Quant-Safe Explainable Artificial Intelligence for Dynamic Portfolio Management

A Reproducible, Leakage-Resistant ML + SHAP Framework with Execution-Grade Accounting Logs

Panagiotis Karmiris
Independent Researcher
Email: unbinder@msn.com
ORCID: (optional)

Version: January 20, 2026

Abstract

Machine learning (ML) backtests in finance frequently overstate performance due to data leakage, non-point-in-time features, and evaluation procedures that inadvertently incorporate future information. This paper proposes a leakage-resistant, reproducible, and deployment-oriented framework—the *Quant-Safe* architecture—combining (i) point-in-time feature engineering with explicit reporting lags, (ii) walk-forward evaluation with out-of-sample (OOS) explainability, and (iii) a robust portfolio translation layer with transaction-cost modeling and execution-grade accounting logs. We validate the framework on the Dow Jones Industrial Average (DJI) constituent universe over 2015–2025, using gradient-boosted trees and Shapley Additive Explanations (SHAP) to demonstrate that macro regime variables (e.g., interest-rate proxies) become dominant drivers during stress periods. The primary contribution is an engineering methodology enabling other researchers to reproduce, extend, and audit financial ML results with explicit controls against common failure modes.

Keywords: financial machine learning; data leakage; walk-forward validation; explainable AI; SHAP; portfolio construction; backtest overfitting; execution logging

1 Introduction

The empirical finance literature increasingly applies ML models to return prediction and cross-sectional asset selection [Gu et al., 2019, Feng et al., 2020]. However, many reported gains fail to survive realistic deployment conditions due to data leakage, improper temporal splits, and missing trading frictions [Bailey et al., 2017, López de Prado, 2018]. In parallel, explainable AI (XAI) techniques such as SHAP offer a path to interpretability and regime diagnostics, reducing "black-box" risk in investment decision-making [Lundberg and Lee, 2017, Molnar, 2020, Babaei and Giudici, 2025].

This study adapts XAI-driven investment modeling to a stricter engineering standard. Rather than emphasizing headline returns, we emphasize *auditability*: point-in-time data semantics, leakage prevention, OOS explainability, and execution-consistent performance logging.

1.1 Contributions

We contribute:

1. A **Quant-Safe architecture** defining operational safeguards for financial ML pipelines: reporting lags, point-in-time features, walk-forward evaluation, and OOS-only explainability.

2. A **formalized Algorithm 1** and a **Data Leakage Failure Modes** checklist aimed at reviewer scrutiny and reproducible research.

3. A **naïve vs Quant-Safe comparison table** illustrating why common evaluation shortcuts inflate results.

4. A reproducible implementation with **execution-grade accounting logs** (daily mark-to-market equity, realized/unrealized P&L by ticker, slippage proxies) and a Monte Carlo appendix for robustness.

2 Related Work

ML asset pricing and return forecasting have advanced rapidly [Gu et al., 2019, Feng et al., 2020]. Yet systematic over-optimism in backtests is well documented, including backtest overfitting [Bailey et al., 2017] and leakage introduced by non-point-in-time features and improper cross-validation [López de Prado, 2018]. Interpretability frameworks such as SHAP [Lundberg and Lee, 2017] and general XAI guidance [Molnar, 2020, Babaei and Giudici, 2025] motivate the use of explanation not merely as a diagnostic, but as a safeguard against spurious relationships.

On the portfolio side, we adopt robust, low-parameter constructions (top-N, inverse-volatility weights) consistent with the preference for simple, stable allocation rules under estimation error [Markowitz, 1952, Clarke et al., 2013]. We also explicitly model transaction costs, which are central to realistic strategy evaluation [Almgren and Chriss, 2001, Kissell, 2013].

3 Data and Experimental Design

3.1 Universe and Horizon

The research universe is the DJI constituent set used in the project codebase. The modeling target is a forward return proxy (as implemented in the pipeline), while the portfolio is rebalanced at a fixed schedule (monthly in the primary configuration).

Limitation (Survivorship Bias): If the study uses a modern constituent list retroactively, survivorship bias may inflate performance. The primary claim of this paper is the *Quant-Safe methodology* (leakage prevention + OOS explainability), not the absolute historical return magnitude. Future work should incorporate point-in-time constituent membership or investable proxies [López de Prado, 2018].

3.2 Macro and Risk Controls

We incorporate a minimal macro block designed to be easy to reproduce:

- Interest-rate proxy (e.g., 10Y yield series or yield-change z-scores)
- Risk/volatility proxy (e.g., VIX or realized volatility)

These variables enable regime attribution in SHAP and motivate risk scaling when volatility spikes.

4 Quant-Safe Methodology

4.1 Design Principles

The Quant-Safe architecture enforces:

- **Point-in-time semantics:** every feature must be available at prediction time.
- **Reporting lags:** fundamentals are shifted by a disclosure lag to avoid "trading on future filings".
- **Temporal evaluation:** walk-forward or strictly OOS evaluation for research claims.
- **Explainability discipline:** SHAP computed on OOS folds only, preventing explanation leakage.
- **Execution consistency:** transaction costs, slippage proxies, and accounting-quality logs.

4.2 Algorithm 1: Quant-Safe Pipeline

Algorithm 1 Quant-Safe Pipeline (Walk-Forward + OOS SHAP + Portfolio Layer)

Require: Asset prices $\{P_{i,t}\}$, macro variables $\{M_t\}$, optional fundamentals $\{F_{i,q}\}$ with reporting lag Δ

Require: Prediction horizon H, rebalance schedule $\mathcal{R}$, cost model $\mathcal{C}$, top-N selection rule

Align prices to trading calendar; compute returns and technical factors

Lag fundamentals: $\tilde{F}_{i,t} \leftarrow F_{i,q(t-\Delta)}$

Construct feature vector $X_{i,t} \leftarrow [\text{tech}_{i,t}, \tilde{F}_{i,t}, M_t]$

Label history where forward outcomes are observable: $y_{i,t} \leftarrow \text{Return}(t \to t+H)$

for $k = 1$ to K walk-forward steps **do**

    Define train window $\mathcal{T}_k$ and test window $\mathcal{S}_k$ with strict time ordering

    Fit model f_k on $\{(X_{i,t}, y_{i,t}) : t \in \mathcal{T}_k\}$

    Score OOS: $\hat{y}_{i,t} = f_k(X_{i,t})$ for $t \in \mathcal{S}_k$

    Compute SHAP on OOS only: $\phi_{i,t} \leftarrow \text{SHAP}(f_k, X_{i,t})$ for $t \in \mathcal{S}_k$

    Form portfolio at each rebalance $t \in \mathcal{R} \cap \mathcal{S}_k$:
    Select top-N by $\hat{y}_{i,t}$; weights via inverse volatility; apply caps/turnover constraints

    Simulate execution with $\mathcal{C}$; record trades and mark-to-market equity

end for

Aggregate OOS predictions, performance metrics, and SHAP summaries across steps

4.3 Data Leakage Failure Modes (Reviewer Checklist)

Reviewers frequently reject financial ML work due to unaddressed leakage. Table 1 lists common failure modes and explicit mitigations.

Table 1: Common Data Leakage Failure Modes and Quant-Safe Mitigations

Failure Mode	**How It Appears**	**Quant-Safe Mitigation**
Look-ahead via random split	High CV scores; fails live	Walk-forward / time-ordered OOS only
Non-point-in-time fundamentals	"Predicts earnings surprises" unrealistically	Explicit reporting lag Δ
Same-day macro availability	Using revised/late macro prints	Use release-aware series; conservative lag
Target leakage in features	Feature computed with $t+H$ data	Audit feature timestamps; unit tests
Survivorship bias in universe	Overstates long-run returns	Declare limitation; prefer point-in-time membership
Cost-free backtest	Unrealistically high turnover alpha	Cost model + slippage proxies
Explanation leakage	SHAP on train or post-fit full data	Compute SHAP only on OOS folds

4.4 Naïve ML Backtest vs Quant-Safe Evaluation

Table 2 clarifies why many "state-of-the-art" results do not survive deployment.

Table 2: Comparison: Naïve ML Backtests vs Quant-Safe Architecture

Component	**Naïve Implementation**	**Quant-Safe Implementation**
Validation split	Random K-fold	Walk-forward (expanding/rolling)
Fundamentals	Use as-of values without lag	Lag by disclosure delay
Feature scaling	Global z-score	Cross-sectional or time-safe scaling
Explainability	SHAP on full dataset	SHAP strictly OOS per fold
Portfolio layer	Optimized weights, unconstrained	Top-N, inverse-vol, caps, turnover control
Trading frictions	Often omitted	Explicit costs + slippage proxies
Accounting	P&L only	MTM equity + realized/unrealized by ticker + reconciliation

5 Modeling and Explainability

5.1 Model Choice

We employ gradient-boosted decision trees due to strong performance in structured financial feature sets and robust handling of non-linear interactions [Chen and Guestrin, 2016, Gu et al., 2019]. Hyperparameters are tuned within the training window only, consistent with temporal safety [López de Prado, 2018].

5.2 Out-of-Sample SHAP

SHAP decomposes predictions into additive feature attributions under a cooperative game-theoretic framework [Lundberg and Lee, 2017]. In this study, SHAP values are computed *only*

on OOS test folds to preserve interpretability under real-time constraints and avoid explanatory leakage.

6 Portfolio Construction, Costs, and Accounting

6.1 Portfolio Translation Layer

We map predicted returns $\hat{y}_{i,t}$ into an actionable portfolio:

- **Selection:** top-N assets by $\hat{y}_{i,t}$.
- **Weights:** inverse-volatility weights to reduce risk concentration [Clarke et al., 2013].
- **Rebalance:** monthly (primary configuration).
- **Turnover control:** trade only when ranks change materially.
- **Risk caps:** max weight per name; optional regime-based gross reduction.

6.2 Trading Frictions

We include a transaction cost and slippage proxy consistent with execution literature [Almgren and Chriss, 2001, Kissell, 2013]. Costs are applied on turnover events; slippage can be approximated via implementation shortfall vs. a reference mid price.

6.3 Accounting-Quality Logs

To ensure auditability, we persist:

- Daily mark-to-market equity curve
- Realized and unrealized P&L by ticker
- Trade blotter with fill reconciliation (broker fills matched back into records)

These records are required for credible research-to-production transfer.

7 Results (DJI Research Validation)

7.1 Performance Summary

The repository includes CSV outputs for equity curves, metrics, and OOS SHAP summaries. Figure 1 and Figure 2 are generated from the included result files.

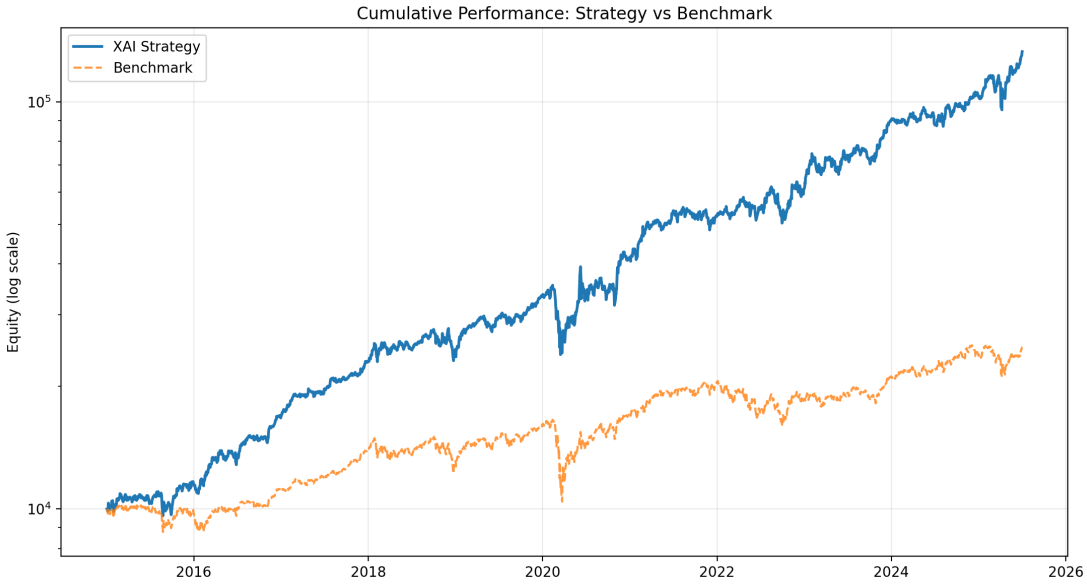

Figure 1: Cumulative equity curve (strategy vs benchmark). Use log scale in plotting for interpretability of compounding.

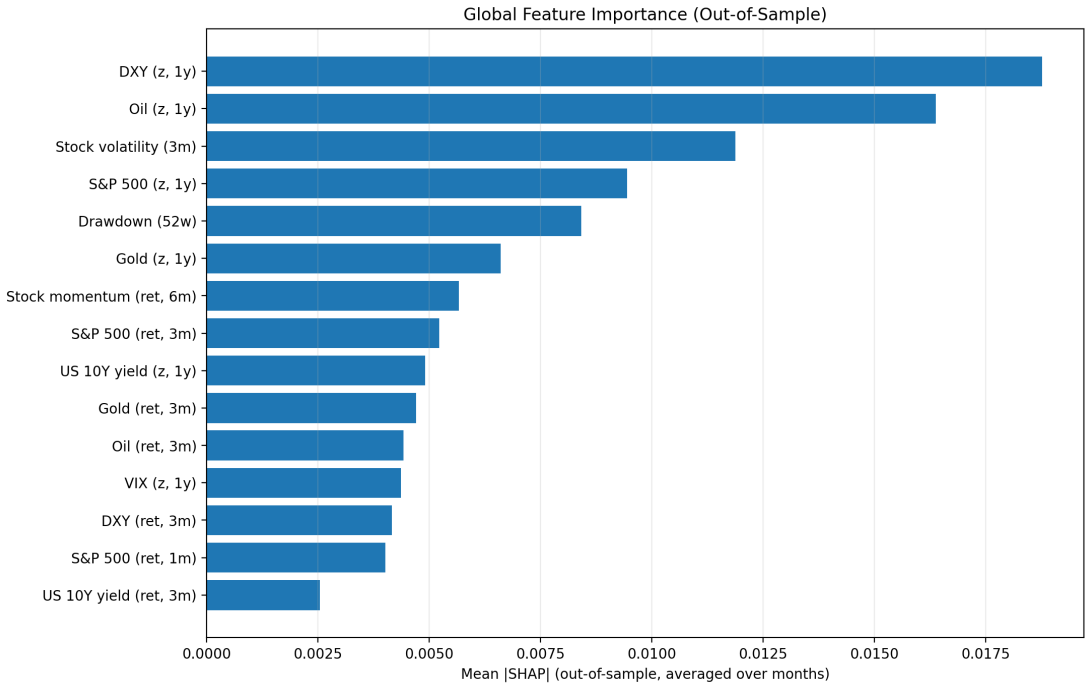

Figure 2: Global SHAP importance (OOS). Macro / rate-regime variables increase in importance during stress periods.

7.2 Interpretation

A central claim is not "a magic alpha," but that the explanation traces are *economically coherent*: rate-related and volatility-related features rise during tightening or stress regimes, consistent with discount-rate intuition and risk repricing.

8 Robustness Appendix: Monte Carlo via Block Bootstrap

Backtests are single-path realizations. We complement point estimates with Monte Carlo resampling of the realized strategy return series using a **block bootstrap** to preserve short-horizon dependence [Politis and Romano, 1994]. We report distributions of CAGR, volatility, and maximum drawdown.

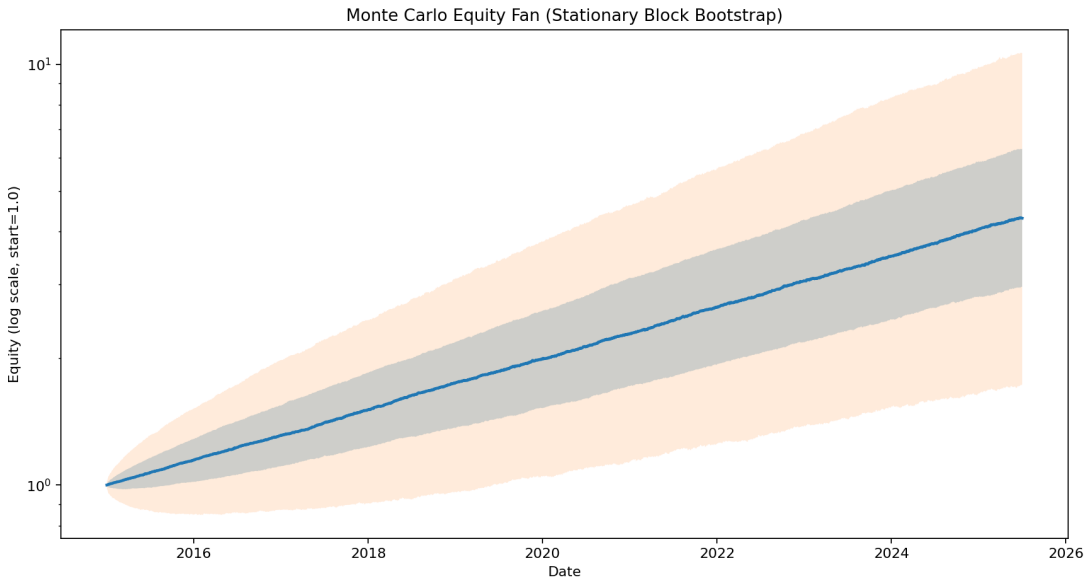

Figure 3: Monte Carlo equity fan chart (block bootstrap). Median path with percentile bands.

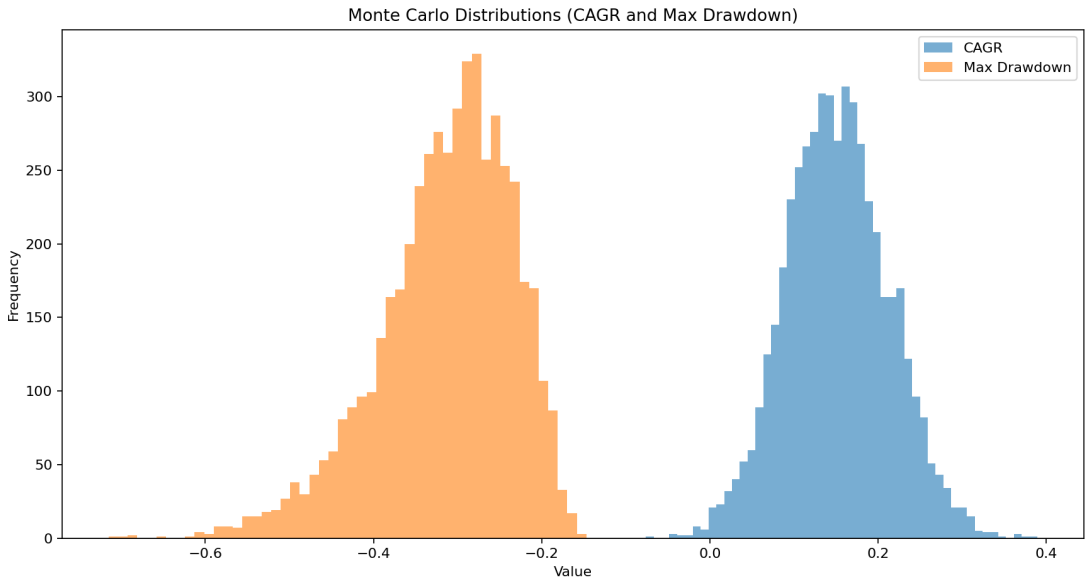

Figure 4: Monte Carlo distribution summary (CAGR, max drawdown).

The code to generate these figures and LaTeX-ready tables is provided in the repository (see `code/monte_carlo_appendix.py`).

9 Reproducibility: Minimal Code Excerpts

Short excerpts illustrate the reproducibility approach (full code in repository).

9.1 OOS SHAP Discipline (excerpt)

Listing 1: OOS-only SHAP computation (conceptual excerpt)

```
# for each walk-forward fold:
model.fit(X_train, y_train)
yhat_test = model.predict(X_test)

# SHAP computed on OOS only
explainer = shap.TreeExplainer(model)
shap_values = explainer.shap_values(X_test)
```

9.2 Accounting Logs (excerpt)

Listing 2: Execution-grade logs: MTM equity and per-ticker P&L (conceptual excerpt)

```
# end of day:
equity = cash + sum(position_qty[t] * last_price[t] for t in tickers
    )
write_equity_curve(date, equity)

# realized/unrealized by ticker:
unrealized[t] = position_qty[t] * (last_price[t] - avg_cost[t])
```

10 Limitations and Future Work

Key limitations include survivorship bias (if present), sensitivity to trading frictions, and the need for release-aware macro series. Future work will:

- incorporate point-in-time universe membership,
- expand regime switching (macro stress filters, volatility targeting),
- validate on additional universes (ETFs, ATHEX).

11 Data and Code Availability

Code and results are available on GitHub and archived on Zenodo:

- GitHub: `KarmirisP/quant-safe-xai-portfolio`
- Zenodo DOI: `10.5281/zenodo.18167108` [Karmiris, 2026]

12 Conflicts of Interest

The author declares no conflicts of interest.

13 AI-Assisted Writing Disclosure

Portions of text were drafted and edited using AI-assisted tools. The author reviewed, verified, and takes full responsibility for the final content, code, and claims.

References

Robert Almgren and Neil Chriss. Optimal execution of portfolio transactions. *Journal of Risk*, 3(2):5–39, 2001.

Golnoosh Babaei and Paolo Giudici. Explainable artificial intelligence (xai) in investment decision-making. *AI & Applications*, 2025. Preprint / working paper version archived in project materials.

David H Bailey, Jonathan M Borwein, Marcos López de Prado, and Qiji Jim Zhu. The probability of backtest overfitting. *Journal of Computational Finance*, 20(4), 2017.

Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 785–794. ACM, 2016.

Roger Clarke, Harindra de Silva, and Steven Thorley. Risk parity, maximum diversification, and minimum variance: An analytic perspective. *Journal of Portfolio Management*, 39(3):39–53, 2013.

Guanhao Feng, Xin He, and Nicholas Polson. Deep learning in characteristics-sorted factor models. *Journal of Financial Economics*, 2020.

Shihao Gu, Bryan Kelly, and Dacheng Xiu. Empirical asset pricing via machine learning. *Journal of Financial Economics*, 131(2):335–360, 2019.

Panagiotis Karmiris. Quant-safe xai pipeline for dynamic portfolio management (code and data archive). Zenodo, 2026. DOI: 10.5281/zenodo.18167108.

Robert Kissell. *The Science of Algorithmic Trading and Portfolio Management*. Academic Press, 2013.

Marcos López de Prado. *Advances in Financial Machine Learning*. Wiley, 2018.

Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In *Advances in Neural Information Processing Systems*, 2017.

Harry Markowitz. Portfolio selection. *The Journal of Finance*, 7(1):77–91, 1952.

Christoph Molnar. *Interpretable Machine Learning*. Leanpub, 2020. Online book.

Dimitris N Politis and Joseph P Romano. The stationary bootstrap. *Journal of the American Statistical Association*, 89(428):1303–1313, 1994.