Alessya Visnjic

CEO @ WhyLabs.ai

Engineer, architect, aspiring data scientist
Traveler, Beat Saber fan, aspiring baker

Past: Allen Institute for AI, Amazon/AWS

Connect: Linkedin, Twitter, Virtual Coffee

Goals:

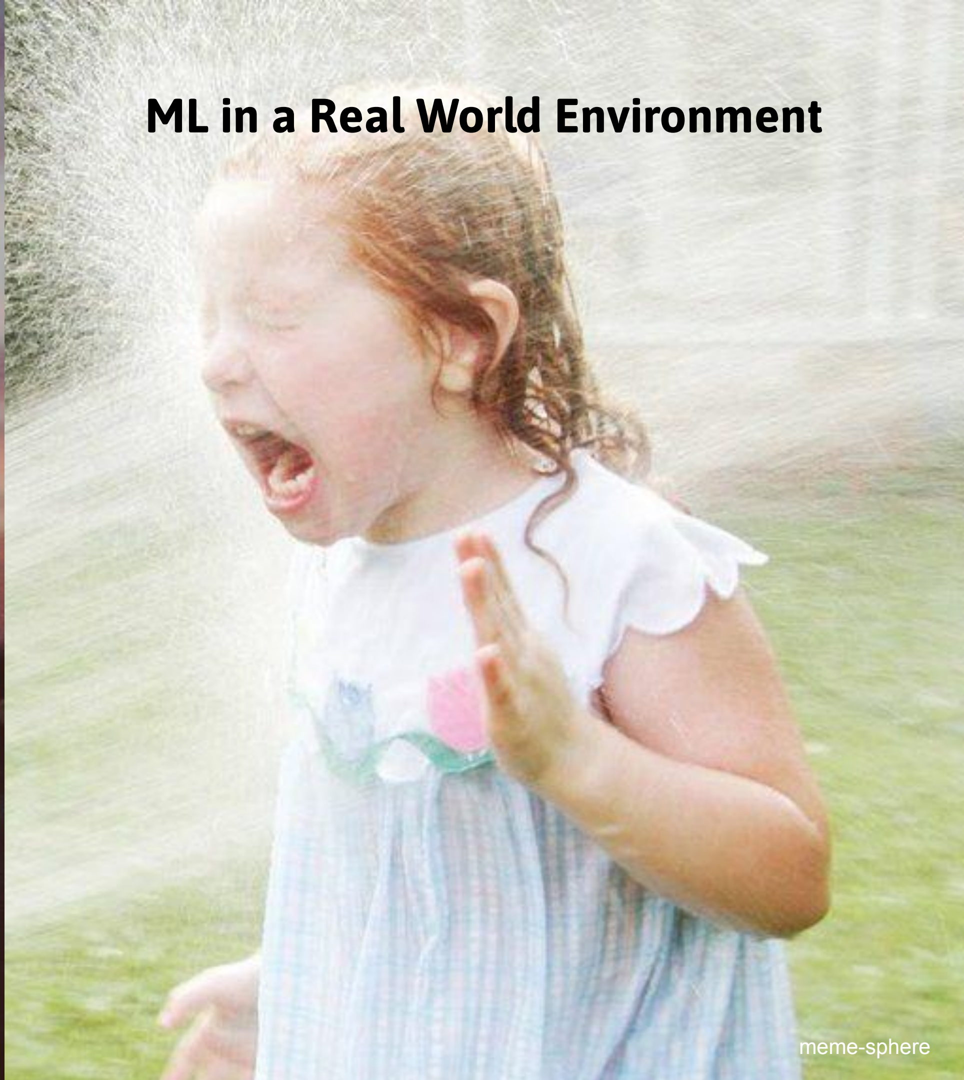
1. ***Develop an intuition for ML/AI observability systems***

2. ***Explore a logging component of an observability system***

Agenda

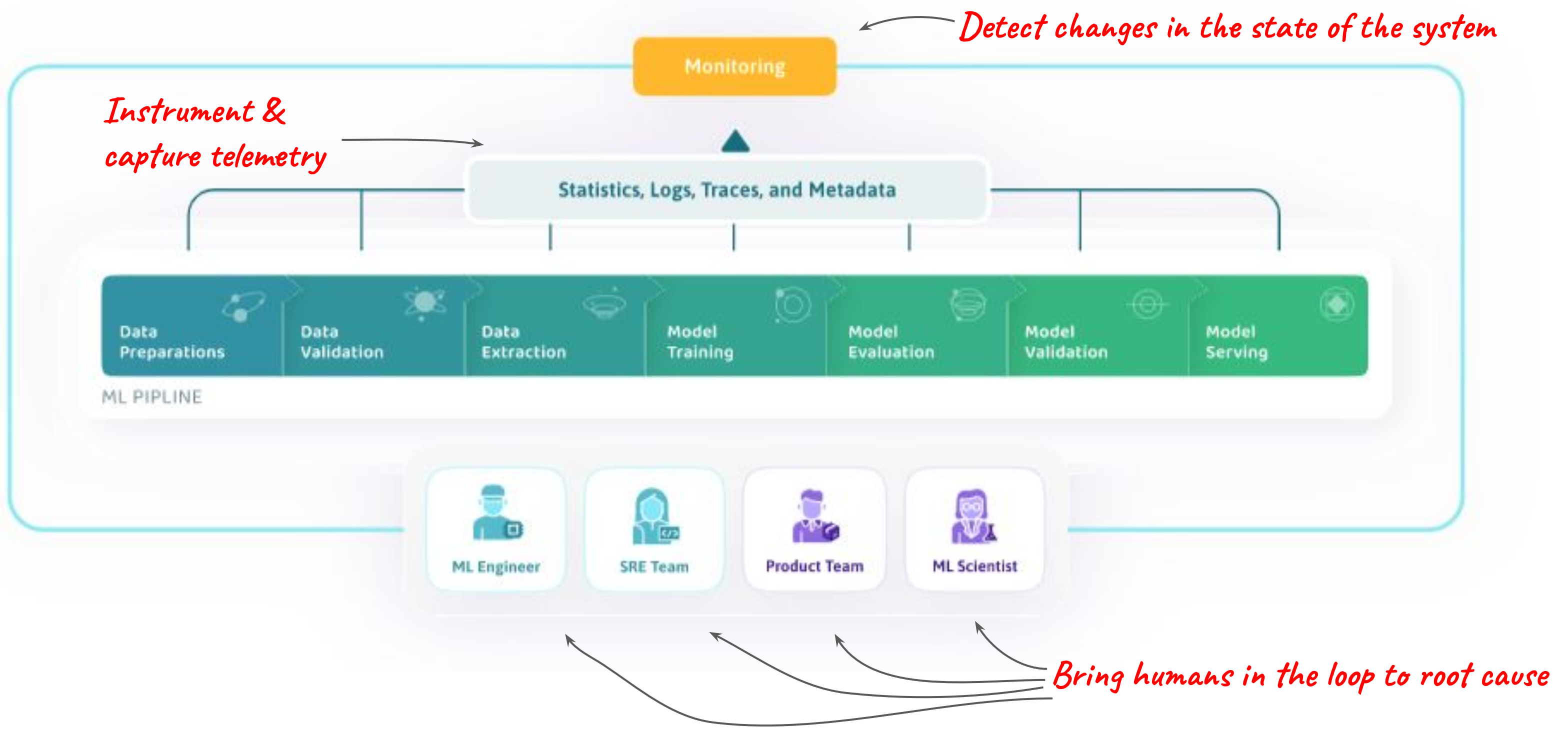
1. Monitoring production ML applications
2. From monitoring to observability
3. Designing an ML monitoring system
4. From monitoring to observability
5. ML Observability system architecture
6. ML Telemetry (whylogs demo)
7. ML Observability system overview (WhyLabs' AI Observatory demo)


ML in an Experimentation Environment
@jhong8 on unsplash


ML in a Real World Environment
meme-sphere

Monitoring production ML applications: simple task?

Simple task: Detect and root cause changes in the behavior of an ML application



Monitoring production ML applications: systems considerations

The ML application is part of a greater software system and this system is alive…

- State over time: tracking past, current, and future states of the application
- Upstream systems: what changes your state/behavior?
- Downstream system: how do you change the state/behavior of others?

Monitoring production ML applications: scalability considerations

Monitoring an ML application shouldn't cost more than running the ML application…

- Latency: how much time do you have to measure and notify?
- Scale: how measurements will be processed?
- Cost: how much should you spend per measurement?

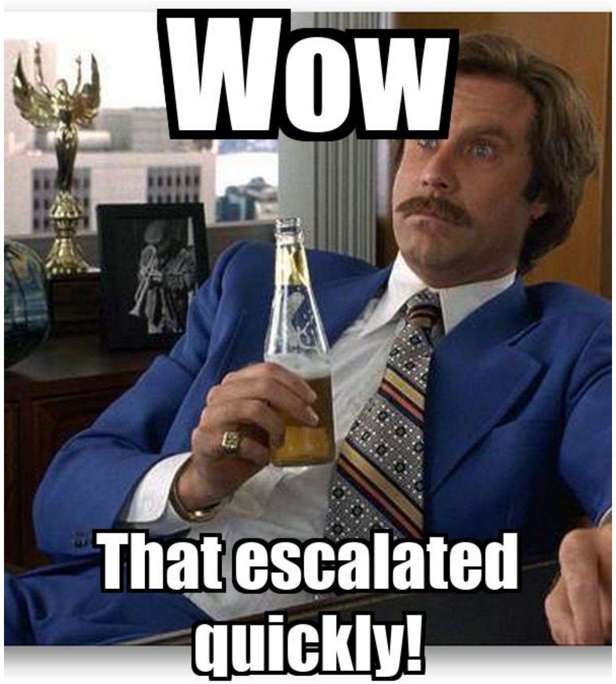
Monitoring production ML applications: ownership considerations

Monitoring solutions in any team setting require long term ownership…

- Automation: how much can be automated?
- Maintenance: how would the system be maintained over time and by who?

Monitoring production ML applications: *real world considerations!*

- State over time: tracking past, current, future states of the application
- Upstream systems: what changes your state/behavior?
- Downstream system: how do you change the state/behavior of others?
- Latency: how much time do you have to measure and notify?
- Scale: what is the volume of the measurements to be made?
- Cost: how much should you spend per measurement?
- Automation: how much can be automated?
- Maintenance: how would the system be maintained over time?



Monitoring production ML applications: a few design principles

- Decouple the process of capturing telemetry from the process of acting upon it
- Put telemetry capturing as close to the data & model as possible
- Make telemetry capturing support both batch and streaming systems
- Make telemetry capturing processes platform and model agnostic
- Design telemetry artifacts to be lightweight (support formats for storage & consumption)
- Design telemetry artifacts to be extensible & configurable
- Design telemetry to be megrable (over time and across instances/partitions)
- Design telemetry storage to support massive cardinality
- Design monitoring system to support a wide range of forecasting & anomaly detection methods
- Design monitoring system to support correlation and lineage artifacts

Telemetry design matters A LOT!

From monitoring to observability: how different are the systems*?

Monitoring:

- Capture metrics
- Measure change
- Notify

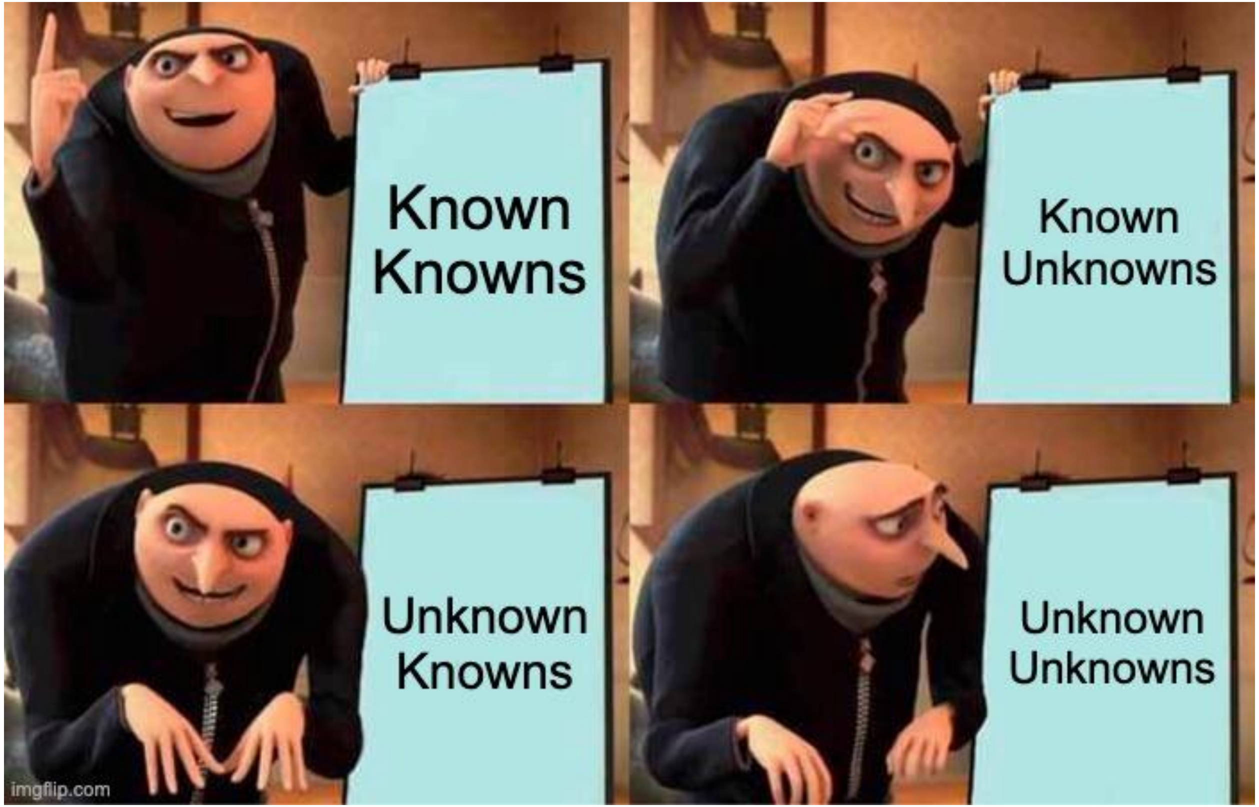
Observability:

- Capture internal state (including metrics)
- Measure change
- Notify
- Root cause

Telemetry is at the root of the difference between the two

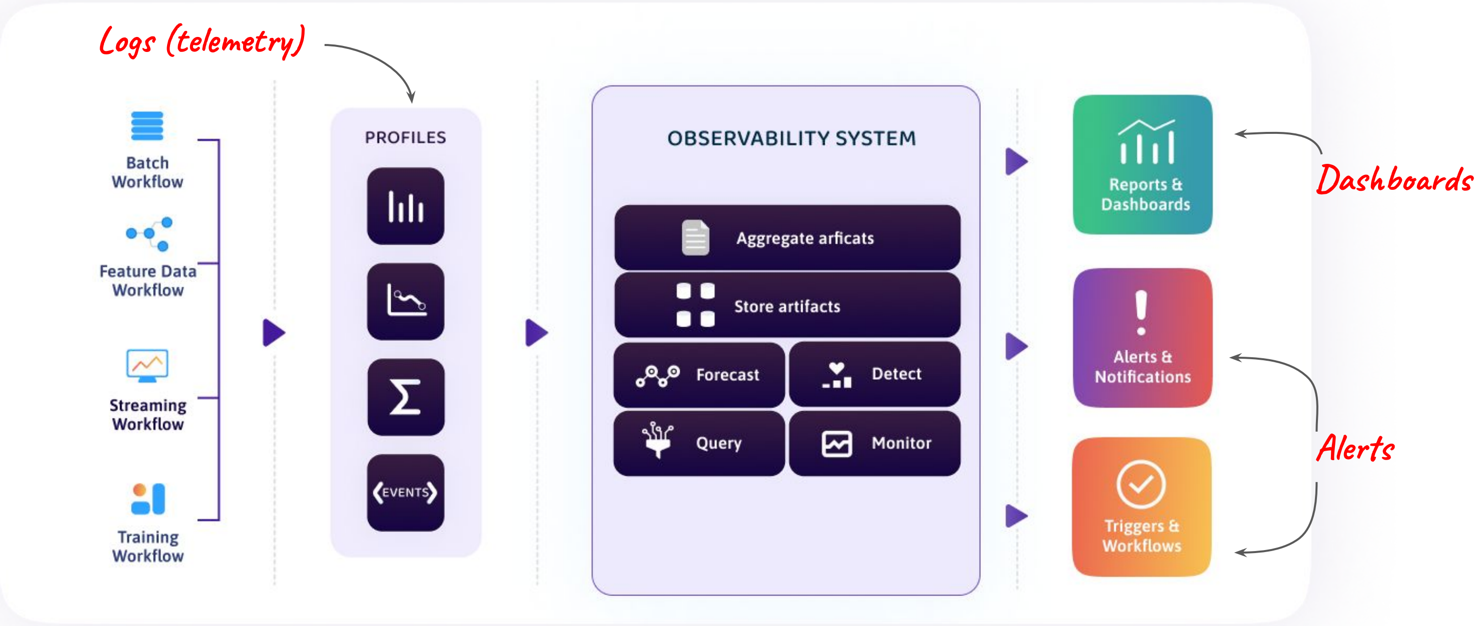
* Extremely simplified view of the world

From monitoring to observability: unknown unknowns!



From monitoring to observability: detection is not enough!

ML Observability system architecture: high level overview



* Monitoring toolbox components from the perspective of users. Source: [Data Distribution Shifts and Monitoring]().

ML Telemetry (Logs)

Telemetry = information that captures the state of an ML application

One “unit” of telemetry = profile

What to capture:

- Lineage Metadata
- Schema
- Counts
- Summary statistics
- Distributions
- Stratified samples

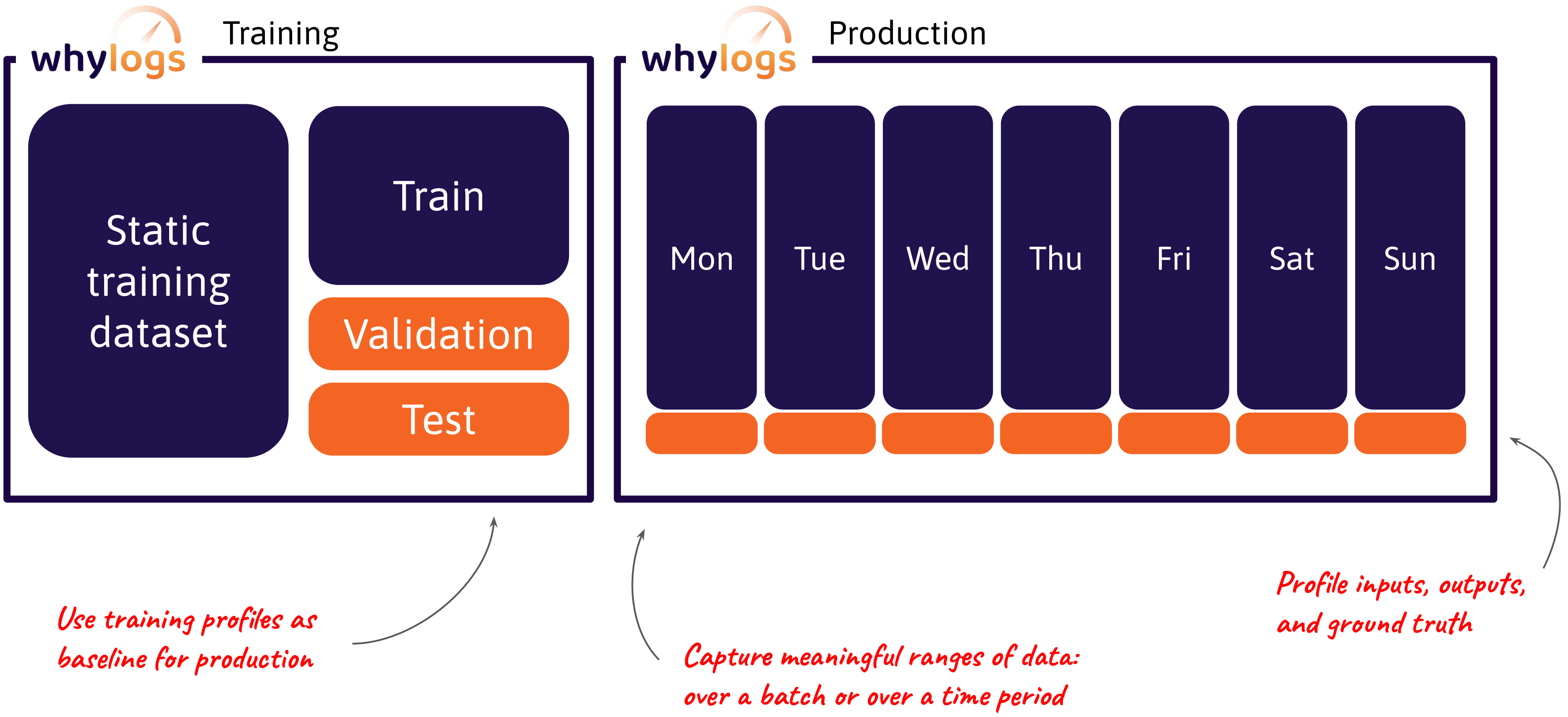
ML Telemetry: standardizing ML and data telemetry

whylogs

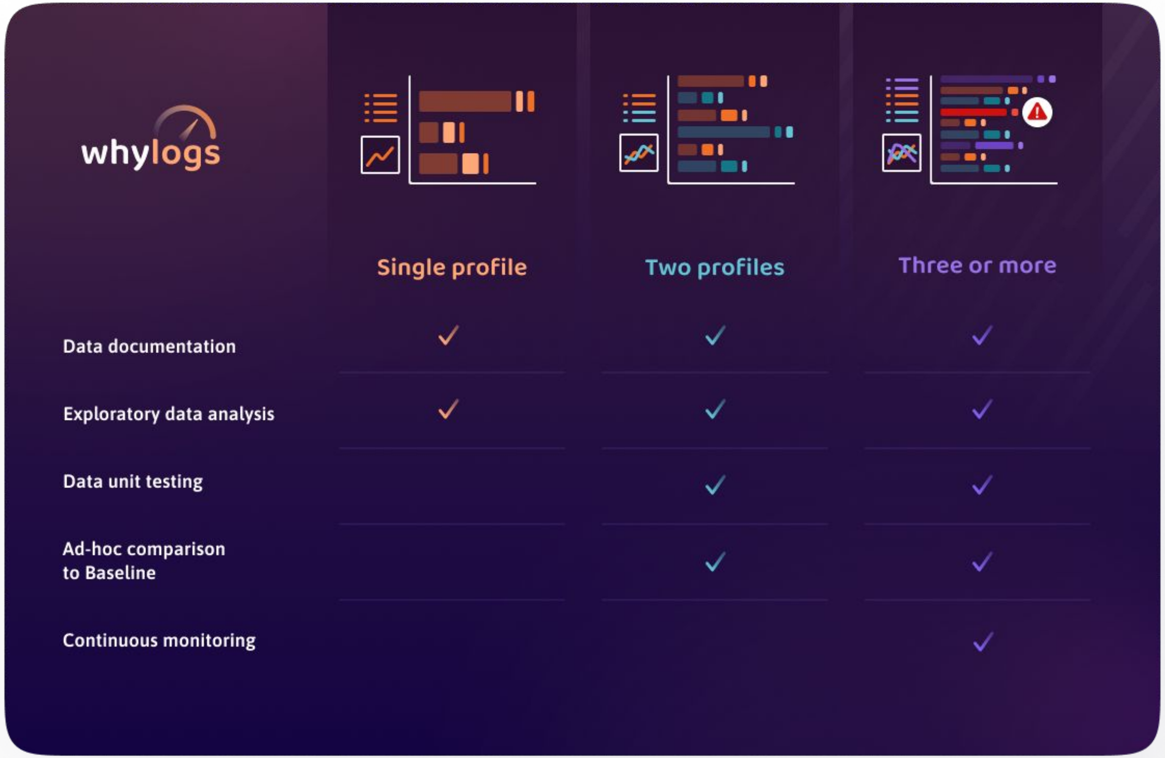
bit.ly/whylogs:

Telemetry for the ML stack

ML Telemetry: profiling during training and during inference



ML Telemetry: best practices, one profile at a time



	Single profile	Two profiles	Three or more
Data documentation	✓	✓	✓
Exploratory data analysis	✓	✓	✓
Data unit testing		✓	✓
Ad-hoc comparison to Baseline		✓	✓
Continuous monitoring			✓

It better be easy, cheap, and fast to capture these profiles

ML Telemetry: easy to capture - no configuration necessary

```
from whylogs import get_or_create_session
import pandas as pd

session = get_or_create_session()

df = pd.read_csv("path/to/file.csv")

with session.logger(dataset_name="my_dataset") as logger:

    #dataframe
    logger.log_dataframe(df)

    #dict
    logger.log({"name": 1})

    #images
    logger.log_images("path/to/image.png")
```

Log rotation included

You can even profile images and other unstructured data!

ML Telemetry: cheap to capture & cheap to store

whylogs captures statistics using stochastic streaming algorithms, which enables a few important properties:

Dataset	Size	No. of Entries	No. of Features	Est. Memory Consumption	Output Size (uncompressed)
Lending Club	1.6GB	2.2M	151	**14MB**	7.4MB
NYC Tickets	1.9GB	10.8M	43	**14MB**	2.3MB
Pain pills in the USA	75GB	178M	42	**15MB**	2MB

Scales w/ the number of features/statistics

Near-constant memory footprint

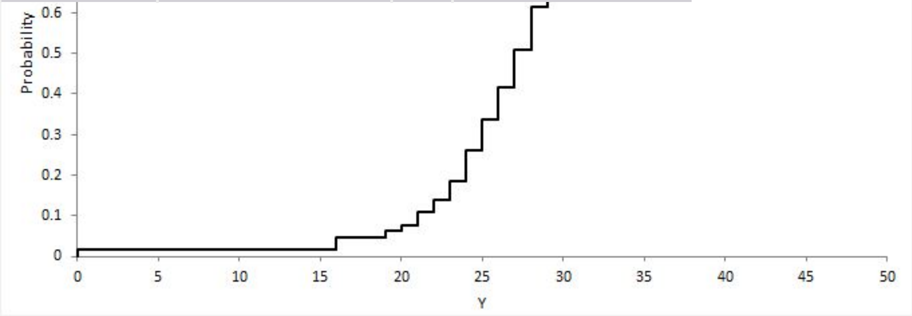
Tiny output, even smaller when compressed

ML Telemetry: accurate statistics, density functions, and error bars

Feature name	count	max	min	stddev	nunique	null_count	quantile_0.0000	...	quantile_1.0000
chlorides	1199.0	0.611	0.012	0.044	134.0	0.0	0.012	...	0.611
quality	1199	8.000	3.000	0.785	6.0	0.0	3.000	...	8.000
alcohol	1199	14.900	8.400	1.060	65.0	0.0	8.400	...	14.900
density	1199	1.004	0.997	0.001	390.0	0.0	0.990	...	1.004
pH	1199	4.010	2.890	0.153	82.0	0.0	2.890	...	4.010

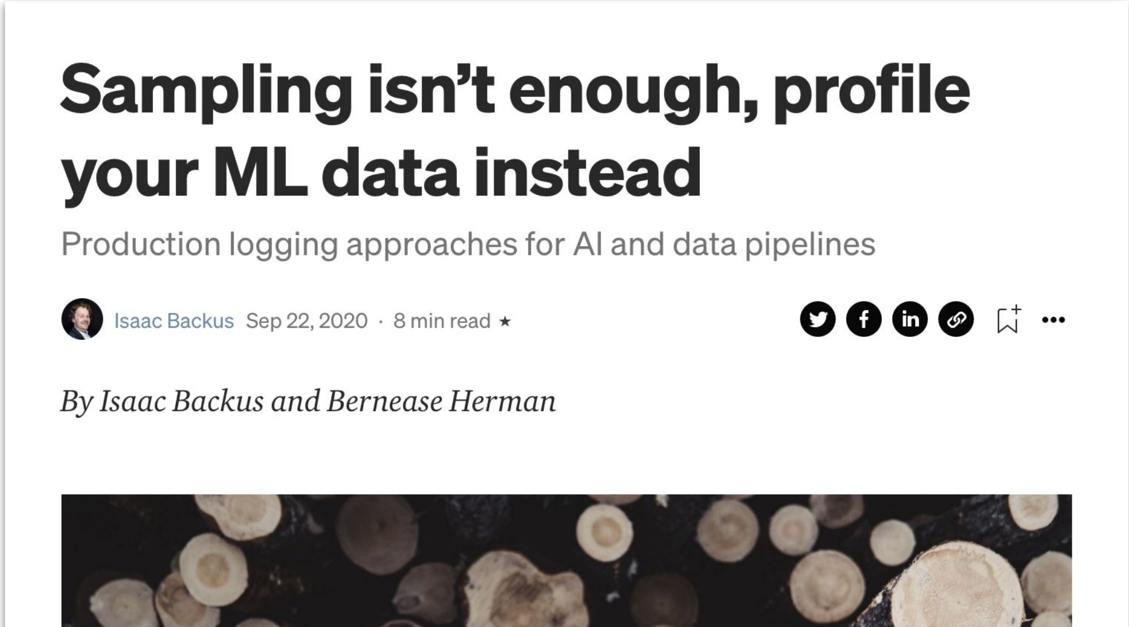
All the stats you need!

Unless you need more... Then customize!



ML Telemetry: capture more accurate insights than sampling

Whylogs profiles 100% of the data to accurately capture distributions. Capturing distributions from sampled data is significantly less accurate. This chart presents median errors for distributions estimated with whylogs vs. from sampled data.

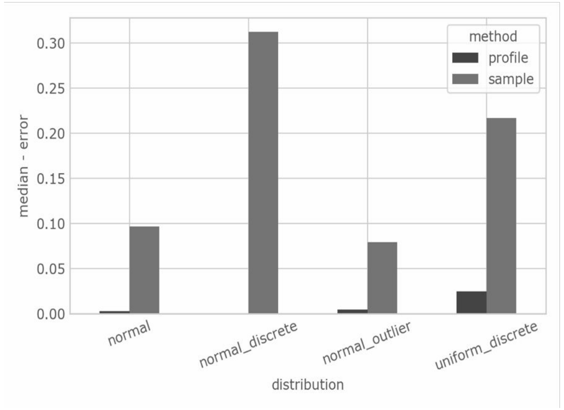


Sampling isn't enough, profile your ML data instead

Production logging approaches for AI and data pipelines

Isaac Backus Sep 22, 2020 · 8 min read ★

By Isaac Backus and Bernease Herman



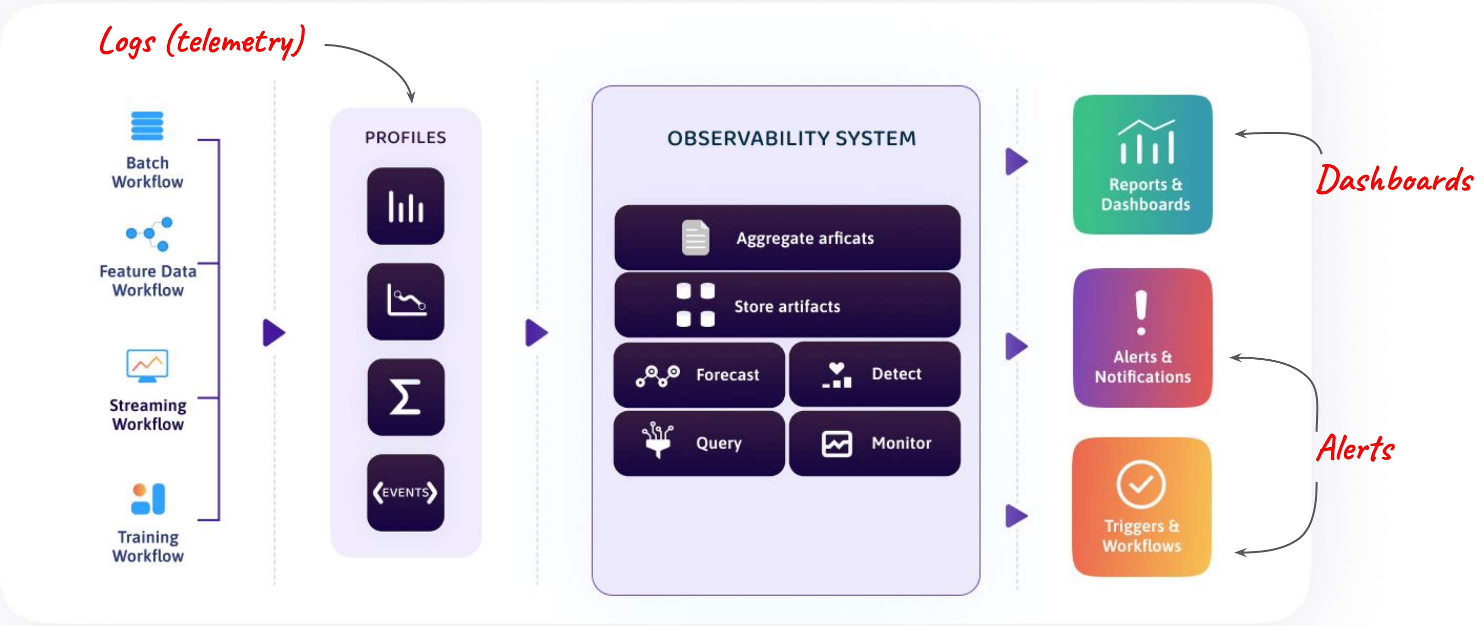
Sampling dynamic data to capture an accurate distribution is <u>very hard</u>

ML Telemetry: whylogs simple demo

```
pip install -U whylogs
```

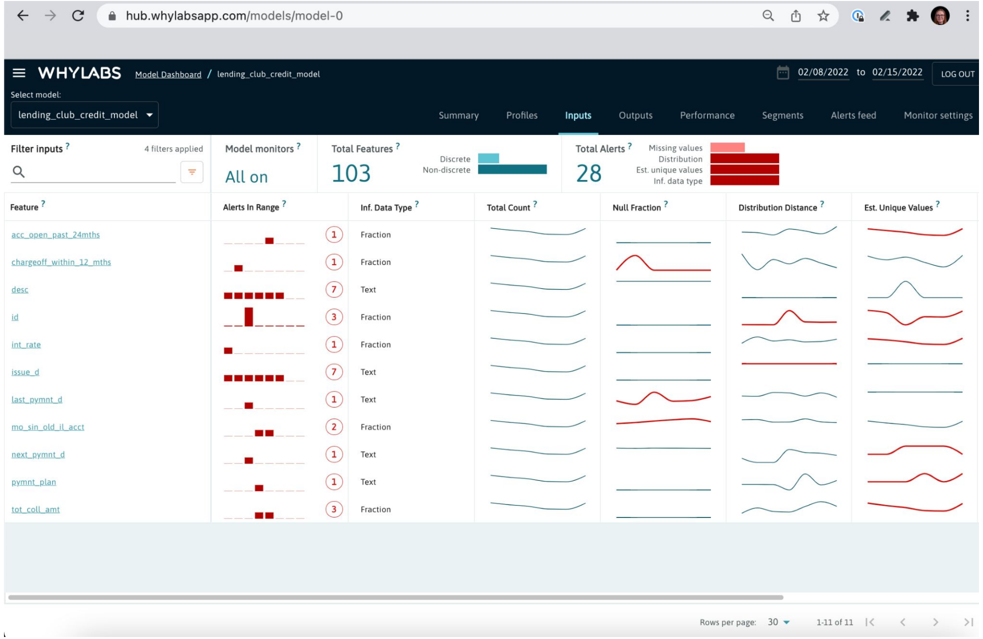
Follow along with this [interactive notebook](bit.ly/CS329s-whylogs)
(`bit.ly/CS329s-whylogs`)

ML Observability system architecture: what's next after telemetry



* Monitoring toolbox components from the perspective of users. Source: Data Distribution Shifts and Monitoring.

ML Telemetry: Observatory demo



Follow along with this interactive notebook
(bit.ly/CS329s-whylogs)

Final thoughts:

- ★ Production ML applications are evolving rapidly and monitoring requirements evolve along
- ★ Observability is yet to be fleshed out for ML/AI applications
- ★ A lot of big, interesting, and important problems yet to be solved
- ★ We are at the very inception of the MLOps toolchain:
  - Great area for establishing expertise/career
  - Great area for new startups

Help define the telemetry standard for ML & data applications:

github.com/whylabs/whylogs

join.slack.whylabs.ai

Thank you!

alessya@whylabs.ai

@zalessya

linkedin.com/in/alessya