VAAS-X SDK · INDUSTRIAL

Predictive Maintenance Without the Integration Project

How Bootstrap gives fleets of industrial machines operational memory that catches multi-channel precursor patterns threshold alarms can't see — validated on the field's own benchmark, zero configuration required.

100%	24	14ms	0
CMAPSS BLIND ID	CHANNELS, ZERO CONFIG	CPU RETRIEVAL, 1.18M EPS	GPU REQUIRED

VAAS-X Ltd (SC873624) · July 2026 · vaasx.com

The problem: alarms tell you after it's already too late

Threshold-based monitoring only fires once a single reading crosses a fixed line. By the time a temperature or vibration sensor trips an alarm, the fault it's flagging has often already been developing for hours or days across several correlated channels at once — the kind of multi-channel precursor pattern a single-variable threshold structurally cannot see.

Building a model to catch those patterns has historically meant a real integration project per site: label historical failure data, design a schema for each sensor array, train and retrain as equipment or configurations change. For a 200-machine production line, that's 200 integration projects, or one generic model nobody trusts near their specific equipment.

What Bootstrap actually does

Bootstrap is the ingestion layer of the `vaas-x` SDK. Point it at a sensor stream and it profiles each channel statistically, classifies the data's structure, and starts accumulating **state-action-outcome episodes** — what the system observed, what happened, and what the outcome was — without you writing a schema or labelling a single reading first.

```
from vaasx import Bootstrap

brain = Bootstrap(api_key="YOUR_API_KEY", device_id="line_3_press_7")

# Any http(s), ws(s), mqtt, local file (.jsonl/.csv/.json), pandas
# DataFrame, or plain Python generator.
brain.connect("mqtt://press-7.internal/channels")

hits = brain.query("vibration rising with temperature drift", k=5)
for hit in hits:
    print(hit["score"], hit["episode"])

# Tell the system whether a past episode's flagged pattern was a
# real fault -- future queries weight toward what actually mattered.
brain.outcome(episode_id=hits[0]["id"], success=True)
```

Every stored episode is retrievable by similarity in milliseconds, and results are weighted toward episodes with a recorded successful outcome by default — the system surfaces what actually preceded a real fault, not just what looks statistically similar to the current reading.

Validated: NASA CMAPSS FD001 blind trial

CMAPSS FD001 is the standard public benchmark dataset for predictive maintenance research — turbofan engine run-to-failure sensor data used across the field to compare approaches on equal footing. Bootstrap was presented with all 24 sensor channels with every identifier replaced by an opaque numeric label: no channel names, no physical units, no prior information about what any signal represented.

> **RESULT**
>
> Bootstrap correctly classified all 24 channels — separating stable operational baselines from the channels carrying dynamically significant, fault-relevant information — with zero configuration and zero domain knowledge supplied in advance.

That's the same zero-config profiling a real deployment gets: point Bootstrap at your actual sensor array, and it works out the structure from the data itself, the same way it did on a dataset it had never seen with labels it was never given.

Deployment model

Bootstrap's profiling, classification, and episode accumulation all run locally, on your own hardware — on-premise, air-gapped, or cloud, your choice. Only the resulting episodes cross the network to reach retrieval. There's no cloud dependency for the equipment itself to keep running, and no GPU requirement: retrieval runs at 14ms mean latency across 1.18 million stored episodes on a standard CPU server, with no specialist hardware.

Pricing is infrastructure-based, not per-query: a fixed monthly cost per tier regardless of how many queries your fleet makes or how many episodes it accumulates, so connecting a hundred machines doesn't multiply your bill the way per-token AI pricing would.

> **SCOPE, HONESTLY**
>
> What ships and works today: Bootstrap ingestion, retrieval, and the outcome-grounded query loop above — all production, all benchmarked. Encrypted search over your data is architected into the platform but is currently a roadmap item for custom Enterprise deployments, not part of the standard hosted service — check current availability before relying on it for a compliance requirement. This whitepaper only describes what's actually shipping.

Get started

Free tier: instant API key, no credit card — `pip install vaas-x`, connect a source, query your first episode in under ten minutes.

Full guide: vaasx.com/docs/getting-started.html | Enterprise: hello@vaasx.com