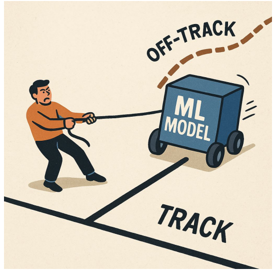
Monitoring

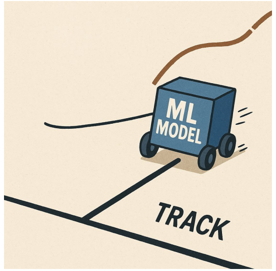
Agenda

1. What is **Monitoring** in MLOps?
2. **Concept** Drift, **Data** Drift, **Model** Drift
3. Intro to Monitoring Tools: **Prometheus**, **Grafana**

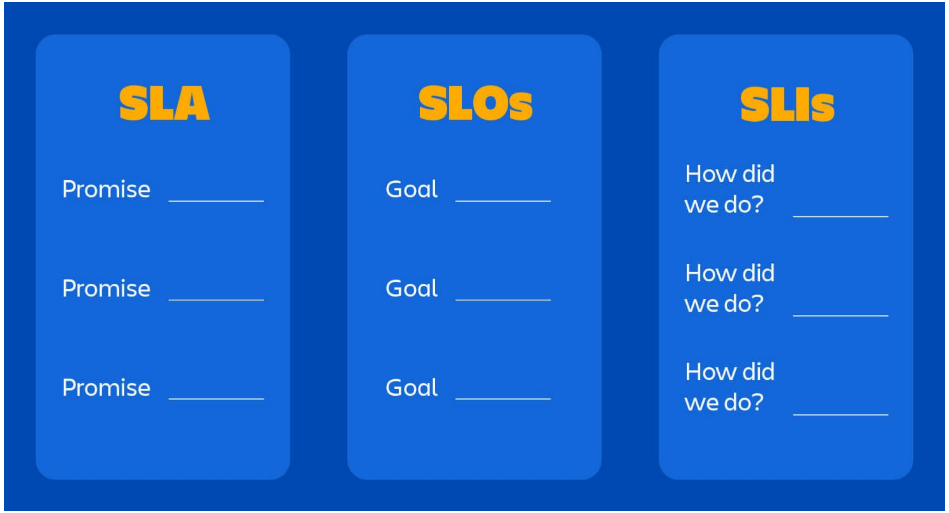
What is Monitoring? Why do we care?

- ML systems don't stay static, data evolves
- You can’t fix what you don’t observe
- SLA, SLO, SLI?





What is Monitoring? Why do we care?



SLA

Promise ________

Promise ________

Promise ________

SLOs

Goal ________

Goal ________

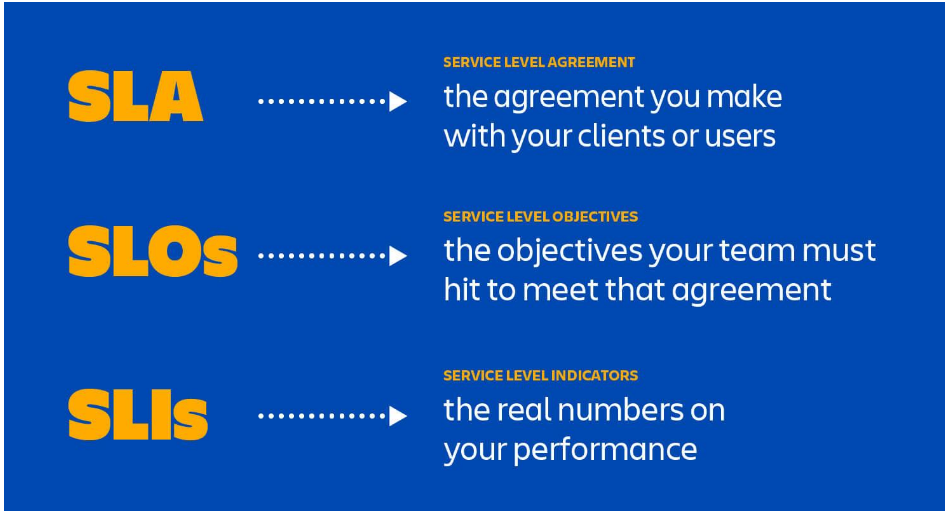
Goal ________

SLIs

How did we do? ________

How did we do? ________

How did we do? ________



SLA → SERVICE LEVEL AGREEMENT
the agreement you make with your clients or users

SLOs → SERVICE LEVEL OBJECTIVES
the objectives your team must hit to meet that agreement

SLIs → SERVICE LEVEL INDICATORS
the real numbers on your performance

What to Monitor?

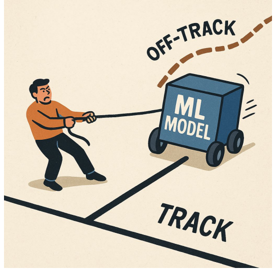
- Model Metrics
    - Are the predictions accurate? => Model outputs
- System Metrics
    - Is it working? => Latencies, Memory, CPU usage
- Resource Metrics
    - Is the data what is expected? => Model Inputs

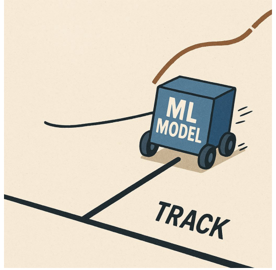
Drift Happens: Monitoring in MLOps

Concepts, Chaos, and Catching Your Models Misbehaving

Real-world Example: The Self-Driving Car That Mistaked a Moon for a Yellow Light

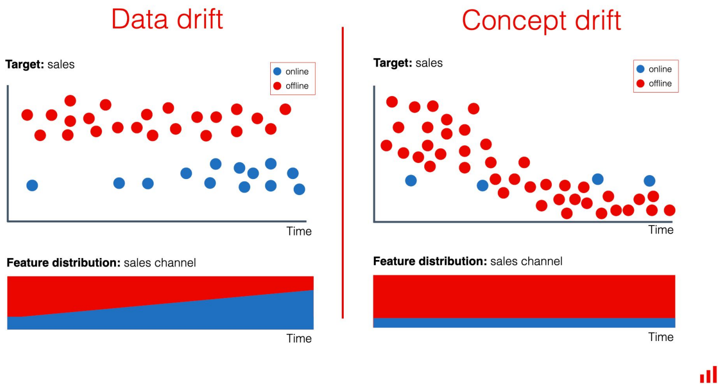
- Story: In 2021, Tesla’s Autopilot misinterpreted a bright full moon as a yellow traffic signal, causing unnecessary slowdowns. (Source: AI Incident Database)
- Moral: If your model only learns from streetlights, it might brake for the moon.

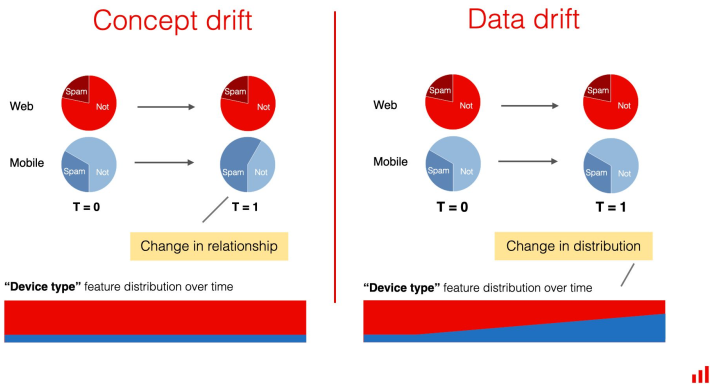




Meet the Drifts

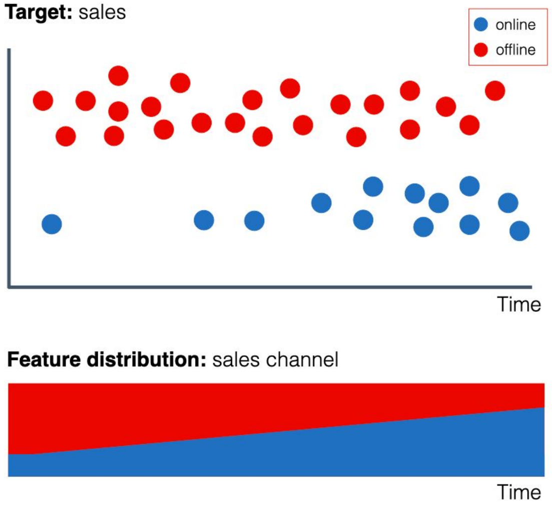
1. Concept Drift: When the relationship between input and output changes
   - Ex: Users start clicking more on videos at night than day
2. Data Drift: When input data distribution changes
   - Ex: Feature values shift due to seasonality or new users
3. Model Drift: Degradation in performance over time
   - Ex: Accuracy drops silently, no one notices... until it’s too late



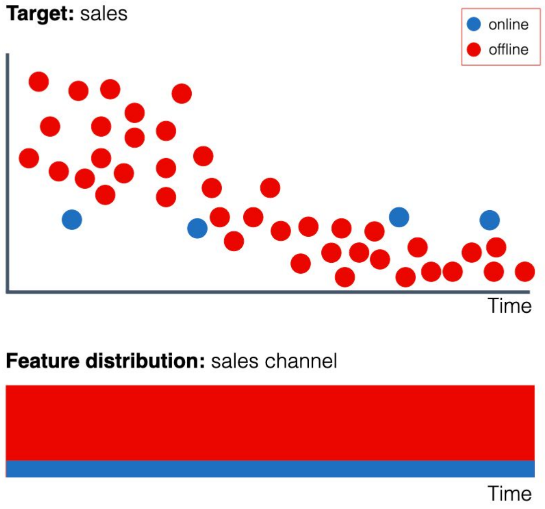


Data vs Concept Drift

Data drift

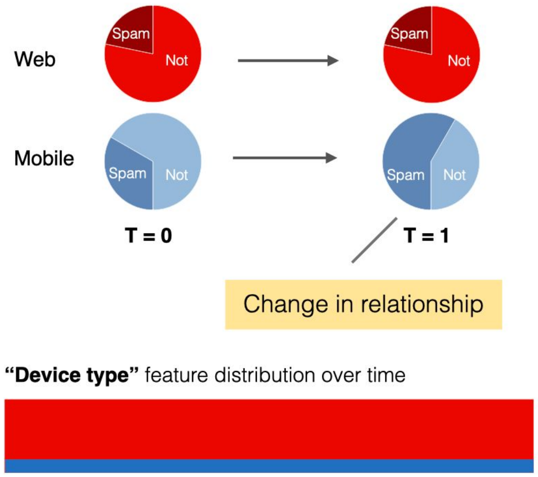


Concept drift



Data vs Concept Drift

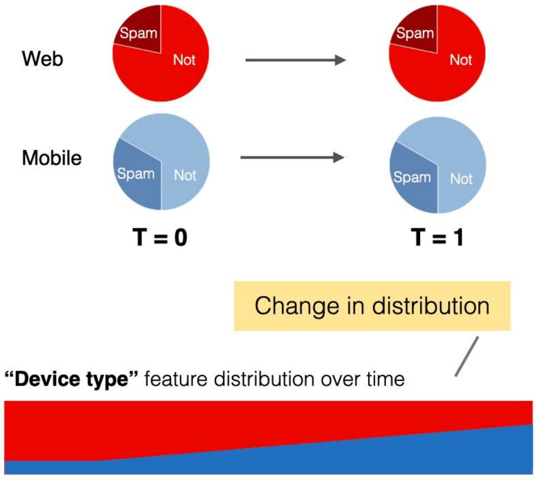
Concept drift



Change in relationship

“Device type” feature distribution over time

Data drift



Change in distribution

“Device type” feature distribution over time

Back to the tesla examples

What type of drift is it?

Story: In 2021, Tesla’s Autopilot misinterpreted a bright full moon as a yellow traffic signal, causing unnecessary slowdowns.

Discussion Prompt

- What kind of drift do you think is most dangerous?
- Can one drift cause another?

Game Time!!

Which of these is what?

TODO: [Quizizz Link]

Triggers of ML Model Drift

Real Change

- Label or feature distribution changes
  e.g. product launch in a new market
- Concept can change
  e.g. competitor launching a new service

May require a new model

Data Integrity Issues

- Correct data enters at source but faulty data engineering.
  e.g. debt-to-income values & age values are swapped in the input
- Incorrect data enters at source
  e.g. due to a front-end issue, a website form accepts leaving a field blank.

What to do?

- Detect , how? Do I have labels?

1. Model-Based Detection: Use the model itself to detect performance degradation.
2. Statistical Tests: Compare distributions of input features over time.

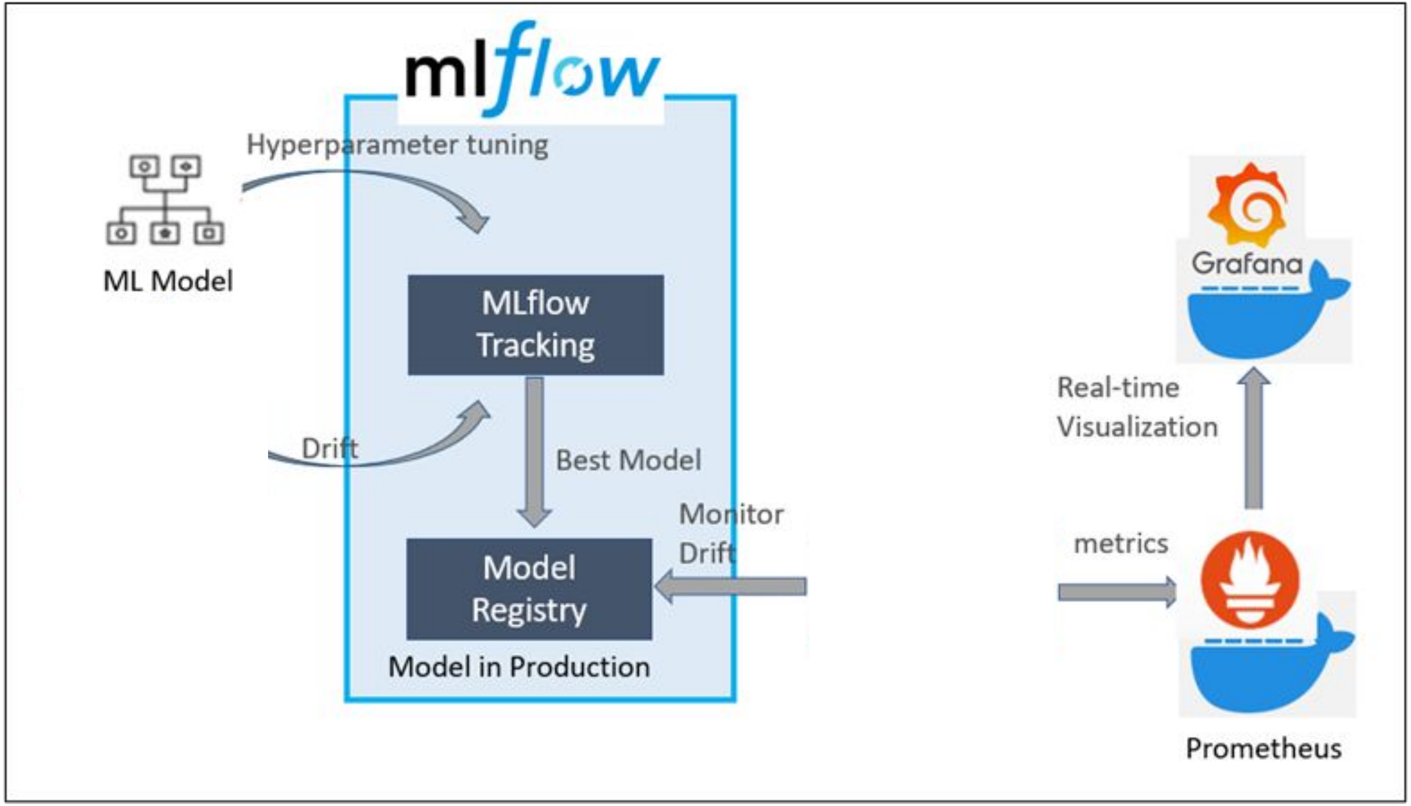
- Analyze
- Get to the root cause:
  - **Data Integrity:** Integrity & Outlier Monitoring
  - **Drift Analytics:** What’s Important & Why
- Resolve
  - Update Pipeline
  - Retrain/Adapt model

Statistical Tests

Metric	Drift Type	Measures	Symmetric?	Good For
K-S Test	Data Drift	Max CDF distance	Yes	Continuous features
PSI	Data Drift	Distribution shift in bins	No	Business monitoring
Page-Hinkley	Concept Drift	Change in metric mean	N/A	Model error detection over time
KL Divergence	Data Drift	Info loss between distributions	No	Distributional changes
JS Divergence	Data Drift	Smoothed, symmetric KL	Yes	Stable comparison of distributions

Demos

Tool Time: Monitoring Stack Intro

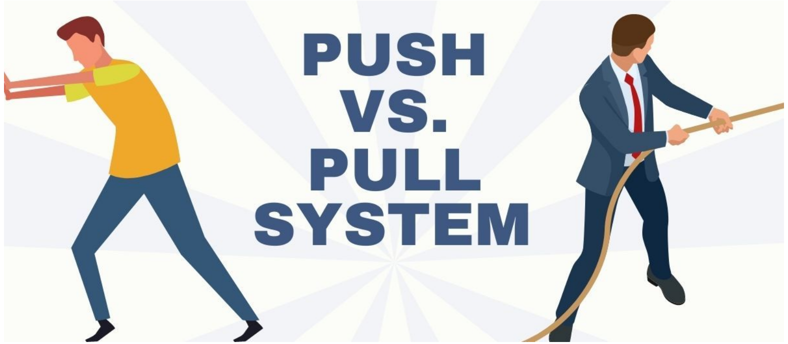


Tool Time: Monitoring Stack Intro

- **Prometheus**: Collects and stores time-series metrics (like accuracy, latency) from your system and services. *Think of it as the data collector.*
- **Grafana**: Visualizes metrics from tools like Prometheus using *customizable dashboards*, perfect for spotting trends, issues, or just making things look cool.

Prometheus Components

- Server
- Metric Storage Database
- User Interface
- API
- Alerting (alertmanager)
- Query Language (PromQL)



Prometheus is primarily pull-based (scraping) but it can have a push gateway

How Prometheus Collects Data

- Prometheus pulls metrics over HTTP from targets
- Targets expose metrics at /metrics endpoint
- Example: http://myapp:9100/metrics

prometheus.yml

```
global:
  scrape_interval: 15s  # How often to scrape targets
  evaluation_interval: 15s  # How often to evaluate rules

scrape_configs:
  # Scrape a custom application exposing metrics
  - job_name: 'my_app'
    static_configs:
      - targets: ['localhost:8080']
```

Prometheus Metric Types

- A **metric** = a measurable piece of information

- **Counter:** Can only go up or reset back to 0. (e.g., total HTTP requests)
- **Gauge:** Go up, down, reset. (e.g., memory usage)
- **Summaries & Histograms**. (e.g., request duration buckets)
  (e.g., quantiles of durations)

PromQL

/metrics

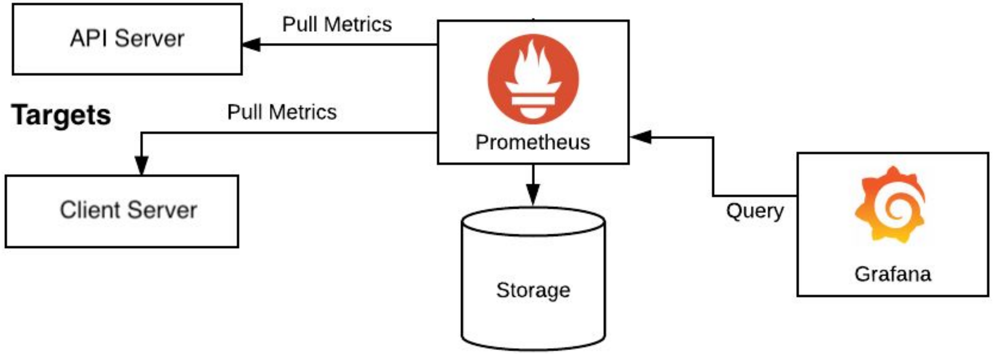
```
# HELP model_accuracy Current model accuracy
# TYPE model_accuracy gauge
model_accuracy{model="resnet50"} 0.89
```

PromQL

```
avg_over_time(model_accuracy[5m])
```

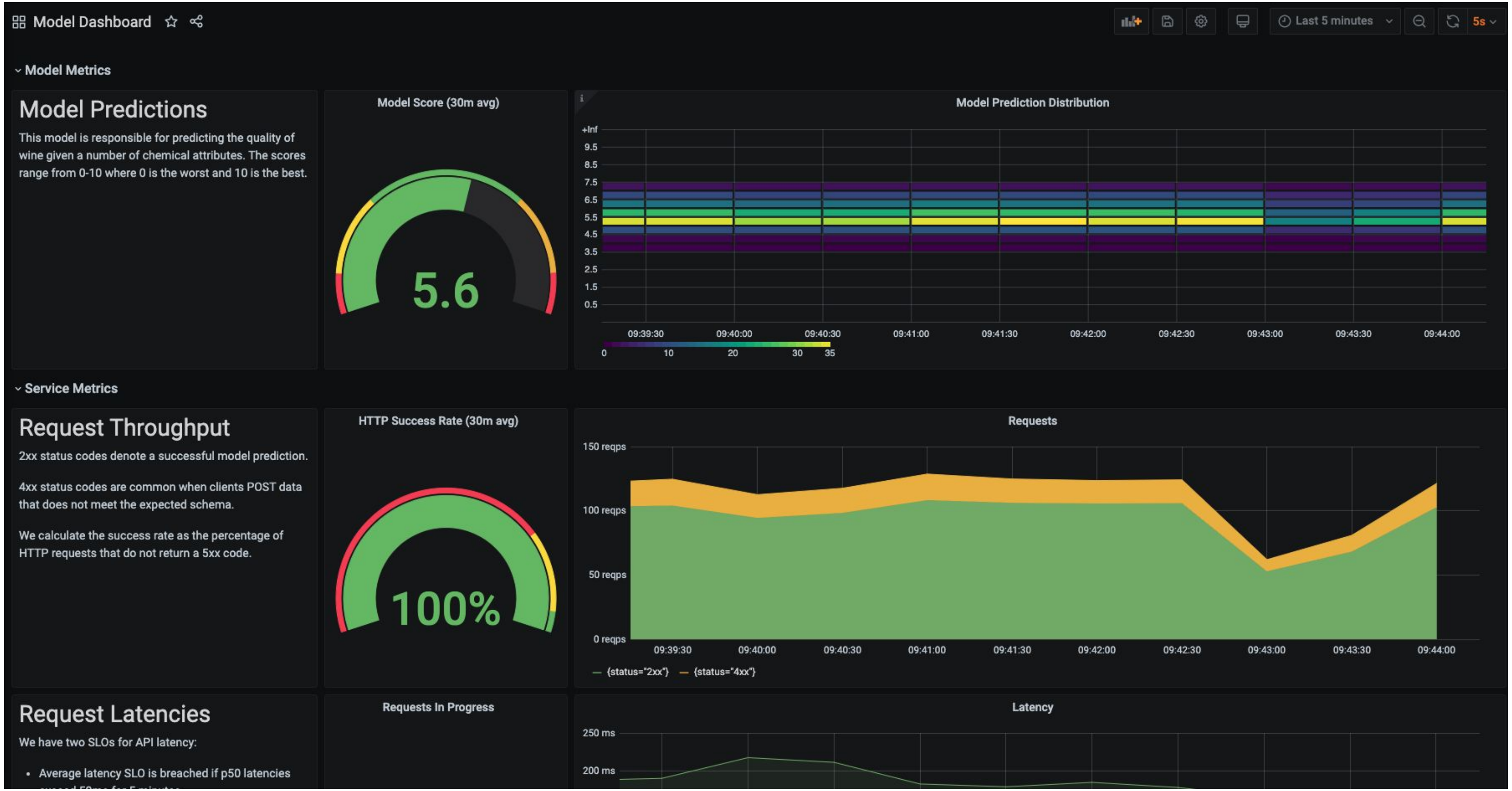
Grafana

- Grafana queries Prometheus.
- Shows you beautiful dashboard.
- **Categories:**
  - Metrics
  - Logs



Demo

Grafana



Model Predictions

This model is responsible for predicting the quality of wine given a number of chemical attributes. The scores range from 0-10 where 0 is the worst and 10 is the best.

Request Throughput

2xx status codes denote a successful model prediction.

4xx status codes are common when clients POST data that does not meet the expected schema.

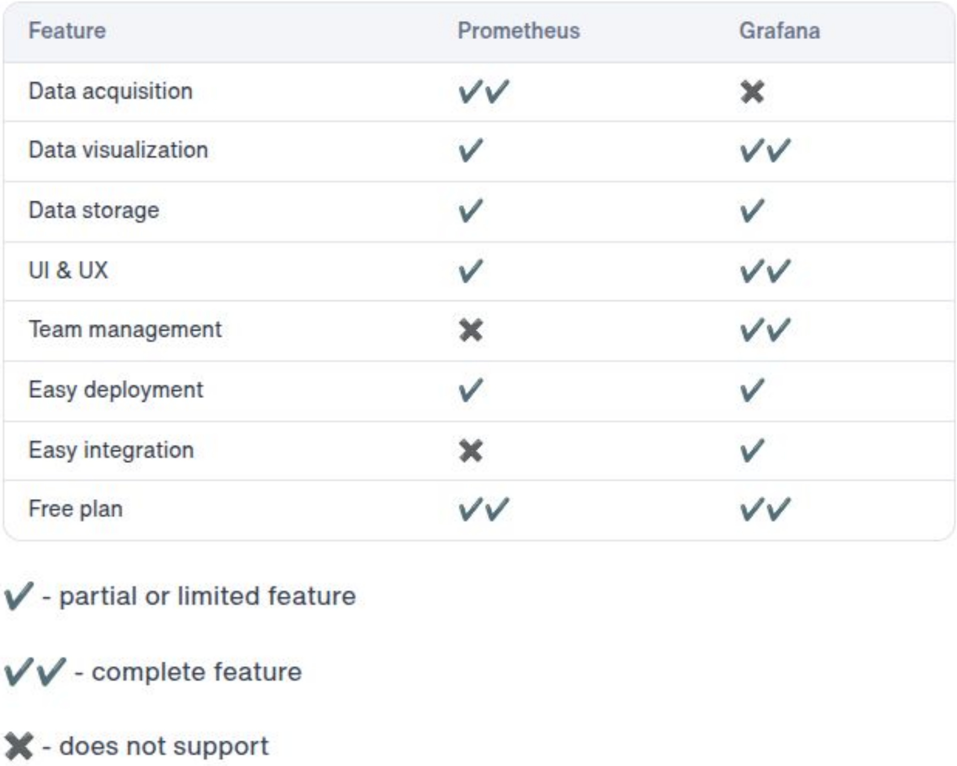
We calculate the success rate as the percentage of HTTP requests that do not return a 5xx code.

Request Latencies

We have two SLOs for API latency:

- Average latency SLO is breached if p50 latencies

Comparison



Feature	Prometheus	Grafana
Data acquisition	✓✓	✖
Data visualization	✓	✓✓
Data storage	✓	✓
UI & UX	✓	✓✓
Team management	✖	✓✓
Easy deployment	✓	✓
Easy integration	✖	✓
Free plan	✓✓	✓✓

✓ - partial or limited feature

✓✓ - complete feature

✖ - does not support

Alerting

Generally it's recommended to use the Prometheus internal alerting. Grafana alerting, while "easy", has a number of downsides.

- It's hard to make HA, since it depends on an instance of Grafana server.
- It doesn't have any way to de-duplicate if you run multiple Grafana servers.
- It's dependent on networking between Grafana and Prometheus, making it less reliable than Prometheus internally executed rules.

Visualizing

Generally it's recommended to use the Prometheus internal alerting. Grafana alerting, while "easy", has a number of downsides.

- It's hard to make HA, since it depends on an instance of Grafana server.
- It doesn't have any way to de-duplicate if you run multiple Grafana servers.
- It's dependent on networking between Grafana and Prometheus, making it less reliable than Prometheus internally executed rules.

Lab

- User docker to:
  - Expose your FastAPI app
  - Expose Prometheus
  - Expose Grafana
- Build a nice dashboard
  - Explain in a comment in the assignment why you chose those metrics & what it means