Practical ML Monitoring

O'Reilly Online Training

Shreya Shankar
University of California, Berkeley 🐻

Introduction

About me 👩🏽‍💻

- Grew up in Texas 🤠
- BS & MS from Stanford Computer Science 🌲
- Adversarial ML research at Google Brain 🧠
- First ML engineer at an applied ML startup 🦄
  - Worked with terabytes of time series data
  - Built infrastructure for large-scale ML and data analytics
- PhD student at UC Berkeley 🐻

Evolution of my Interests 🧬

- In school and research, I trained *many* models and cared about *robustness*
  - Fairness
  - Generalizability to unknown distributions
  - Security
- In industry, I wanted to train *few* models but do *lots* of inference
- What happens beyond the validation or test sets?

The Depressing Truth about ML IRL 💔

- Most data science projects don't make it to production
- Data in the "real world" is not clean and balanced like canonical datasets (e.g., ImageNet) ⚖️
- Data in the "real world" is always changing 🌎
- Things break in prod
  - So...we need ops 🕵🏽‍♂️

Breakout Discussion

- What brought you here?
- What do you monitor for your ML pipelines?
- What tools do you use to monitor?
- What are you hoping to learn?

Course Overview and Learning Goals ✅

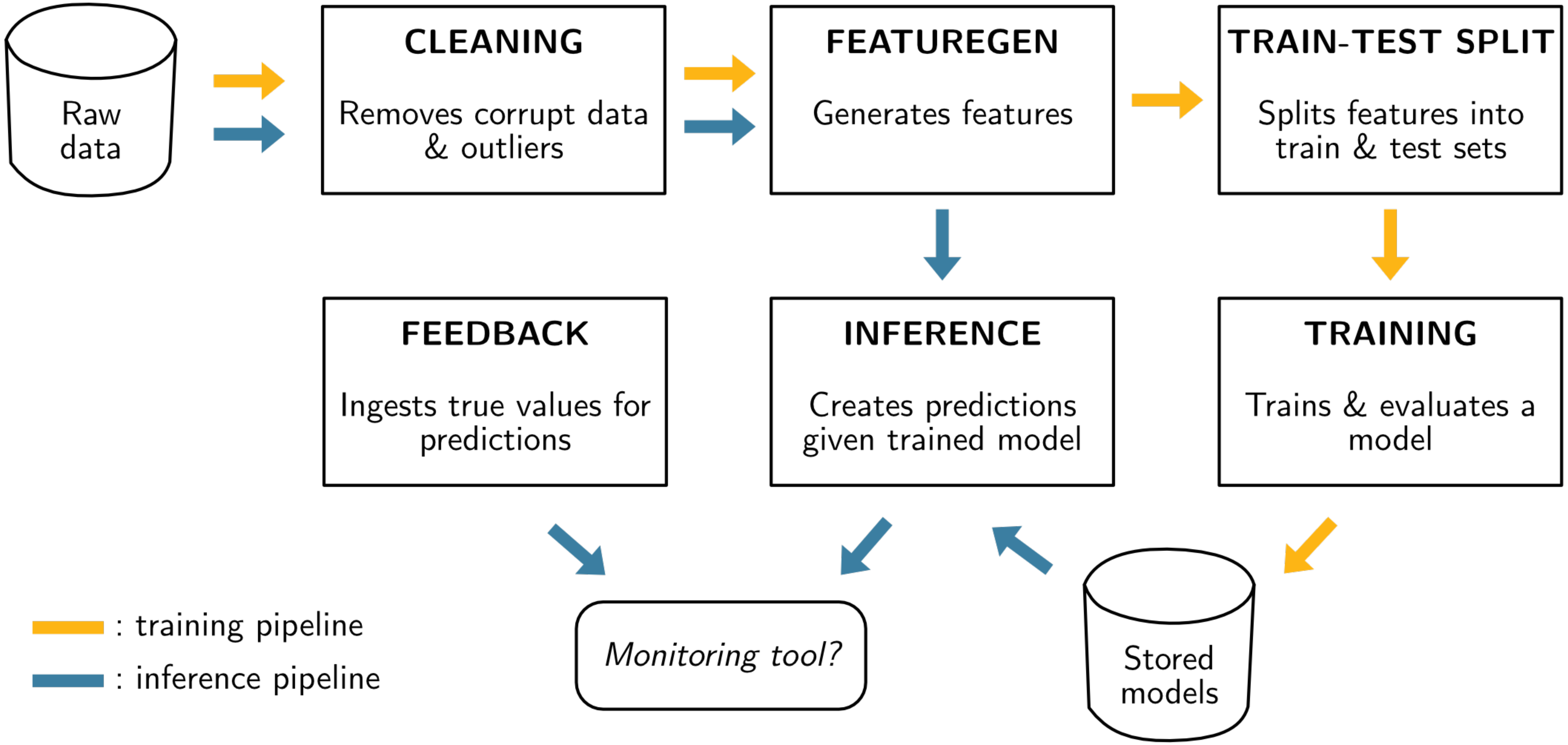
1. What is an end-to-end ML pipeline?
2. ML monitoring
    1. What is it?
    2. Why is it hard?
3. Performance drift detection
4. Building a data management system for ML monitoring

Toy ML Task: Running Example 🪀

Task familiarization 🚕

- Binary classification task: predict whether a passenger in a NYC taxi ride will give the driver a "reasonable" tip (>10% of fare)
- Using NYC Taxi & Limousine Commission public dataset
- Using `pd.DataFrame` and `sklearn` Random Forest Classifier
- Evaluating **accuracy**

Pipeline familiarization 👷🏽



Notebook #1 📓

Familiarizing Ourselves with the ML Pipeline

- Prereqs: Python 3.9+, unix-based shell
- Clone monitoring repository
  - `pip install -r requirements.txt`
- Go through first notebook

ML Performance Monitoring: Challenges and Solutions 👷

Machine Learning is Everywhere

But Hard to Operationalize

Harvard Business Review

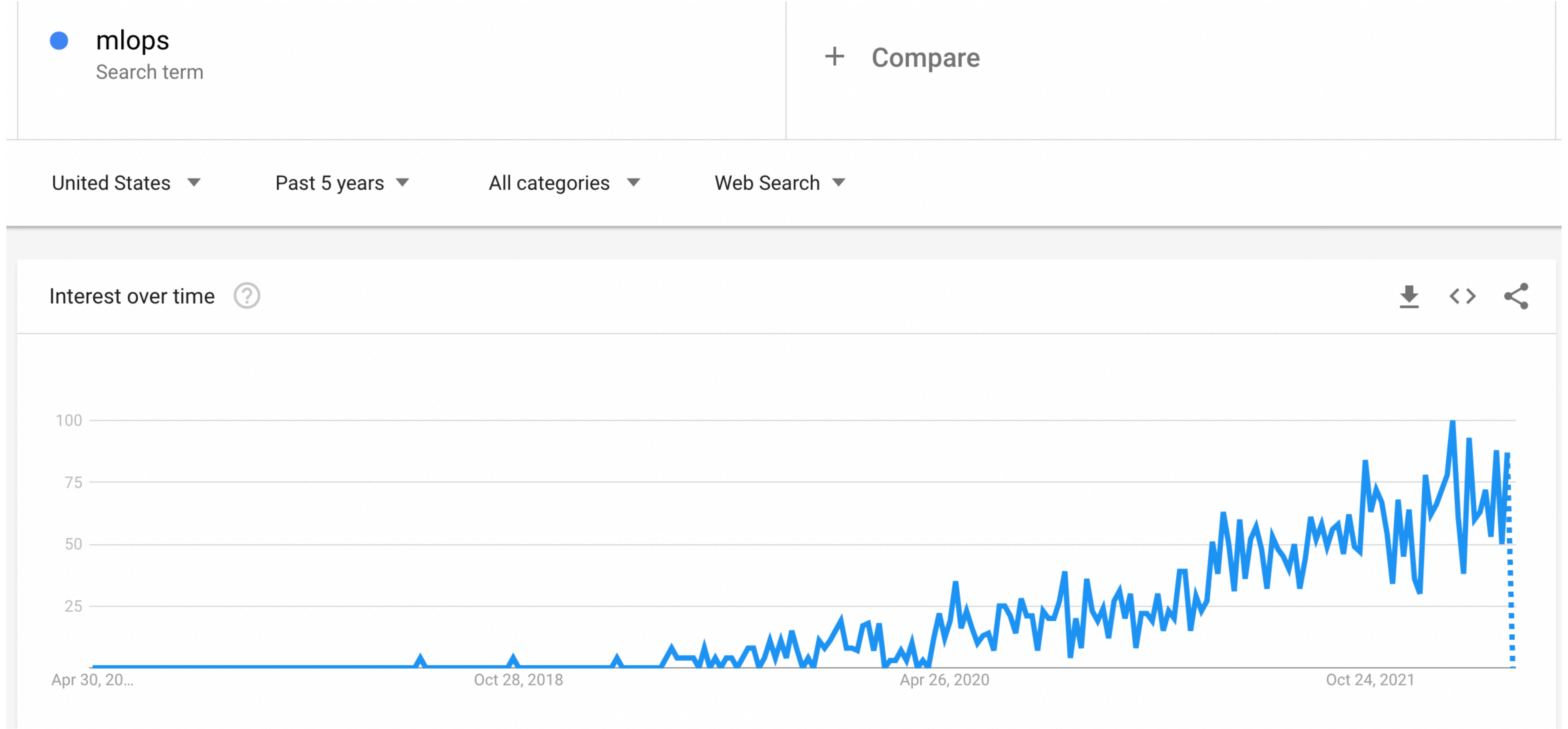
Subscribe Sign In

Technology And Analytics

When Machine Learning Goes Off the Rails

by Boris Babic, I. Glenn Cohen, Theodoros Evgeniou, and Sara Gerke

From the Magazine (January–February 2021)



CONFESSIONS OF A DATA GUY

Home About Contact Resources

Just because a ML model is "complex" doesn't mean the MLOps surrounding it has to be black box as well.

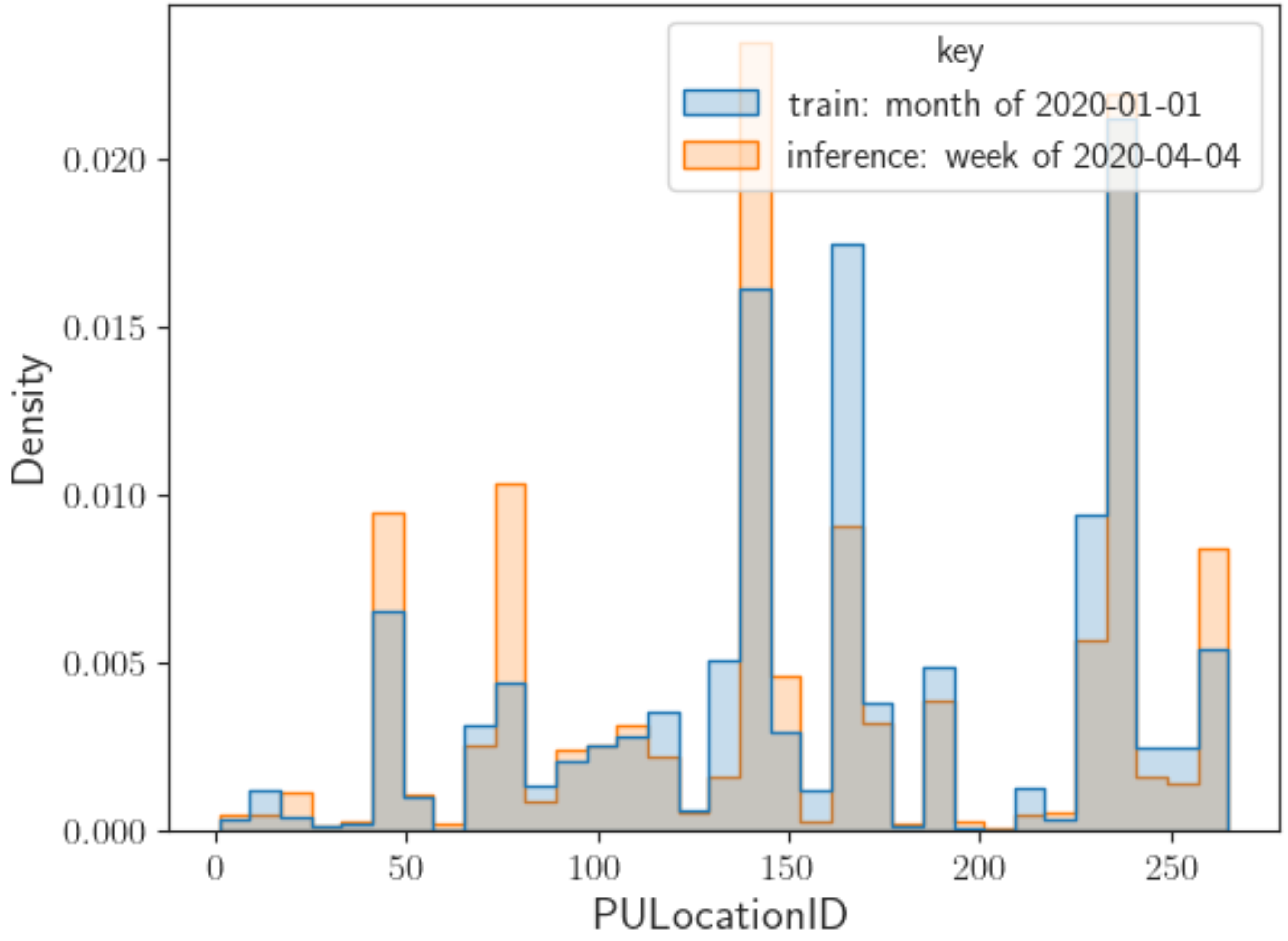
reality

MLOps is hard because once you try to put a system in place around a ML model, the reality starts to set in. The whole point of MLOps is to make some ML model lifecycle productionized and hardened, ready for the real world without someone constantly babysitting.

Need for Monitoring

Data “drifts”... 🤔

- $P(X_{train}) \neq P(X_{test})$
- $P(Y_{train} \mid X_{train}) \neq P(Y_{test} \mid X_{test})$
- Model performance drops over time



Need for Monitoring

Case Study: EIS for US police departments 👮

- **Why?**
  - Model performance drops
  - Severe consequences — allocating interventions to wrong officers, missing allocations to high-risk officers
- **What?**
  - Data integrity checks
  - Anomaly Detection
  - Model performance metrics
- **How?** Customized

Ackerman et al. 2018

Why is ML Monitoring Hard?

- Complicated pipelines
  - Distributed architecture
  - Many components
- Lack of *feedback*, or ground-truth data
- Lots of different errors

Tiered Approach to Model Monitoring

1. Basic data validation
2. Anomaly detection
3. Performance drift detection (course focus)

Basic Data Validation

Solution: Low-Latency Online Checks

- Encode constraints for features
    - Type checks
    - Value checks (e.g., nonnegativity)
    - Set membership checks (e.g., for categorical variables)
- Log results somewhere!
- Can run in background to minimize latency

Anomaly Detection

Solution: Z-Score and Outlier Checks

- Compare to historical aggregations, e.g. assert:
  - Within 2 standards deviations of historical mean
  - Within IQR (interquartile range) of historical median
- Run on:
  - Input & output values (features, predictions)
  - Fraction of missing values

Performance Drift Detection

Solutions: TBD

- Distance metrics (Kolmogorov-Smirnov test)
  - Hard to scale
  - Alert fatigue 🚨
- Approximate ML metric that uses labels (e.g., accuracy, precision)
  - How to do without labels?

Performance Drift Detection 📉

Estimating Accuracy for Unlabeled Predictions

Importance Weighting ⚖️

- **Problem**: no labels
- **Solution**: importance-weight training bucket accuracy
  - Split train set into buckets
  - Create criteria for buckets
  - Determine training accuracy for each bucket

Estimating Accuracy for Unlabeled Predictions

Importance Weighting ⚖️

- At inference, classify feature vector into bucket & aggregate training accuracies
- Example
  - Buckets FiDi and Midtown have training accuracies of 80% and 50%
  - 100 FiDi rides, 500 Midtown rides
  - Estimated inference accuracy $= 0.8 \times 100 + 0.5 \times 500 = \dfrac{80 + 250}{600} =$ **55%**

Estimating Accuracy for Unlabeled Predictions

Bucketing Strategy Matters!

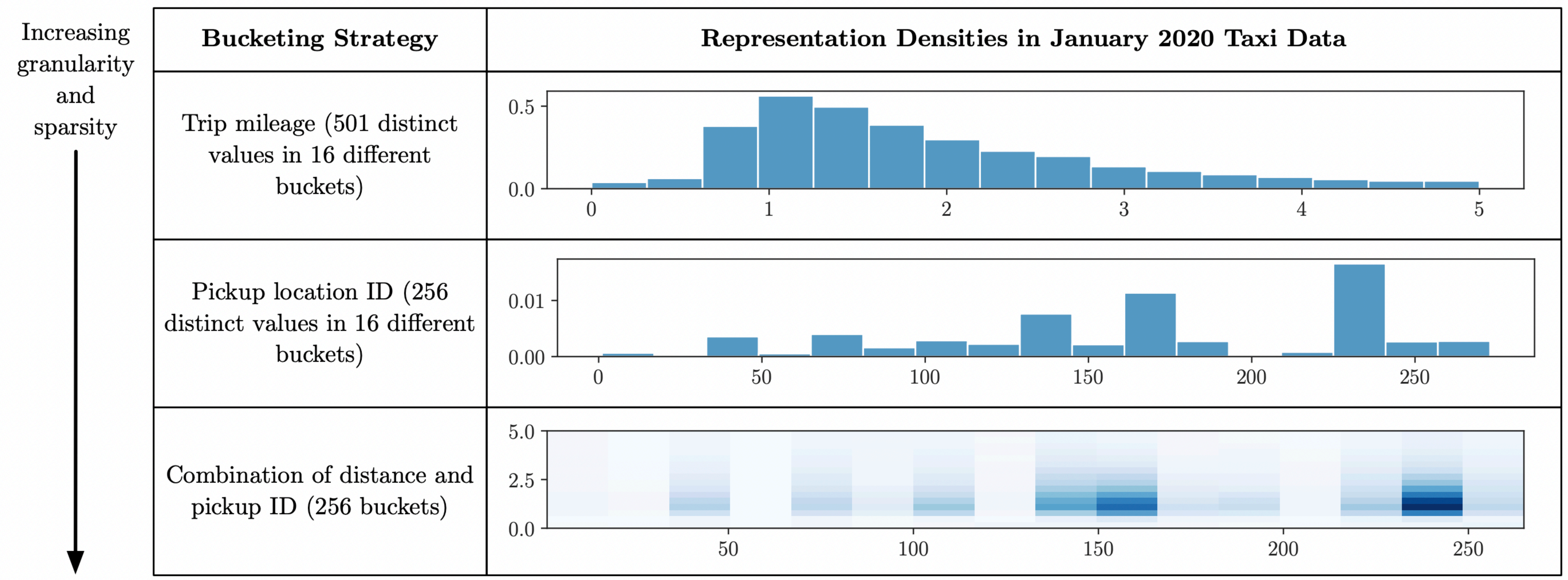


Figure 2: Bucketing strategies based on pickup location and trip distance. 1-D histograms are normalized to show density. As buckets become more finer-grained, they also become sparse.

Notebook #2

Familiarizing Ourselves with Importance Weighting

- Prereqs: Python 3.9+, unix-based shell
- Clone monitoring repository
    - `pip install -r requirements.txt`
- Go through second notebook

Building and Integrating a Monitoring System 🛠️

Goal: Estimate Real-Time ML Performance

Proposed Secret Sauce 🍝

- Logging & triggers framework to maintain:
  - Predictions
  - Labels
  - Metric estimates
    - Importance weighting to estimate performance for unlabeled predictions

Logging & Triggers Framework

Interface

- User-defined metric functions over different window sizes

```python
import math
import numpy as np

def metric_fn(y_true: np.ndarray, y_pred: np.ndarray):
  return math.sqrt((y_true - y_pred) ** 2)
```

- `log_training_set`, `log_prediction` and `log_feedback` functions
- `get_metrics` function

Logging & Triggers Framework

Interface

```
import math
import numpy as np

from monitoring import Task

def metric_fn(y_true: np.ndarray, y_pred: np.ndarray):
  return math.sqrt((y_true - y_pred) ** 2)

task = Task("test")
task.register_metric("mse", metric_fn, [60, 60*60])

task.log_prediction("ABC", ... )
task.log_feedback("ABC", ... )
print(task.get_metrics())
```

Naive Implementation 👶

Task Initialization

- Create empty predictions & feedback tables
- Compute buckets from training set
  - Gaussian Mixture Model
- Initialize bucket counters for each window to 0

Naive Implementation 👶

`log_prediction` Trigger

- Given: ID, prediction timestamp, features, prediction
- Add prediction to predictions table
- Run bucketing function on features to get bucket ID
- For all windows containing prediction timestamp
  - Increment window[bucket ID] counter
- Recompute metrics

Naive Implementation 👶

`log_feedback` Trigger

- Given: ID, feedback timestamp, feedback
- Add feedback to feedbacks table
- Get prediction timestamp, bucket ID from predictions table
- For all windows containing prediction timestamp
    - Decrement window[bucket ID] counter
- Recompute metrics

Naive Implementation 👶

Storage: DuckDB

- In-memory
  - Metric function (e.g., accuracy)
  - Bucketing function
  - Windows
    - Start timestamp
    - Window size (s)
    - Predictions
    - Labels
- Disk
  - Training set
  - All predictions
  - All labels
  - Historical metric values
  - Prediction <> label view?

Notebook #3

Integrating Monitoring System into ML Monitoring Workload

- Prereqs: Python 3.9+, unix-based shell
- Clone monitoring repository
  - `pip install -r requirements.txt`
- Read code in db/task.py
- Go through third notebook

Wrapping Up 🛠️

Breakout Discussion

How to React when ML Performance Goes Down?

- Responses
  - Fix data errors
  - Retrain model
  - Change objective
- Response “playbook”

Breakout Discussion

Wrapping up

- Learning outcomes
- Lessons learned from performance drops
- Further readings
- Q&A