VerifIA : OSS Domain-Aware AI Model Verification Tool

Presenters: Houssem Ben Braiek and Ahmed Haj Yahmed

at

SEMLA x SEMTL

Workshop Day

VerifIA A Canadian Non-Profit Organization

Our Mission is to ensure "**every AI system in production is fully verified**".

Open-Source Tools

Industry Partnerships

Workshops & Training

By promoting transparency, safety, and compliance, we're building a collaborative ecosystem where **verified, human-aligned AI becomes the norm—not the exception.**

Founding Team



Foutse Khomh,
SE Professor



Mohamed El Abed,
BI Advisor



Houssem Ben Braiek,
ML Tech Lead



Ahmed Haj Yahmed,
MLOps Specialist

Need for Domain-Aware Verification

Perf. metric gaps, adopted mitigations, and domain-aware verification

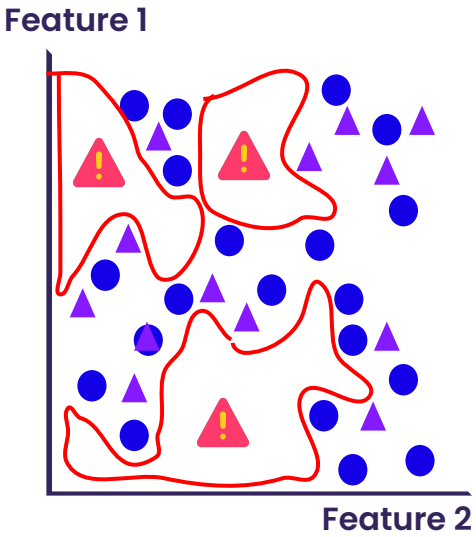
Conventional AI Model Testing



Performance-only testing on finite samples is often misleading and risky.

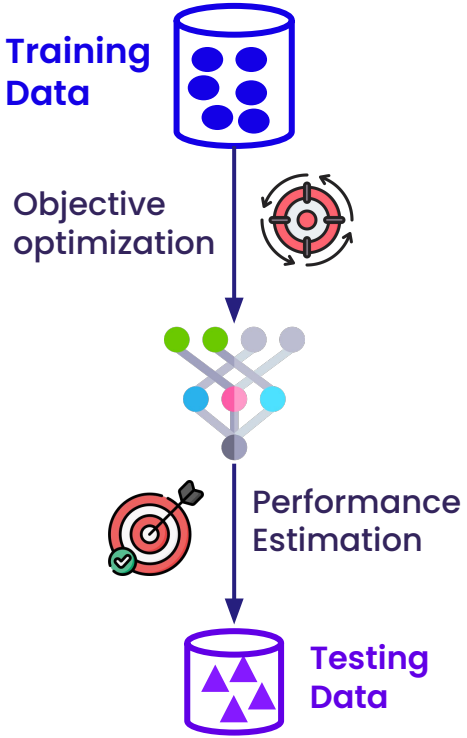
Performance Testing Pitfalls

Data Bias



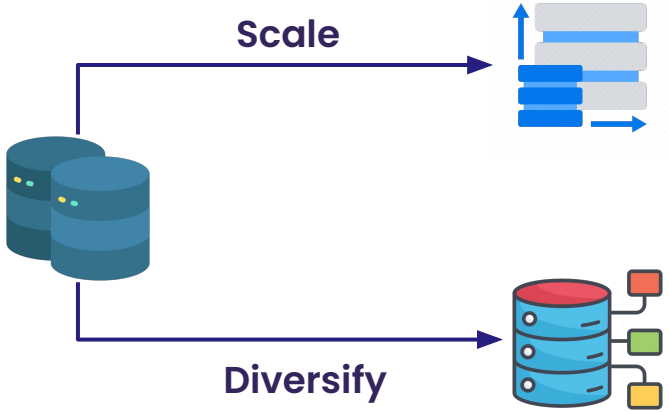
Sampling or selection bias lead models to misrepresent the problem space.

Shortcut Learning



Deep networks exploit shortcuts in training data, converging fast with misleading high test scores.

Common Practices to Avoid Perf. Testing Pitfalls



Most ML problems involve long-tail distributions, where rare cases dominate and are hard to cover.

In high-stakes ML, researchers rely on **domain-aware stress tests** to uncover model issues—but these tests are **manually-tailored** for each use case.

We propose a **framework** that equips engineers with a **systematic, reusable** approach to **design domain-aware stress tests**.

The Value Prop. of Domain-Aware Verification

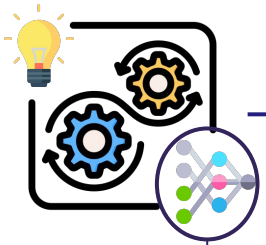


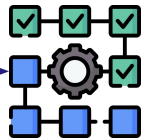
Benchmarks



Domain Knowledge:
- Application requirements
- Apriori knowledge
- System properties
- etc.

Domain-Aware Verification





Consistency Rate

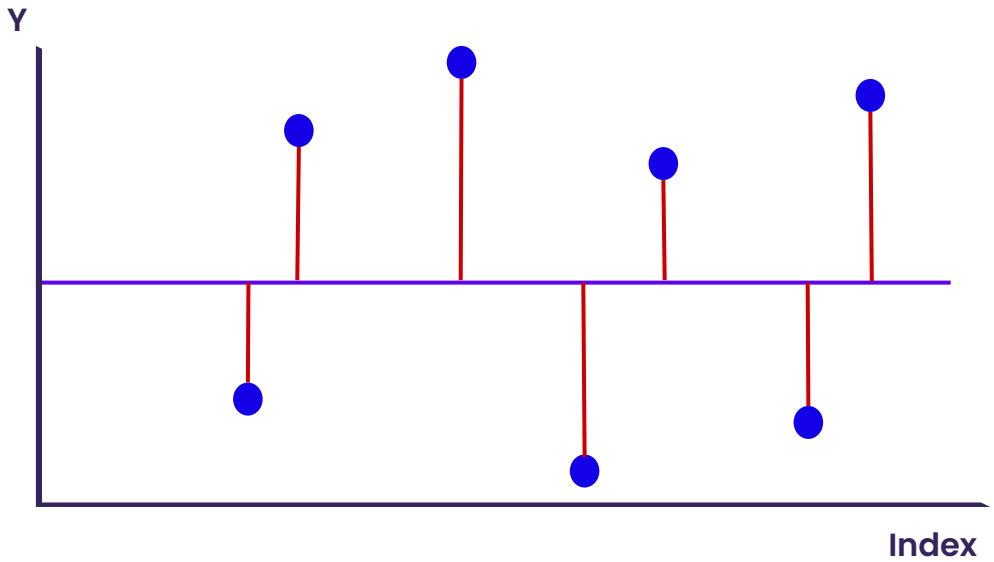
VerifIA v0.1

Forecasting models
y or P(y=1)

Foundations of Domain-Aware AI Forecasting Model Verification

Origins, Manifestations, Test Space, Coverage

Foundations: AVG. Baseline Error

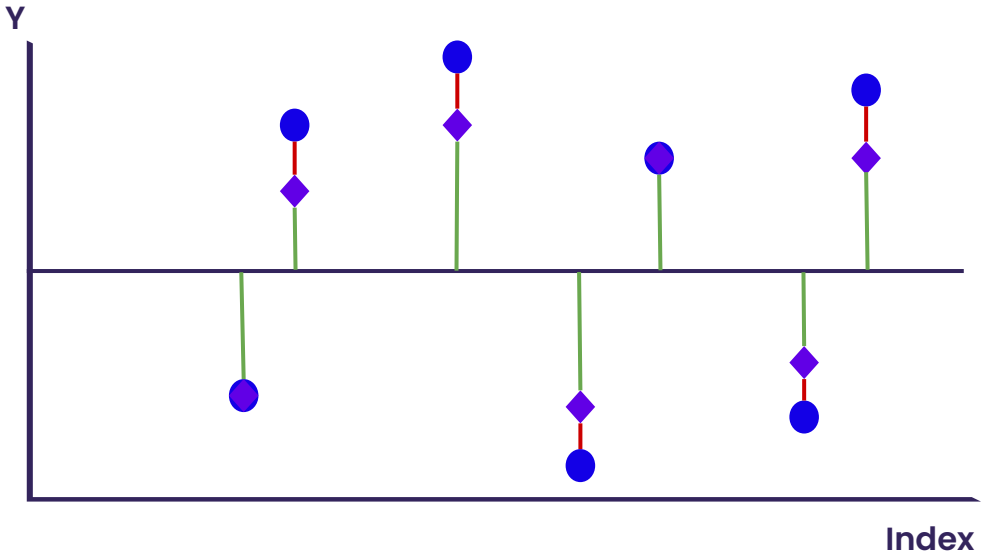


avg(y) ~ Baseline Prediction

● Groundtruth (y)

– var(y) ~ Baseline Error

Foundations: Explained Variance



- Groundtruth (y)
- Prediction (y')

– Unexplained deviations ~ Model Error

– Variance of (y) explained by (y')

Quantifying unexplained deviations requires ground truth labels—hence the need for labeled benchmarks.

If we knew the true value, we'd replace the model—not evaluate it.

Given domain knowledge, can we describe the nature of this variance without specifying its exact value?

Foundations: Invariance & Directional Expectation



Car Trades Datasets

learn



predict

Car Pricing Model

X: Car Features

Brand
Model
Year
Mileage
Grade
Location
...

y': 25k

avg(y) + dev(avg(y), y')

Data Intelligence

derive

X': Derived Car Features

Brand
Model
Year +/- 1
Mileage +/- 10k
Grade
Location
...

Expected **y'': 25k +/- 2k**

y' + dev(y', y'')

where,
-2k < dev(y', y'') < 2k

Invariance Rule

X'': Derived Car Features

Brand
Model
(Year+1) ↗
(Mileage -10k) ↘
Grade
Location
...

Expected **y''': 27k+**

y' + dev(y', y''')

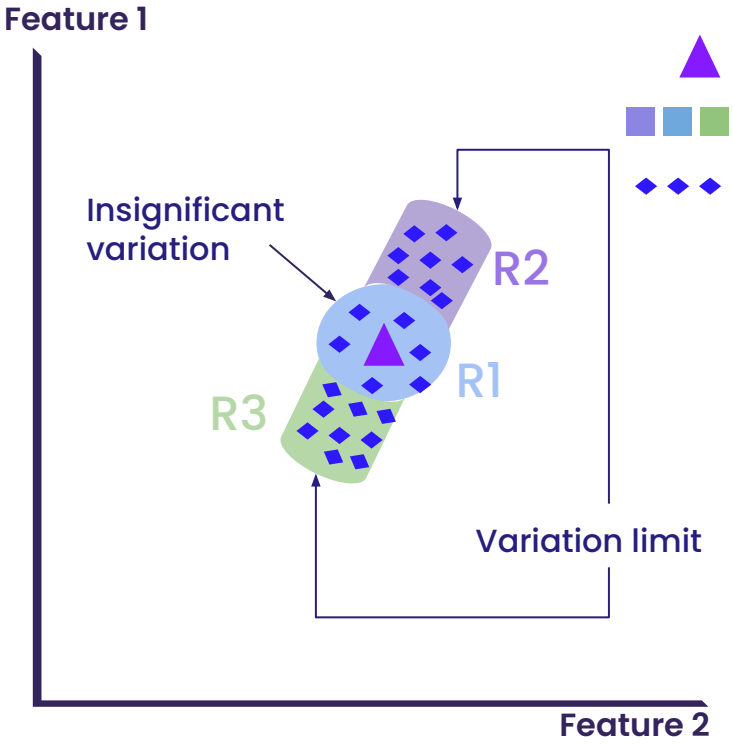
where dev(y', y''') > 2k

Directional Expectation Rule

Domain Intelligence

⇒ 2k: insignificant variation

Foundations: Rules-constrained input sub-spaces



Original Input

Rule-constrained input space

Derived (Synthetic) Inputs

Enumerating all rule-based derivations quickly leads to combinatorial explosion.

To search the rule-constrained input space for inconsistency-revealing cases, ***VerifIA*** uses:



Evolution-Based metaheuristics: GA



Swarm-Based metaheuristics: PSO, CSA, BAT, GWO, MFO, WOA, MVO, FFA, and SSA.

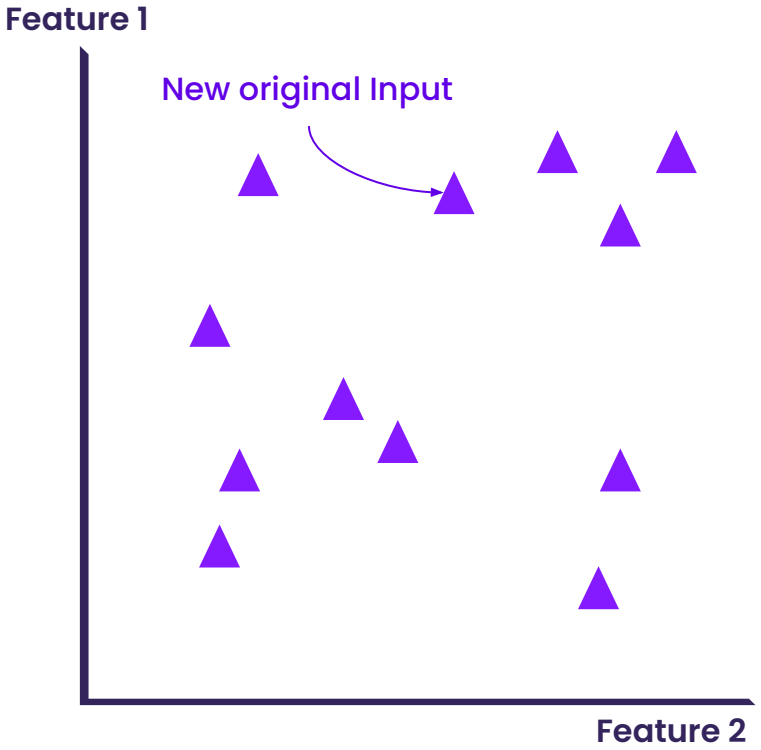
Particle Swarm Opt. (PSO);
Genetic Algorithm (GA);

Multi-Verse Opt. (MVO);
FireFly Algo. (FFA);

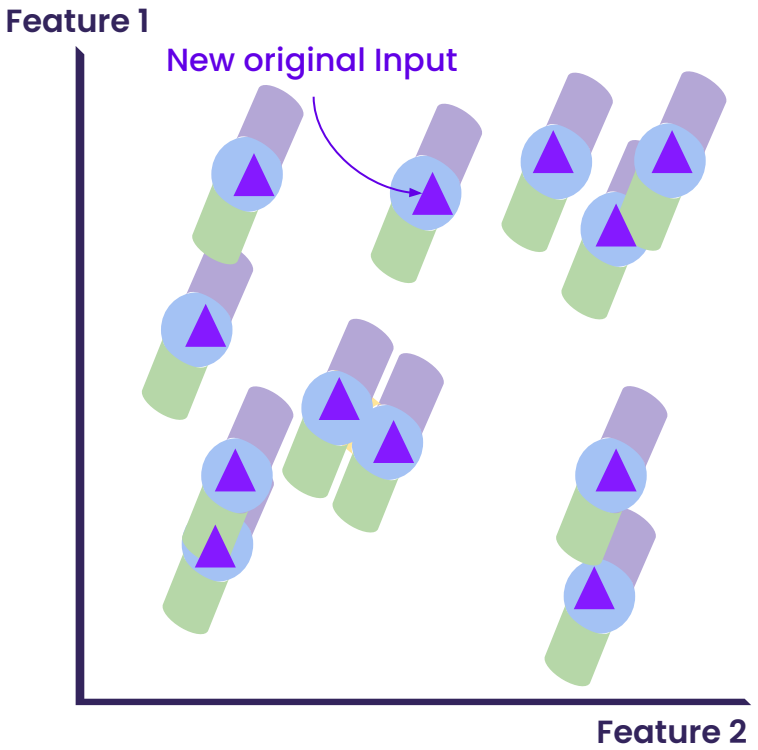
Moth-Flame Opt.(MFO);
Whale Opt. Algo. (WOA);

Grey Wolf Opt. (GWO);
Salp Swarm Algo. (SSA);

Foundations: Perf. & Consistency Test Coverage

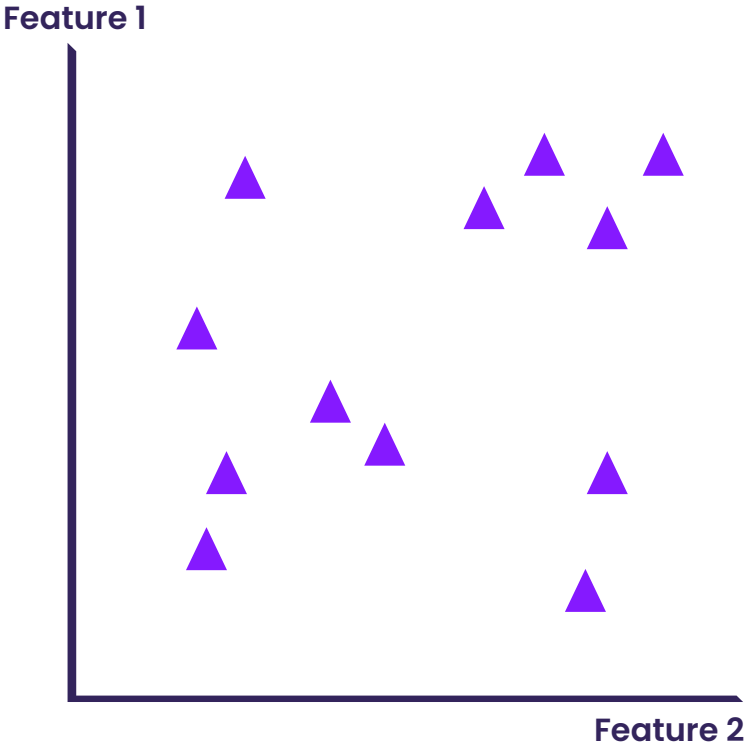


Performance Test Coverage

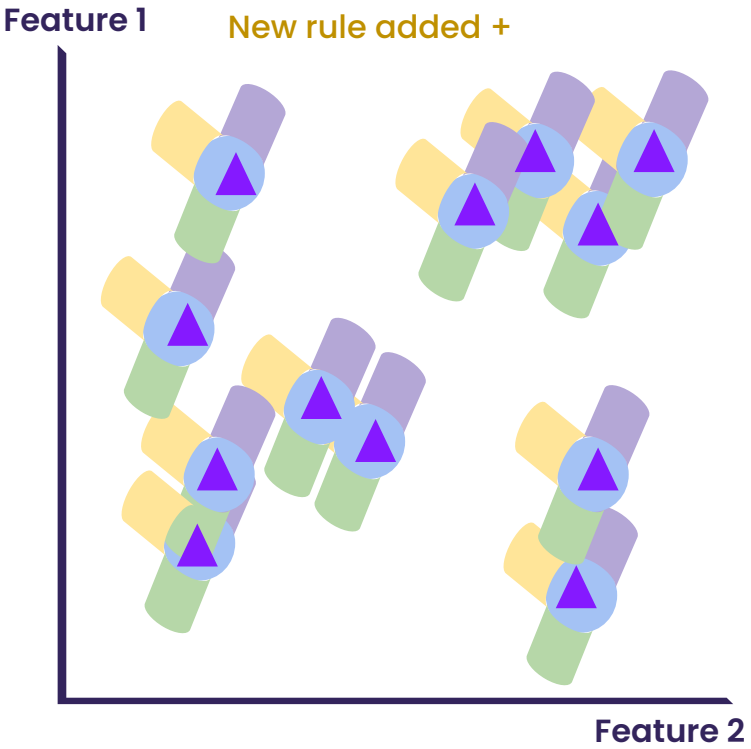


Consistency Test Coverage

Foundations: Perf. & Consistency Test Coverage

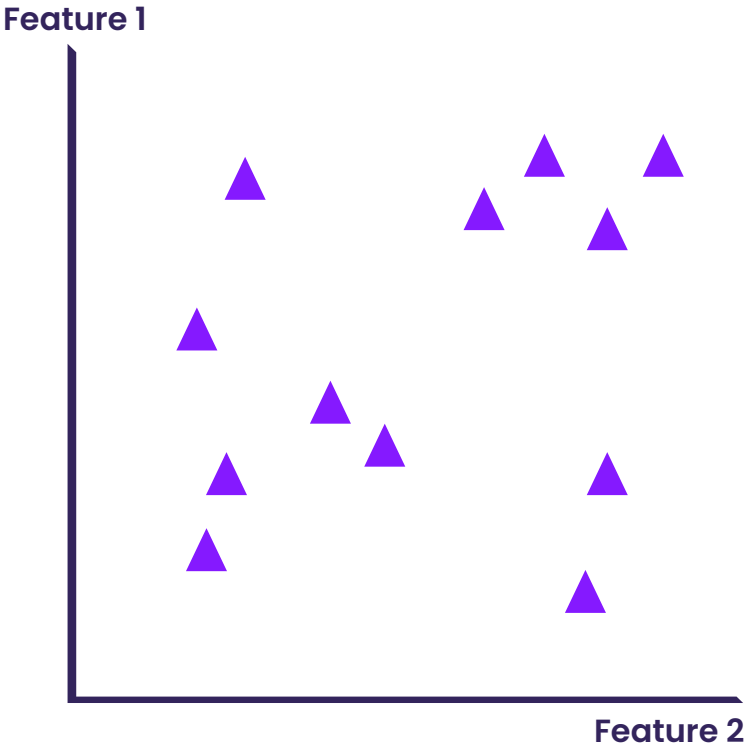


Performance Test Coverage

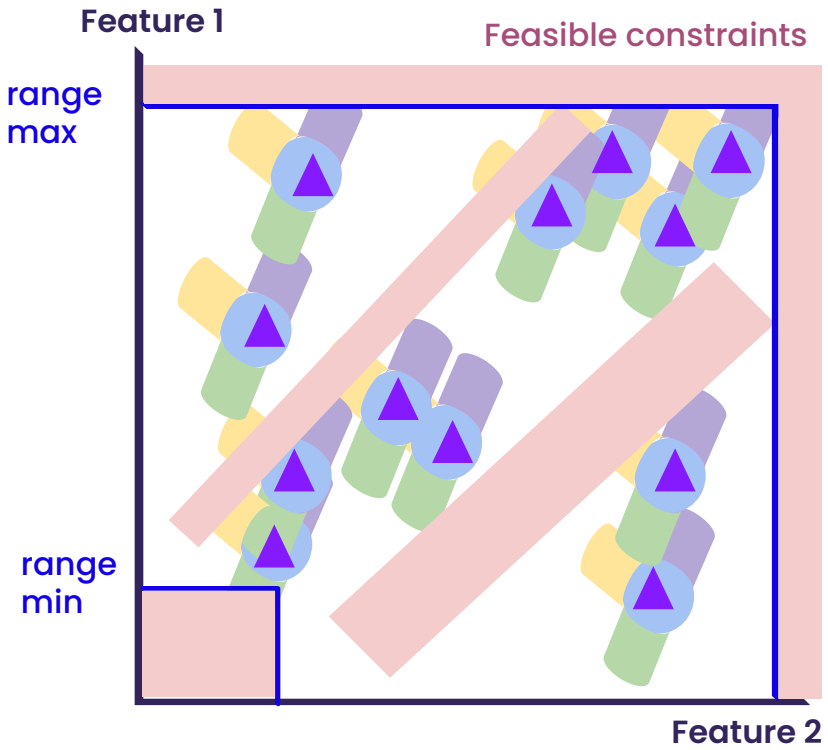


Consistency Test Coverage

Foundations: Perf. & Consistency Test Coverage



Performance Test Coverage



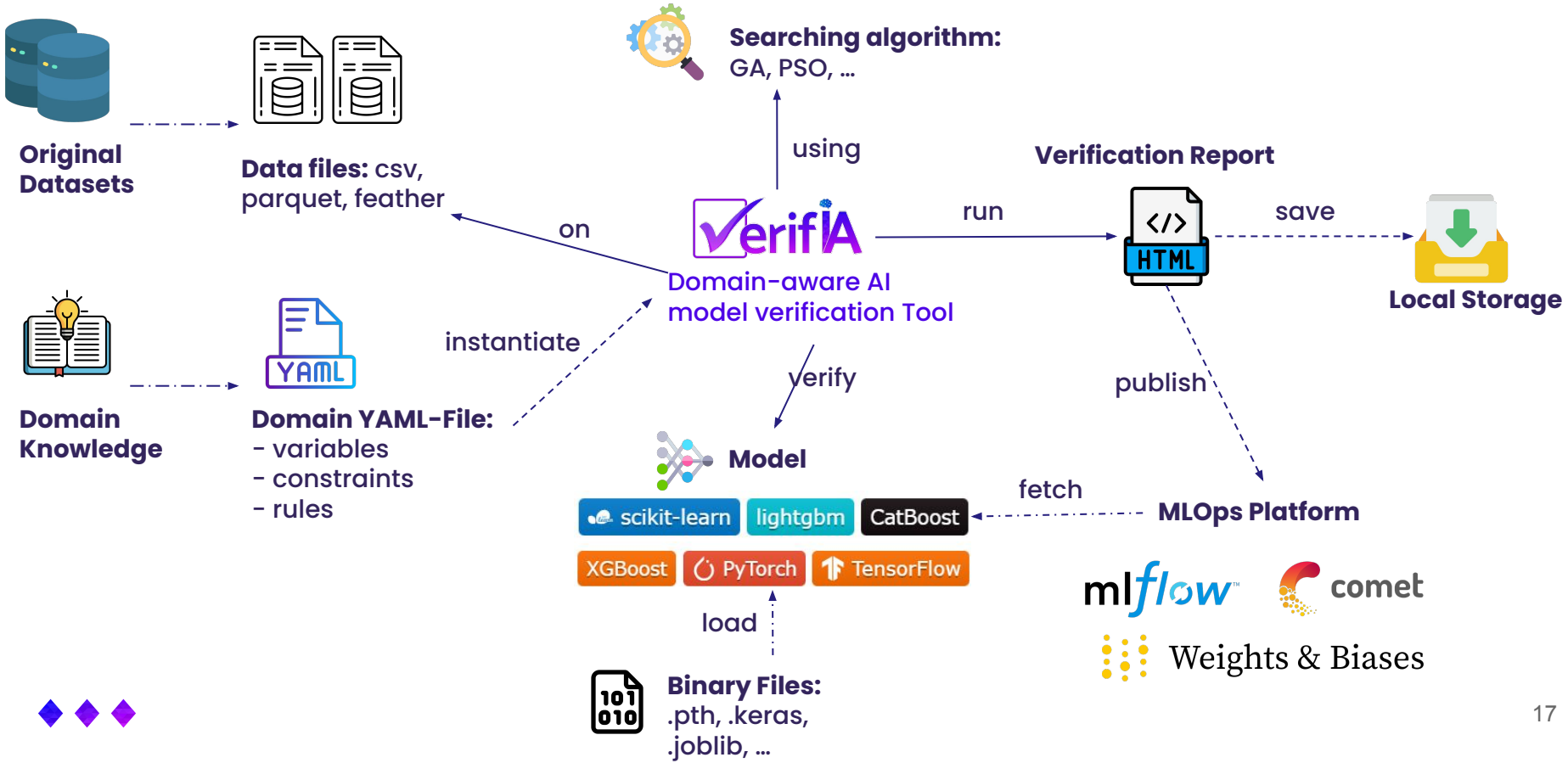
Consistency Test Coverage



The Open-Source Python Tool

Structure, Quickstart, Verif. Report and Domain Definition

VerifIA Tool: Inputs, Outputs, and Components



VerifIA Tool: Quickstart Code Example



domain_rules.yaml

```yaml
variables:
 pos_imp_1:
   type: FLOAT
   range: [-3, 3]
   variation_limit: 0.5
   insignificant_variation: 0.02
 pos_imp_2:
   type: FLOAT
   range: [-3, 3]
   variation_limit: 0.5
   insignificant_variation: 0.02
 target:
   type: FLOAT
   range: [-100, 100]
   insignificant_variation: 0.05

constraints: {}

rules:
 R_inc_pure:
   premises: { pos_imp_1: inc, pos_imp_2: inc }
   conclusion: { target: nodec }

 R_dec_pure:
   premises: { pos_imp_1: dec, pos_imp_2: dec }
   conclusion: { target: noinc }
```

```python
from verifia.verification import RuleConsistencyVerifier

# 1. Arrange: load your domain rules
verifier = RuleConsistencyVerifier("domain_rules.yaml")

# 2. Act: attach model (card or dict) + data
report = (
    verifier
      .verify(model_card_fpath_or_dict="model_card.yaml")
      .on(data_fpath="test_data.csv")        # .csv, .json,
      .using("GA")                           # RS, FFA, MFO,
      .run(pop_size=50, max_iters=100)       # search budget
)

# 3. Assert: save and inspect your report
report.save_as_html("verification_report.html")
```

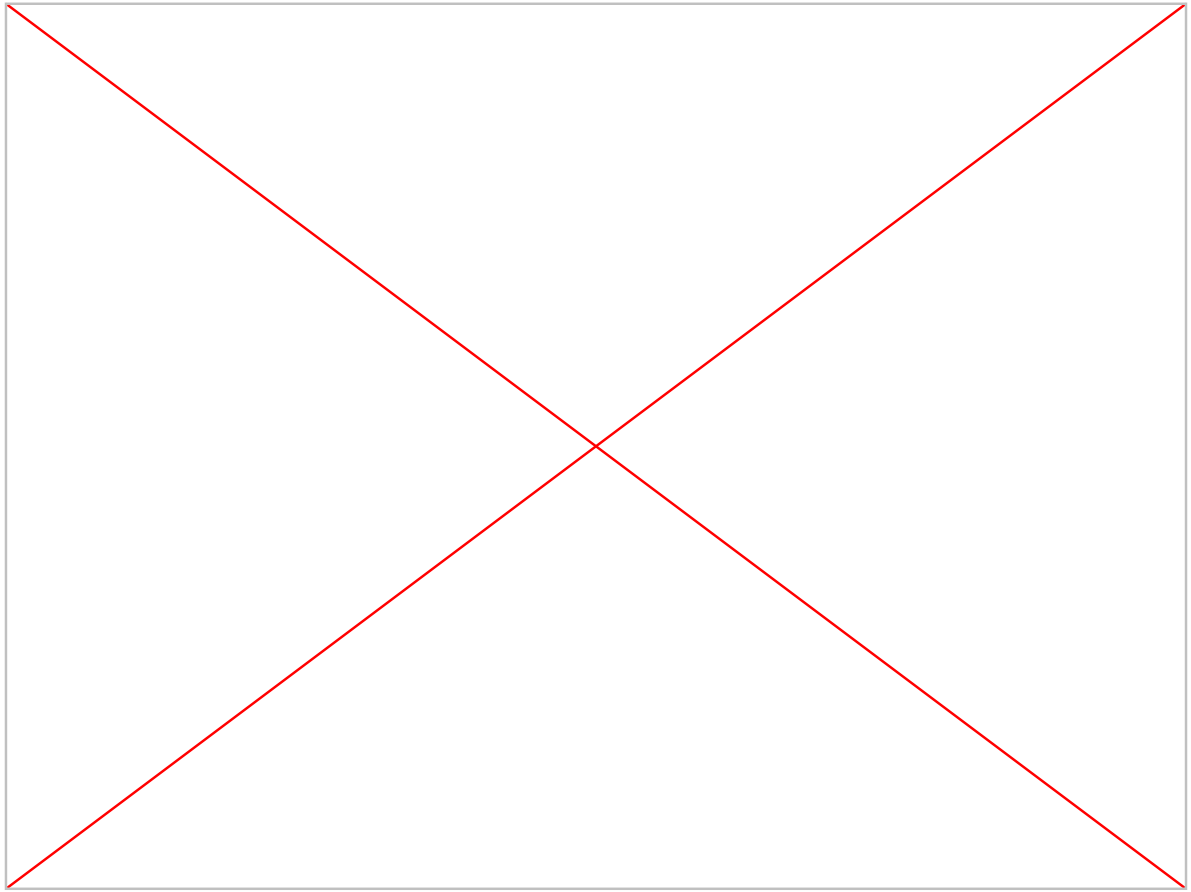
model_card.yaml

```yaml
name: your_model
version: '1'
type: regression
framework: sklearn
feature_names:
 - fearture_1
 - fearture_2
 - ...
target_name: target
local_dirpath: models
```

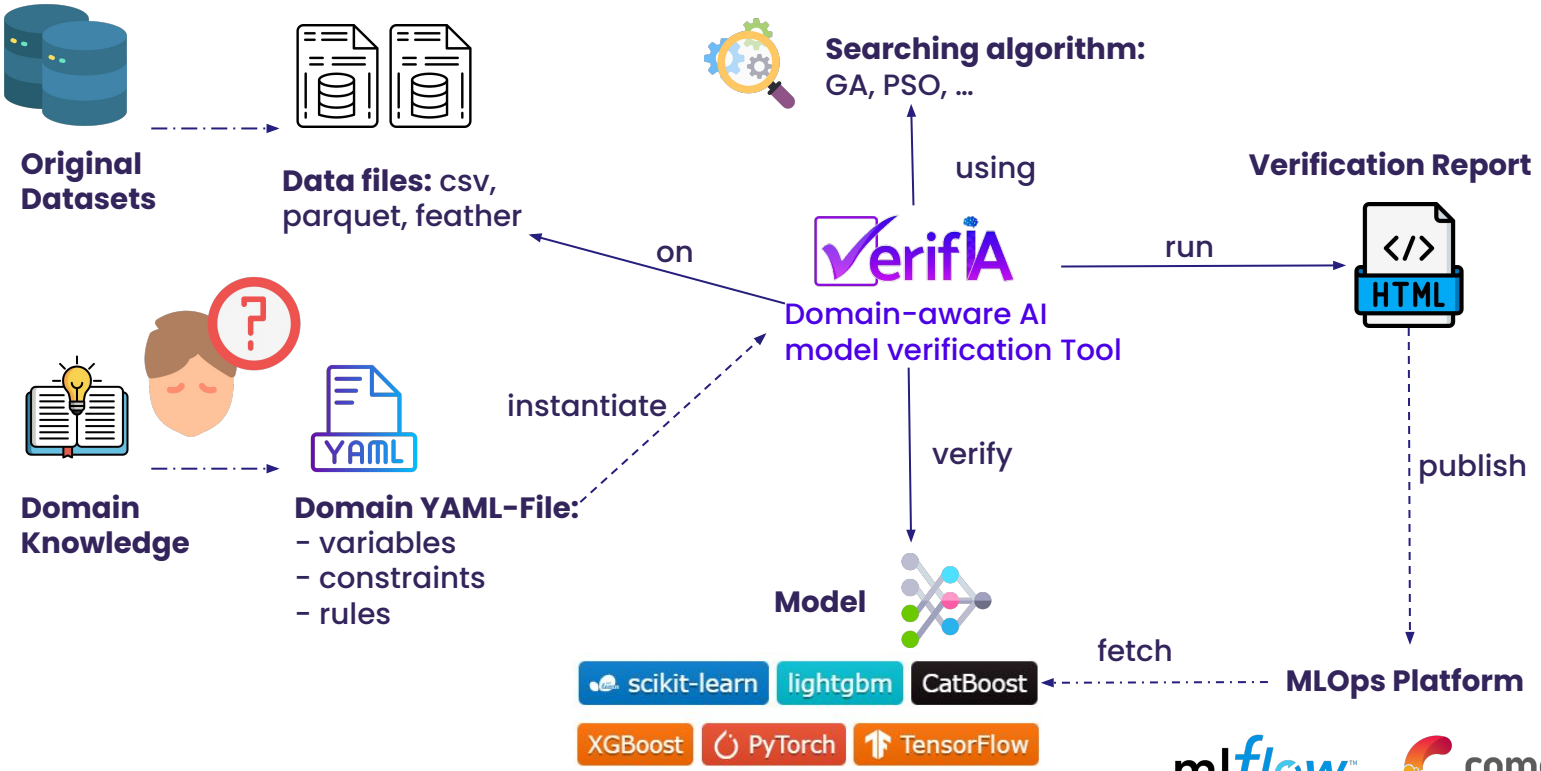
```
> pip install verifia
```

https://github.com/VerifIA/verifia

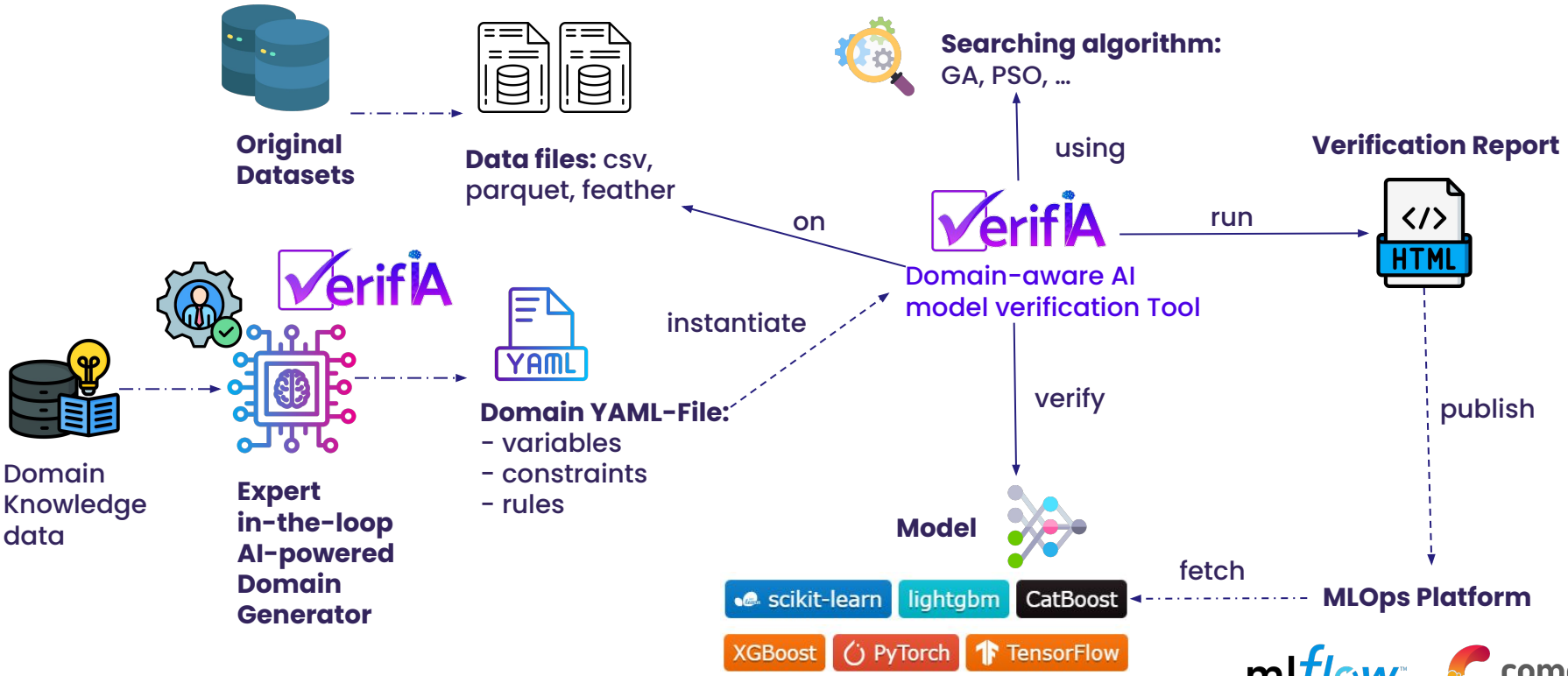
VerifIA Tool: Verification Report



VerifIA Tool: The Domain File is a Prerequisite



VerifIA Tool: AI-Assisted Domain File Generation

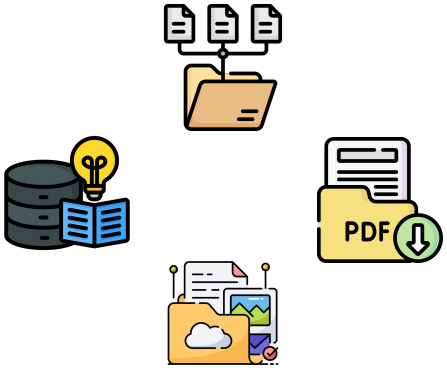


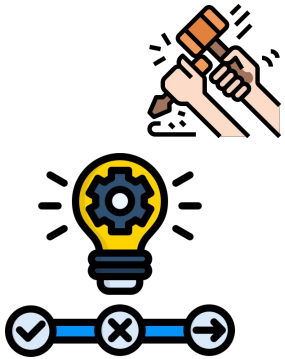


Expert in-the-loop AI-powered Domain Generator

Why AI-Assisted Domain File Generation?

Domain Knowledge data







Domain YAML-File:
- variables
- constraints
- rules

VerifIA

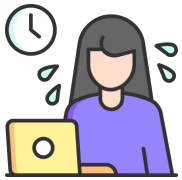
Domain-aware AI model verification Tool



Error-Prone

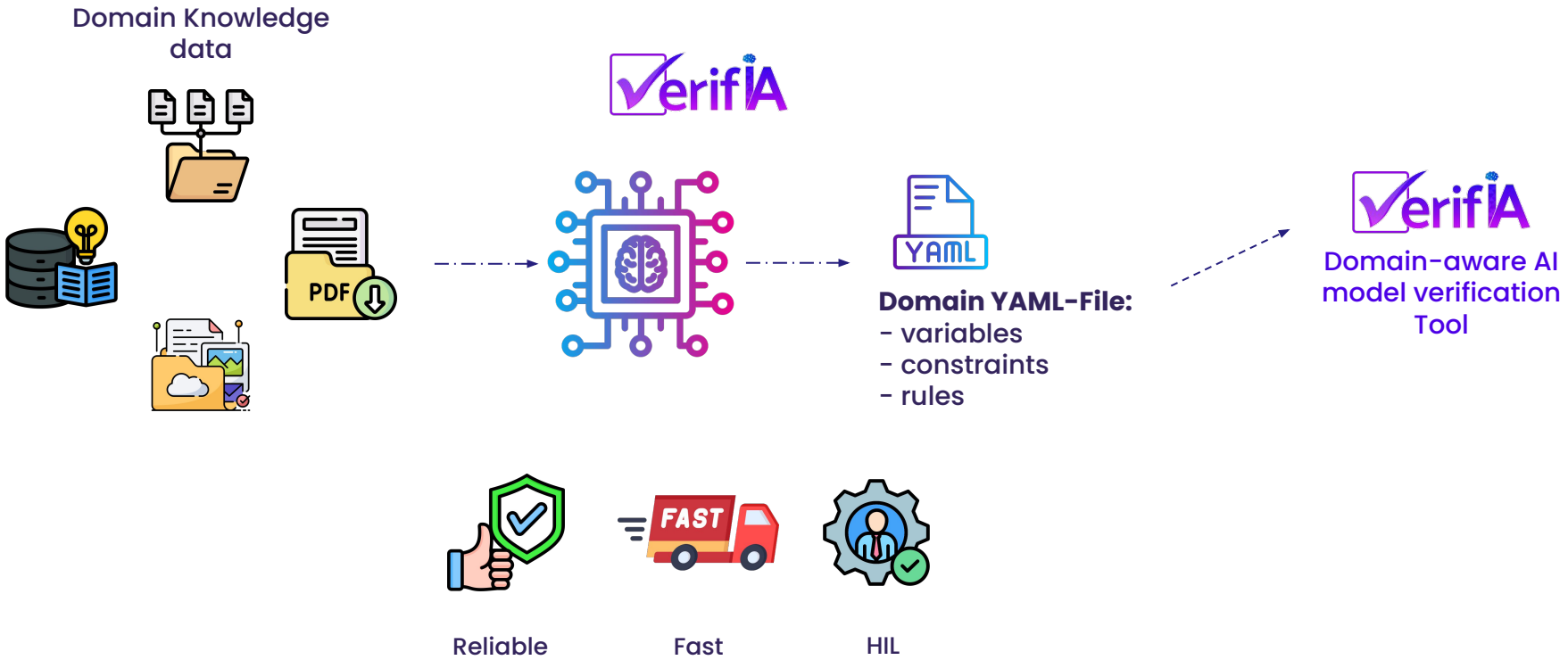


Time-Consuming

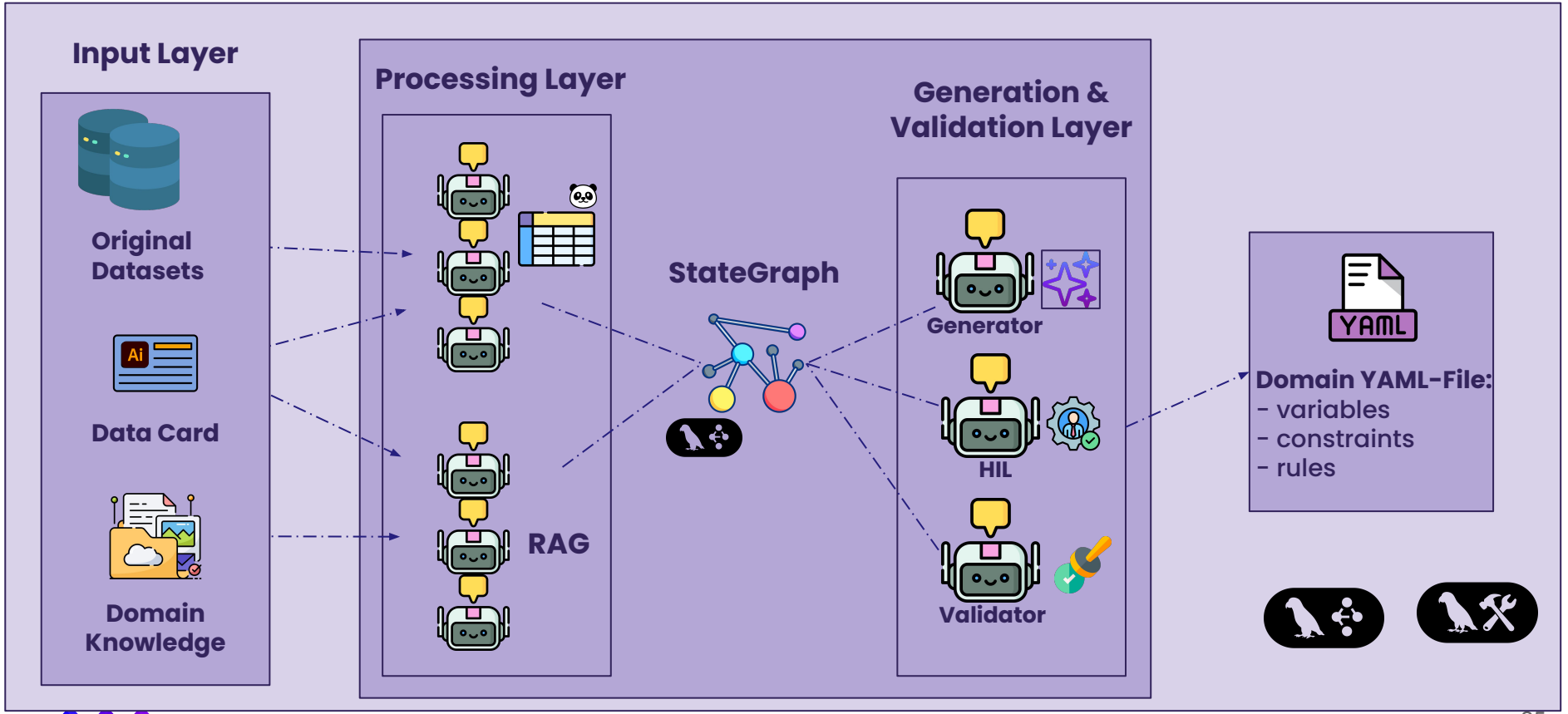


Hard

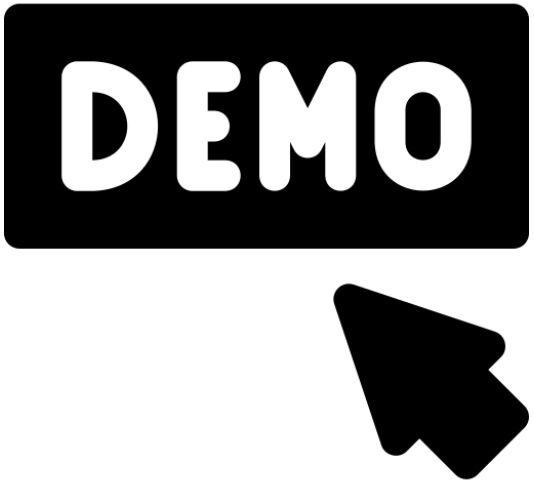
Why AI-Assisted Domain File Generation?



Domain-GenFlow Conceptual Overview



Domain-GenFlow UI



Call to Action

If you believe in VerifIA's mission:



Try VerifIA



Star us on GitHub



Share VerifIA



Follow us

www.verifia.ca