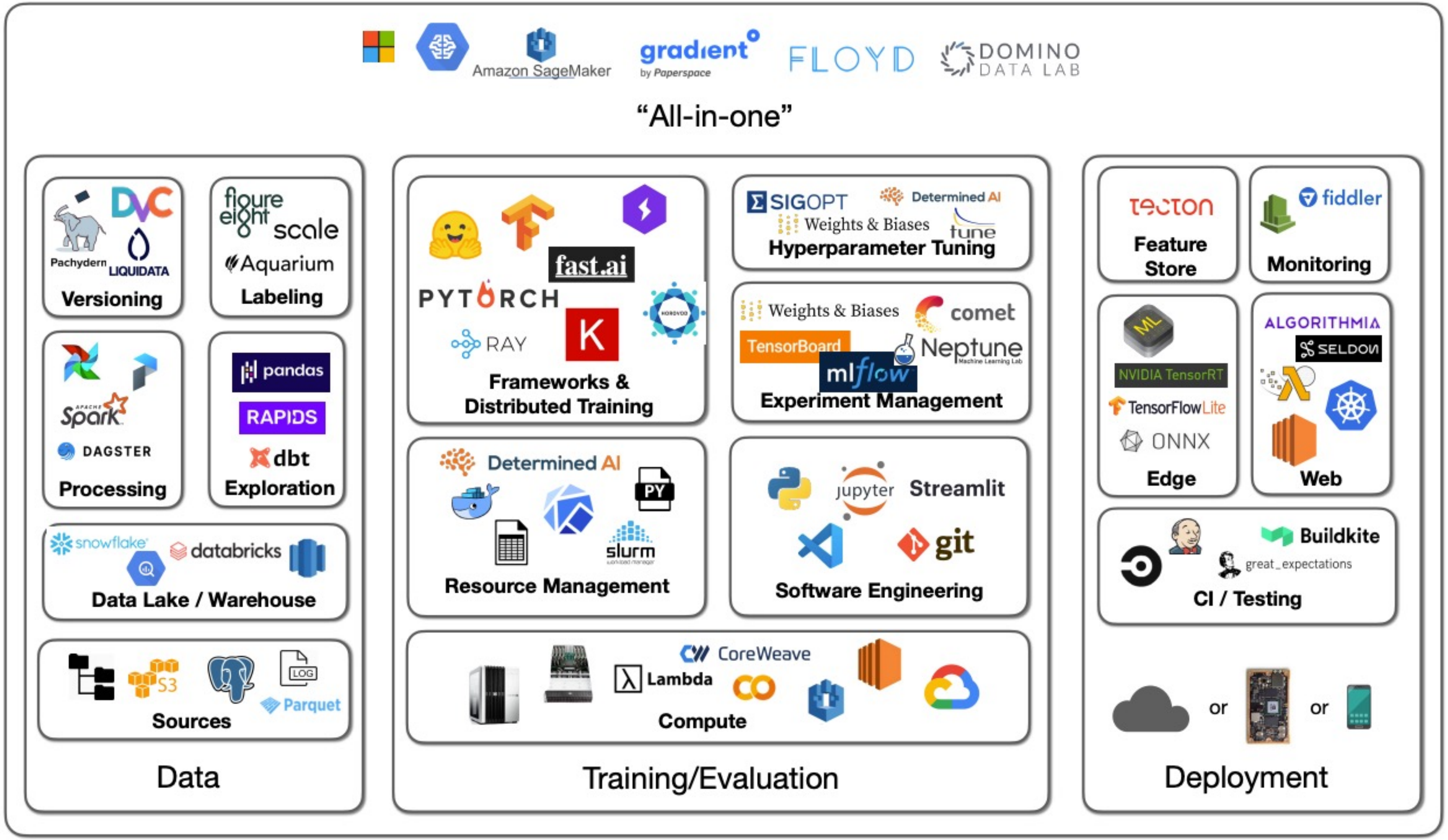
What I learned from monitoring more than 30 Machine Learning Use Cases

Pydata, 11.04.2022

HEY, I AM LINA WEICHBRODT

- Lead Machine Learning Engineer
  at German Internet Bank DKB
- Ex Senior Research Engineer at Zalando
- >30 models: Recommender Systems, Personalization, NLP in Customer Service, Finance
- Large scale production systems

Machine Learning Tooling can be a little overwhelming



Agenda

Add ML Monitoring

Implement Basic Software Monitoring

Use simple tooling

Agenda

Add ML Monitoring

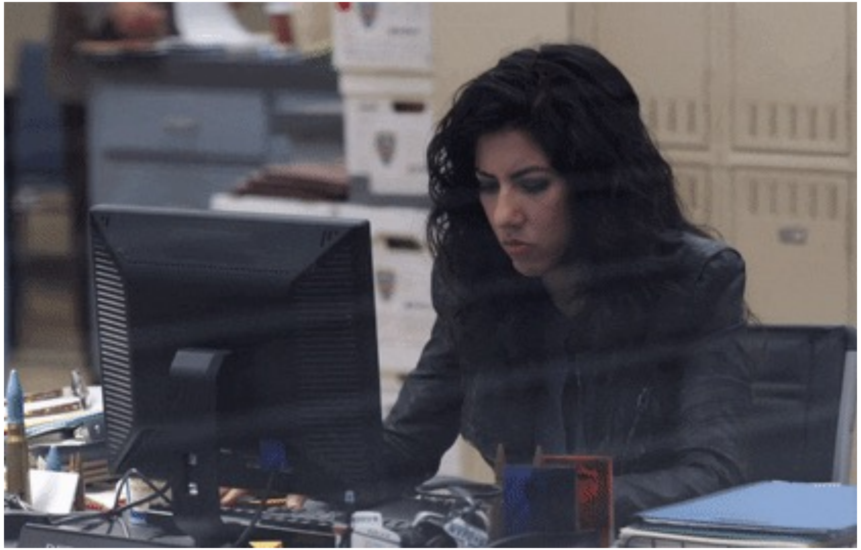
Implement Basic Software Monitoring

Use simple tooling

Software Monitoring Basics

Everybody(!) experiences regular problems

- [Google Cloud Health]
- [AWS Health]



Typical: bugs, human error

You **cannot avoid** problems, just **detect** them **fast** and **act based on severity**

Use the four golden signals

Latency: the time it takes to serve a request

Traffic: the total number of requests

Errors: the number of requests that fail

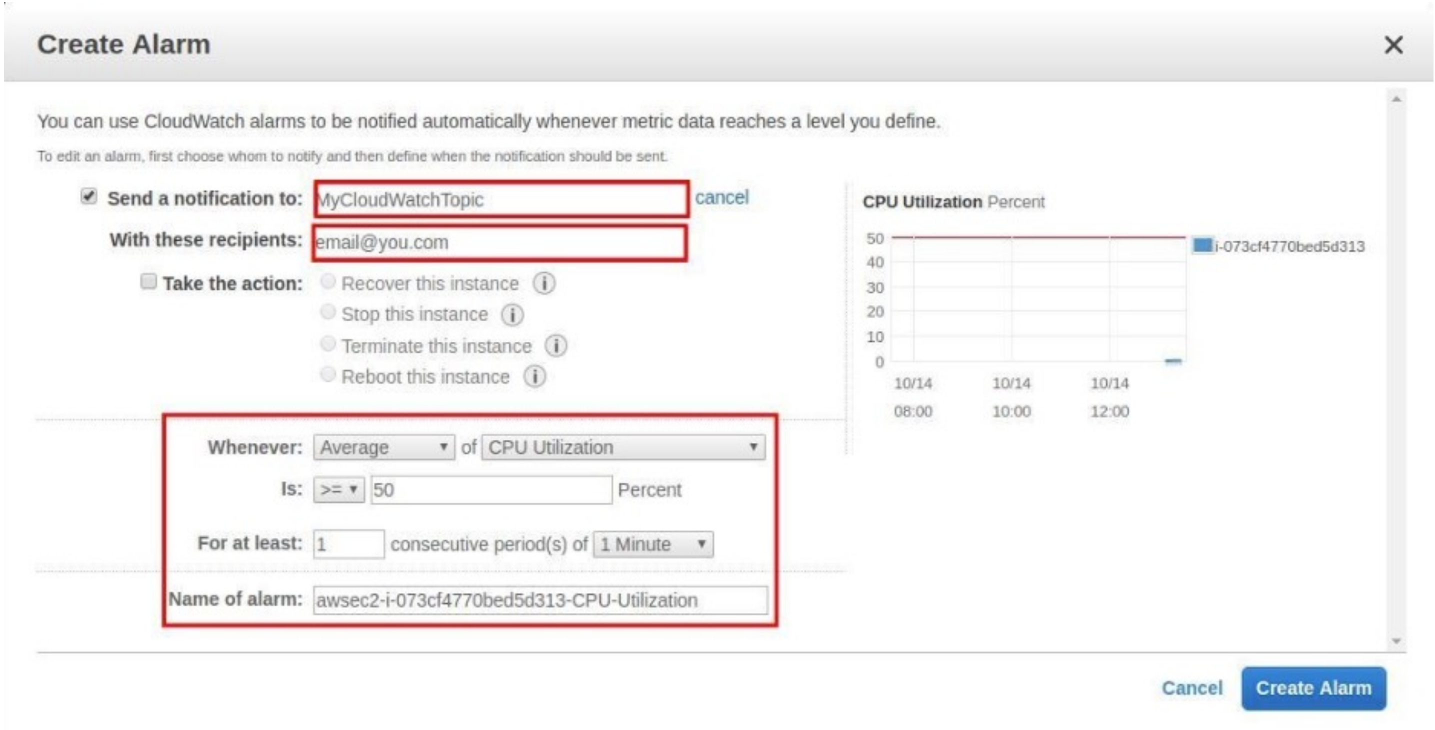
Saturation: the load on your network and servers

- → we focus on **symptoms**, meaning **end-user pain**, not causes
- → Use them for all of your products

Monitoring in practice: Live Dashboards

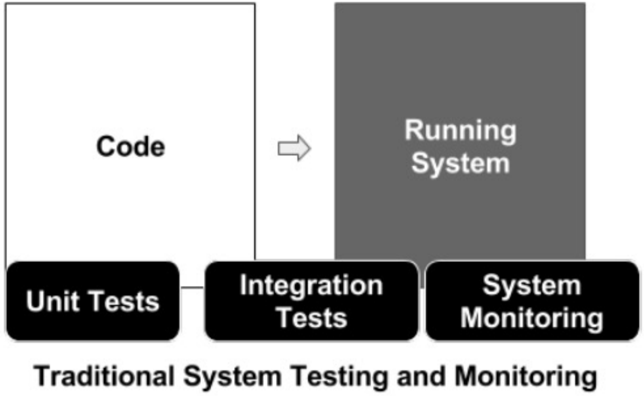
Monitoring in practice: Get notified if a metric is too low or high

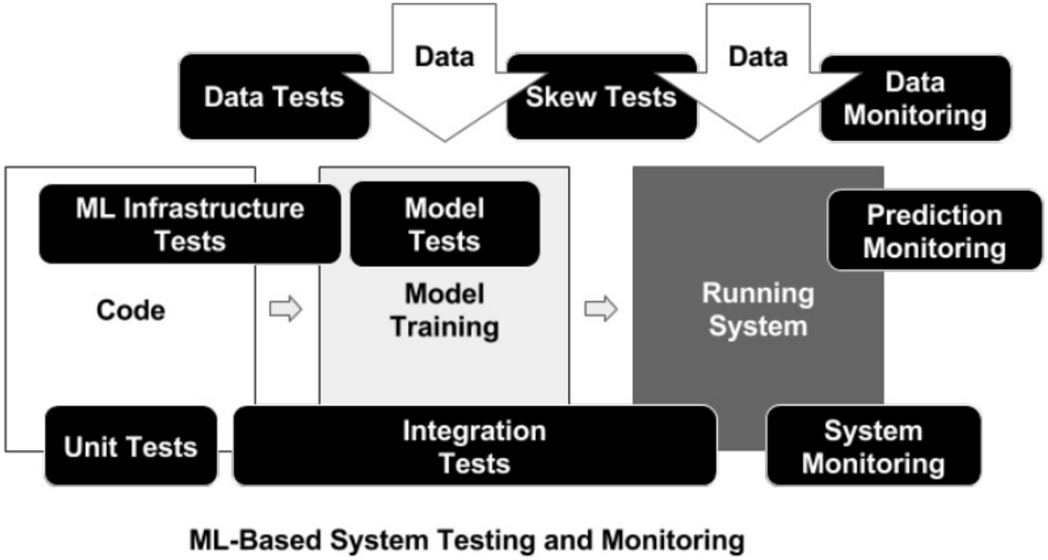
Example with AWS Cloudwatch (many vendors offer this needed functionality)



Is traditional software monitoring enough?

Google paper „ML Test Score“ shows the higher complexity





Silent failures causes huge commercial impact

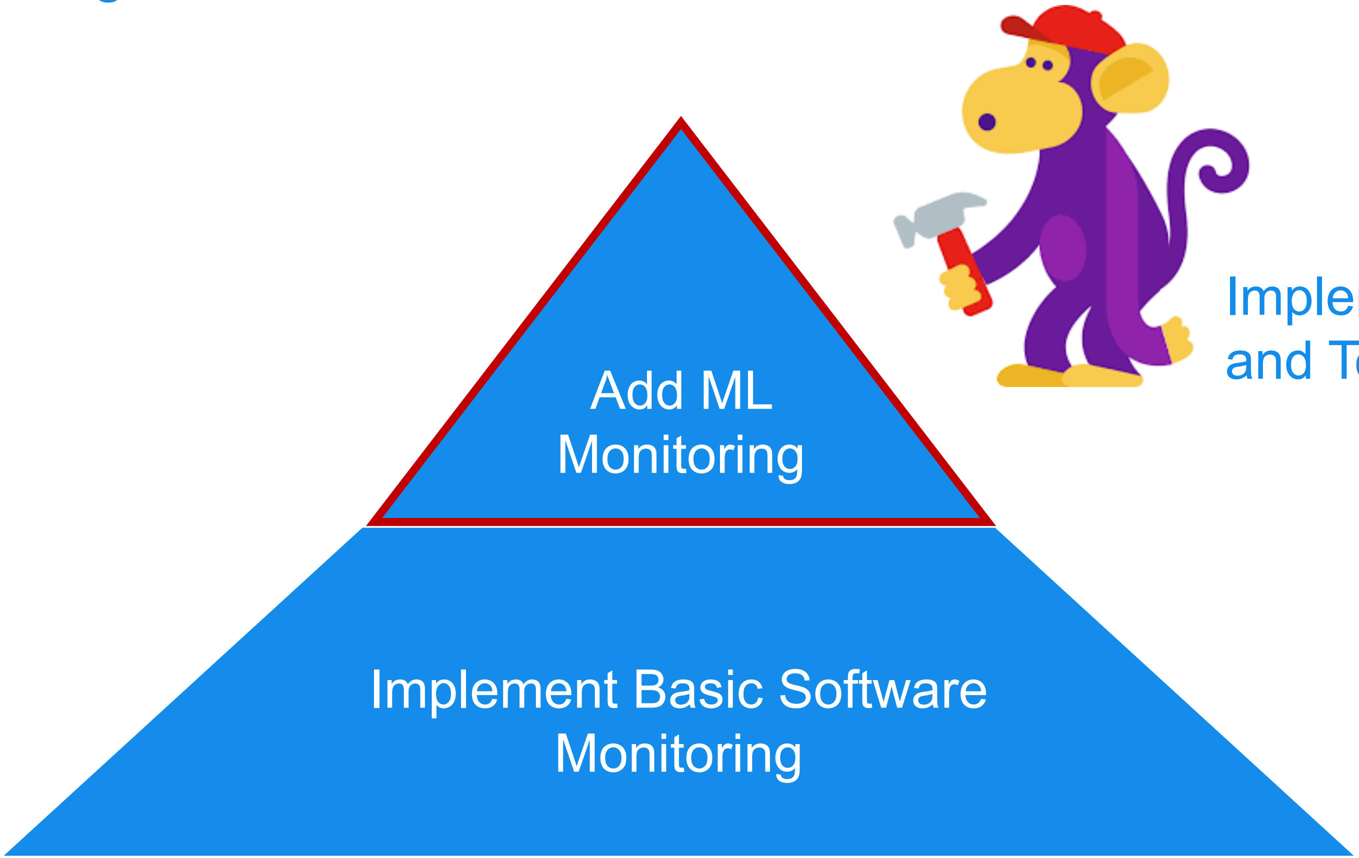
Examples of silent failure I personally experienced:

- Input data changes
  - Client calling us e.g. … sec
  - External s… oss
- Aggressive p…
  - Optional fi…
  - Filter "on sa…
- Bugs in our ow… es
- Model is automa… model is worse
- Tensorflow versi… d, we got a faulty version
- Client changes the way the product works without telling us, e.g. product is now used by not logged in users

None of these things create an error, slowness, saturation. SILENT!

Bigger $$$ impact than most model improvements

Agenda



Implementation
and Tooling

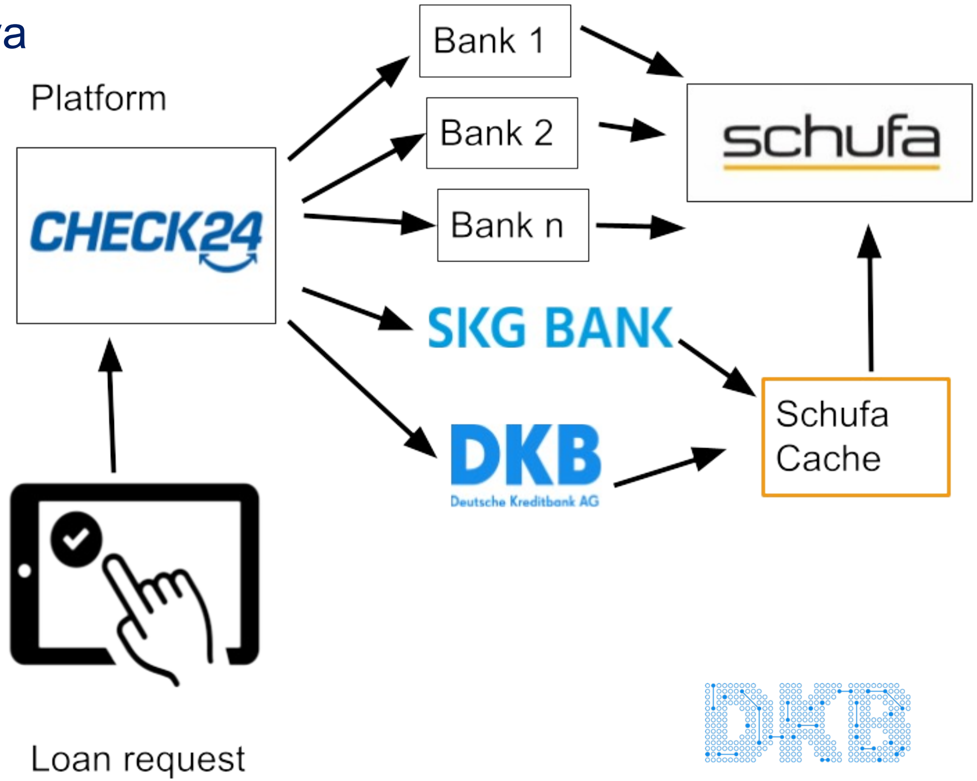
Practical Example: Process Cost Optimization for Private Loans

More than **90% of requests for private loans** come from platforms like Check24, Smava

Platforms call many banks which call the „Schufa“ (German credit score agency) → low take-up rate

High monthly cost for credit reports

Can we predict **with high certainty** which applications are ultimately rejected and **reject before calling the Schufa**?



Practical Example: Process Cost Optimization for Private Loans

Predict: Loan request rejected/accepted

Data:

- Application data from the last n months
- Unknown: credit agency response (existing loans at other banks, credit score, special events like account cancelations)

Input fields:

- Application information, e.g. income, rent, family status, employment
- Process data e.g. current risk configuration, age limits, other configuration
- Past applications by same person
- Other fields: is_customer, platform, type of loan,…

Symptom based monitoring: prioritize backwards from output

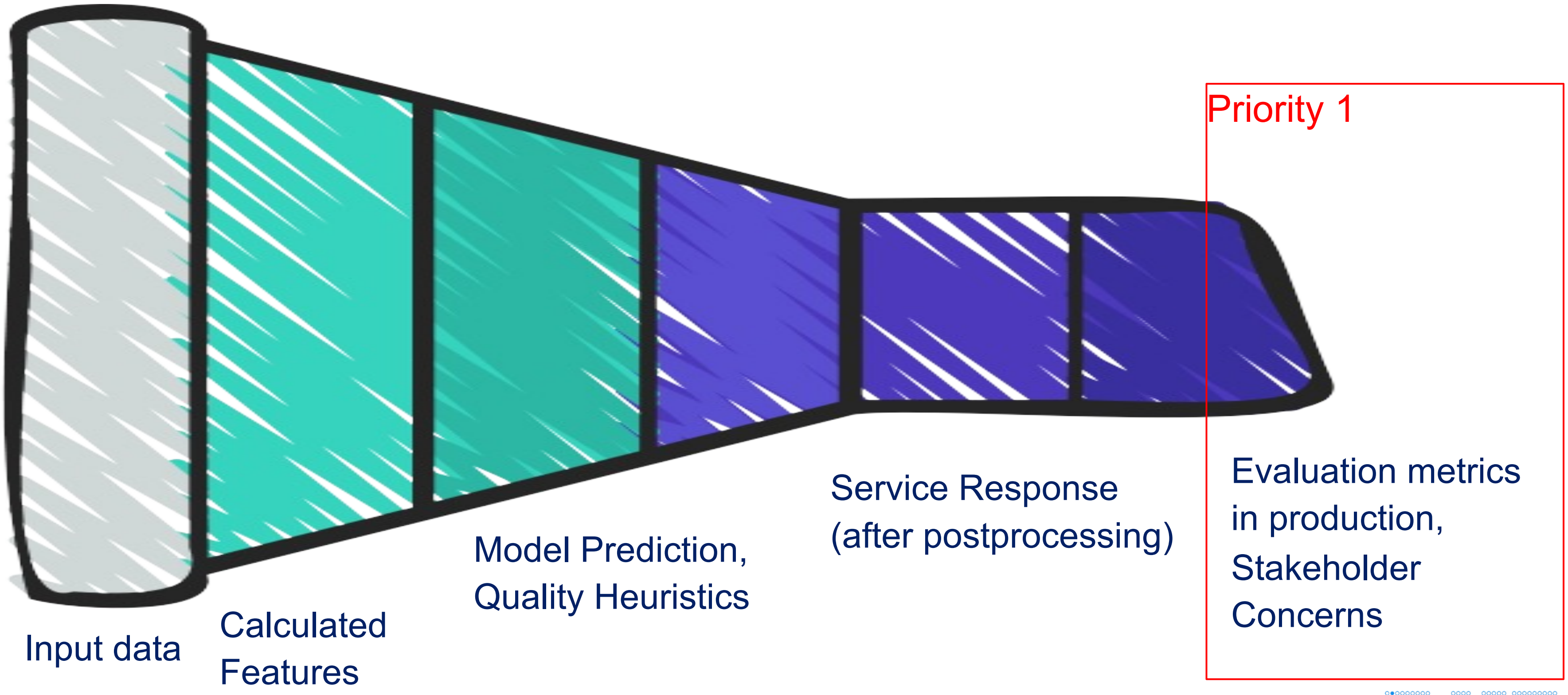
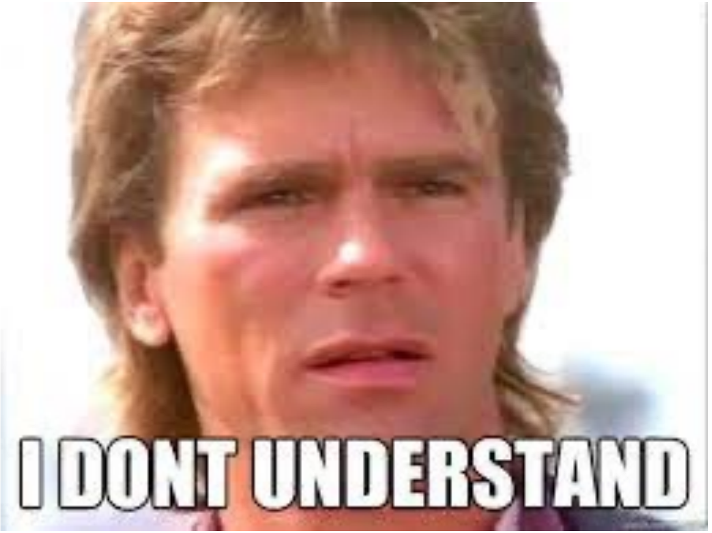


Image source: Flaticon

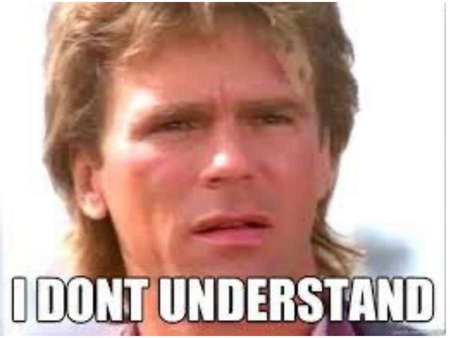
Monitoring Priority 1: Evaluation Metrics in Production

Data Scientist:

> "Can I monitor my evaluation metrics in production?"



Monitoring Priority 1: Evaluation Metrics in Production



Answer: Maybe!

Do you know the correct target close in time? Common problems:

- Unknown result e.g. if an application is rejected because of a high fraud probability, we don‘t know if we made an error
- Delayed result, e.g. if we predict the delivery time for a package we know the true delivery time days later
- Filter bubble effect, e.g. algorithm decides what to show the customer. Unseen options cannot be evaluated

Monitoring Priority 1: Evaluation Metrics in Production

If you can, monitor the evaluation metrics in production

→ Store prediction and target

→ Calculate the metrics used during evaluation, e.g. batch job and or create an endpoint to receive a feedback call

→ Add metrics to dashboard and create an alert

Loan Rejection Prediction:

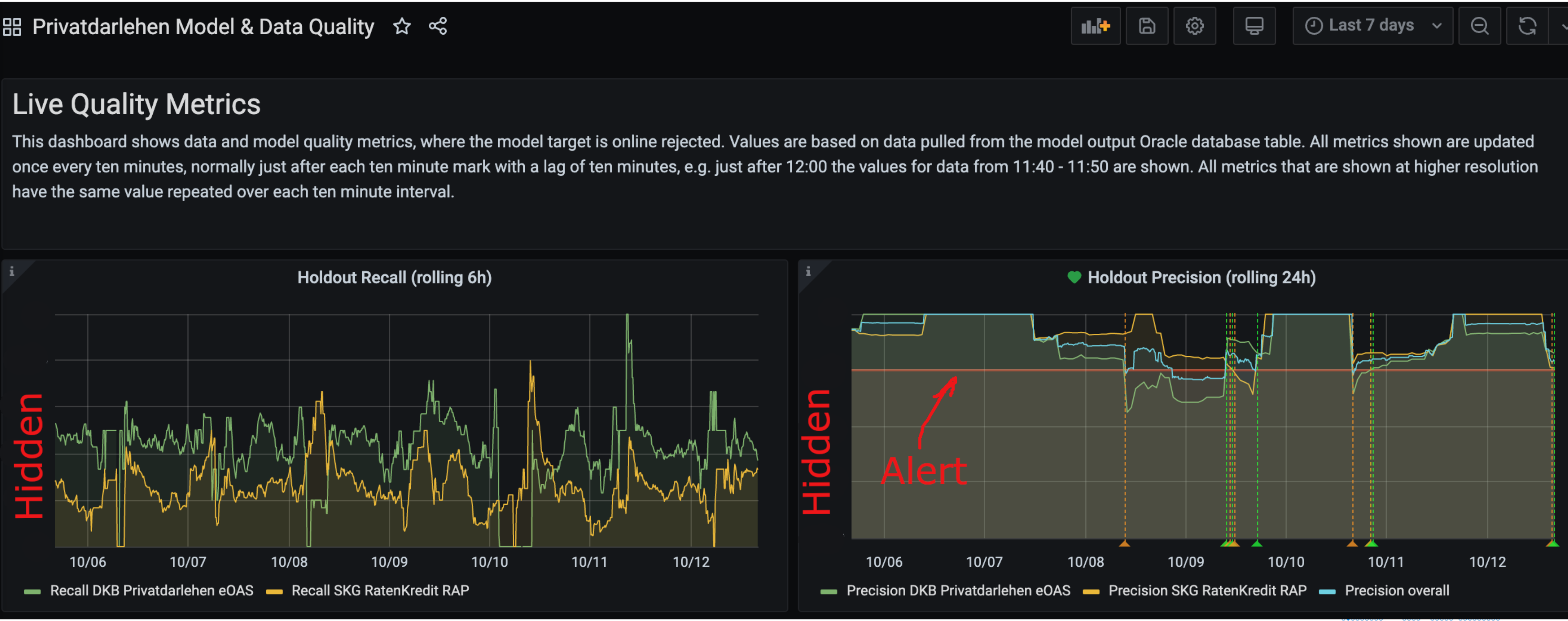
→ If we reject, the correct target is unknown.

→ We create "production evaluation data" where the model is called but not used for decision making

→ For this data we know the prediction and the correct, actual result

→ Calculate the metrics and compare them to the expected performance

Real-Time Dashboard: Evaluation Metrics in Production



Monitoring Priority 1: Stakeholder Fear Signals

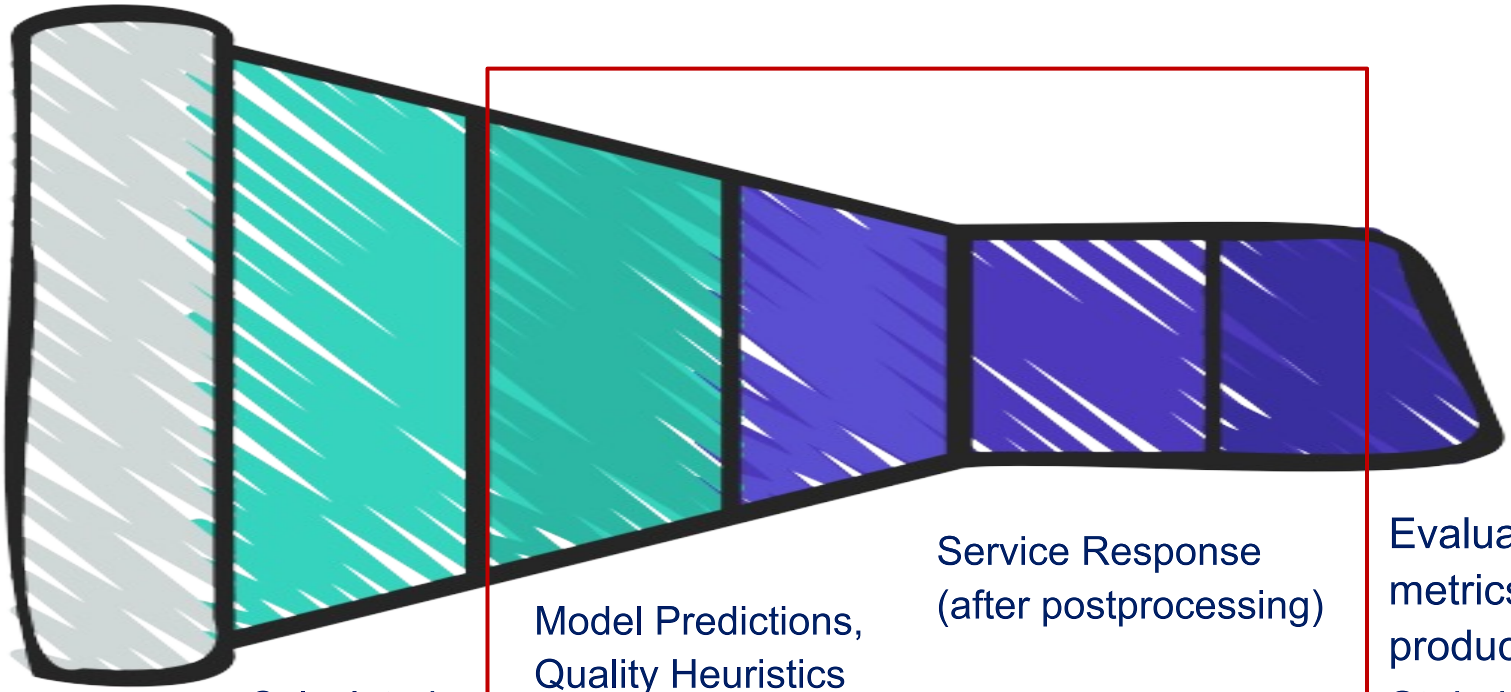
Monitor **what the stakeholders want to avoid**

- Machine Learning Applications need trust → ask stakeholders for their worst-case scenarios, e.g. service makes wrong decision, is uncertain, doesn't answer
- Put these fears into metrics to make sure you would detect these scenarios
- Add metrics to dashboard and alert

Loan Rejection Prediction:

- Fear: unfairly reject applications → alert on precision <95%
- Fear: Make application slow → alert on p95 speed <300 msec

Symptom based monitoring: prioritize backwards from output



Input data

Calculated Features

Model Predictions, Quality Heuristics

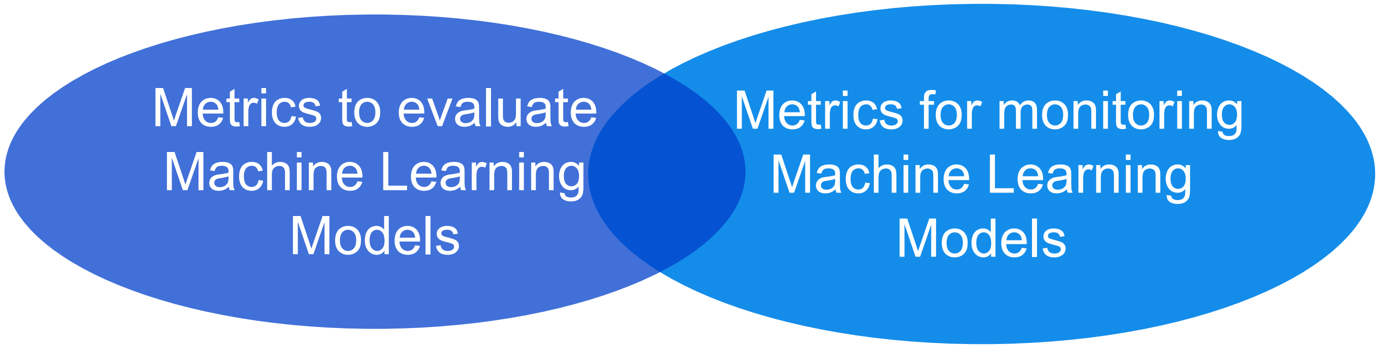
Service Response (after postprocessing)

Priority 2

Evaluation metrics in production, Stakeholder Concerns

Image source: Flaticon

Insight: A lot of Machine Learning Monitoring is done without the evaluation metrics



- Measured to evaluate model quality, e.g. precision, recall, NDCG, …
- To calculate evaluation metrics we compare the prediction against outcome in production
- Often not available or not available close in time

- Measured in order **detect** a problem, not to capture model quality
- Detection Metrics are easier to implement

Monitoring Priority 2: Response distribution

Monitor the response distribution

- monitoring the output is a good „catch all“ technique, needed if you cannot calculate evaluation metrics in production or if there is a delay between prediction and outcome
- Detect slow or sudden shifts of response distribution
- Often easy to do (just one or few outputs)
- The importance of a change is more clear compared to input monitoring (an input field‘s change might be not relevant to the output)

→ Rule Based Distance Metrics: Median, Quantiles, Share of empty/insufficient outputs

→ Statistical distance metrics: Kolmogorov-Smirnov Statistic, D1 Distance, Population Stability Index

Monitoring Priority 2: Response distribution

Example distribution distance metric: D1

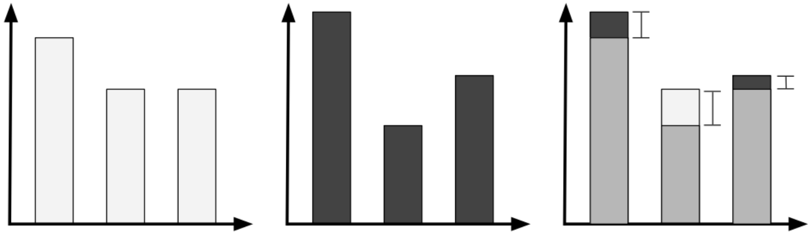


Figure 8: Two distributions, white and black are compared. When overlaying them, the difference can be "seen". The sum of the magnitude of these visible differences is the d_1 distance.

$$d_1(p,q) = \sum_{i=1}^{n} |p_i - q_i|$$

→ Sum of Distances of Probability Density Functions

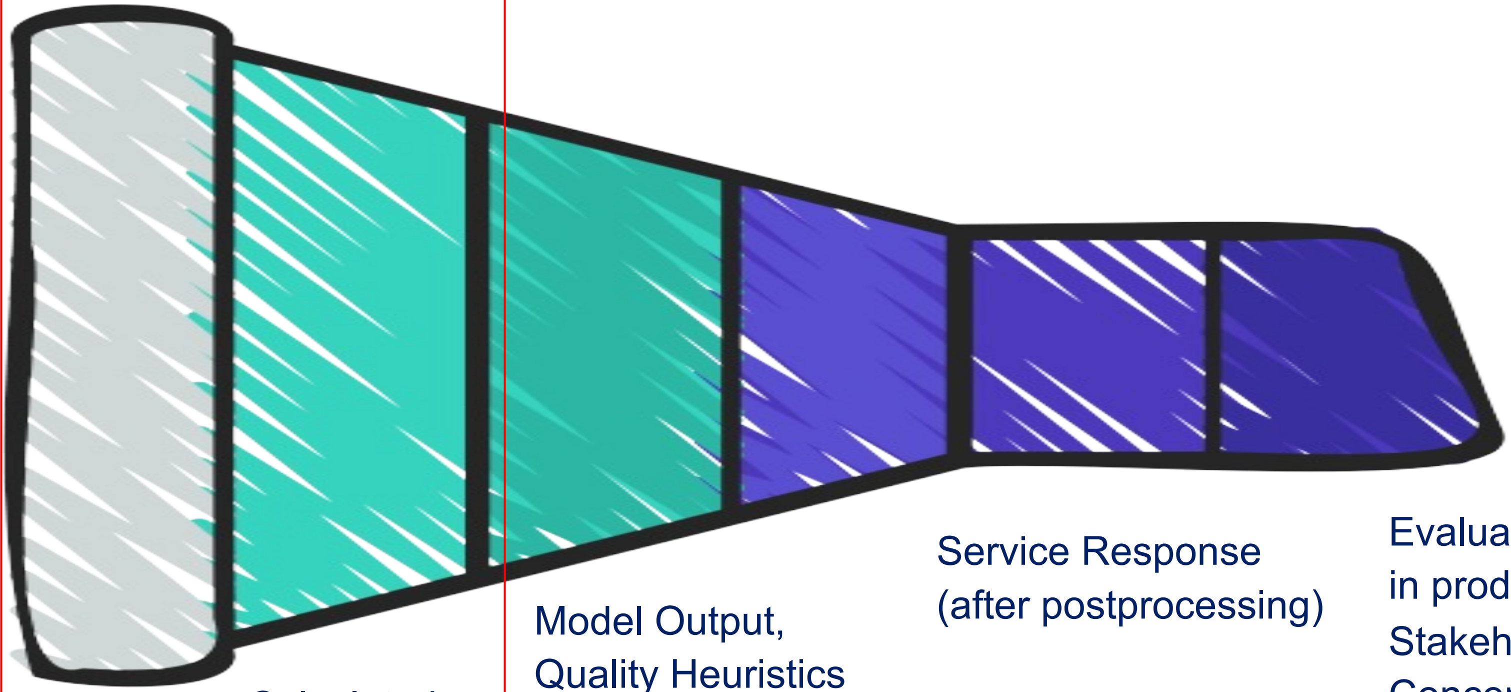
Source: Google Paper: Data Validation for Machine Learning 2019

Monitoring Priority 2: Heuristic Quality Metrics

Heuristic Quality Metrics

- Create use-case-specific, human understandable quality indicators, e.g. heuristic for a „really good“ or „bad“ response and common sense heuristics
- The metric doesn’t have to be a great quality indicator, just go down if quality goes down (do not aim to measure objective quality!)
- E.g.
  - Common-Sense-Metric for a personalized algorithm: Share of personalized responses
  - Bad responses metric: Share of empty responses/fallback responses
  - Common-Sense-Metric for a personalized home page ranking: What is the rank of a user‘s most used carousel?

Symptom based monitoring: prioritize backwards from output



Input data

Calculated Features

Model Output, Quality Heuristics

Service Response (after postprocessing)

Evaluation metrics in production, Stakeholder Concerns

Priority 3

Image source: Flaticon

Monitoring Priority 3: Input and Feature Data distribution

Monitoring Input and feature distribution

- Compare difference between training and serving or train on features you logged (Google Rules of Machine Learning, Rule #29)
- Compare the serving distribution over time: a sudden shift indicates a problem

→ Rule Based Distance Metrics: Median, Quantiles, Share of empty/insufficient inputs

→ Statistical distance metrics: Kolmogorov-Smirnov Statistic, D1 Distance, Population Stability Index

Agenda

Add ML Monitoring

Implement Basic Software Monitoring

Implementation and Tooling

Survey: Do I need an ML Ops Monitoring Tool?

ML Ops Monitoring Tools:

- Focus on monitoring and/or explainability, e.g. aporia, superwise
- Monitoring as part of full featured tool, e.g. Seldon, Sagemaker

Survey under practitioners in the ML Ops Community Slack: "How are your production experiences using dedicated machine learning monitoring tools?"

→ many companies with models in production do not have a custom framework in use

Start simple, re-evaluate later

Pure Data Science Product, starting from scratch or very advanced product

→ evaluate full featured machine learning platforms

Your company offers services for monitoring and alerting, only some of your services are machine learning services

→ use existing tools for metric collection and alerting, e.g. Prometheus, Grafana, a job scheduler

Grafana Prometheus

Monitoring: The stack

Advantages of using existing monitoring and alerting stack:

- No new tool(s) needed
- Immediate start
- Usually sufficient (unless you do a lot of ML debugging)
- Integration with other metrics on same dashboard
- Machine Learning Tools develop fast at the moment → many are not mature yet

Monitoring: Example Implementation

- Add metrics to your inference code:

```
from prometheus_client import Histogram
h = Histogram('model_prediction', 'Model output score distribution')
...
prediction_score = model.predict(request)
h.observe(prediction_score)
```

- For complicated calculations: log response to storage e.g. s3, run a script every 10 mins to calculate (raw) metric components

- Create a dashboard and create alerts

Takeaways

- Monitor golden signals + add machine learning monitoring
- Prioritize monitoring output metrics (user impact!) like response monitoring and if available evaluation metrics in production
- You often don't need a new tool, use the tools you already have and add a few metrics

Talk to me

We are looking for a **Machine Learning Lead Engineer**, feel free to ask my any questions here or in slack.

Join the ML Ops Slack Channel to talk to others working on Machine Learning in production.

BACKUP