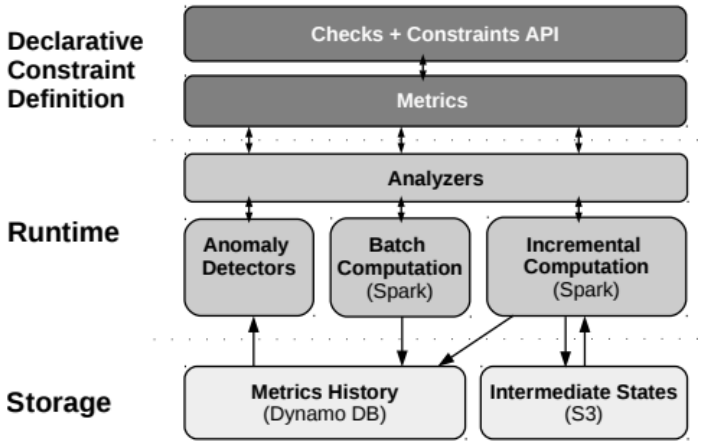
- **Orchestrator for Data Selection and Retraining**
  - Users declaratively specify triggering and selection policies
  - Continuous evaluation and drift detection
  - Sampling-aware data partitioning to reduce data access overhead

* Modyn: Data-Centric Machine Learning Pipeline Orchestration. SIGMOD 2025.

Data Validation

Automating Large-Scale Data Quality Verification

constraint	arguments
dimension *completeness*	
`isComplete`	column
`hasCompleteness`	column, udf
dimension *consistency*	
`isUnique`	column
`hasUniqueness`	column, udf
`hasDistinctness`	column, udf
`isInRange`	column, value range
`hasConsistentType`	column
`isNonNegative`	column
`isLessThan`	column pair
`satisfies`	predicate
`satisfiesIf`	predicate pair
`hasPredictability`	column, column(s), udf



AWS Glue

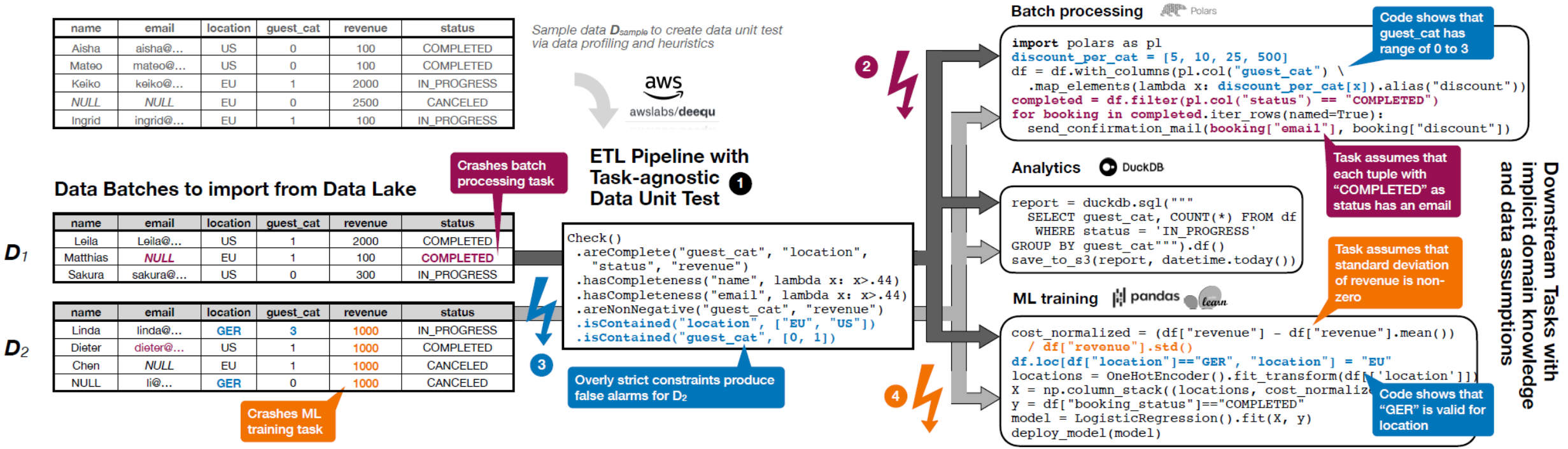
great expectations

- **Amazon's Deequ**
  - API combining quality constraints with user-defined validation code, which enables data unit tests
  - Efficient execution of the constraints

```
checks += Check(Level.Error)
 .isComplete("customerId", "title",
  "impressionStart", "impressionEnd",
  "deviceType", "priority")
 .isUnique("customerId", "countryResidence",
  "deviceType", "title")
 .hasCountDistinct("title", _ <= numTitles)
 .hasHistogramValues("deviceType",
   _.ratio("phone") <= maxExpectedPhoneRatio)
```

* Automating Large-Scale Data Quality Verification. PVLDB 2018.

Task-aware Data Unit Test Generation



name	email	location	guest_cat	revenue	status
Aisha	aisha@...	US	0	100	COMPLETED
Mateo	mateo@...	US	0	100	COMPLETED
Keiko	keiko@...	EU	1	2000	IN_PROGRESS
NULL	*NULL*	EU	0	2500	CANCELED
Ingrid	ingrid@...	EU	1	100	IN_PROGRESS

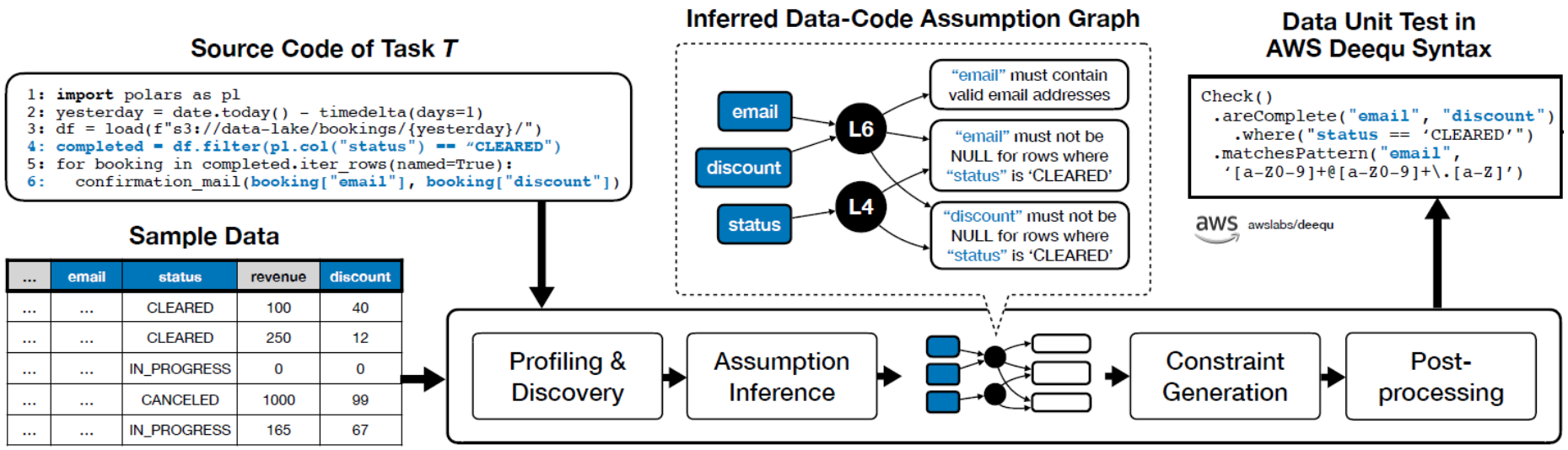
name	email	location	guest_cat	revenue	status
Leila	Leila@...	US	1	2000	COMPLETED
Matthias	*NULL*	EU	1	100	COMPLETED
Sakura	sakura@...	US	0	300	IN_PROGRESS

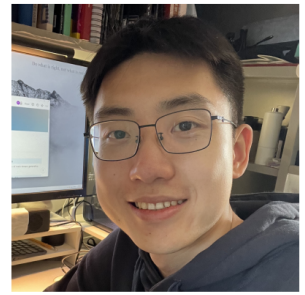
name	email	location	guest_cat	revenue	status
Linda	linda@...	GER	3	1000	IN_PROGRESS
Dieter	dieter@...	US	1	1000	COMPLETED
Chen	*NULL*	EU	1	1000	CANCELED
NULL	li@...	GER	0	1000	CANCELED

Data unit tests must reflect the **assumptions** of **each** downstream task

* PrismaDV: Automated Task-Aware Data Unit Test Generation. arXiv. 2026.

PrismaDV: Task-aware Data Unit Test Generation



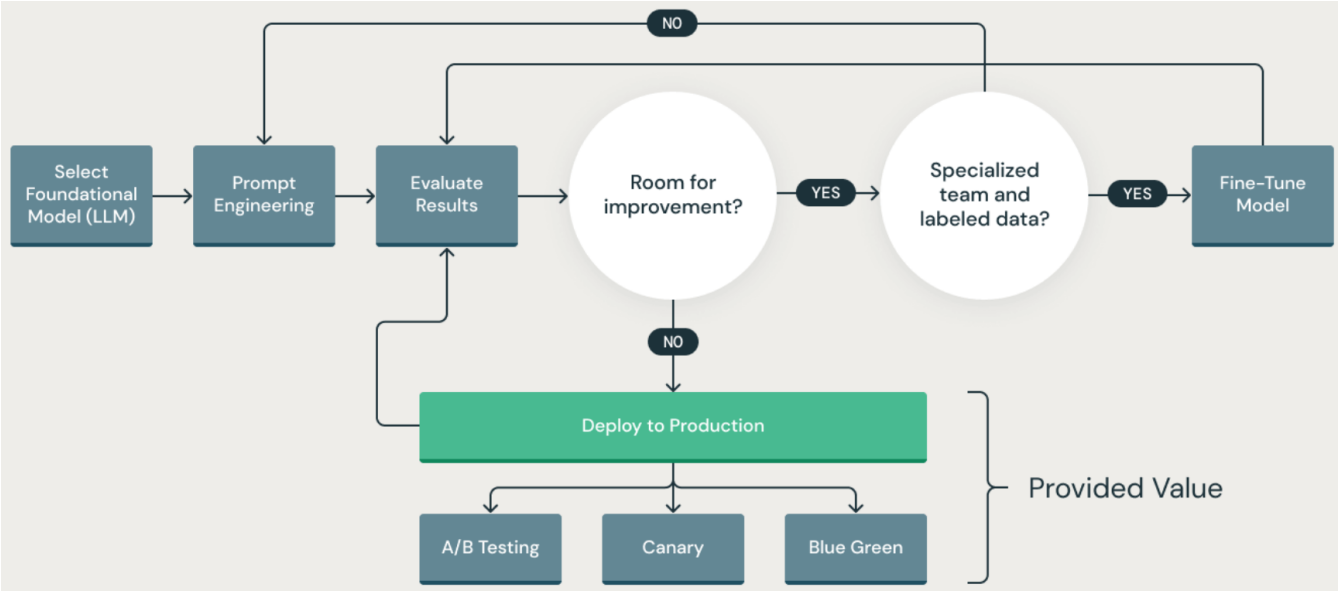


Compound AI System

- **PrismaDV**
  - Jointly analyzes downstream task code and data
  - Builds a "**data-code assumption graph**" linking data columns to code-level assumptions
  - Synthesizes executable data unit tests for deployment
  - Compound AI System

LLMOps

LLMOps



- **Why LLMOps?**
  - Training and fine-tuning LLMs require much more computational resources than traditional ML
  - Complex performance metrics (e.g., BLEU, ROUGE)
  - Prompt engineering and chains of LLM calls (compound AI system)
  - Data Ingestion: chunking of text, embedding, RAG
  - Fine-tuning and transfer learning

* https://www.databricks.com/blog/what-is-llmops

Summary

- MLOps principles and components
- Production ML challenges and end-to-end ML platforms
- ML lifecycle management for domain experts
- Post-deployment continuous training
- LLMOps

Please read the recommended papers