Operationalizing Machine Learning: An Interview Study

Shreya Shankar*, Rolando Garcia*, Joseph M. Hellerstein, Aditya G. Parameswaran

{shreyashankar,rogarcia,hellerstein,adityagp}@berkeley.edu

EPIC DATA lab
UC Berkeley

What do ML Engineers Actually Do?

We conducted an interview study of 18 ML engineers across organizations of different sizes and sectors. We found **four** high-level tasks in their workflows.

"I look for features from data scientists, [who have ideas of] things that are correlated with what I'm trying to predict."

Data Collection
Cleaning, preprocessing, wrangling

Experimentation
Model training, adding or removing features

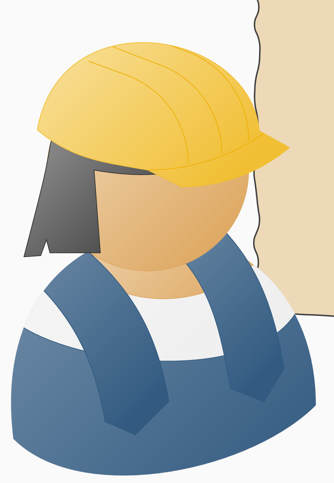
Evaluation and Deployment
Hold-out dataset, good accuracy, dynamic validation sets, A/B testing

Monitoring and Response
Rule-based guardrails, on-call rotations

"You have this classic issue where most researchers are evaluat[ing] against fixed data sets...[but] most industry methods change their datasets."

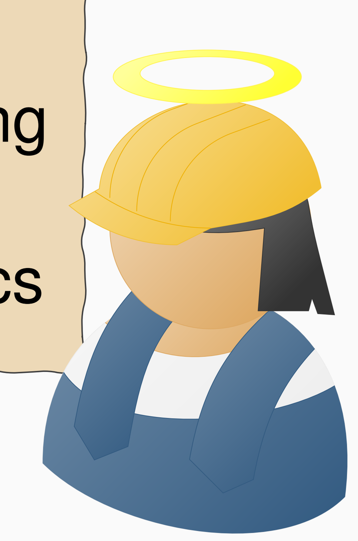
"Monitoring is both metrics and then a predicate over those metrics that triggers alerts. That second piece doesn't exist—not because the infrastructure is hard, but because no one knows how to set those predicate values."

Evaluation: an Active Organizational Effort

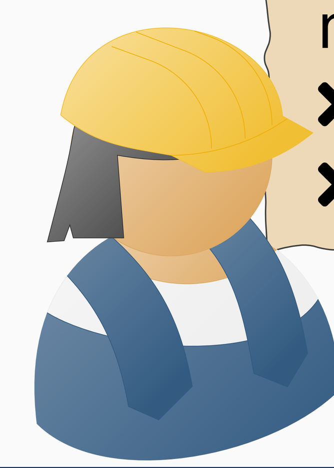


Myths
- ✗ Static hold-out set
- ✗ A single global metric suffices
- ✗ ML metrics only

Tips
- ✓ Dynamic datasets
- ✓ Check underperforming populations
- ✓ ML *and* product metrics

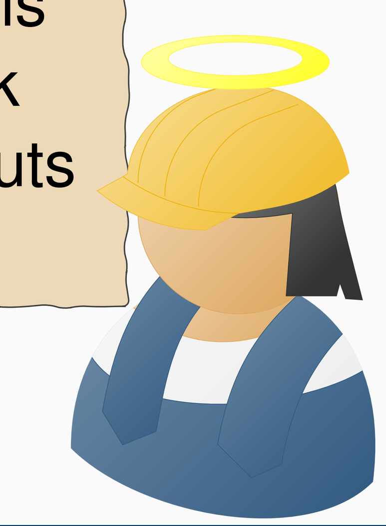


Non-ML Rules for Reliable ML Deployments



Myths
- ✗ Build models "robust" to data distribution shift
- ✗ Always serve the latest model's predictions
- ✗ Validate only the outputs
- ✗ Set and forget

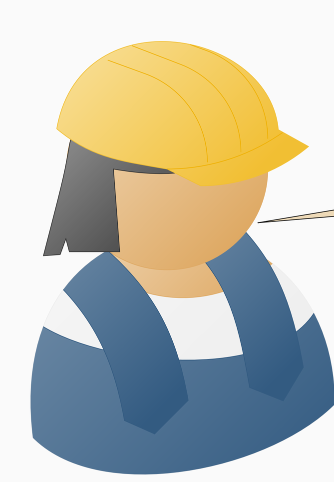
Tips
- ✓ Frequently retrain models on latest data
- ✓ Maintain simple models and heuristics for rollback
- ✓ Validate inputs & outputs
- ✓ On-call rotations



Challenges and Future Work

ML engineers identified several pain points in their workflows, which we plan to automate while building future systems and frameworks:

- Mismatch between development and production environments: how do we minimize errors in the process of promoting to production?
- Alert fatigue in data validation: how do we precisely determine when to reject a prediction because of poor data quality?
- No two ML bugs are the same: how do we tame the long tail of ML bugs?



ML [bugs] don't get caught by tests or production systems and just silently cause errors [that man- ifest as] slight re- ductions in performance. This is why **[you] need to be paranoid when you're writing ML code.**