Rampart: Reproducible Benchmarking of Data Processing Paradigms with Automated Anti-Leakage Verification

Eos Xavier, Lívia R. Duarte Ripardo, Fernando M. da Mota, Bruno M. Nogueira, Vanessa A. Borges

[1]Faculdade de Computação (FACOM)
Universidade Federal de Mato Grosso do Sul (UFMS) – Campo Grande, MS – Brasil

{eos.xavier, rosa.livia, fernando.mota,
bruno.nogueira, vanessa.a.borges}@ufms.br

Abstract. *The growing adoption of data-processing engines for machine-learning pipelines has increased the need for rigorous benchmarks that ensure validity and reproducibility. However, many studies still lack mechanisms to prevent data leakage. Rampart is presented as an open-source framework executing the same walk-forward validation pipeline across three data-processing paradigms while enforcing anti-leakage invariants. Rampart treats prediction equivalence as a validity clause for cross-paradigm latency comparisons and produces bitwise-reproducible executions. Evaluated on two public time-series panels, the framework reveals a scale-dependent latency crossover and supports inclusion of new paradigms with low engineering overhead.*[1]

Keywords: software frameworks; cross-paradigm benchmarking; data leakage; reproducibility; experimental methodology.

1. Introduction

Although data-processing paradigms are often treated as implementation choices that affect only performance and cost, the CACE principle suggests that such changes may also alter learned behavior in non-obvious ways [Sculley et al. 2015]. Two main problems arise in ML pipelines on analytical engines: reproducibility and data leakage. Reproducibility concerns the ability to obtain consistent results when the same ML pipeline is executed under the same experimental conditions, a property that data-science pipelines often violate due to inadequate provenance tracking [Rupprecht et al. 2020]. Data leakage occurs when information that should be unavailable during training, validation, or model selection is inadvertently incorporated into the pipeline [Kapoor and Narayanan 2023].

Recent studies show that these issues are widespread. Kapoor and Narayanan (2023) identified data leakage in 294 papers across 17 research fields, Gundersen and Kjensmo (2018) found AAAI and IJCAI papers documented few reproducibility-relevant variables, and Pineau et al. (2021) reported under half of NeurIPS 2018 submissions shipped the code needed for reproduction. In time-series scenarios, Semmelrock et al. (2025) further note that leakage prevention via walk-forward validation is rarely automated in benchmarking tools. The growing use of specialized analytical engines compounds this, since each paradigm requires re-implementing methodological guarantees [Kiehn et al. 2022].

[1]Artifacts: https://doi.org/10.5281/zenodo.20749082

To mitigate these limitations, **Rampart** is introduced as an open-source framework for comparing ML pipelines across data-processing paradigms. Rampart runs a walk-forward validation pipeline over three paradigms: (i) SQL-compiling engines, such as DuckDB[2]; (ii) DataFrame libraries, such as Polars[3]; and (iii) task-graph schedulers, such as Dask[4]. These paradigms differ in architecture and execution semantics, and performance-centered comparisons leave open whether they also affect ML *predictions*. Rampart investigates this while enforcing a five-property anti-leakage protocol (P1–P5). It is applied to two public time-series panels, World Bank and INEP. Results show that paradigms affect performance and benchmark validity: prediction equivalence is needed for fair latency comparisons, while Polars performs better at smaller scales and Dask at larger scales.

2. Related Work

Industry-standard ML benchmarks and tracking layers cover ML pipeline execution but not paradigm equivalence: TPCx-AI [Brücke et al. 2023] validates each reference implementation against independent accuracy thresholds rather than verifying cross-paradigm prediction equivalence, and MLflow [Zaharia et al. 2018] records reproducibility metadata without enforcing leakage prevention. Traditional database benchmarks address adjacent problems: TPC-DS [Nambiar and Poess 2006] targets analytical queries without ML pipelines, and Harby and Zulkernine (2025) benchmark data lake, warehouse, and lakehouse architectures on analytical queries with no model-training stage; within SBBD, Soares et al. (2021) recommend data systems via query profiling and Oliveira et al. (2020) extend self-tuning within a single relational engine. None aligns data-processing paradigms within a fixed ML pipeline.

Since none of these verifies cross-paradigm prediction equivalence under enforced anti-leakage, Rampart fills this gap by unifying benchmarking, cross-paradigm execution, anti-leakage validation, and reproducibility within a single artifact.

3. The Rampart Framework

A Rampart execution takes as input a raw time-series panel (indexed by entity and time), a target column, a pipeline specification, and one or more paradigms under evaluation. As output, it produces per-fold predictions and metrics; per-stage latency and resource-consumption measurements; anti-leakage reports; and serialized Parquet/JSON artifacts with execution-environment snapshots.

The three paradigms represent distinct execution models, not an exhaustive set. Apache Spark is excluded because it targets multi-node clusters, whose shuffle, partition skew, and broadcast joins reflect deployment infrastructure rather than the execution paradigm; pandas because it belongs to the same DataFrame family as Polars and serves as the common in-memory layer into which all paradigms materialize before model fitting.

Rampart is organized into a layered architecture (Figure 1): raw data normalization into immutable Parquet artifacts; paradigm-specific ETL for standardized fold inputs;

[2]DuckDB: `https://duckdb.org/`
[3]Polars: `https://pola.rs/`
[4]Dask: `https://www.dask.org/`

anti-leakage verification of P1–P5, halting on P1–P4 violations; and benchmark execution with latency/resource instrumentation, statistical analysis, and Parquet/JSON report persistence. Configuration parameters, random seeds, and environment metadata are centralized to reduce hidden technical debt [Sculley et al. 2015] and support reproducibility. Fold generation and anti-leakage verification are shared across all paradigms and cannot be overridden.

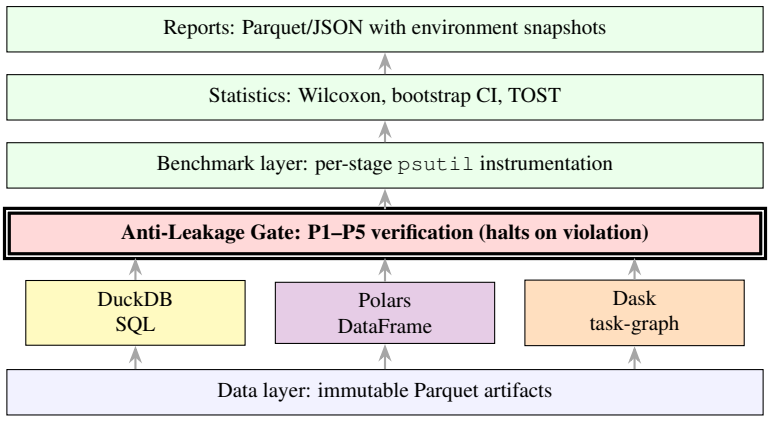

Figure 1. Rampart's layered architecture.

Experimental reproducibility is ensured through centralized random seeds, sequential training, version-pinned environments, and a numerical tolerance of 10^{-9}. Wall-clock latency is measured at each stage boundary (exact); CPU/memory/I/O are sampled at 100 ms intervals, so per-stage resource attribution is approximate for sub-100 ms stages.

3.1. Anti-Leakage Protocol

Rampart implements walk-forward temporal validation, a block cross-validation strategy for time-structured data [Roberts et al. 2017], and verifies five anti-leakage properties extending the Kapoor–Narayanan taxonomy [Kapoor and Narayanan 2023] and the preprocessing analysis of Semmelrock et al. [Semmelrock et al. 2025]. Each fold k splits the timeline into training, validation, and test windows, written $[t_s^k, t_e^k]$, $[v_s^k, v_e^k]$, and $[s_s^k, s_e^k]$, where subscripts s and e denote the start and end of a window and g the gap between them. Temporal leakage then arises whenever features over $[t_s^k, t_e^k]$ use observations from $t > t_e^k$, or when these windows are not separated by at least g.

The five properties are: **P1 (temporal ordering)**, requiring $t_e^k < v_s^k$ and $v_e^k < s_s^k$ for each fold; **P2 (gap sufficiency)**, requiring at least g temporal units between splits, with $g = 2$ years for the annual panels used here; **P3 (data separation)**, ensuring disjoint temporal periods and excluding features derived from the target or strongly correlated with it, using a configurable Pearson cutoff of $|r| > \rho_{\max}$, with $\rho_{\max} = 0.80$ in this study; **P4 (feature-selection scope)**, restricting correlation and collinearity filters to data available only up to t_e^1 of the first fold; and **P5 (preprocessing scope)**, ensuring that scaling and imputation statistics are fitted exclusively on training data.

Hyperparameter selection uses only the validation set, preventing leakage during model selection. After this stage, the final model is retrained using both training and validation data before evaluation on the test set. P1–P4 violations halt execution and generate

diagnostic outputs (*hard enforcement*); P5 is handled through documented contracts and unit tests (*soft enforcement*).

4. Experimental Evaluation

For each paradigm, the same pipeline runs across walk-forward folds. It has four stages: *Processing* loads Parquet files, creates temporal features (lags, rolling means, and year fixed effects), and applies correlation-based feature selection; *Setup* defines train/validation/test splits for entity-year prediction; *Baseline* fits global mean, linear trend, naive lag-based $\hat{y}_t = y_{t-1}$, and cross-entity mean models; and *Hierarchical* fits a panel Ridge regression model [Hoerl and Kennard 1970] with α selected by walk-forward cross-validation over $\{10^{-3}, 10^{-2}, 10^{-1}, 1, 10\}$, with entity fixed effects represented by *dummy* variables and a shared coefficient capturing the cross-entity signal.

Experiments use an Azure `D4s_v3` VM (4 vCPUs, 16 GB RAM) running Linux and Python 3.11. Each stage runs $n = 10$ times after two warm-ups, with paradigm order randomized via Fisher–Yates and `gc.collect()` invoked between runs. Statistical analyses use Wilcoxon signed-rank with Bonferroni correction across 12 stage-by-paradigm-pair tests ($\alpha = 0.004$), Cohen's d_z effect sizes, 10,000-resample bootstrap CIs, and two one-sided tests (TOST) for equivalence [Lakens et al. 2018].

Rampart is evaluated on two public time-series education panels (Table 1). The World Bank panel[5] covers macroeconomic education indicators for Latin American countries. The INEP Censo Escolar panel[6] covers Brazilian municipal education indicators. In both panels, short temporal gaps are filled with entity-level *forward fill* using only past observations, and records with missing targets are removed.

Table 1. Dataset dimensions and walk-forward fold configurations.

	World Bank	**INEP Censo Escolar**
Domain	Latin American macro education	Brazilian municipal education
Entities × years	32 × 24 (2000–2023)	5,564 × 18 (2007–2024)
Observations / features	768 / 22	94,061 / 22
Target	secondary completion rate	lower-secondary dropout rate
Folds (train/val/test, gap)	9 (8/2/2, 2 yr)	8 (5/1/1, 2 yr)
Missing rows	<1%	≈ 6%

4.1. Latency Analysis Across Paradigms

Latency is analyzed across data-preparation and model-training stages to assess how each paradigm behaves as dataset scale increases. These results can be observed in Table 2.

Polars leads the Processing stage on both panels by more than one order of magnitude through its *lazy* planner, which pushes filters, joins, and projections to Arrow kernels. DuckDB incurs buffer-materialization costs; Dask adds task-graph overhead even for in-memory workloads.

On World Bank (768 rows), all paradigms convert to pandas before scikit-learn, limiting ML-stage differences to $1.6\times$; Dask underperforms because scheduler initialization dominates. On INEP (94K rows), Dask leads both ML stages ($2.0\times$ and $2.2\times$) by

[5]World Bank Open Data: `https://data.worldbank.org/topic/education`

[6]Censo Escolar (INEP): `https://www.gov.br/inep/pt-br/areas-de-atuacao/pesquisas-estatisticas-e-indicadores/censo-escolar`

Table 2. Latency per stage on WB and INEP panels ($n = 10$, mean $\pm$ SD, seconds). Bold: per-stage winner.

Dataset	Stage	DuckDB	Polars	Dask	p
WB	Processing	0.30±0.26	**0.011±0.0003**	1.33±0.04	<0.001
	Setup	0.42±0.03	**0.083±0.002**	306.7±0.5	<0.001
	Baseline	2.56±0.03	**1.89±0.02**	3.05±0.03	<0.001
	Hierarch.	**33.6±0.09**	33.8±0.09	35.4±0.11	<0.001
	Total	36.9	**35.8**	343.5	<0.001
INEP	Processing	1.55±0.05	**0.072±0.003**	470.6±9.0	<0.001
	Setup	0.91±0.04	**0.38±0.01**	483.8±2.1	<0.001
	Baseline	910.3±8.6	886.1±6.4	**445.1±4.0**	<0.001
	Hierarch.	2233.3±15.1	2146.3±18.4	**998.4±7.2**	<0.001
	Total	3146.1	3033.0	**2397.9**	<0.001

p: Wilcoxon signed-rank with Bonferroni correction ($\alpha = 0.004$); peak RSS for WB: 460–480 MB.

caching partitions across folds, amortizing setup costs and yielding the best end-to-end latency ($1.3\times$ gain).

4.2. Prediction Equivalence as a Validity Clause

Latency comparisons are meaningful only when pipelines produce identical predictions. $\Delta = 0$ is treated as necessary: under the same data, seeds, and environment, deterministic ETL and ML stages should yield identical outputs, so any difference indicates a silent pipeline inconsistency.

During Rampart's development, this validation halted execution after a fold-ordering inconsistency in Dask's `dask.compute()` produced $\Delta \approx 0.04$ in Baseline R^2. After the fix, all paradigms produced bitwise-identical predictions on both datasets ($\Delta = 0.0$ across folds, baseline models, and metrics, including R^2, MASE [Hyndman and Koehler 2006], and WAPE). Bootstrap 95% confidence intervals were $[0.0, 0.0]$ within a smallest effect size of interest (SESOI) of $\delta_{R^2} = 0.01$, defined as the smallest practically relevant R^2 change for educational outcome prediction at this scale.

5. Discussion

The most informative outcome is not the latency profile itself, but what Rampart's anti-leakage gate caught during development: a fold-ordering inconsistency in Dask's `dask.compute()` produced misaligned predictions and $\Delta \approx 0.04$ in Baseline R^2 (about $4\times$ the SESOI) before execution was halted. After the fix, all paradigms produced bitwise-identical predictions across folds, baselines, and metrics, validating prediction equivalence as a discriminating condition that exposed the bug at the equivalence-check stage rather than leaving it as an unrecognized error in the latency tables.

The scale-dependent crossover near 10^4 rows aligns with the COST observation [McSherry et al. 2015] that distributed infrastructure imposes fixed overheads unjustified at small scale. Dask is inefficient for small workloads due to scheduler start-up, but amortizes this cost in iterative ML workloads through cached partition reuse.

Paradigm selection depends on dataset scale: in-process engines (Polars/DuckDB) suffice below $\approx 10^4$ rows; above that range, Dask dominates ML stages, though Polars still leads data preparation. The crossover is bracketed by only two points (WB at 768

and INEP at 94,061 rows), so $\sim 10^4$ is an order-of-magnitude indication rather than a calibrated boundary; Rampart can refine it with intermediate workloads.

Several threats to validity remain. Internally, $n = 10$ leaves the Bonferroni threshold ($\alpha = 0.004$) close to the minimum reachable Wilcoxon p-value ($\sim$0.002); the OS page cache is not flushed between runs, so latencies reflect sustained throughput rather than cold-cache behavior, and the single-node VM may suffer noisy-neighbor effects, partly mitigated by the low CV of the model-training stages. Construct validity is bounded by stage-level latency and peak RSS, which miss tail behavior and may underestimate off-heap allocations such as Arrow buffers and memory-mapped Parquet files. Externally, both panels are annual education data, leaving streaming, skewed OLAP joins, high-cardinality keys, and graph workloads outside the evaluated scope. Finally, despite 10,000 bootstrap resamples, inference is bounded by $n = 10$, so equivalence indicates compatibility with $\Delta = 0$ within the tested folds rather than proof beyond them [Lakens et al. 2018].

6. Conclusion and Future Work

Rampart was presented as an open-source framework for benchmarking machine-learning pipelines across data-processing paradigms, combining automatic leakage prevention, bitwise-reproducible execution, and latency comparison under equivalent conditions. Prediction equivalence acts as a validity clause for cross-paradigm comparison.

Experiments on two public time-series panels revealed scale-dependent behavior: Polars led data preparation through Arrow-based *lazy* execution, while Dask led end-to-end above $\approx 10^4$ rows through cached partition reuse. The anti-leakage gate also caught a non-deterministic fold-ordering bug in Dask; after correction, all three paradigms produced bitwise-identical predictions ($\Delta = 0.0$), demonstrating the discriminating role of prediction equivalence.

Adding Dask to the DuckDB+Polars base required 18.2% additional SLOC with zero base-class changes. Multi-node evaluation, datasets larger than main memory, and workloads beyond the education domain remain open.

Use of Generative AI

Claude (Anthropic) supported language revision and drafting; the authors remain responsible for the scientific content, design, analysis, and references.

Acknowledgments

This work was supported by the Brazilian research funding agency FUNDECT (Call No. 42/2024).

References

Brücke, C., Härtling, P., Palacios, R. D. E., et al. (2023). Tpcx-ai - an industry standard benchmark for artificial intelligence and machine learning systems. *Proc. VLDB Endow.*, 16(12):3649–3661.

de Oliveira, R., Lifschitz, S., Kalinowski, M., et al. (2020). Evaluating database self-tuning strategies in a common extensible framework. In *Anais do XXXV Simpósio Brasileiro de Bancos de Dados*, pages 97–108, Porto Alegre, RS, Brasil. SBC.

Gundersen, O. E. and Kjensmo, S. (2018). State of the art: Reproducibility in artificial intelligence. *Proceedings of the AAAI Conference on Artificial Intelligence*, 32(1).

Harby, A. A. and Zulkernine, F. (2025). Data lakehouse: A survey and experimental study. *Information Systems*, 127:102460.

Hoerl, A. E. and Kennard, R. W. (1970). Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12(1):55–67.

Hyndman, R. J. and Koehler, A. B. (2006). Another look at measures of forecast accuracy. *International Journal of Forecasting*, 22(4):679–688.

Kapoor, S. and Narayanan, A. (2023). Leakage and the reproducibility crisis in machine-learning-based science. *Patterns*, 4(9):100804.

Kiehn, F., Schmidt, M., Glake, D., et al. (2022). Polyglot data management: State of the art & open challenges. *Proceedings of the VLDB Endowment*, 15(12):3750–3753.

Lakens, D., Scheel, A. M., and Isager, P. M. (2018). Equivalence testing for psychological research: A tutorial. *Advances in Methods and Practices in Psychological Science*, 1(2):259–269.

McSherry, F., Isard, M., and Murray, D. G. (2015). Scalability! but at what cost? In *Proceedings of the 15th USENIX Conference on Hot Topics in Operating Systems*, HOTOS'15, page 14, USA. USENIX Association.

Nambiar, R. O. and Poess, M. (2006). The making of tpc-ds. In *Proceedings of the 32nd International Conference on Very Large Data Bases*, VLDB '06, pages 1049–1058. VLDB Endowment.

Pineau, J., Vincent-Lamarre, P., Sinha, K., et al. (2021). Improving reproducibility in machine learning research (a report from the NeurIPS 2019 reproducibility program). *Journal of Machine Learning Research*, 22(164):1–20.

Roberts, D. R., Bahn, V., Ciuti, S., et al. (2017). Cross-validation strategies for data with temporal, spatial, hierarchical, or phylogenetic structure. *Ecography*, 40(8):913–929.

Rupprecht, L., Davis, J., Arnold, K., et al. (2020). Improving reproducibility of data science pipelines through transparent provenance capture. *Proceedings of the VLDB Endowment*, 13(12):3354–3368.

Sculley, D., Holt, G., Golovin, D., et al. (2015). Hidden technical debt in machine learning systems. In Cortes, C., Lawrence, N., Lee, D., Sugiyama, M., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc.

Semmelrock, H., Ross-Hellauer, T., Kopeinik, S., et al. (2025). Reproducibility in machine-learning-based research: Overview, barriers, and drivers. *AI Magazine*, 46(2):e70002.

Soares, E., Souza, R., Thiago, R., et al. (2021). A recommender for choosing data systems based on application profiling and benchmarking. In *Anais do XXXVI Simpósio Brasileiro de Bancos de Dados*, pages 265–270, Porto Alegre, RS, Brasil. SBC.

Zaharia, M. A., Chen, A., Davidson, A., et al. (2018). Accelerating the machine learning lifecycle with mlflow. *IEEE Data Eng. Bull.*, 41:39–45.