NETDATA

Practical AI with Machine Learning for Observability in Netdata

Costa Tsaousis

AI for observability is tricky

Google:

All of Our ML Ideas Are Bad

(and We Should Feel Bad)


:URL

Wednesday, 2 October, 2019

Todd Underwood, Google

The vast majority of proposed production engineering uses of Machine Learning (ML) **will never work**. They are structurally unsuited to their intended purposes. There are many key problem domains where SREs want to apply ML but most of them do not have the right characteristics to be feasible in the way that we hope. After addressing the most common proposed uses of ML for production engineering and explaining why they won't work, several options will be considered, including approaches to evaluating proposed applications of ML for feasibility. **ML cannot solve most of the problems most people want it to**, but it can solve some problems. Probably.

Question 1

Hey AI,
how is my infrastructure today?

Question 1: typical AI's plan

Hey AI, how is my infrastructure today?

AI's plan:

1. Get raised alerts.
2. Get top nodes by CPU, memory, disk pressure.
3. Get the last few error logs.
4. If you are lucky: attempt to discover metrics about errors or latency, pick a few of them and make a couple of queries in hope they will show something.
5. Conclude.

The problem:
AI does not have a context (what is important for you), and does not have any means to surface infrastructure level issues easily, apart from alerts.

Question 1: Netdata's Answer

Hey AI, how is my infrastructure today?

Netdata provides 2 tools, that require from the LLM to know just the time-window of interest:

1. **find_anomalous_metrics**
   Netdata trains 18 ML models per metric, at the edge, providing ~54 hours of rolling behavioral patterns.
   Anomaly detection occurs in real-time during data collection. A data point is flagged as anomalous only when all 18 models reach consensus. The result is ordered by anomaly rate.

2. **find_correlated_metrics**
   Score metrics based on how much they changed between two time-windows, using either KS2 or Volume.

Both of these tools return an ordered list of metrics, and the LLM can select how many it wants.

Design of ML in Netdata

Netdata ML Anomaly Detection Overview

Netdata trains ML at the edge, on each server!

The Challenge

- Netdata monitors 3k - 10k metrics per server
- Collects metrics in real-time, per-second

The Goal

- Detect genuine anomalies without drowning in false positives
- No manual configuration or training required
- No more than 2-5% CPU utilization of a single core
- No more than a few KiB per metric

The Solution

- Unsupervised k-means clustering (k=2)
- 18 models per metric trained on different time windows
- Consensus mechanism: ALL models must agree
- Min-max normalized anomaly scoring

How Individual Model Detection Works

Real-time anomaly detection!

Training a new model (every 3 hours)

- Take 6 hours of high-resolution metric data (per-second)
- Create feature vectors (differenced, smoothed, 5 lags)
- K-means clustering finds 2 behavior patterns
- Record min and max distances from cluster centers

Anomaly Detection per model (real-time)

- Collect raw metric value
- Apply differencing → smoothing → lagging to create 6D feature vector
- Calculate average Euclidean distance to the 2 cluster centers
- Anomaly Score = 100 × (distance - min) / (max - min)
- If score ≥ 99 → Model says "anomalous"

Key: This is **not a percentile**! It's normalized distance relative to training extremes.

The Consensus Mechanism

18 models consensus!

18 Models Per Metric

- Each trained on 6-hour windows
- Staggered at 3-hour intervals
- Covers ~57 hours of patterns

For a point to be anomalous

- Model 1 (trained on hours 0-6): ✓ Anomalous
- Model 2 (trained on hours 3-9): ✓ Anomalous
- Model 3 (trained on hours 6-12): ✓ Anomalous
- ... ALL 18 models must agree!

Result: Random false positives virtually eliminated!

Why This Achieves Ultra-Low False Positives

Without any false positives!

- **Self-adapting thresholds**
  Each metric's "normal" is based on its own historical patterns
- **Multi-timescale validation**
  Short-term spikes/dives must be anomalous across ALL time windows to be flagged anomalous
- **Extreme outlier focus**
  Only flags behavior at or exceeding historical maximums

Real-world result:

- NOT 1% false positive rate
- 1% false positive per model ^18 models = 10^-34% or: 0.00000000000000000000000000000001% false positives per metric

Host Level Anomaly Detection Events

Reliable host level anomalies!

What is anomaly_detection.detector_events?

- A special chart that tracks host-level anomaly events
- Created for each host running ML anomaly detection
- A state chart: 0 or 1
- Triggers when ≥1% of host metrics get anomalous **together**

What 1% Threshold Means in Practice

- Using the binomial probability
- n = 3,000 metrics
- p = 10^-36 (probability of one metric being a false positive)
- k = 30 (need at least this many)

$P(X \geq 30) \approx (3000 \text{ choose } 30) \times (10^{-36})^{30}$

The probability is: 10^{-1050} (in other words: really a lot of zeros after the dot :)

Host-level false positives are not just "extremely unlikely"
They're mathematically impossible in the lifetime of the universe.

And this how it looks

On a real outage

Up to 5% of host metrics anomalous simultaneously

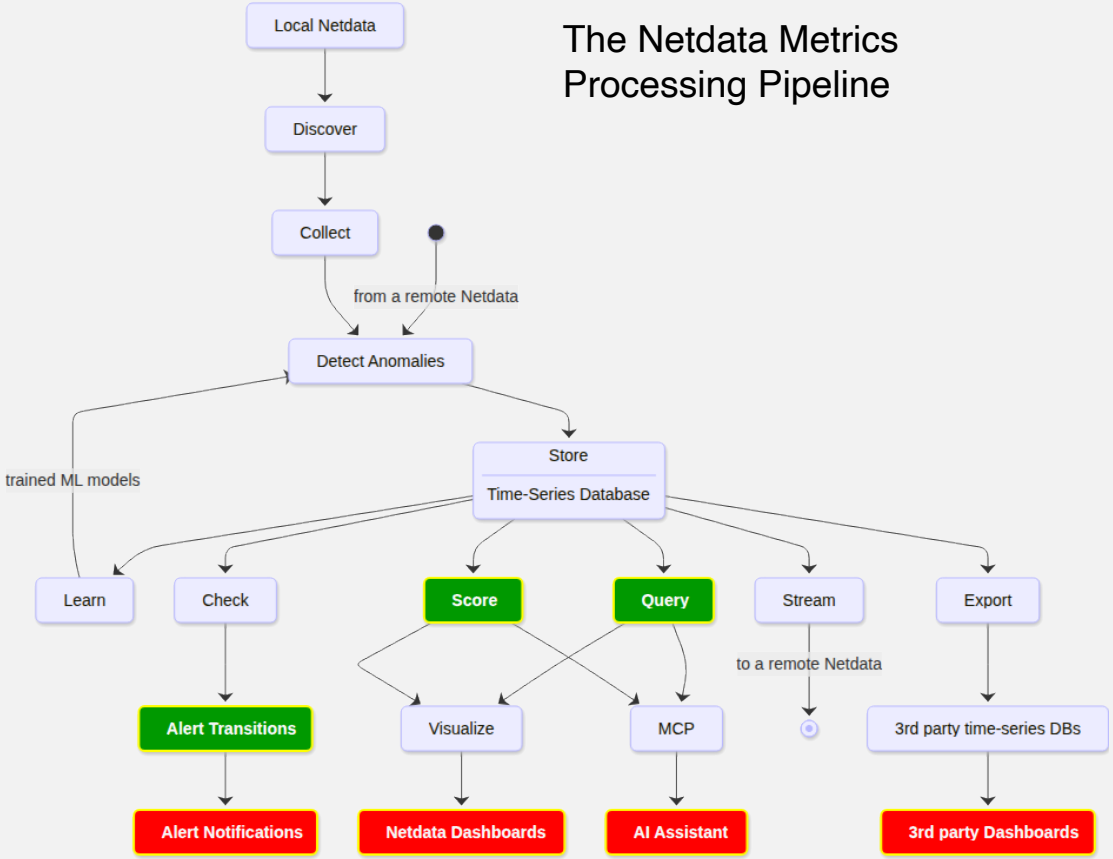
Up to 1500 concurrently anomalous metrics per host

Host level anomalies identified

ML is trained at the edge!

The Metrics processing pipeline in Netdata, automatically trains ML models and detects anomalies.

The Netdata Metrics Processing Pipeline



ML CPU Utilization

Sliced training and careful consideration of the metrics that benefit from ML, allows Netdata to be lightweight.

- **Many metrics are usually zero.**
  Like hardware errors, rare exceptions, etc. These are usually covered by alerts that check for non-zero values. So, all such metrics do not need to be trained.
- **Other metrics are usually constant.**
  Like the memory size on a server, or the pool of connections size of a database server. These metrics do not need to be trained either.
- **Sliced training for the rest.**
  For the rest of the metrics, we need to train a model every 3 hours.

Netdata usually trains 1-4 metrics per second.
At scale, Netdata needs ~1 CPU core, for training ML models and detecting anomalies, for **500k metrics/s**.

ML Memory Footprint

Remarkably efficient memory footprint for real-time ML on thousands of metrics

- **Sliding Window Buffer**
  9 × 8 bytes (double) = 72 bytes
- **K-means Models**
  18 models per metric, 120 bytes per model
  Total for 18 models: 18 × 120 = 2,160 bytes
- **Feature Vector Buffer**
  ~72 bytes
- **K-means Working Structure**
  2 × 48 bytes + metadata = ~152 bytes

Total: 2.4KiB per metric

  - For 3k metrics/server = 7.2 MiB
  - For 10k metrics/server = 24 MiB

ML Storage Footprint

Netdata stores anomalies together with the samples, so anomaly based queries are possible.

- **The anomaly bit is stored in the db.**
  Anomaly information is stored in the database together with each sample collected.

  We developed a **custom floating point number**, which includes the anomaly bit (much like IEEE 745 stores the sign of floating point numbers), ensuring that there is **no storage overhead at all**.

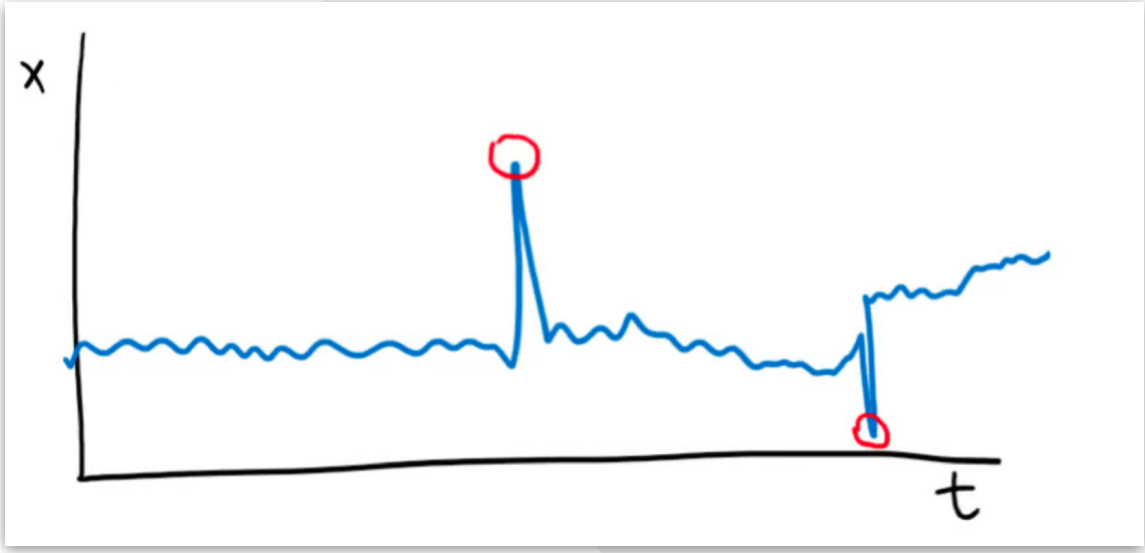
- **Anomaly rate is calculated on the fly.**
  The Netdata query engine calculates the anomaly rates for all metrics, on the fly, in one go.

- **Aggregated anomaly rate.**
  The Netdata query engine calculates aggregated anomaly rates when combining multiple metrics in the same query, providing a high level anomaly rate for each chart.

What ML can Detect?

What it can detect? (1/5)

Point Anomalies or Strange Points: Single points that represent very big or very small values, not seen before (in some statistical sense).

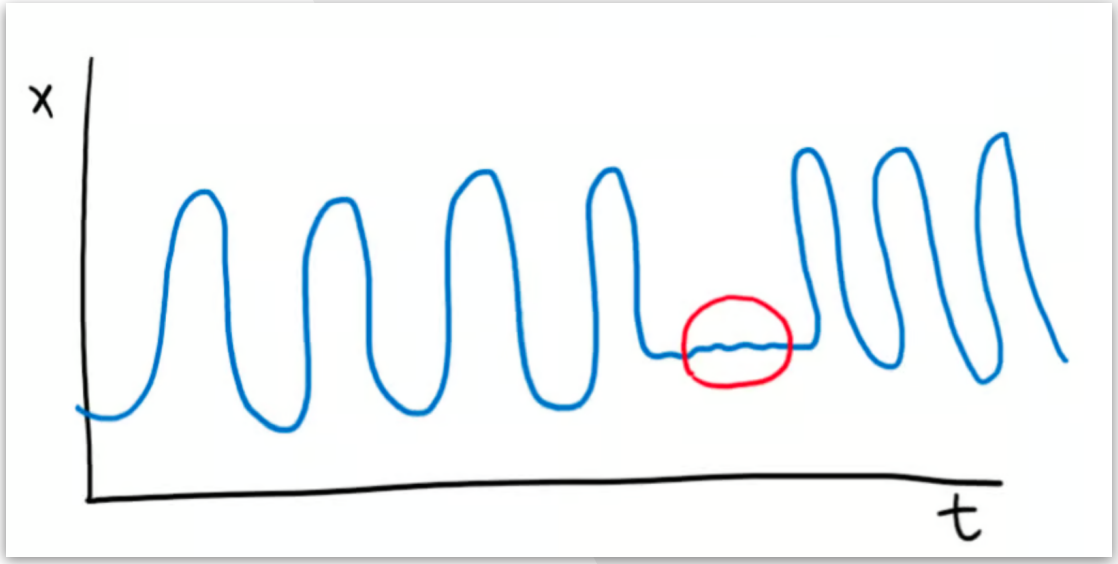


Examples:

- A sudden, extreme spike in the number of failed transactions for your database server.
- An unexpected, moment of high CPU utilization or sudden memory spike for your application server.

What it can detect? (2/5)

Contextual Anomalies or Strange Patterns: Not strange points in their own, but unexpected sequences of points, given the history of the time-series.

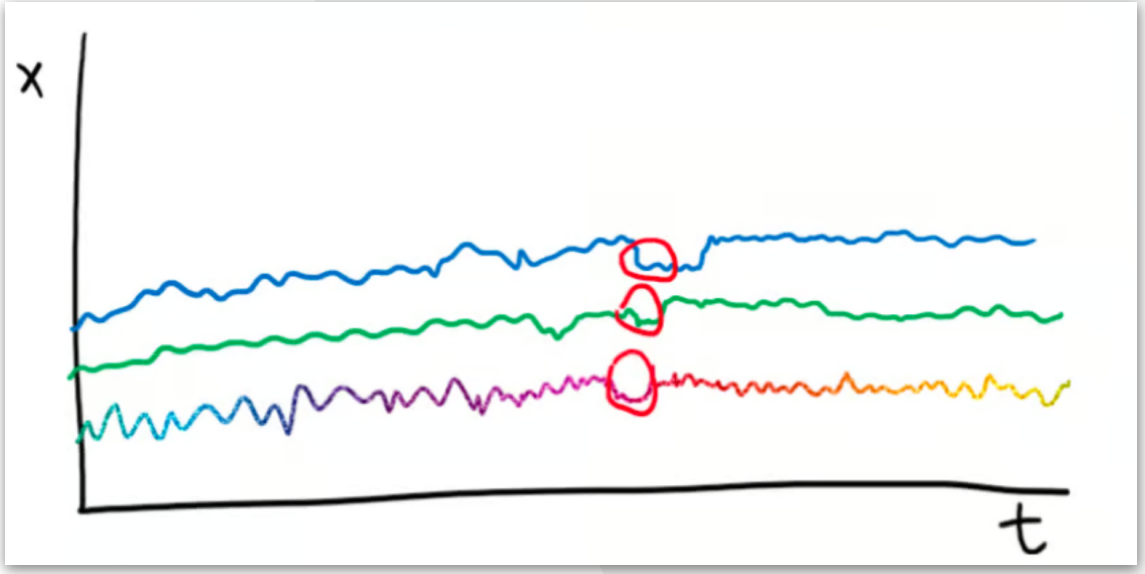


Examples:

- A regular database job, or a backup that did not run.
- A cap on the number of web requests received.

What it can detect? (3/5)

Collective Anomalies or Strange Multivariate Patterns: Neither strange points nor strange patterns, but in global sense something looks off.

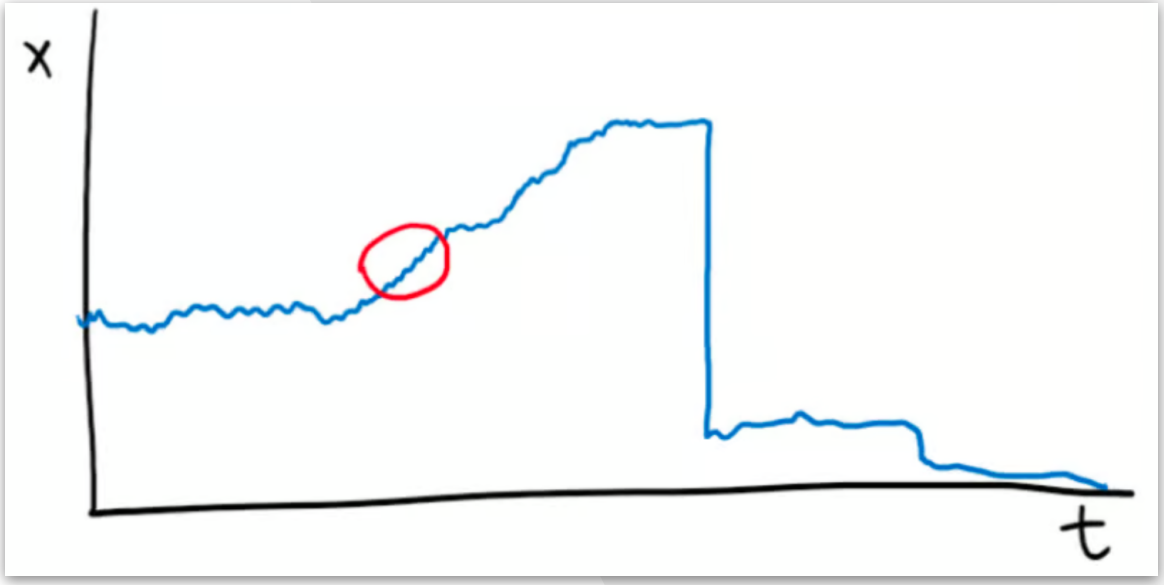


Examples:

- A network issue that introduces a lot of retransmits, lowers the throughput of the web server or the workload on the database server.

What it can detect? (4/5)

Concept Drifts or Strange Trends: A slow and steady drift to a new state.

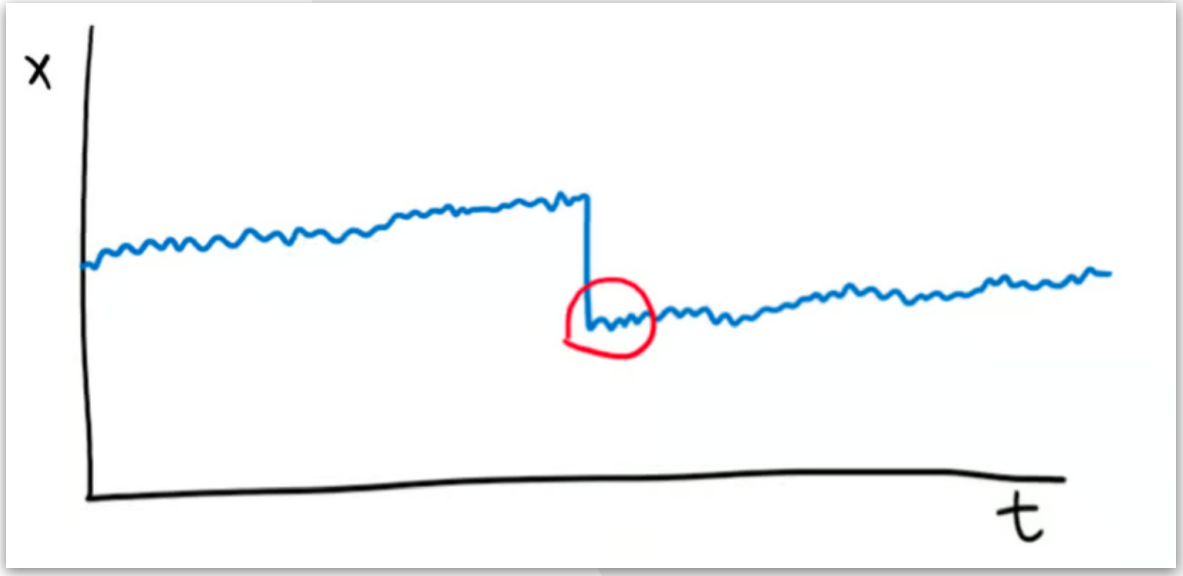


Examples:

- A memory leak in an application.
- An attack that is gradually increased to its full load.
- A gradual increase in response time latency.

What it can detect? (5/5)

Change Point Detection or Strange Step: A shift occurred and gradually a new normal is established.



Examples:

- A faulty deployment that does not serve all the workload.

What ML cannot Detect?

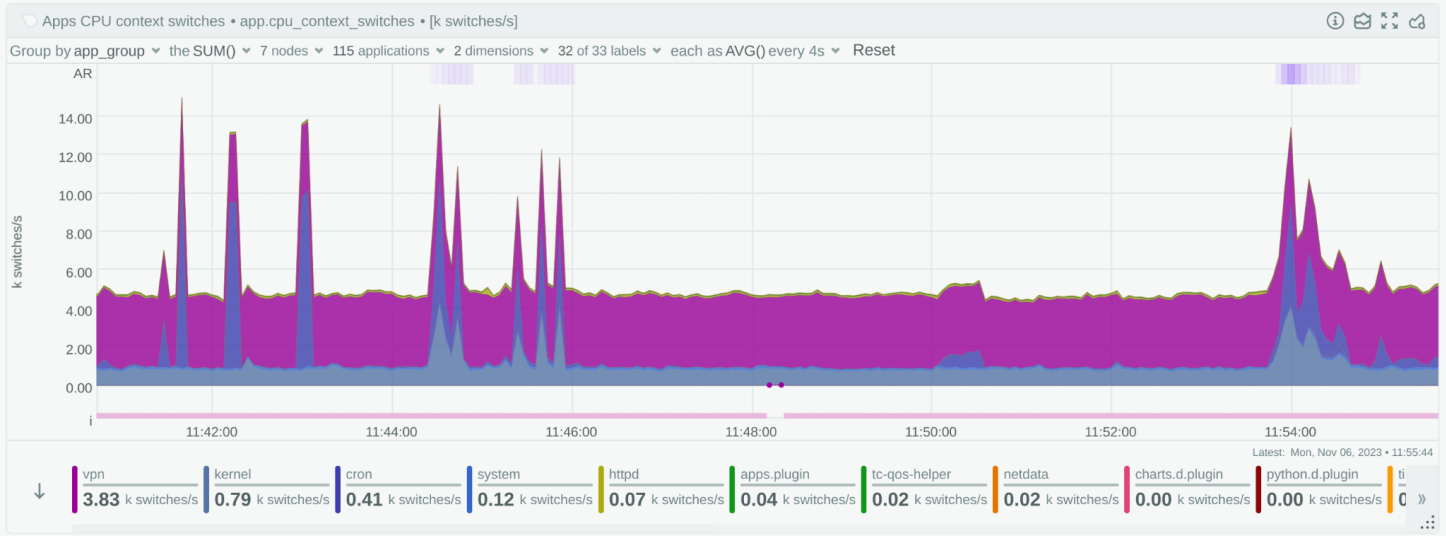
ML Limitations & Trade-offs

It's a powerful **first-line detection** system but needs complementary approaches for complete coverage

- **Short lived workloads**
  CI/CD jobs, cron jobs, short lived containers: the system may not get a chance to train models for them.
- **Crashed services**
  No data = no anomaly detection on the failed services. But it will detect anomalies in dependent services.
- **Slow degradation below radar**
  Models may adapt incrementally to slow degradation. "Boiling frog" problem - never crosses anomaly threshold.
- **Constant metrics that suddenly change**
  They have been excluded from training to optimize performance. This includes binary/state metrics.
- **Learns anomalies too**
  An anomaly becomes “normal” in 3 hours and again “anomaly” in 57 hours.

Integrated Everywhere

A Netdata Chart



:Netdata Parent Demo URL

Netdata Cloud Live Demo URL:

A Netdata Chart - controls

NIDL Controls - review data sources and slice/filter them
(NIDL = Nodes, Instances, Dimensions, Labels)

Aggregation across metrics

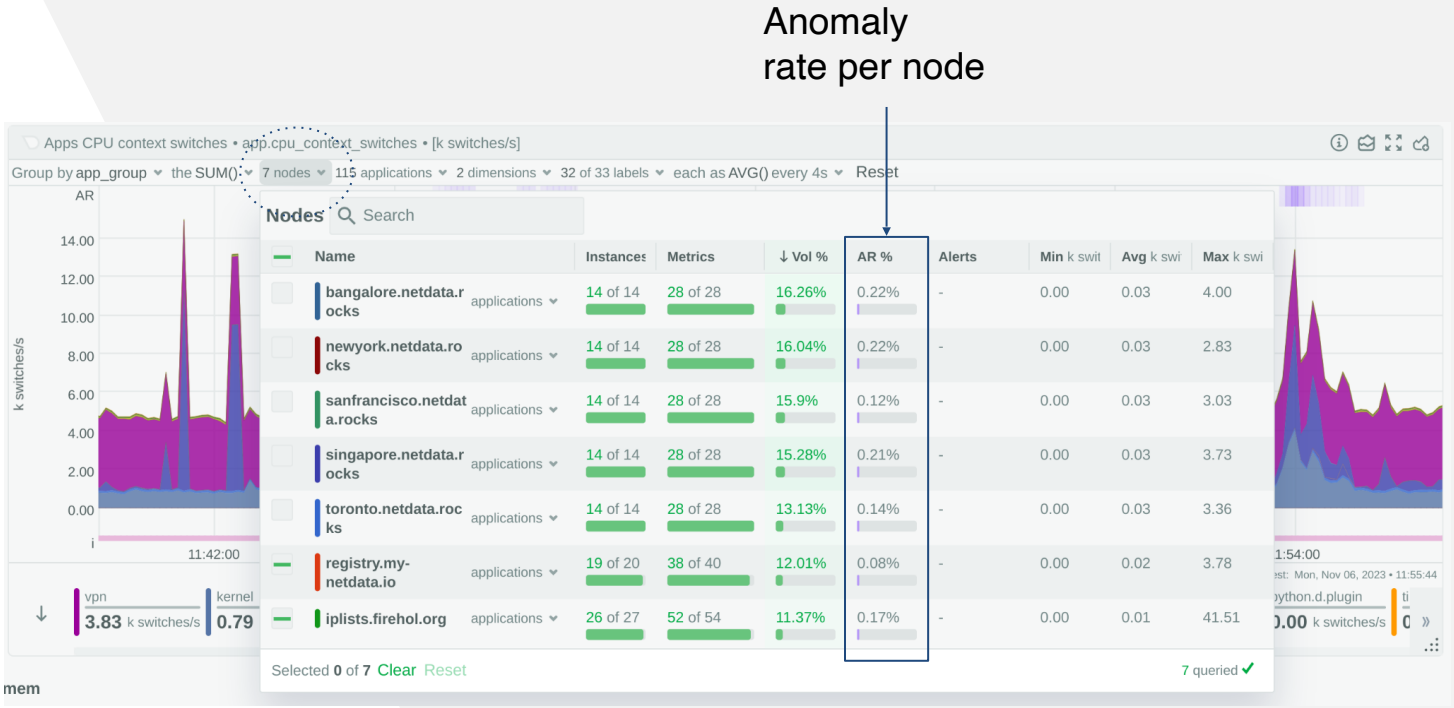
Dice the data

Aggregation across time

Anomaly rate ribbon

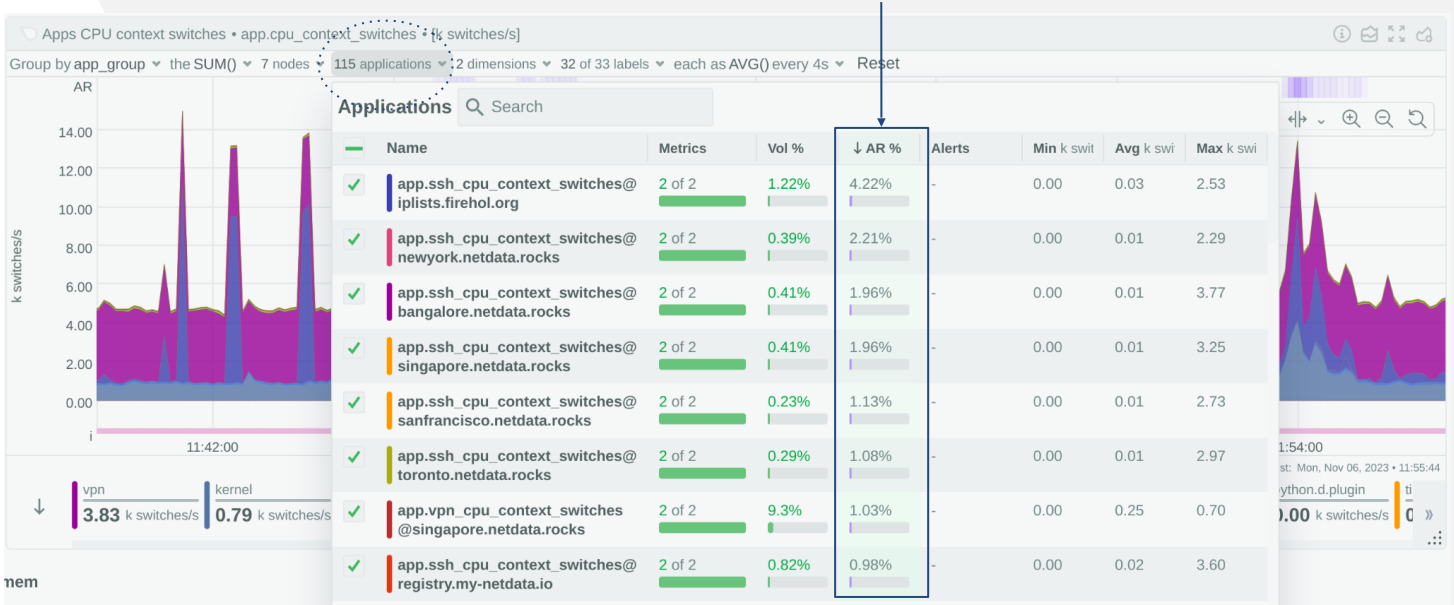
Info ribbon

A Netdata Chart - anomaly rate per node

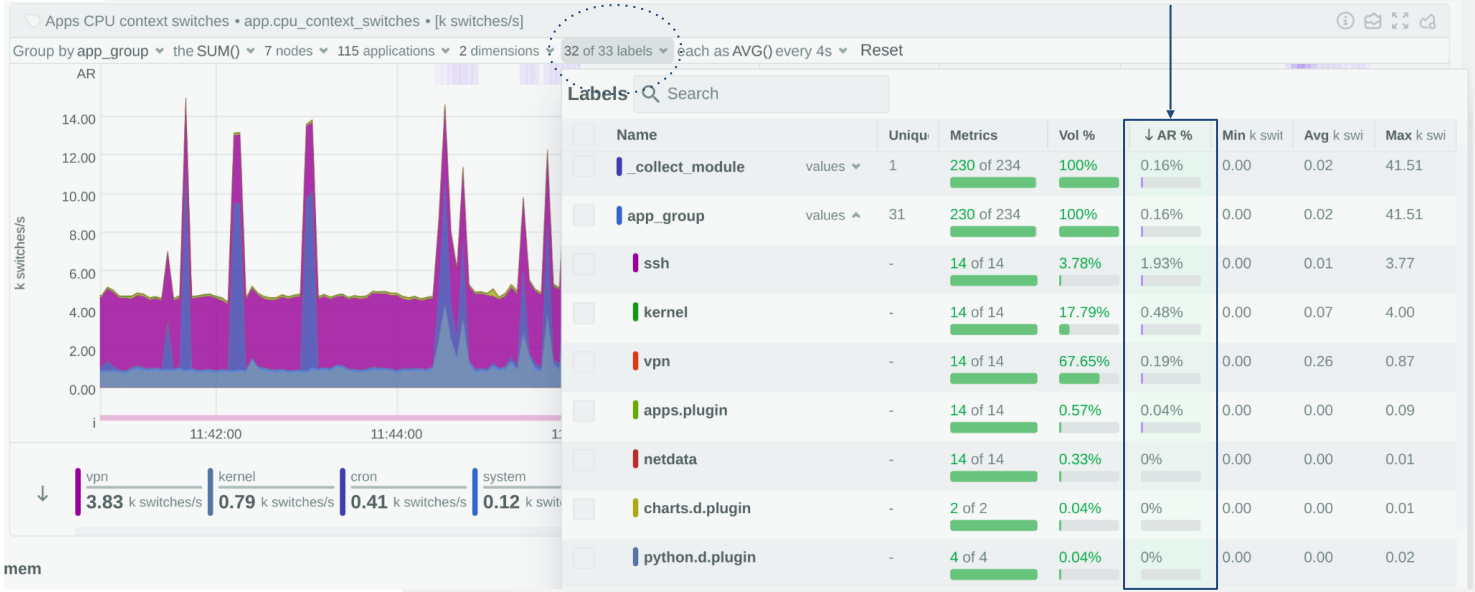


A Netdata Chart - anomaly rate per instance

Anomaly
rate per application instance

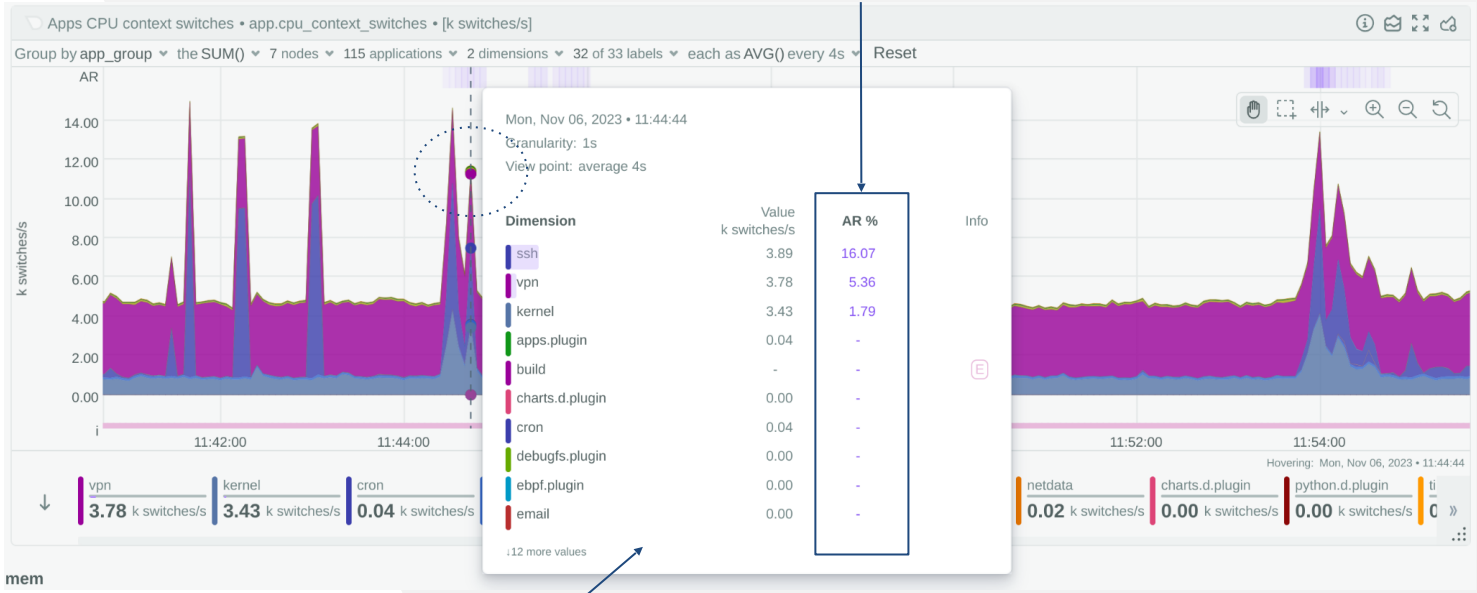


A Netdata Chart - anomaly rate per label



A Netdata Chart - anomaly rate per point

Anomaly rate per point on the chart



Popover per point while hovering the chart

Netdata has a Scoring Engine

Netdata's scoring engine

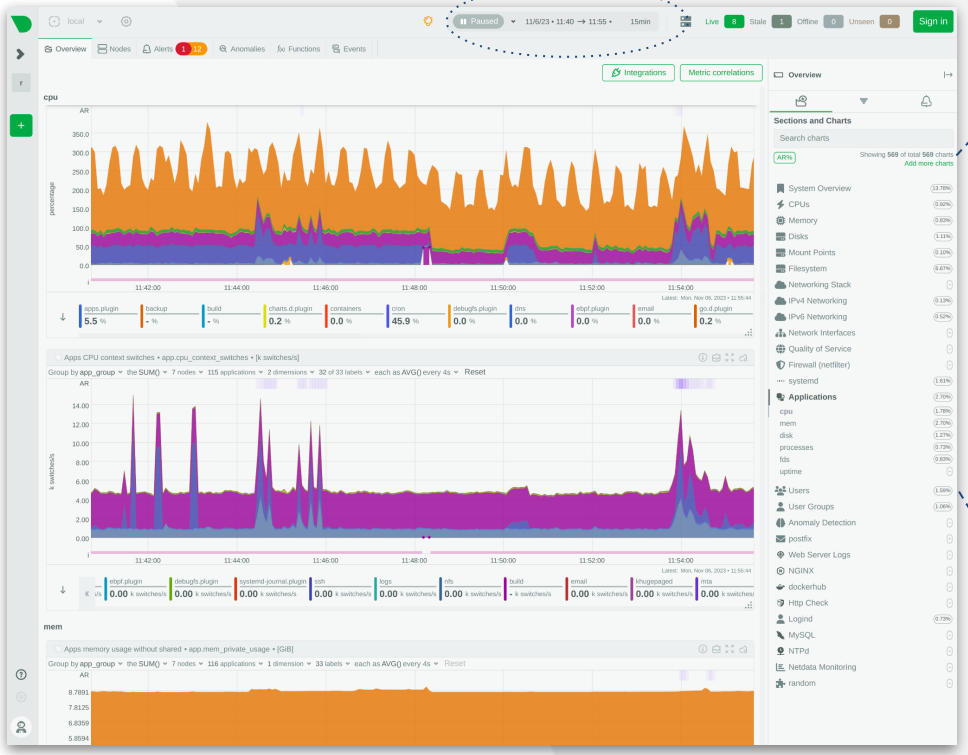
Netdata can score all metrics based on their anomaly rate for any given time-frame!

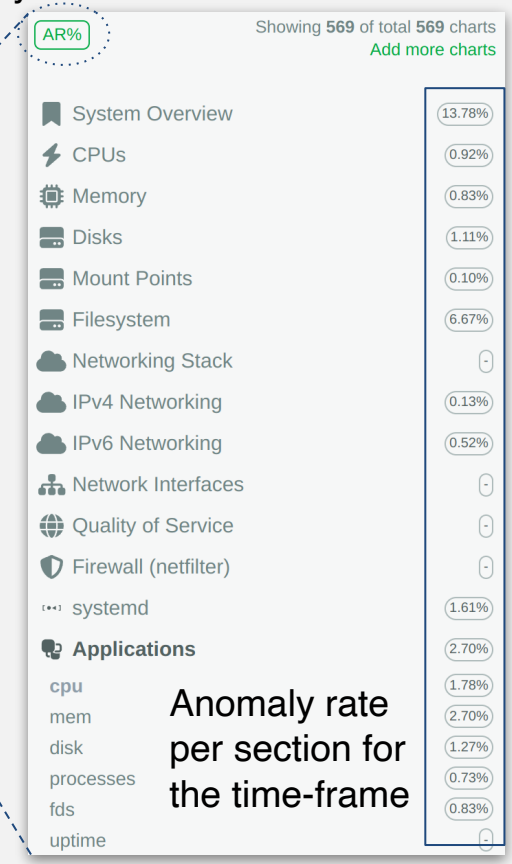
- A **scoring engine**, a unique feature of Netdata, across monitoring systems.
- All metrics, independently of their context, can be scored across time, based on various parameters, including their anomaly rate.
- **Metrics correlations** is a subset of the scoring engine, that can score metrics based on their rate of change (data), anomaly rate of change (anomaly rate), but also based on volume, similarity, etc.

A Netdata Dashboard - what is anomalous?

Time-frame picker

Anomaly Rate button





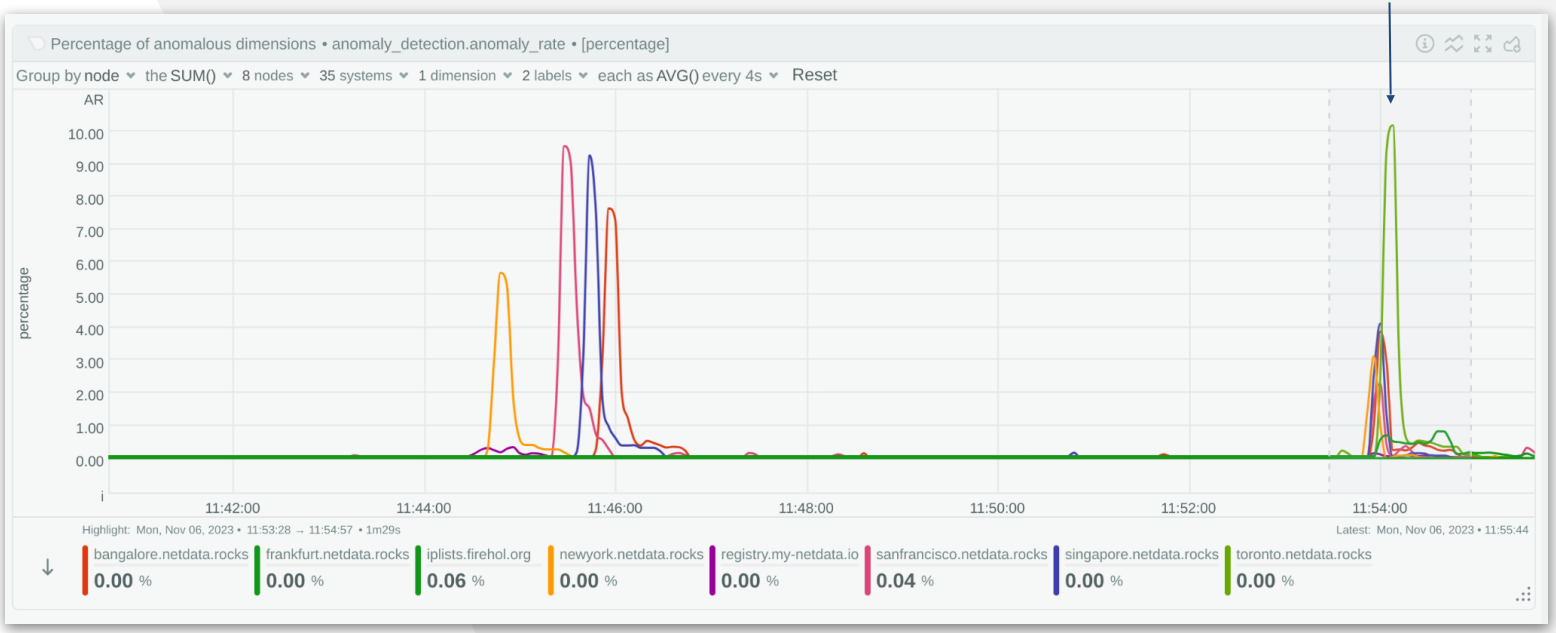
Anomaly Advisor - Root Cause Analysis

Percentage of Host Anomaly Rate

Number of metrics concurrently anomalous

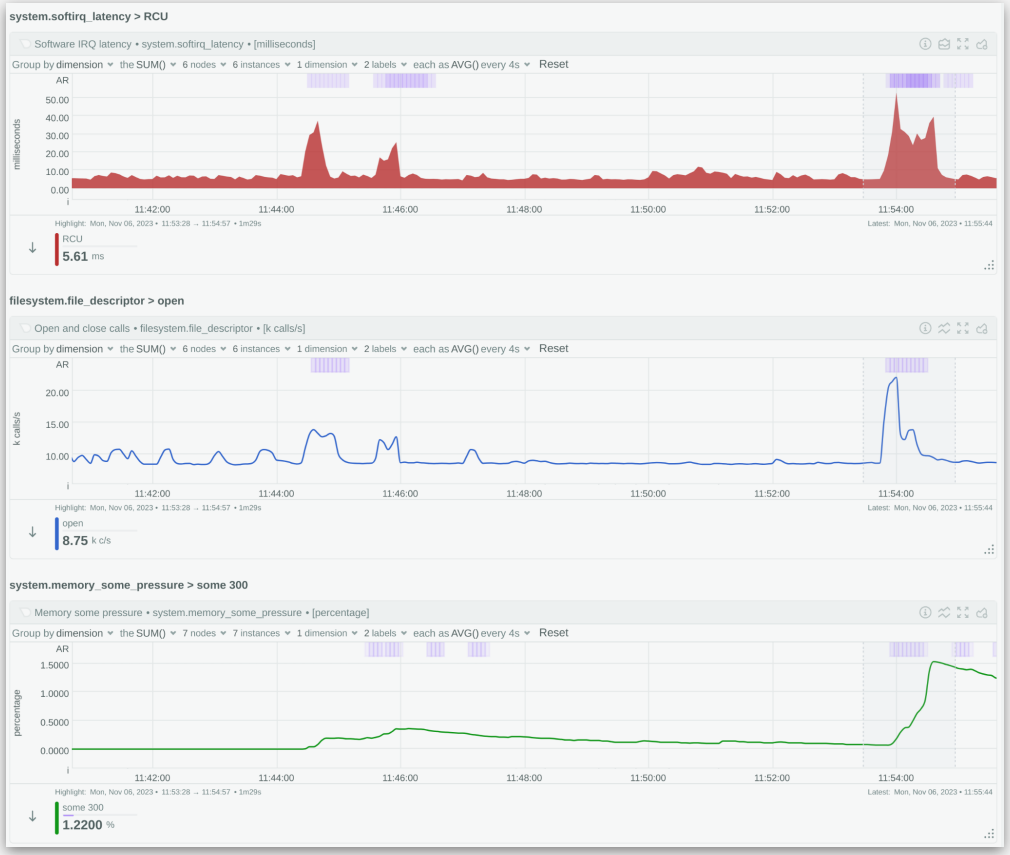
Anomaly Advisor - Find the Root Cause

Highlighting an area on the chart, triggers the analysis



Anomaly Advisor - The Report

Anomaly advisor presents a **sorted list of all metrics**, ordered by their **anomaly rate**, during the highlighted time-frame.



Highlights

Highlights of ML in Netdata

ML is an advisor.

To help you understand the data better, and identify metrics of interest.

- **Unsupervised** anomaly detection
  (you don't need to configure it, or do anything about it - just use it).
- **For all metrics**, individually
  (learning their behavior, from their past data, including the workload).
- **Available everywhere**
  (on every dashboard, on every chart, on every metric, to help you understand the data better).
- **Scoring engine**
  (to help you find what is important among the thousands of metrics available).
- **Root Cause Analysis**
  (to find correlations among even unrelated data).

About Netdata

- **75k Github Stars!**

  Netdata is the most starred observability project.

- **Leading the Observability category in CNCF!**

  In terms of Github stars: Netdata is 1st, then is Elasticsearch with 73k stars, Grafana has 69k, Prometheus has 60k, etc.

- **1.5 million downloads every day!**

  Docker hub is counting 650M pulls so far.
  Cloudflare reports 51TB of transferred data per month.

GitHub URL:

What makes Netdata unique?

- **Easy**

  Just install it on your servers and you are done!
  Dashboards and alerts are automatically created for you.

- **Real-Time**

  Per second resolution of all data.
  Just 1-second data collection to visualization latency!

- **A.I. everywhere**

  Machine Learning learns the patterns of all your data and detects anomalies without any manual intervention or configuration!

- **Cost Efficient**

  Designed to be used out-of-the-box and significantly lower operational costs.

Questions?

:GitHub URL, https://github.com/netdata/netdata

Costa Tsaousis

The fastest path to AI-first, full stack observability, even for lean teams!

netdata.cloud