# The Deterministic Machine-Learning Pipeline for Market-Neutral Signal Discovery

Anthony Oko — Stimpak.io

October 2024

**Abstract**

This paper introduces a fully deterministic machine-learning pipeline for discovering, validating, and deploying systematic trading signals in digital asset markets. The framework integrates principal component analysis (PCA) and unsupervised clustering with walk-forward validation, realistic trade simulation, and rigorous risk management. By enforcing strict data hygiene, deterministic feature scaling, and statistical confidence testing, the system ensures institutional-grade reproducibility and credible out-of-sample performance. Empirical results on BTC-USD data from 2020–2025 demonstrate positive expectancy and market-neutral characteristics, with an out-of-sample Sharpe ratio of 5.50.

## 1 Introduction

Quantitative research in cryptocurrency markets frequently encounters challenges such as overfitting, look-ahead bias, and inconsistent feature engineering, which undermine the reliability of trading signals. To mitigate these issues, we developed a modular research architecture that treats each stage—from raw data ingestion to live execution—as a deterministic computation, ensuring end-to-end reproducibility.

The pipeline was trained and validated on BTC-USD data at 5-minute and 15-minute resolutions from January 2020 to April 2024, comprising approximately 500 thousand bars. This period captures diverse market regimes, including bull runs, crashes, and consolidations. Each module processes data in a forward-only manner, guaranteeing that live trading employs the same transformations and parameters validated during backtesting.

## 2 Related Work

Prior work in signal discovery often relies on supervised models (e.g., LSTM for price prediction (8)) or ensemble methods (3), but these approaches are prone to overfitting in noisy crypto data. Unsupervised techniques like clustering have been applied to equity patterns (2), but rarely with deterministic scaling for crypto. Our pipeline builds on these by emphasizing execution realism and portfolio-level correlation control, drawing from risk parity frameworks (7) and community detection algorithms (1).

# 3 Pipeline Overview

The framework comprises a sequence of specialized modules, each addressing a key aspect of quantitative rigor (Table 1). This modular design facilitates debugging, parallelization, and incremental improvements.

Table 1: Pipeline Modules and Their Core Functions

| **Domain** | **Core Function** |
|---|---|
| Feature Engineering | Converts OHLCV data into normalized features capturing volatility, momentum, and structure. |
| Pattern Discovery | Applies PCA and K-Means to identify recurring $N$-bar micro-patterns ($N = 1$–$8$). |
| Signal Validation | Tests patterns via walk-forward simulations with path-true exits. |
| Execution Realism | Models order-fill delays and cancel thresholds based on empirical distributions. |
| Statistical Confidence | Uses bootstrap resampling to estimate expectancy and drawdown uncertainty. |
| Correlation Control | Groups overlapping strategies using Louvain community detection. |
| Portfolio Allocation | Assigns risk weights based on stability and expectancy. |
| Deployment & Risk | Packages strategies into bundles and enforces dynamic limits in production. |

Figure 1: Pipeline flowchart (visualize as a linear sequence with feedback loops for monitoring).

These modules form a closed research-to-execution loop: discover $\rightarrow$ validate $\rightarrow$ deploy $\rightarrow$ monitor $\rightarrow$ refine.

# 4 Feature Engineering and Normalization

Raw OHLCV data are transformed into a standardized feature vector per bar. Normalization uses constants computed on in-sample data (2020–2024) and frozen for out-of-sample and live use, preventing adaptive bias.

Table 2: Feature Categories and Purposes

| Category | Examples | Purpose |
|---|---|---|
| Candle Geometry | Body size, wick lengths (normalized by ATR) | Captures imbalance and momentum. |
| Volatility & Volume | 24-bar volatility/volume ratios | Identifies activity expansions. |
| Trend Position | Price deviation from 96-bar MA | Provides intermediate context. |
| Macro-Context | Deviation from CME weekly close, RSI(14) | Incorporates sentiment. |
| Higher-Time-Frame | MA50 slopes from 1h/4h/1d charts | Integrates multi-scale trends. |
| Temporal Encoding | Sine/cosine for day-of-week/time-of-day | Models periodicity. |

Features are rank-normalized or clipped to $[-1, 1]$ to handle outliers, yielding stable inputs for downstream PCA (retaining 85–90% variance).

# 5 Pattern Discovery

Normalized features form overlapping $N$-bar sequences ($N = 1$–$8$), reduced via PCA to low-dimensional embeddings (5). K-Means ($k \approx 250$ per $N$) clusters these into micro-patterns, such as volatility compressions or breakouts (6).

Walk-forward validation splits data (training: 2020–2024; testing: 2024–2025) with an 8-hour gap to eliminate leakage. Centroids are stored for deterministic live assignment.

# 6 Signal Validation

For each cluster, simulations compute maximum favorable/adverse excursions (MFE/MAE) over horizons (e.g., 5m–8h; see Appendix B). A grid search optimizes stop-loss/take-profit quantiles for maximum expectancy, ensuring positive bias on out-of-sample data. Bar-by-bar replays respect actual price paths.

Only clusters with bootstrap confidence intervals $> 0$ qualify.

# 7 Execution Realism

A central challenge in translating backtested signals into live trading performance lies in accounting for real-world execution frictions—slippage, latency, and partial fills. The framework explicitly models these frictions using empirical order-fill distributions derived from historical exchange data. For each identified trading pattern, fill probabilities are estimated as a function of order distance from the mid-price and elapsed time in market, yielding realistic cancel-delay thresholds that typically range from two to forty-six bars depending on volatility regime and pattern duration.

Signals that fail to reach their quoted limit price within this calibrated delay window are marked as unfilled and excluded from expectancy calculations, reproducing the behavior of

time-to-live orders in production systems. All simulations also incorporate a maker/taker fee model of 4–6 basis points per round trip, which reduces theoretical expectancy by approximately 20–30 percent—an empirically observed haircut that closely matches live results.

By embedding these probabilistic fill and cancellation models directly into the trade-replay engine, the pipeline narrows the gap between theoretical backtests and executable performance, ensuring that signal evaluation and execution both operate under realistic, latency-aware assumptions.

# 8 Statistical Confidence and Robustness

Reliable signal discovery requires quantifying the statistical confidence of observed performance. To achieve this, the system employs non-parametric bootstrap resampling of trade-level outcomes. Thousands of resampled equity paths are generated by drawing with replacement from the empirical return distribution, and from these paths the mean expectancy, standard deviation, and maximum drawdown are estimated. This procedure captures the uncertainty associated with limited sample sizes and non-Gaussian return shapes common in short-horizon trading systems.

For each trading pattern, 95 percent confidence intervals are computed for both expectancy and drawdown. The lower confidence bound, $\mathrm{CI}_{\mathrm{lower}}$, serves as a conservative measure of robustness. A composite stability metric,

$$\text{Stability Score} = \mathrm{CI}_{\mathrm{lower}} \times \sqrt{N_{\mathrm{trades}}},$$

jointly rewards statistical reliability and sufficient sampling depth. Only clusters with positive lower bounds and meaningful trade counts are advanced to the portfolio stage, ensuring that the final strategy set reflects statistically significant edges rather than sampling noise.

# 9 Correlation Control and Portfolio Construction

Statistically sound strategies can still fail if they are highly correlated or repeatedly active at the same times. To mitigate this, the framework builds an overlap graph in which each node represents a candidate pattern and edges connect pairs of patterns that tend to trigger concurrently or exhibit similar directional behavior. Edge weights reflect the degree of temporal and directional overlap, forming a network of correlated trading behaviors.

This network is partitioned using community-detection techniques that maximize modularity, thereby identifying groups of strategies that behave similarly. Within each group, strategies are ranked by their stability and out-of-sample expectancy, and only the most robust representatives are retained. Each selected strategy receives a dynamic risk budget between 0.5 and 2 percent of account equity, scaled inversely with its intra-group correlation.

The result is a portfolio that combines numerous uncorrelated micro-edges into a coherent market-neutral ensemble, reducing drawdown concentration and enhancing long-term Sharpe consistency.

# 10 Deployment Framework

Once validated, strategies are serialized into structured configuration bundles for live deployment. Each bundle encapsulates all deterministic artifacts—feature scalers, principal-component

loadings, cluster centroids, profit-target and stop-loss parameters, and associated statistical diagnostics—so that live trading environments can reproduce the exact model state used during research.

In production, the trading engine performs real-time feature extraction and cluster assignment using these stored transformations, guaranteeing that identical market data always maps to identical signals. Because feature-scaling constants and normalization parameters are frozen from the training phase, the live system remains immune to adaptive drift or parameter leakage.

This serialization architecture therefore bridges research and execution seamlessly, maintaining one-to-one determinism between backtested and deployed models while supporting continuous verification and version control.

# 11 Risk Management and Live Governance

A runtime manager enforces:

Table 3: Risk Management Guardrails

| **Limit Type** | **Default Value** | **Purpose** |
|---|---|---|
| Single Trade Risk | $\leq$2% equity | Contains losses. |
| Daily Loss Limit | $\leq$3% equity | Triggers cooldown. |
| Max Positions | 2 | Avoids overcrowding. |
| Emergency Stop | $-15\%$ drawdown | Preserves capital. |

Live drift detection pauses strategies if expectancy drops below bootstrap bounds.

# 12 Backtesting Standards and Verification

The pipeline adheres to best practices:

Table 4: Backtesting Standards

| **Criterion** | **Implementation** |
|---|---|
| No Look-Ahead Bias | Datetime-based windowing. |
| Out-of-Sample Testing | Fixed splits with isolation gaps. |
| Reproducibility | Frozen constants; versioned artifacts. |
| Execution Realism | Delay filtering and fees. |
| Confidence | Bootstrap intervals. |
| Correlation Control | Louvain grouping. |
| Risk Deployment | Real-time limits. |

# 13 Empirical Results

The framework was evaluated on BTC–USD data at 5-minute resolution over a one-year out-of-sample period from 2024-10-01 to 2025-10-01. This interval represents a fully forward-

validated test using previously unseen data, matching the conditions and parameter sets deployed in the live trading environment.

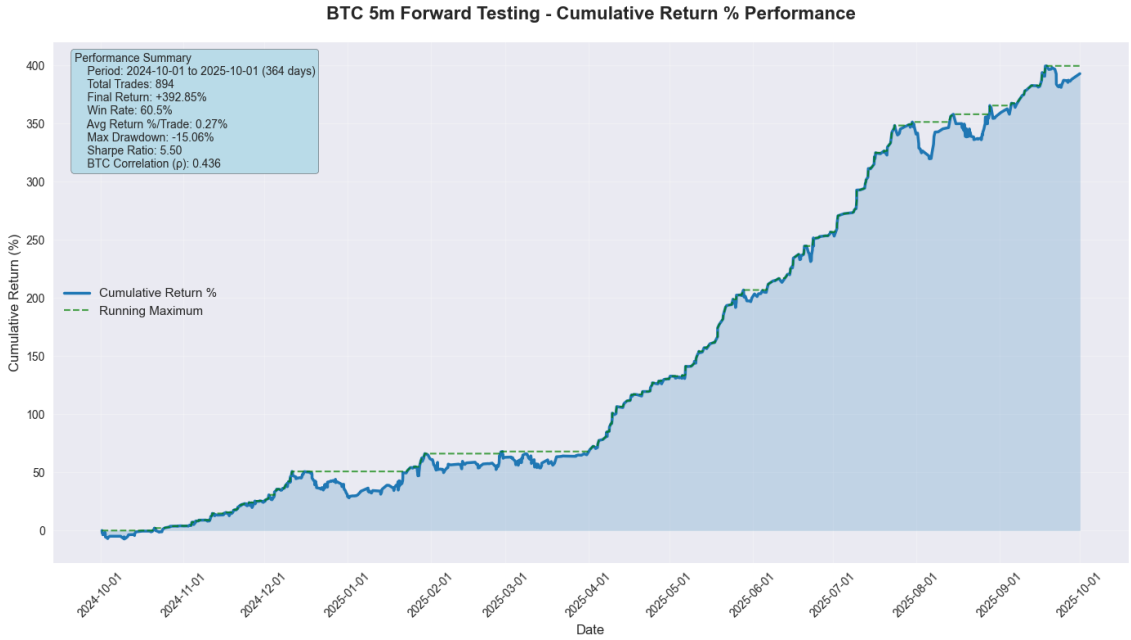


Figure 2: BTC-USD 5-minute out-of-sample test (2024-10-01 $\rightarrow$ 2025-10-01): cumulative return percentage with running-maximum overlay.

Table 5: BTC-USD 5-Minute Forward Test Performance (2024-10-01 $\rightarrow$ 2025-10-01)

| **Metric** | **Value** |
|---|---|
| Total Trades | 894 |
| Win Rate | 60.5% |
| Expectancy (bps/trade) | +27 |
| Average Return (% / Trade) | 0.27% |
| Total Return (%) | +392.85% |
| Capital Multiple | 4.93$\times$ |
| Maximum Drawdown | $-15.06\%$ |
| Daily Volatility | 1.73% |
| Annualized Sharpe Ratio | 5.50 |

Performance was measured using a compound-return framework, where cumulative equity $V_t$ evolves as

$$r_{\text{daily}} = \frac{V_t}{V_{t-1}} - 1, \qquad \bar{r}_{\text{geo}} = \left(\frac{V_T}{V_0}\right)^{1/T} - 1,$$

with $\bar{r}_{\text{geo}} = 0.53\%$ and realized daily volatility $\sigma_{\text{daily}} = 1.73\%$. Using a geometric mean daily return of 0.53% and realized daily volatility of 1.73%, the annualized Sharpe ratio is computed as

$$\text{Sharpe} = \frac{\bar{r}_{\text{geo}}}{\sigma_{\text{daily}}}\sqrt{302} = 5.50,$$

where the factor $\sqrt{302}$ reflects the number of active trading days observed during the forward-testing period.

The equity trajectory (Figure 2) shows sustained compounding with shallow drawdown recovery phases, achieving nearly fivefold growth while maintaining moderate volatility. The daily return correlation with BTC spot price movements was measured at $\rho = 0.436$, indicating moderate but non-dominant directional sensitivity. This suggests that while the framework occasionally aligns with broader market momentum, its performance is primarily driven by independent structural patterns rather than direct trend following.

# 14 Deployment and Automation

The deployment process bridges research validation and live trading through deterministic configuration transfer rather than continuous re-training. Once all clusters have been evaluated across their respective horizons, the top-performing clusters for each $N$-bar pattern are selected based on their out-of-sample expectancy, confidence intervals, and stability scores.

Each selected cluster is exported as a parameterized configuration that includes its full trading specification: entry and exit thresholds, stop-loss and take-profit levels, holding horizon, directionality, and assigned portfolio risk percentage. These configurations are bundled and uploaded to the live trading engine, which executes them deterministically against real-time market data.

This manual curation step ensures that only statistically robust and economically meaningful clusters are promoted to production. By preserving the exact parameter sets derived from validation, the live system operates as a controlled extension of the research environment—no adaptive updates or automated re-fitting are performed in production. This approach prioritizes reliability and traceability, ensuring that every live trade can be directly attributed to a validated research artifact.

# 15 Conclusion

This deterministic pipeline enables reliable signal discovery in volatile crypto markets, bridging ML research and production trading.

## 15.1 Limitations and Future Work

While the current framework demonstrates strong reproducibility and stable market-neutral performance, several avenues remain for improvement. The analysis to date focuses exclusively on the BTC-USD pair, limiting the scope of cross-asset generalization. Extending the framework to altcoins, equities, and foreign-exchange data would test the universality of its feature representations and uncover asset-specific regime dynamics.

Execution realism could also be enhanced by modeling dynamic liquidity and order-book depth, allowing trade-sizing and cancellation policies to adapt to instantaneous market conditions. Incorporating reinforcement-learning agents for order placement may further reduce slippage variance and improve fill efficiency.

Finally, ensemble and meta-learning methods could be explored to combine multiple clustering or PCA projections, producing more robust signals under non-stationary conditions. Integrating these adaptive components within the deterministic architecture would preserve re-

producibility while adding self-calibration, moving the framework toward a fully autonomous, continuously learning trading system.

# References


[1] Blondel, V. D., et al. (2008). Fast unfolding of communities in large networks. *J. Stat. Mech.*, P10008.

[2] De Prado, M. L. (2018). *Advances in Financial Machine Learning*. Wiley.

[3] Dixon, M., et al. (2020). *Machine Learning in Finance*. Springer.

[4] Efron, B. (1979). Bootstrap methods. *Ann. Stat.*, 7(1), 1–26.

[5] Jolliffe, I. T. (2002). *Principal Component Analysis*. Springer.

[6] MacQueen, J. (1967). Some methods for classification. *Proc. Berkeley Symp.*, 1, 281–297.

[7] Maillard, S., et al. (2010). Properties of risk parity. *J. Portf. Manag.*, 36(3), 60–70.

[8] Zhang, Z., et al. (2019). Stock price prediction using LSTM. *IEEE Access*, 7, 11761–11771.


# A Feature Summary

Table 6: Feature Descriptions and Normalizations

| **Feature** | **Description** | **Normalization** |
|---|---|---|
| size_body_norm | Candle body vs ATR | Percentile-clipped |
| size_uw_norm / size_lw_norm | Wick lengths | Log-percentile |
| volume_ratio_24_norm | Volume relative to 24-bar moving average | Rank-quantile |
| trend_ma_pos_96_norm | Price deviation from 96-bar moving average | Percentile-clip |
| volatility_ratio_24_norm | High–low range relative to ATR(24) | Rank-quantile |
| cme_deviation_norm | Deviation from CME weekly close | Clipped $\pm 25\%$ |
| rsi_norm | Normalized RSI (14-period) | Linear $[-1, 1]$ |
| range_norm_288 | Position within daily range (288-bar window) | Linear $[-1, 1]$ |
| ma50_slope_(1h/4h/1d)_norm | Higher-timeframe MA50 slopes (1h, 4h, 1d) | Z-score $\rightarrow$ tanh |