MASTER

MATHEMATICAL FINANCE

MASTER'S FINAL WORK

PROJECT

BITCOIN DIRECTION PREDICTION USING KOLMOGOROV-ARNOLD NETWORKS WITHIN THE AFML FRAMEWORK

PETR TERLETSKIY

MAY 2026

MASTER

MATHEMATICAL FINANCE

MASTER'S FINAL WORK

PROJECT

BITCOIN DIRECTION PREDICTION USING KOLMOGOROV-ARNOLD NETWORKS WITHIN THE AFML FRAMEWORK

PETR TERLETSKIY

SUPERVISION:

PROF. JOÃO AFONSO BASTOS

MAY 2026

To my family and God.

GLOSSARY

ADF Augmented Dickey-Fuller.
AFML Advances in Financial Machine Learning.
ANN Artificial Neural Network.
AR Autoregressive.
ATR Average True Range.
BTC Bitcoin.
CCI Commodity Channel Index.
CNN Convolutional Neural Network.
CPCV Combinatorial Purged Cross-Validation.
CUSUM Cumulative Sum.
DSR Deflated Sharpe Ratio.
EMH Efficient Market Hypothesis.
EWMA Exponentially Weighted Moving Average.
FFD Fixed-width Window Fractional Differentiation.
GRU Gated Recurrent Unit.
HAR Heterogeneous Autoregressive.
JEL Journal of Economic Literature.
KAN Kolmogorov-Arnold Network.
LR Logistic Regression.
LSTM Long Short-Term Memory.
LSTNet Long- and Short-term Time-series Network.
MACD Moving Average Convergence Divergence.
MDA Mean Decrease Accuracy.
MFI Money Flow Index.
MFW Master's Final Work.
MLP Multi-Layer Perceptron.
MVRV Market Value to Realized Value.
OBV On-Balance Volume.
OHLCV Open-High-Low-Close-Volume.
PBO Probability of Backtest Overfitting.
RF Random Forest.
RNN Recurrent Neural Network.
RSI Relative Strength Index.
SADF Supremum Augmented Dickey-Fuller.
SMT Sub- and Super-Martingale Test.
SVM Support Vector Machine.
TA Technical Analysis.
TBL Triple Barrier Labelling.
TCN Temporal Convolutional Network.
VIX CBOE Volatility Index.
XGBoost Extreme Gradient Boosting.

ABSTRACT, KEYWORDS, AND JOURNAL OF ECONOMIC LITERATURE (JEL) CODES

[1]

This dissertation evaluates whether Kolmogorov-Arnold Networks can predict Bitcoin price direction and extract interpretable symbolic formulas, benchmarked against five models within López de Prado's Advances in Financial Machine Learning framework. Using Combinatorial Purged Cross-Validation with the Deflated Sharpe Ratio and Probability of Backtest Overfitting, the study finds that no model achieves statistically significant predictive performance, consistent with market efficiency. The core contribution is KAN symbolic expression extraction, deriving closed-form decision functions from the trained network.

KEYWORDS: Bitcoin; Kolmogorov-Arnold Networks; Symbolic Extraction; Triple Barrier Labelling; Combinatorial Purged Cross-Validation; Financial Machine Learning.

JEL CODES: C45; C53; C58; G11; G17.

[1] Code, data, and LaTeX source are publicly available at https://github.com/pterletskiy/BTC_KAN_AFML.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

I would like to begin by thanking ISEG for the opportunity to pursue the Master's in Mathematical Finance, and for the academic environment in which this work could take shape.

I am deeply grateful to my supervisor, Professor João Afonso Bastos, for his support, guidance, and crucial insights he offered throughout the development of this challenging work.

To my Master's colleagues Rodrigo, Mariana and Jamila, thank you for the companionship that made the demanding moments lighter, and, above all, to Tiago, for being there from the start.

To the gym crew, thank you for always pushing my limits under the bar and beyond and, in doing so, helping me become a better, more disciplined, and more resilient human being.

To Henriques, Alves, Mixão, Chica, Afonso, Miguel, and Tomás, thank you for standing at my side and supporting me for more than a decade. Friendships that last that long become a foundation.

To Paulo, thank you for being so supportive and so patient, for teaching me so much along the way, and for your constant presence. What I have learned from you I will carry with me for life, and I am profoundly grateful for every conversation we have had.

To Filipa and Leonor, thank you for always being ready to help me, for supporting me, and for offering your insights when I needed them the most.

To my family supporting me from Russia, my grandparents Larissa, Yuriy, and Anatoliy, my uncle Nikita, my step-mother Alexandra, my aunt Lyubov, my brother Maxim and sister Liza, and my cousin Fyodor, thank you for the love, the kind words, and the support from afar.

To my father Mikhail, thank you for being so supportive from so far away, for always reminding me that I am capable of everything, and for reminding me of my last name. I hope I make you proud.

To my step-father Francisco, thank you for supporting my journey, for always being ready to give your helping hand, for the precious advice you have always shared, and for listening even when my difficulties were hard to put into words.

To my brother Nicolas, thank you for listening to my complaints, for your simple, clarifying insights, for silly jokes and moments that lightened the load, and for your unconditional support.

To grandmother Maria, whose warmth welcomed me as one of her own from my first day in Portugal, thank you for your love, for the moments shared, for the small lessons, and for every meal that turned a new country into home. I called you avó because you were one. I hope I make you proud.

To my babushka Tatiana, who accompanied me through the first years of my life and loved me like no one else has, thank you for holding me close, and for seeing me as your light at the end of the tunnel. I hope, from wherever you watch, I am still that light and that I make you proud.

To my mother Elena, who lived this work alongside me, who absorbed my frustrations and steadied my doubts, who has always known me better than I know myself, and who always reminds me of who I am, of what I have already achieved, and that I am capable of everything. I am endlessly grateful for the food, the comfort, the knowledge, and, above all else, for your unconditional love. Thank you for everything and for making anything possible! Your pride has been one of the quiet forces that carried me forward. This work, and the person who wrote it, exist because of you.

In the interest of transparency, I acknowledge the use of AI-based tools during the preparation of this dissertation. Their use was confined to language refinement and code optimisation. The research design, the methodology, the implementation, the analysis, and the conclusions are entirely my own.

BITCOIN DIRECTION PREDICTION USING KOLMOGOROV-ARNOLD NETWORKS WITHIN THE AFML FRAMEWORK BY PETR TERLETSKIY

1 INTRODUCTION

Recent machine-learning studies report striking accuracies for Bitcoin (BTC) direction prediction, yet many rely on evaluation pipelines whose safeguards against information leakage are incomplete, leaving open how much of the reported predictability is genuine. This thesis re-examines that question under a leakage-free protocol and pairs it with a concern that the financial machine-learning literature rarely treats alongside prediction: interpretability. It applies a Kolmogorov-Arnold Network (KAN) to BTC direction prediction within the Advances in Financial Machine Learning (AFML) framework (López de Prado, 2018), benchmarks the KAN against five baselines under identical conditions, and extracts a closed-form symbolic decision function for the probability that BTC closes higher than its entry price ($P(\text{Up})$) as its novel deliverable. The guiding stance of this work is that interpretability is a contribution separable from predictive accuracy.

1.1 Research Context and Motivation

The cryptocurrency asset class emerged with Bitcoin (BTC), presented to the world on 31 October 2008 by Satoshi Nakamoto. Since its first transaction on 12 January 2009, BTC has grown into the anchor asset of a market whose total capitalisation reached approximately $4.27 trillion at its October 2025 peak and stands near $2.2 trillion in mid-2026, with BTC alone accounting for 56 to 58 percent of that total. Market maturation has accelerated since January 2024, when the U.S. Securities and Exchange Commission approved the first eleven spot Bitcoin exchange-traded funds; these products have absorbed roughly $58 billion in cumulative net inflows by mid-2026.

This thesis studies BTC in isolation for four reasons: it has the longest continuous price history in the asset class, the largest market capitalisation and highest liquidity, the deepest on-chain data coverage among cryptocurrencies, and returns to which altcoins exhibit high beta, so BTC captures the systematic cryptocurrency factor through a single instrument.

Bitcoin's daily direction is a classic multivariate signal-extraction problem, driven by feature families whose functional forms are unknown. Unlike classical econometric methods that impose functional forms a priori, machine learning (ML) models can infer structure directly from the data. However, most ML models, and neural networks in particular, operate as black boxes whose internal logic cannot be directly interpreted. Liu et al. (2024b) propose KANs as a multi-layer perceptron (MLP) alternative with univariate learnable activation functions on the edges of the network rather than fixed activations on the nodes; these edge functions can be distilled into closed-form symbolic expressions, making the trained model as auditable as a linear regression while retaining nonlinear representational power. To the author's knowledge, no prior work applies KANs to BTC price direction prediction or extracts symbolic formulas from a classification KAN under the AFML framework (López de Prado, 2018). This gap defines the four research questions of this thesis: the six-model benchmark (Q4) provides the methodological scaffolding, and the closed-form formula (Q3) is the primary novel contribution.

1.2 Problems and Research Questions

Q1. Most cryptocurrency ML studies rely on one or two feature families, typically technical indicators or price histories (Bourday et al., 2024), and there is no consensus on whether macroeconomic, crypto-macro, or on-chain features add information beyond price-derived inputs. This thesis asks which families carry signal under a leakage-free evaluation. Multi-model permutation importance across the 28 Combinatorial Purged Cross-Validation (CPCV) folds provides the answer.

Q2. Many recent BTC ML studies that apply traditional ML pipelines to financial time series report daily-direction accuracies of 80 to 90 percent (Omole and Enke, 2024), but these figures rest on pipelines with fixed-horizon labels, overlapping label spans, and train-test splits that never purge concurrent observations, so the reported accuracies may be inflated by an unknown margin. The second question revisits whether BTC price direction remains predictable once labels and features stop leaking the future, tested through three corrections on the six-model cross-section: the Deflated Sharpe Ratio (DSR), the Probability of Backtest Overfitting (PBO), and pairwise Area Under the Curve (AUC) comparisons via the DeLong test.

Q3. Existing KAN symbolic-extraction work in finance targets regression problems on relatively predictable series (Cho et al., 2025; Oad et al., 2025; Gao et al., 2025), and no published work has extracted a closed-form classification formula from a CPCV-trained KAN in a weak-signal regime. This thesis asks whether a human-readable expression for $P(\text{Up})$ can be extracted from a CPCV-trained KAN while preserving most of its predictive accuracy.

Q4. KANs have been applied to VIX forecasting (Cho et al., 2025), stock prediction (Oad et al., 2025), and cryptocurrency regression (Gao et al., 2025), but never to BTC direction classification under a uniform protocol. The fourth question places the KAN in a six-model cross-section (the KAN plus five baselines) against Autoregressive (AR) Logistic, Logistic Regression, Random Forest, Extreme Gradient Boosting (XGBoost), and Long Short-Term Memory (LSTM) under identical CPCV splits, features, sample weights, and metrics.

1.3 Contributions

C1. This thesis assembles a 73-feature universe spanning four families: 25 technical, 9 mathematical, 29 external (macroeconomic, crypto-macro, on-chain), and 10 autoregressive lag features, all competing in multi-model permutation importance selection across the 28 CPCV folds with no family privileged a priori.

C2. This thesis applies the complete AFML stack to BTC direction prediction: Cumulative Sum (CUSUM) event sampling, triple-barrier labelling (TBL), sample weighting, fixed-width window fractional differentiation (FFD), CPCV with purging and embargo, and the DSR, PBO, and DeLong AUC corrections, in a single reproducible workflow that implements every leakage and multiple-testing safeguard the framework prescribes.

C3. This thesis extracts a human-readable expression for $P(\text{Up})$ from the trained KAN through a pruning and symbolic-extraction pipeline, the first closed-form classification formula derived from a CPCV-trained KAN in a weak-signal regime and the primary novel contribution of this work.

C4. Chapter 4 benchmarks the KAN against the five baselines under identical CPCV splits, features, sample weights, and metrics, with pairwise DeLong AUC tests flagging significant cross-section differences.

2 LITERATURE REVIEW

2.1 Kolmogorov-Arnold Networks

2.1.1 Mathematical foundation

The Kolmogorov-Arnold representation theorem decomposes any continuous multivariate function into a finite sum of continuous univariate functions. Kolmogorov (1957) and Arnold (1958) established that any continuous $f : [0, 1]^n \to \mathbb{R}$ admits the representation

$$f(x_1, \ldots, x_n) = \sum_{q=1}^{2n+1} \Phi_q \left(\sum_{p=1}^{n} \varphi_{q,p}(x_p) \right), \tag{1}$$

where $\varphi_{q,p}$ and Φ_q are continuous univariate functions, so addition is the only multivariate operation.

The theorem was long considered useless for neural-network design because Kolmogorov's inner functions $\varphi_{q,p}$ can be highly non-smooth and even fractal, with no universal inner family compatible with C^1 regularity (Liu et al., 2024b). Liu et al. (2024b) resolve this on two fronts: they parameterise each univariate function as a learnable B-spline rather than insisting on the pathological inner functions of the existence proof, and they generalise the architecture beyond the two-layer, $2n + 1$-width form to arbitrary widths and depths. The justification is empirical: most functions of practical interest are smooth and have sparse compositional structure, so splines can approximate the decomposition even if they cannot realise it exactly. The authors prove a B-spline approximation rate that is independent of input dimension, avoiding the curse of dimensionality.

2.1.2 How KANs work

Where an MLP places fixed activations on nodes and learns scalar weights on edges, a KAN places learnable activations on edges and lets the nodes do nothing but sum (Liu et al., 2024b). Each edge parameterises its activation as a B-spline

$$g(x) = \sum_{k=0}^{K} a_k B_k(x), \tag{2}$$

of order K over G grid intervals, with coefficients a_k trained alongside the rest of the network. A KAN is specified by a width vector $[n_{\text{input}}, n_{\text{hidden}}, n_{\text{output}}]$, with every directed edge between layers carrying one trainable univariate spline rather than a scalar weight.

Training runs Adam first to escape shallow local minima, then L-BFGS to exploit second-order curvature once Adam has positioned the model. The loss carries two sparsity penalties on top of the prediction term: an L_1 term on edge activations that drives small-magnitude splines toward zero, and an entropy term that concentrates importance on a few edges. Together they let a trained KAN be pruned cleanly and make the surviving edges amenable to symbolic replacement.

The property exploited in this thesis is what happens after training. Each retained edge's spline is a numerically defined univariate function, and the symbolic-extraction pipeline provides a library of symbolic candidates ($\sin$, $\cos$, $\exp$, $\tanh$, x^2, and others) that match against the learned shape (Liu et al., 2024b). Once every retained edge is replaced by its best symbolic match, the network collapses to a closed-form expression in the input features. No other neural family decomposes cleanly into

univariate functions whose shapes can be read off, named, and substituted.

2.1.3 KAN variants

A growing ecosystem of KAN variants has emerged since the original release, surveyed by Yamak et al. (2025), who note that symbolic distillation applied to financial time series remains underexplored. Two variants matter for this thesis. KAN 2.0 (Liu et al., 2024a) introduces multiplication nodes alongside the original addition nodes, enabling discovery of multiplicative feature interactions (for example, an interaction between RSI and the Stochastic oscillator) through the same train-prune-symbolify pipeline; it is cited as future work rather than implemented. The Temporal Kolmogorov-Arnold Network (TKAN) of Genet and Inzirillo (2024) replaces an LSTM cell's dense gates with KAN-parameterised gates, gaining temporal memory at the cost of breaking the static feedforward structure that the symbolic-extraction pipeline requires.

2.1.4 KANs in finance

Cho et al. (2025) apply KANs to forecasting the CBOE Volatility Index (VIX) and introduce the four-step symbolic-extraction pipeline this thesis adapts: train a small KAN with L_1 and entropy regularisation, prune low-importance edges, symbolify each surviving spline against a candidate library, and fine-tune the affine parameters introduced during symbolification. Their extracted formulas recover well-known stylised facts of the VIX with explicit coefficients. The caveat is that the VIX is mean-reverting and structurally predictable, while BTC price direction is binary classification in a weak-signal regime, which makes the same pipeline substantially harder to satisfy.

Oad et al. (2025) sit closest to this thesis on the joint axis of KAN methodology, finance applications, and interpretability. Their KASPER framework stacks two KAN layers, the first performing soft regime detection via a Gumbel-Softmax classifier, the second performing regime-conditioned spline forecasting, reporting $R^2 = 0.89$ and Sharpe 12.02 on Yahoo Finance stock data. The methodological setup differs from this thesis on several axes: KASPER targets stock-price regression with a regime-detection layer, derives interpretability from Shapley attributions rather than closed-form expressions, and uses conventional walk-forward splits.

Gao et al. (2025) are the closest related work on the asset-class axis. DecoKAN combines a discrete wavelet transform with KAN-based temporal and feature mixers, extracting closed-form expressions for detail-branch dynamics with $R^2 > 0.99$ on Bitcoin, Ethereum, and Monero daily series. The methodological setup differs from this thesis despite the shared asset class: DecoKAN targets multivariate price-level regression at multiple horizons and uses chronological splits, with evaluation reported in R^2 terms rather than through the Deflated Sharpe Ratio, Probability of Backtest Overfitting, or multiple-testing corrections that the AFML framework introduces for classification and trading tasks.

Three intersecting gaps emerge, and this thesis closes all three. No prior work applies KANs to BTC price direction prediction (the literature targets regression on the VIX, stock prices, or crypto price levels). No prior work extracts symbolic formulas from a classification KAN: the two precedents that distil cleanly (Cho et al. (2025) on VIX, Gao et al. (2025) on crypto price levels) are both regression on more predictable targets. Finally, no prior work combines KAN symbolic extraction with the AFML framework, since every KAN-in-finance application surveyed evaluates under chronological splits without purging, embargo, multiple-testing correction, or backtest-overfitting checks.

2.2 Bitcoin Price Prediction with Machine Learning

2.2.1 Market efficiency and technical trading rules in cryptocurrency

The Efficient Market Hypothesis (Fama, 1970) posits that asset prices fully reflect available information, so past-price signals should carry no exploitable predictive content in an efficient market. Brock et al. (1992) provided the seminal empirical test of this position for technical trading rules, applying moving-average and trading-range-break strategies to 90 years of Dow Jones Industrial Average data and finding that the rules generated returns inconsistent with the random-walk null under bootstrap inference. Their methodology (a broad rule universe evaluated against a resampled null distribution) established the empirical template that cryptocurrency researchers extended to Bitcoin from 2016 onwards, once its price history had accumulated enough observations to support the same tests.

Urquhart (2016) conducted the first formal weak-form efficiency test on Bitcoin, applying autocorrelation, unit-root, non-linearity, and long-memory tests to daily returns from 2010 to 2016. The random-walk null was rejected across the full sample, though the second subperiod showed some evidence of efficiency, opening a research programme on how Bitcoin's informational efficiency evolves over time. The subsequent literature is surveyed comprehensively in Bariviera and Merediz-Solà (2021), which converges on an adaptive-efficiency framing: Bitcoin's departures from efficiency are largest in its early years and narrow as trading volume, exchange coverage, and institutional participation grow, with the specific empirical verdict conditional on the test choice and sample window.

The natural companion question, whether the residual inefficiencies translate into net-of-costs profits, was addressed comprehensively by Hudson and Urquhart (2021). Following the Brock et al. tradition, they evaluated nearly 15,000 technical trading rules from five rule families on two Bitcoin markets and three other cryptocurrencies, applying multiple-hypothesis-testing corrections to protect against data snooping. Their in-sample results showed significant predictive power across all rule classes, with estimated breakeven transaction costs well above prevailing exchange fees. In the out-of-sample analysis, however, no predictability survived on the two Bitcoin markets, though it persisted for the three other cryptocurrencies. Their finding provides an informative benchmark for this thesis: even a data-snooping-corrected exhaustive search over technical rules did not identify an out-of-sample tradable edge on Bitcoin, which sets the context in which the six ML architectures compared in Chapter 4 are evaluated. The framing that this thesis adopts is therefore not that predictability is absent, but that identifying it requires the evaluation discipline introduced in Section 2.3.

2.2.2 Technical analysis and ML for BTC

Máté et al. (2024) provide the most directly relevant precedent for the technical-indicator portion of this thesis's feature set. They train four machine-learning models (Artificial Neural Network (ANN), Convolutional Neural Network combined with LSTM (CNN-LSTM), Support Vector Machine (SVM), Random Forest) on 27 technical indicators to classify BTC daily direction, reporting accuracies of 81 to 82%. Two of their findings carry into the design choices made here: the Price Rate of Change (ROC) emerges among the top feature-importance contributors, justifying the inclusion of ROC in this thesis's 25-indicator set with the 14-day window calibrated separately, and the four-model comparison shows that multi-indicator strategies consistently outperform single-indicator rules, justifying engineering an indicator universe rather than committing to one canonical signal. The evaluation setup is the standard ML protocol at the time of publication: standard train-test splits with

fixed-horizon labels (which assign $+1$ or -1 based purely on whether the price is up or down a fixed number of days ahead), no purging or embargo, and equal sample weights. Under the AFML evaluation protocol adopted in this thesis, headline accuracies from such setups sit on a different evaluation basis and are not directly comparable to those reported in Chapter 4, a distinction returned to at the literature level in Section 2.2.5.

2.2.3 Deep learning for BTC direction

Omole and Enke (2024) target the same problem as this thesis: predicting the direction of BTC with a deep-learning classifier. They compare CNN-LSTM, Long- and Short-term Time-series Network (LSTNet), and Temporal Convolutional Network (TCN) architectures on on-chain blockchain indicators, with three feature-selection algorithms (Boruta, genetic algorithm, LightGBM) addressing the curse of dimensionality. Their best configuration (Boruta + CNN-LSTM) reaches 82.44% test accuracy on a fixed-horizon binary label, and backtested trading on these predictions reports annualised returns in the thousands of percent. The evaluation follows the standard ML protocol of the crypto-DL literature: fixed-horizon labels with no CUSUM event filtering, chronological split without purging or embargo, and equal sample weights. Under the AFML protocol adopted in this thesis, such setups produce accuracy and backtest-return figures that are not directly comparable to those reported in Chapter 4 (Section 2.2.5).

2.2.4 Crypto-DL surveys and selective abstention

Beyond the closely related work, the broader crypto deep-learning landscape contains dozens of further studies surveyed in recent reviews. Wu et al. (2024) compare seven architectures (Recurrent Neural Network (RNN), LSTM, bidirectional LSTM, Gated Recurrent Unit (GRU), 1D-CNN, TCN, Transformer) on Bitcoin, Ethereum, Litecoin, and Ripple across pre-COVID and COVID windows; Bourday et al. (2024) survey hybrid Transformer-plus-ML approaches with similar coverage. Both document the same recurring characteristics of the field: small datasets, evaluation on train-set metrics, backtest returns reported without transaction costs or slippage, limited use of risk-adjusted performance metrics, and diverse evaluation protocols across studies. Each of these characteristics is addressed by a specific AFML correction, and their prevalence in the literature explains why this thesis treats the 80 to 90 percent accuracy band cited in Section 1.2 as a literature-level rather than a model-level pattern.

A separate strand argues that, in noisy financial regimes, a model should be allowed to skip predictions on observations where it lacks confidence rather than being forced to make a prediction on every observation. Nabar and Shroff (2023) train cascaded classifiers that decline to predict on observations where the model's Gini impurity exceeds a confidence threshold, reporting that the selective predictions achieve higher accuracy and higher risk-adjusted returns at the cost of lower support. The principle directly motivates the bet-sizing threshold this thesis adopts in Section 3.10.1: predictions with calibrated probability near 0.5 produce a bet size near zero, which is structural abstention. The architectures differ (one classifier with continuous bet-sizing rather than a cascade of selective classifiers), but the insight is the same: a model that abstains when its confidence is low outperforms one that predicts on every observation.

2.2.5 Evaluation conventions in the literature

The work surveyed in Sections 2.2.2 through 2.2.4 shares a recurring set of evaluation conventions that differ from the AFML protocol adopted here in five specific ways: (i) chronological splits without purging or embargo; (ii) fixed-horizon labelling at $t + 1$ rather than triple-barrier labelling; (iii) equal sample weights across observations rather than uniqueness-weighted sample weights; (iv) accuracy reported without class-balance or economic-significance adjustment; and (v) no multiple-testing correction for selected Sharpe ratios. Under the AFML framework surveyed next, headline accuracies from these conventions sit on a different evaluation basis and are not directly comparable to results reported under AFML evaluation.

2.3 The AFML Methodology

2.3.1 Motivation

The five conventions synthesised in Section 2.2.5 are not isolated to any single paper but recur across the broader ML-in-finance tradition. López de Prado (2018) proposes an asset-agnostic framework that treats financial ML as a methodology-first discipline, motivated by structural features of financial time series (label overlap, non-stationarity, selection bias in strategy comparison) that general-purpose ML protocols do not accommodate by default. The components fall into two groups: data-side corrections that change how the training tuples (X, y, w, t_1) are constructed (Section 2.3.2), and evaluation-side corrections that change how cross-validation and reported performance are computed in the presence of multiple-testing and overfitting risks (Section 2.3.3).

2.3.2 Data-side corrections

The CUSUM event filter reduces the training set to informationally meaningful events. It tracks a cumulative deviation and triggers whenever the deviation crosses a volatility-calibrated threshold, resetting after each trigger (López de Prado, 2018, Snippet 2.4). Quiet observations are discarded; the classifier sees only bars where the series has moved meaningfully.

Triple-barrier labelling replaces fixed-horizon labels with a rule that reflects actual trading outcomes. Each event is followed by three barriers (upper take-profit, lower stop-loss, vertical time limit), and the label is whichever barrier is touched first. The resolution timestamp t_1 on each event lets downstream purging identify training observations whose labels overlap the test period.

Sample weights compensate for the redundancy that overlapping labels introduce. When two events fire close enough in time that their barrier windows share calendar days, the realised returns over those shared days are mechanically correlated, so the two events are not independent observations. AFML's weights combine three components: a uniqueness factor that discounts events whose return windows overlap concurrent labels, a return-attribution factor that emphasises events with larger absolute returns, and an optional time-decay factor for older observations.

Fractional differentiation addresses a stationarity-versus-memory dilemma: integer differencing achieves stationarity but destroys long memory, while price levels preserve memory but are non-stationary. Fixed-width Window Fractional Differentiation (FFD) finds the minimum exponent $d^* \in]0, 1[$ that achieves stationarity under an Augmented Dickey-Fuller (ADF) test while preserving as much memory as possible, computed via a fixed-width window. In this thesis d^* is estimated per fold on training data only, inside the CPCV loop.

2.3.3 *Evaluation-side corrections*

Combinatorial Purged Cross-Validation (CPCV) replaces standard k-fold cross-validation, which is invalid for financial ML because overlapping labels leak across folds. CPCV partitions the data into N contiguous groups, selects k as the test set, and yields $\binom{N}{k}$ splits and $\binom{N-1}{k-1}$ backtest paths. Purging removes training observations whose labels overlap the test period under three sufficient conditions (López de Prado, 2018, Ch. 7), and an embargo removes a buffer of bars after each test group to prevent serial-correlation leakage. The output is a matrix of out-of-sample predictions over multiple paths, enabling distribution-level analysis of the Sharpe ratio. The Sharpe ratio, defined as a strategy's mean return divided by the standard deviation of its returns (Sharpe, 1966), is the canonical risk-adjusted performance metric in finance.

The Deflated Sharpe Ratio (DSR) corrects observed Sharpe ratios for the fact that the maximum across many trials is systematically higher than a single pre-registered Sharpe. DSR deflates for selection bias across n_{trials} configurations and for non-normal returns, whose negative skewness and excess kurtosis inflate naive estimates. A DSR above 0.95 indicates a result that survives multiple-testing correction at the 5% level. The Probability of Backtest Overfitting (PBO) is computed via combinatorially symmetric cross-validation. The procedure repeatedly splits the backtest paths into in-sample and out-of-sample halves and checks whether the in-sample best underperforms the out-of-sample median. A PBO near zero means in-sample selection is reliable; a PBO above 0.5 means selection is adversarial, with the in-sample best systematically losing out-of-sample.

2.3.4 *AFML in practice and the fitting-scheme finding*

Ślepaczuk and Bieganowski (2024) apply FFD and triple-barrier labelling to supervised autoencoders on BTC, ETH, and LTC with walk-forward d^* estimation, reporting that the data-side AFML corrections transfer to crypto and improve risk-adjusted returns. The evaluation setup differs from this thesis on the axis identified in Section 2.2.3: walk-forward splits rather than CPCV, and neither DSR nor PBO reported. Their contribution validates the data-side corrections for crypto; the evaluation-side corrections remain to be tested.

Chassot and Audrino (2026) show that a properly fitted linear Heterogeneous Autoregressive (HAR) model (Corsi, 2009) outperforms Random Forest, lasso, gradient-boosted trees, and a feed-forward neural network across 1,445 US equities for realised volatility forecasting, tracing prior reports of ML superiority to suboptimal fitting schemes that handicapped the baseline. The finding directly anticipates this thesis's Chapter 4: under proper AFML evaluation all six models sit well below the DSR significance threshold, the outcome Chassot and Audrino predict for a weak-signal regime where the fitting scheme has been done correctly.

No surveyed study combines AFML and KANs, and none reports DSR and PBO for KAN-based predictions. This thesis is the first to apply the complete pipeline to BTC direction prediction with a KAN classifier and five baselines.

3 METHODOLOGY

The methodology unfolds in three sequential phases. Phase I (pre-CPCV) transforms the raw 4,208 daily bars into the aligned tuple (X, y, w, t_1) through data acquisition (Section 3.1), event filtering and triple-barrier labelling (Section 3.2), AFML sample weighting (Section 3.3), and feature engineering (Section 3.4).

Phase II runs the CPCV pipeline. It assembles per-split training and test folds (Section 3.5), applies per-fold preprocessing (Section 3.6), trains six models with nested hyperparameter tuning (Sections 3.7 and 3.8), and calibrates predicted probabilities (Section 3.9) to produce a test-fold prediction stream across all 28 splits.

Phase III is the post-CPCV evaluation and interpretability layer. It bet-sizes predictions, stitches paths, and computes the DSR, the PBO, and pairwise DeLong AUC tests across the seven backtest paths (Section 3.10), and extracts the closed-form symbolic decision function from the trained KAN (Section 3.11). The full pipeline is shown in Figure A.1.

3.1 Data

The base instrument is Bitcoin priced in US dollars (BTC-USD). Daily Open-High-Low-Close-Volume (OHLCV) bars are retrieved from Yahoo Finance for 2014-11-01 to 2026-05-09 (4,208 daily bars), with the closing-price series plotted in Figure 3.1. The start date is set to satisfy the longest-warmup feature in the engineered set, the 50/200 exponential moving average ratio at 200 days, so every shorter-warmup feature inherits a clean starting point.

All rolling windows operate on the BTC 24/7 trading calendar: 7-day weeks, 30-day months, 90-day quarters, 180-day semesters, and 365-day years. The 365-day convention flows downstream into the Sharpe annualisation of Section 3.10.2 and the DSR reference distribution.

Three external sources extend the feature set beyond price and volume, detailed in Section 3.4. Macroeconomic series (yield curves, the CBOE Volatility Index, commodity prices, the dollar index) come from Yahoo Finance and the Federal Reserve Economic Data (FRED) database. The ETH/BTC ratio comes from the CoinMetrics Community API with Yahoo Finance as fallback. Eight on-chain features (active addresses, transaction count, hash rate, MVRV, net exchange flow, fee per transaction, exchange supply fraction, native-unit issuance) come from CoinMetrics. External series align onto BTC's calendar through a backward-looking join, using only the most-recent observation timestamped at or before each BTC trading day. CoinMetrics observations shift forward by one trading day to respect the API's end-of-day reporting convention, so a metric timestamped at t enters the feature matrix at $t + 1$.

3.1.1 Sample-period price dynamics

The labelling window (2015-08-08 to 2026-05-02, established in Section 3.2.2) covers four complete Bitcoin cycles, plotted in Figure 3.1. The 2017 retail bull cycle peaked at approximately \$19,000 in December 2017 before the year-long 2018 collapse to \$3,000. Recovery began in 2019 and was interrupted by the March 2020 COVID-19 crash to \$4,000, followed by a rapid rebound as Federal Reserve liquidity expansion supported risk assets across the board. The 2021 institutional bull cycle, marked by treasury allocations from Tesla and MicroStrategy and by early speculation on a US spot-ETF approval, peaked at approximately \$69,000 in November 2021. The 2022 crypto winter

followed, driven by the Terra/Luna collapse in May, the Celsius Network bankruptcy in July, and the FTX collapse in November, with prices bottoming near $16,000. Recovery through 2023 and 2024 was punctuated by two structural events: the SEC's January 2024 approval of eleven spot Bitcoin ETFs, and the fourth halving in April 2024 (a 50% reduction in the block reward, further tightening the supply schedule). The 2025 rally, supported by the maturation of the ETF ecosystem and expanding institutional participation, peaked at approximately $120,000 in October 2025 before a correction to $79,000 by the labelling window's end in May 2026.

The price trajectory spans roughly four orders of magnitude in absolute terms across the sample, but the modelling task is direction prediction on triple-barrier-labelled returns (Section 3.2.3), which are scale-invariant by construction. The AFML sample-weighting scheme of Section 3.3, which discounts events sharing calendar days with concurrent labels and applies linear time decay, further attenuates the influence of any single cycle on the trained models. The 2018 and 2022 bear cycles are represented in the aligned event set at their natural frequency, so the classifier trains against genuine downside regimes rather than a bull-market-only sample.

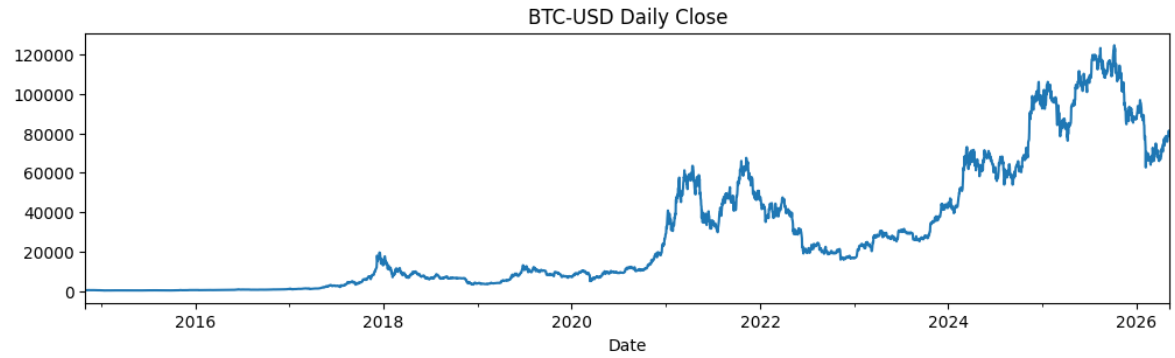


FIGURE 3.1: BTC-USD daily closing price from 2014-11-01 to 2026-05-09. The labelling window (2015-08-08 to 2026-05-02) covers four complete Bitcoin cycles: the 2017 retail bull, the 2020 COVID crash and recovery, the 2021 institutional bull, and the 2022 crypto winter, followed by the 2023 to 2025 ETF-driven rally and the early-2026 correction. ↩

3.2 Labelling

3.2.1 Daily volatility

The daily volatility series feeds two downstream steps. It is estimated as the Exponentially Weighted Moving Average (EWMA) standard deviation of log returns with span 50 (López de Prado, 2018, Snippet 3.1):

$$\sigma_t^2 = (1-\alpha)\,\sigma_{t-1}^2 + \alpha\, r_t^2, \qquad \alpha = \frac{2}{\text{span}+1}, \tag{3}$$

with $\sigma_t = \sqrt{\sigma_t^2}$. The span is shortened from the AFML default of 100 days for equities. The shorter span tracks BTC's faster regime transitions, driven by the four-year halving cycle and by regulatory shocks that compress volatility responses into days rather than months. The half-life of approximately 17 days gives the estimator a memory of roughly two trading weeks (Figure A.2). The CUSUM threshold in Section 3.2.2 takes a multiple of the series' mean, so the event filter fires at a roughly constant signal-to-noise ratio across the sample. The triple-barrier widths in Section 3.2.3 scale by the daily volatility at each event, so barriers expand and contract with the prevailing regime rather than

sitting at a fixed return percentage.

3.2.2 CUSUM event filter

The CUSUM filter selects the timestamps at which the cumulative drift in log returns crosses a volatility-scaled threshold and retains the bars at those timestamps for triple-barrier labelling, discarding everything in between. Two accumulators are maintained in parallel: an upward accumulator S_t^+ that adds positive returns and floors at zero, and a downward accumulator S_t^- that adds negative returns and ceils at zero:

$$S_t^+ = \max\bigl(0,\, S_{t-1}^+ + r_t\bigr)\,, \qquad S_t^- = \min\bigl(0,\, S_{t-1}^- + r_t\bigr)\,, \tag{4}$$

with $S_0^+ = S_0^- = 0$. An event fires at the first t for which $S_t^+ \geq h$ or $S_t^- \leq -h$, at which point the offending accumulator resets to zero. The implementation follows AFML Snippet 2.4 (López de Prado, 2018).

The threshold is set to $h = 1.0 \times \overline{\sigma}$, where $\overline{\sigma}$ is the mean of the daily volatility series from Section 3.2.1 over the labelling window. Lower multipliers fire too many events with limited information content per event, while higher multipliers leave long stretches of the sample without any event and undersample the active periods; $h = 1.0 \times \overline{\sigma}$ balances the event count against the resulting class composition after triple-barrier labelling. The mechanism is illustrated over a three-month window in early 2026 (Figure A.3).

The accumulators run on the full raw return series from 2014-11-01 onwards, but the event-firing window is truncated to start on 2015-08-08. Two constraints set this date: the 50/200 EMA ratio needs 200 calendar days to populate (Section 3.1), and the ETH/USD price series, which feeds the ETH/BTC ratio feature, begins on 2015-08-08 (shortly after the Ethereum Frontier launch on 30 July 2015). Events that survive truncation therefore reflect the genuine state of the cumulative drift at firing, not a reset on 2015-08-08. The filter produces 1,270 events that enter triple-barrier labelling in Section 3.2.3.

3.2.3 Triple barrier labelling

The triple-barrier method assigns a label to each CUSUM event by simulating a path-dependent trading position opened at the event timestamp and closed at the first of three barriers it touches (López de Prado, 2018, Snippet 3.2). The upper horizontal barrier (profit-take) and lower horizontal barrier (stop-loss) are placed at $\pm 1.5 \times \sigma_t$ around the entry price, where σ_t is the daily volatility from Section 3.2.1 at the event timestamp. The vertical barrier (time limit) is set 10 calendar days after the event, approximately one and a half weeks on the BTC continuous calendar. The label is $+1$ on an upper-barrier hit and -1 on a lower-barrier hit. When the vertical barrier is reached first (neither horizontal barrier was touched within the 10 days), the label is the sign of the realised return at expiry if the absolute return exceeds the 2% minimum-return threshold, and is set to zero otherwise; the zero-class events are dropped in the removal step below so the downstream task is strictly binary. The symmetric profit-take and stop-loss multipliers reflect the absence of a directional bias in the labelling step. The prediction problem remains directional and is delegated to the models in Section 3.7.

The output is a table indexed by event timestamp with three columns: the realised return ret (entry-to-exit price change), the label $bin \in \{-1, 0, +1\}$ (which barrier was touched first, modulated

by the minimum-return threshold for vertical-barrier hits), and the barrier-touch timestamp t_1 (the calendar day the position closed, used for the purging and embargo steps of Section 3.5.3). Three representative events, one for each barrier type, are shown in Figure A.4 of Appendix A.2.

The empirical distribution validates the parameter choices. The mean holding period is 5.0 days against a maximum of 10, with a median of 4 days, and only 19.7% of events expire at the vertical barrier, so the horizontal barriers close approximately 80% of positions before the time-out. The 5-day mean also keeps the average barrier-touch timestamp clear of the next CUSUM event, which limits the magnitude of the AFML uniqueness correction in Section 3.3. The 102 vertical-barrier ties that received the zero label are then dropped in a zero-class removal step, since the downstream task is binary direction prediction and a third class with no consistent return interpretation would only add noise to training. The pipeline therefore delivers a strict binary classification problem $bin \in \{-1, +1\}$ with no neutral-return events to manage downstream.

Numerically, the 1,270 events that survive CUSUM truncation enter triple-barrier labelling. The minimum-return threshold collapses 102 into class zero; the zero class is then dropped, leaving 1,168 final binary-labelled observations. The pipeline therefore compresses 4,208 daily bars to 1,168 observations, a ratio of roughly 28%. The class balance settles at 56.85% Up against 43.15% Down (Figure A.5), close enough to balanced that no resampling is required beyond the class-balanced weighting argument applied in Section 3.7.

3.3 Sample Weights

The triple-barrier labelling produces overlapping observations. A position opened at event i with a 10-day vertical barrier remains open through days that may host subsequent CUSUM events, so consecutive events sharing part of their barrier windows are partially redundant. As a concrete example, suppose event A fires on day 10 and event B on day 12: if neither hits a horizontal barrier early, both positions remain open between day 12 and day 20, so the realised returns of A and B over those eight shared days are mechanically correlated and the two events are not independent observations. Training without correction overweights densely-clustered events and biases the gradient towards the most active periods of the market. AFML proposes a three-component weighting scheme that addresses this redundancy and the asymmetric information content across events (López de Prado, 2018, Chapter 4), with a fourth step (outlier capping) added here to control the tail. The first component measures redundancy. For each bar t, the concurrent label count c_t records how many event positions remain open on day t. The average uniqueness $\bar{u}_i$ of event i is then the mean of $1/c_t$ over the days inside its barrier window, so an event sharing its span with no others has $\bar{u}_i = 1$ and an event sharing fully with two others has $\bar{u}_i \approx 1/3$.

The second component scales uniqueness by absolute return, giving $w_i = \bar{u}_i \times |ret_i|$, which rewards events with large realised moves since these carry more directional information than near-zero returns. The weights are normalised so that their sum equals the number of events. The third component applies a linear time decay, downweighting old events by a factor that decreases from one at the most recent event to 0.4 at the oldest, with the slope set so that the average decay is unity. The 0.4 floor reflects BTC's regime non-stationarity: the asset has matured substantially over the labelling window, evolving from a primarily retail-driven market in the mid-2010s into one with structural institutional participation by the early 2020s, so recent observations carry more information about

current dynamics than equally-aged observations from earlier periods. The two multiplicative scalings combine into a per-event weight $w_i = \bar{u}_i \times |ret_i| \times d_i$, where d_i is the decay factor at event i.

The fourth step is an outlier cap added here, limiting extreme weights to the 99th percentile of the post-decay distribution. Events combining a large absolute return, high uniqueness, and a recent timestamp accumulate weights several times the mean, and a handful of these can dominate the empirical risk minimisation of an entire fold. The 99th percentile sits at approximately 2.862, and 12 events hit the cap (notably the COVID-19 crash, the FTX collapse, and the early-2019 rally). The weight distribution is shown in Figure A.6 and the full capped-event list is in Table A.1.

The final per-event weight vector enters each model's training procedure through its standard weight-aware fitting interface. The classical ML classifiers (AR Logistic, LR, RF, XGBoost) accept the vector as a per-sample weighting argument. The neural models (LSTM, KAN) apply the weights as multipliers inside the cross-entropy loss, scaling the gradient at observation i by w_i before averaging across the batch.

3.4 Feature Engineering

3.4.1 Feature families

The 25 technical indicators cover returns and volatility (including the range-based Garman-Klass and Yang-Zhang estimators (Garman and Klass, 1980; Yang and Zhang, 2000)), momentum and trend (anchored on RSI, MACD, and the 14-day rate of change (ROC); the inclusion of ROC is supported by Máté et al. (2024), with the 14-day window calibrated separately), volume, distribution shape, and trend ratios. Indicator periods are kept at canonical values rather than optimised over the labelling window, to avoid a second layer of overfitting risk.

The nine mathematical features come from AFML Part 4 (López de Prado, 2018) and capture information content (Shannon entropy, negentropy, Lempel-Ziv complexity), random-walk diagnostics (Hurst exponent, the variance-ratio test of Lo and MacKinlay (1988)), and distributional or structural-break tests (Jarque-Bera, the Supremum Augmented Dickey-Fuller statistic, Sub- and Super-Martingale Tests (SMT) with polynomial-1 and exponential trends).

The 29 external features cover the macroeconomic backdrop (twenty Yahoo Finance and FRED series spanning the dollar index, yield curves, the VIX, and commodity-and-equity returns at 30-day and 14-day windows), the cryptoasset cross-section (the ETH/BTC ratio from CoinMetrics with Yahoo Finance fallback), and on-chain activity (eight CoinMetrics features on network activity and balance-sheet dynamics). All external series enter the feature matrix through the backward-looking join of Section 3.1, with CoinMetrics observations shifted forward by one trading day.

Ten lagged log-return features encode the autoregressive structure of BTC's own returns at lags $\{1, 2, 3, 4, 5, 6, 7, 14, 21, 30\}$ days. Lag features participate in multi-model Mean Decrease Accuracy (MDA) on equal footing with the other 63 engineered features. The full 73-feature universe is catalogued in Table A.2.

3.4.2 Phase I alignment output

Three features receive a log transform before alignment. The Average True Range (ATR) is replaced by its natural logarithm to compress its heavy upper tail, and the On-Balance Volume (OBV) and Chaikin Oscillator are passed through a sign-preserving log transform to bring their unbounded positive and negative cumulative excursions onto a comparable scale.

Feature engineering produces rolling-window outputs with missing values from two sources: rolling-window warmup at the start of the sample, and external-data calendar gaps. Phase I does not impute these values; per-fold missing-value imputation is deferred to inside the CPCV loop (Section 3.6.1) so the imputation policy uses only training-fold information.

The alignment step intersects the indices of the feature matrix, the label vector, the sample weights, and the barrier-touch timestamps to produce the four-tuple (X, y, w, t_1). Hard assertions guard the output: the intersected index must be non-empty, free of duplicates, and monotonic; the four arrays must share lengths and indices; t_1 must be fully populated; weights strictly positive; labels in $\{-1, 0, +1\}$. The output is a $1{,}168 \times 73$ feature matrix paired with a binary label vector $y \in \{-1, +1\}$, a positive weight vector, and a barrier-touch timestamp vector. Phase I therefore closes with the aligned (X, y, w, t_1) dataset, ready for the cross-validation pipeline of Section 3.5.

3.5 Cross-Validation Framework

The standard k-fold cross-validation procedure is invalid for financial time series with overlapping labels: each triple-barrier label spans multi-day intervals (Section 3.2.3), so events in different folds whose barrier windows overlap on the calendar share information through the train-test boundary. Combinatorial Purged Cross-Validation (CPCV) of López de Prado (2018, Chapter 12) prevents this leakage through three mechanisms covered below: purging training events whose barrier windows reach into the test fold, embargoing the calendar days immediately after the test fold, and combining multiple test-group choices into a backtest that consumes each event exactly once.

3.5.1 Why standard CV fails

Triple-barrier labels are multi-day intervals averaging 5 days, so standard k-fold cross-validation routinely places training-fold barrier windows inside the test fold and exposes the model to test-period prices through its training labels. Reported AUC, accuracy, and Sharpe on standard k-fold splits therefore inflate the genuine out-of-sample performance of any model trained on overlapping-label data.

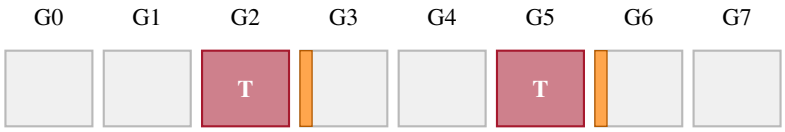
3.5.2 CPCV configuration

The CPCV pipeline partitions the 1,168 aligned observations into $N = 8$ contiguous groups of 146 observations each, with $k = 2$ groups serving as the test set in each split and an embargo of 1% applied to the calendar boundary. The number of splits is $\binom{8}{2} = 28$, and each group appears in $\binom{7}{1} = 7$ test sets, which equals the number of non-overlapping backtest paths into which the splits combine. One such split is illustrated in Figure 3.2.

The choice of $N = 8$ and $k = 2$ balances three competing demands. The 146-event group size is large enough that within-fold Area Under the Curve (AUC) and Sharpe estimates are not dominated by sampling noise. The 28 splits provide the combinatorial diversity required by the PBO test of Section 3.10.3. The 7 paths give the DSR of Section 3.10.3 enough independent draws to populate a stable test-statistic distribution.

The 7 paths are built so that each is a complete out-of-sample sequence covering all 1,168 observations exactly once (López de Prado, 2018, Section 12.4.1), with each path using 4 of the 28 splits. This converts a single backtest into a per-model distribution of Sharpe ratios, AUC, and accuracy that DSR and PBO consume in place of a single point estimate. Across all 28 splits the average training

fold contains 858 observations (73.4% of the labelling window) and the test fold contains 292. Purging drops an average of 1.9 events per split, and the embargo absorbs an average of 16.5. The eight groups span 2015-08-08 to 2026-05-02 and are shown on the BTC price chart in Figure A.7.



Split 12: test = {G2, G5}

FIGURE 3.2: Schematic of one CPCV split with $N = 8$ groups and $k = 2$ test groups (Split 12 shown). Red blocks (labelled T) are the two test groups; grey blocks are training groups; orange strips immediately after each test group mark the 1% embargo removed from training data. Across the $\binom{8}{2} = 28$ splits, every group appears in $\binom{7}{1} = 7$ test sets, which combine into 7 disjoint out-of-sample backtest paths covering the 1,168 events exactly once. ↩

3.5.3 *Purging and embargo*

Purging removes from the training fold any observation whose barrier window overlaps the test fold's calendar range (López de Prado, 2018, Snippet 7.1). Three sufficient conditions cover the overlap geometry, catching respectively the cases where a training event starts inside the test window, resolves inside the test window, or straddles it. The first fires when the entry timestamp $t_{0,i}$ of training observation i falls inside the test window $[t_{\text{start}}, t_{\text{end}}]$. The second fires when the barrier-touch timestamp $t_{1,i}$ falls inside the test window. The third fires when the training observation's barrier window fully encloses the test window, that is, $t_{0,i} \leq t_{\text{start}}$ and $t_{\text{end}} \leq t_{1,i}$. Any training observation satisfying at least one condition is dropped for that split.

The embargo removes a fixed number of training observations immediately after each test group: at 1% of 1,168 total observations, this drops 11 training observations per test-group boundary. The embargo is applied only after the test group because training labels that resolve before the test starts cannot carry information about future test prices, while the post-test boundary can still leak through serial correlation in returns and volatility (López de Prado, 2018, Section 7.4.2). An empirical leakage audit on the 28 splits returns zero training-observation barrier-touch timestamps falling inside their assigned test groups, so the pipeline produces leak-free training and test partitions throughout the labelling window.

3.6 *Per-Fold Preprocessing*

3.6.1 *Fractional differentiation*

An Augmented Dickey-Fuller (ADF) test applied to each of the 73 features inside the training fold at $\alpha = 0.05$ identifies the Average True Range (ATR) as the only feature for which the unit-root null is not rejected. The remaining 72 features are already stationary, either because they are returns and ratios with bounded ranges, or because they are normalised counts and rolling z-scores. Fixed-width window fractional differentiation (FFD) (López de Prado, 2018, Chapter 5) is therefore applied to ATR only, with the objective of removing the price-level trend that ATR inherits from the close-price magnitude while preserving as much of the column's memory as possible. The differencing order d^* is selected per fold by sweeping d over $]0, 1[$ in steps of 0.05 and taking the smallest value for which the ADF test on the training-fold ATR series rejects the unit-root null. The minimum-d rule preserves

the longest available memory among the values that achieve stationarity, with a fallback to $d^* = 1.0$ (standard first difference) if no value in the sweep achieves stationarity. The weight recursion and truncation threshold for the fractional-difference convolution are given in subsection A.7.

Once d^* is fixed on the training fold, the transformation is applied to the full ATR series. The convolution is strictly backward-looking, so applying it to the full series does not leak any test-fold information back into the training fold. This walk-forward use of FFD follows Ślepaczuk and Bieganowski (2024). The lookback creates a short block of missing values at the start of the FFD column, which are dropped from the affected fold rather than forward-filled, since forward-filling FFD output would propagate a stale lookback estimate into observations that should already have a valid one. The number of dropped rows is logged per split and typically lies between two and ten rows. The remaining 72 columns of X can also contain missing values from external-data calendar gaps and rolling-indicator warmups, handled with a forward-then-backward fill applied independently within each fold so that no fill value crosses the train-test boundary. The asymmetric treatment reflects that a rolling-window warmup can be safely propagated, whereas a fractional-differencing lookback estimate is structurally different at every position.

3.6.2 Feature scaling

All 73 features, including the fractionally-differenced ATR, are rescaled with a RobustScaler:

$$\tilde{x}_j = \frac{x_j - \text{median}(x_j)}{Q_3(x_j) - Q_1(x_j)}, \tag{5}$$

where Q_1 and Q_3 are the first and third quartiles respectively, so the denominator is the interquartile range. Both statistics are fitted on the training fold and applied to the test fold without refitting. Most BTC features show fat tails and occasional extreme values from regime breaks (the March 2020 COVID crash, the May 2021 China mining ban, the November 2022 FTX collapse) that displace the mean and inflate the standard deviation. The median and IQR are by construction insensitive to these outliers and centre the bulk of the distribution rather than its tails.

3.6.3 Multi-model MDA feature selection

The feature-selection stage scores each feature using Mean Decrease Accuracy (MDA), a permutation-based importance measure, and selects a fold-specific subset of the training-fold feature matrix (López de Prado, 2018, Section 8.4). MDA is typically computed with a single classifier, whose inductive bias influences the resulting ranking. The implementation here uses two estimators and averages their scores: a Random Forest with 500 trees that captures nonlinear interactions, and a Logistic Regression that captures linear marginal effects, both with balanced class weights. Because the two estimators produce scores on different absolute scales (Random Forest drops F1 by 0.005 to 0.05 per feature, Logistic Regression drops an order of magnitude smaller), and the multi-model MDA is reported as the unweighted mean $\bar{m}_j = (m_j^{\text{RF}} + m_j^{\text{LR}})/2$ of the two F1-drop vectors, the Random Forest signal effectively anchors the ranking, with the Logistic Regression scores acting as a soft tiebreaker that can move a feature in or out of the positive-MDA bucket near the zero boundary.

The MDA computation runs inside an inner 3-fold purged cross-validation on the training fold, using the same t_1-based overlap conditions as the outer CPCV (Section 3.5.3). For each estimator, a baseline cross-validated macro-F1 is recorded, and each feature in turn is permuted to measure the

resulting F1 drop. The mean across the three inner folds gives the feature's MDA score. A feature is retained if its averaged $\bar{m}_j$ is strictly positive, with the set capped at the top 20% of the MDA-ranked features (14 to 15 of the 73) and a minimum floor of five against pathological folds. The cap is anchored in the 80/20 split of Section 3.9: with approximately 686 model-training events and the KAN and LSTM parameter counts of Section 3.7, 14 features already approaches a one-sample-per-parameter regime, so a looser cap would deepen overparameterisation without a corresponding gain in signal.

The AR Logistic model bypasses the MDA stage entirely. As an autoregressive specification, it is restricted to the ten lag features of Section 3.4.1 regardless of the MDA ranking. The five remaining models (Logistic Regression, Random Forest, XGBoost, LSTM, and KAN) receive the post-MDA matrix with between 14 and 15 features per fold.

3.7 Models

All six models share the same training-pipeline scaffolding. The AFML sample weights of Section 3.3 are passed to each estimator as per-sample multipliers, and each model is trained with five seeds per CPCV fold with split-level metrics averaged across the five seeds for every (model, split) cell. Class-balanced weighting compensates for the 56.85% / 43.15% class skew through each estimator's class-rebalancing mechanism, with the exception of XGBoost, which is fit with AFML sample weights alone. The five engineered-feature models also enter the Optuna per-split tuning loop of Section 3.8 with 40 trials per split. Fixed architectures and per-fold search spaces for all six models are listed in Table A.3 in the appendix.

3.7.1 AR Logistic (econometric baseline)

The AR Logistic model is the econometric baseline of the comparison, testing whether pure price-momentum signals carry enough information to forecast the direction of each triple-barrier event. The specification is a logistic regression on the ten autoregressive lags of daily log returns from Section 3.4.1 (lags 1, 2, 3, 4, 5, 6, 7, 14, 21, and 30 days). Hyperparameters are fixed across folds (default L2 penalty at $C = 1.0$, balanced class weights, lbfgs solver), so the AR Logistic does not enter the per-split Optuna loop.

3.7.2 Logistic Regression (linear ML baseline)

The Logistic Regression model is the linear baseline among the engineered-feature classifiers, and the gap between its Sharpe distribution and those of the tree ensembles and neural architectures isolates the contribution of nonlinear interactions. The implementation is scikit-learn logistic regression with balanced class weights, with the regularisation strength and penalty type tuned per split via Optuna.

3.7.3 Random Forest

The Random Forest model (Breiman, 2001) is the nonlinear ensemble baseline, capturing feature interactions and threshold effects through tree splits. The architecture is a Random Forest with the number of trees tuned per fold (Section 3.8), bootstrap sampling, and random feature subsampling at each split, with class imbalance handled by drawing each tree's bootstrap from a class-balanced version of the training fold. Four hyperparameters are tuned per fold; the depth cap at 6 and the minimum-samples-per-leaf floor at 15 events (1.8% of the training fold) prevent overfitting on noisy daily BTC data.

3.7.4 XGBoost

The XGBoost model (Chen and Guestrin, 2016) adds gradient boosting to the ensemble comparison, growing trees sequentially against the residuals of the previous ensemble. The architecture is a 500-tree ensemble with binary logistic objective and early stopping after 20 rounds without improvement on the 20% holdout of Section 3.9. Eight hyperparameters are tuned per fold; the depth cap is set at 3 rather than the Random Forest's 6 because sequential boosting compounds depth nonlinearly across rounds. The shared use of this holdout for early stopping and calibrator fitting is a mild coupling that affects only the final ensemble size.

3.7.5 LSTM

The LSTM model (Hochreiter and Schmidhuber, 1997) is the temporal-dependence model in the comparison, processing the feature matrix as a sequence rather than as independent rows. The architecture is a single-layer LSTM whose final hidden state passes through layer normalisation, dropout, and a linear classifier; the depth is hardcoded at one because two- and three-layer variants overfit on 1,168 events. The input is reshaped into overlapping sequences of length 14, spanning the 10-day vertical barrier with four additional days of context, with feature values normalised by a tanh transformation $z = \tanh((x - \mu)/\sigma)$ fitted on the training fold and applied unchanged to the test fold. The training loop uses AdamW (Loshchilov and Hutter, 2019) with label smoothing 0.1, gradient clipping at norm 1.0, and cosine annealing with warm restarts; Optuna trials run at a reduced 50-epoch budget while the final per-fold refit uses 100 epochs.

3.7.6 KAN

The KAN model (Liu et al., 2024b) is implemented with a single hidden layer of width w, a grid range of $[-1, 1]$ matching the tanh input normalisation, and spline order three. The single-hidden-layer choice is driven by two considerations. First, the symbolic-extraction pipeline of Section 3.11: a two-hidden-layer KAN would compose trigonometric primitives inside trigonometric primitives, producing fourth-order expressions that lose the interpretability that motivates the architecture. Second, the labelled dataset of 1,168 events is small relative to the parameter counts of deeper KAN configurations, so additional layers would risk overparameterisation without a corresponding gain in signal. The cap on w at 6 keeps the symbolic formula humanly readable, since each surviving hidden unit becomes one additive term plus interactions in the closed-form expression. The training loop matches the LSTM's (AdamW, label smoothing 0.1, gradient clipping at norm 1.0, cosine annealing with warm restarts).

3.8 Hyperparameter Tuning

Hyperparameter selection happens entirely inside each CPCV training fold, following the nested-cross-validation discipline of López de Prado (2018, Chapter 7). Within each training fold, Optuna evaluates trial configurations against a purged 3-fold inner cross-validation, with each validation block extended by a positional embargo of 10 observations on either side to drop the overlapping training rows. The 10-observation embargo matches the 10-day vertical barrier of Section 3.2.3, so it covers the same horizon as the outer CPCV's t_1-based purging without the per-event timestamp lookup. The search is driven by the Tree-Structured Parzen Estimator (TPE) sampler (Akiba et al., 2019) with a fixed seed for reproducibility, and a MedianPruner stops trials whose intermediate inner-CV

log-loss falls below the running median of completed trials, freeing compute for more promising configurations. The pruner waits for 5 completed trials on the classical models and 3 on the neural models before it begins to prune, with the lower neural startup threshold reflecting the larger per-trial training cost on LSTM and KAN.

The trial budget is 40 per tuned model per CPCV split, applied uniformly across Logistic Regression, Random Forest, XGBoost, LSTM, and KAN. The uniform budget is a deliberate design choice: giving every tuned model the same number of explorations prevents the between-model comparison from being confounded by asymmetric search effort. The AR Logistic baseline of Section 3.7.1 is not part of the tuning loop, with its hyperparameters fixed by the econometric specification. At the end of each per-fold Optuna study, the best-trial hyperparameters are written into the model's module-level constants before that split's refit, and they are overwritten by the next split's best-trial values before the next refit, so a split's training never sees another split's tuned values.

The Optuna trials produce inner-CV log-loss scores, not test-fold Sharpe estimates, so they do not enter the strategy population over which the Deflated Sharpe Ratio of Section 3.10.3 corrects for multiple testing. The DSR multiple-testing count is the six models in the comparison, not the 5,600 Optuna trials that the tuning loop runs in total.

3.9 Calibration

Bet sizing depends on calibrated probabilities: a model that assigns $P(\text{Up}) \approx 0.7$ to events whose actual Up rate is 55% is overconfident on the long side, and its bets are systematically oversized. Each model's raw probabilities are therefore passed through a per-fold calibrator before they are consumed by the metric computations of Section 3.11. Within each outer training fold, the data are partitioned chronologically into an 80% model-training partition and a 20% holdout. The holdout serves a dual role: it provides the early-stopping signal for XGBoost, the LSTM, and the KAN during model fitting, and it is the partition on which the calibrator itself is then fitted. At $N = 8$, the model-training partition contains approximately 686 events and the holdout contains approximately 172 events, with neither overlapping the outer test fold. The shared use of the holdout for both early stopping and calibration introduces a mild coupling, since the calibrator sees logits from a model that has already been selected on the same partition; the alternative of a third dedicated calibration slice was rejected because it would have shrunk the model-training partition to roughly 600 events and risked underfitting the KAN and LSTM.

The calibration method is selected automatically by model type. The four classical models (AR Logistic, Logistic Regression, Random Forest, XGBoost) are calibrated with Platt scaling (Platt, 1999), which maps the classifier's log-odds output to a calibrated probability through a logistic regression with no regularisation fitted on the holdout. The two neural models (LSTM and KAN) are calibrated with vector scaling, a generalisation of temperature scaling proposed by Guo et al. (2017, Section 4.2): the calibrator fits a temperature T and a per-class bias vector $b \in \mathbb{R}^2$ that minimise the negative log-likelihood of $\text{softmax}((\text{logits} + b)/T)$ on the holdout, with T constrained to $[0.05, 20]$ and the bias entries to $[-5, 5]$, optimised by L-BFGS-B starting from $T = 1$ and $b = 0$. For the LSTM, raw logits are sliced to the valid-index set before calibration so that incomplete-lookback windows do not distort the fitted temperature.

3.10 Evaluation Framework

3.10.1 Trading simulation

The bet-sizing rule converts each model's calibrated probability into a discretised directional position in $\{-0.75, -0.5, -0.25, 0, 0.25, 0.5, 0.75\}$ (López de Prado, 2018, Chapter 10). The direction is set to $+1$ when $P(\text{Up}) > P(\text{Down})$ and to -1 otherwise. The confidence $p = \max(P(0), P(1)) \in [0.5, 1.0]$ enters a z-score $z = (p - 0.5)/\sqrt{p(1-p) + 10^{-10}}$, which is mapped to a raw bet $2\Phi(z) - 1$ on $[-1, 1]$, clipped at ± 0.75, thresholded at 0.05 in absolute value (sub-threshold bets become zero), discretised to the nearest of $\{0, 0.25, 0.5, 0.75\}$, and finally signed. Predictions with p near 0.5 therefore produce bets near zero, giving the rule an implicit abstention mechanism consistent with the Conservative Predictions framework of Section 2.2.3.

Per-event strategy returns are the product of the signed bet and the triple-barrier label return, gross $=$ bet $\cdot\, r_{\text{label}}$. Turnover is $|\Delta\text{bet}|$ between consecutive events, and transaction costs are charged at 0.001 per unit of turnover, giving net $=$ gross $- \, 0.001 \cdot$ turnover. The cost level is conservative relative to the maker and taker fee schedules of the major BTC exchanges (Nabar and Shroff, 2023), with slippage added on top. Returns are annualised at a factor of 365 because BTC trades continuously, and the risk-free rate, the baseline return that a strategy must exceed to compensate for the risk it takes on, is set to zero on the basis that no universally accepted crypto risk-free rate exists.

The path-stitching procedure assembles the 28 per-split prediction streams into the 7 backtest paths of Section 3.5.2. For each path, predictions from each (group, split) pair are filtered to the events within the assigned group's boundaries, sorted by timestamp, and asserted free of duplicates. The five per-seed prediction vectors for each (model, split) cell are averaged at the calibrated-probability stage before any bet sizing, reducing seed-driven variance by approximately $1/\sqrt{5}$.

3.10.2 Path performance metrics

Each of the 7 paths produces a self-contained backtest with its own performance summary. The annualised Sharpe is computed as the per-event mean divided by the per-event standard deviation, scaled by $\sqrt{365}$. Cumulative return is $\prod_t (1 + r_t) - 1$, and the maximum drawdown is the minimum value of $(E_t - M_t)/M_t$ where E_t is the equity curve and M_t its running maximum. The remaining metrics report the trade frequency (traded-event count) and distributional shape (win rate and profit factor).

The $\sqrt{365}$ annualisation factor deserves an explicit note. Strategy returns are sampled at CUSUM events rather than at daily bars, and the labelling window averages approximately 109 events per year, so annualising at the bet-frequency rate $\sqrt{109} \approx 10.4$ would deflate every reported Sharpe by roughly half. The $\sqrt{365}$ convention is chosen to keep absolute Sharpe numbers on the daily-equivalent scale that the finance literature typically uses. The DSR computation of Section 3.10.3 de-annualises by the same $\sqrt{365}$ before applying the multiple-testing correction, so DSR verdicts and model rankings are invariant under the annualisation convention.

3.10.3 Statistical correction layer: DSR, PBO, and DeLong

Three statistical corrections are applied to the path-level performance distribution. The Deflated Sharpe Ratio addresses multiple-testing across the six compared models. The Probability of Backtest Overfitting tests whether the in-sample model-selection generalises out-of-sample. The DeLong test

gives pairwise p-values for the 15 model-pair AUC comparisons.

The Deflated Sharpe Ratio (López de Prado, 2018, Chapter 11) corrects the observed Sharpe against the maximum Sharpe expected under a population of $n = 6$ candidate strategies (the six compared models). The full formula and the expected-maximum and standard-error expressions sit in subsection A.7. Values above 0.95 are taken as significant after the multiple-testing correction.

The Probability of Backtest Overfitting (López de Prado, 2018, Chapter 14) measures the probability that the in-sample best model underperforms the out-of-sample median on the complementary path subset. Enumerating $\binom{7}{3} = 35$ partitions of the 7 paths into a 3-path IS and 4-path OOS, PBO is the fraction of partitions on which the IS-best model's OOS median Sharpe falls below the cross-model median. Values below 0.3 indicate the selection procedure generalises, values above 0.5 indicate the IS-best is anti-predictive OOS.

The DeLong pairwise AUC test (DeLong et al., 1988) compares each of the $\binom{6}{2} = 15$ model pairs. For each (model, split) cell predicted probabilities are seed-averaged across the five seeds; the seed-averaged predictions are pooled across the 28 splits, and the z-statistic (subsection A.7) is converted to a two-sided p-value under the standard normal. The pairwise p-values are reported as a descriptive diagnostic rather than as a selection mechanism.

3.10.4 Stability diagnostics, model comparison, and ranking

Two stability diagnostics complement the main evaluation. The first counts, for each feature, the share of CPCV folds in which it survives the multi-model MDA of Section 3.6.3, flagging features above 80% as stable; a flat selection profile across many features signals a diffuse rather than a concentrated signal (a finding about the data rather than a defect in the selection rule). The second tracks the per-fold d^* used to fractionally-difference ATR (Section 3.6.1), reporting its mean and standard deviation: a standard deviation below 0.1 indicates a consistent stationarity structure across folds, while higher values point to time-varying persistence and motivate a regime-aware refit.

Models are ranked by median path Sharpe across the 7 paths in descending order, with the standard deviation of the path Sharpe distribution as a tiebreaker in ascending order. The standard-deviation tiebreaker prefers a stable model over a spikier one at the same median Sharpe, on the principle that a high-variance backtest distribution is harder to deploy in practice even at the same expected return. The full model-comparison table reported in Chapter 4 lists the median Sharpe and its tiebreaker, the DSR, the mean F1, accuracy, and AUC across paths, and the median values of maximum drawdown, cumulative return, win rate, and profit factor.

3.11 Symbolic Extraction

The symbolic-extraction stage produces a closed-form approximation of the trained KAN model (Liu et al., 2024b) on a single CPCV fold. The output is fold-specific by construction, which Section 5.2 discusses as a limitation to manage rather than a flaw of the method.

The stage operates after all 28 CPCV splits have been evaluated. It selects the last split (split 27 in the CPCV ordering, whose test set covers the most recent observations and is therefore closest to a deployment scenario), reconstructs the fold's preprocessing state from the stored d^*, RobustScaler, and selected feature list, reduces the input dimensionality to the top three features by 28-fold MDA selection frequency, and re-trains a fresh KAN on the resulting preprocessed inputs. The architecture is determined by a data-aware safety envelope: a samples-per-parameter floor that ensures the model has

enough events to support its parameters, clamped against the same hidden-layer width ceiling used in the CPCV loop. When per-fold tuned width and grid values are available, the envelope honours them within these bounds; otherwise the safety-envelope choice becomes the realised architecture.

The trained model is then processed through the four-step symbolic-extraction algorithm of Cho et al. (2025) (see also Noorizadegan et al. (2026)). Step 1 trains the model under a three-phase schedule (Adam, LBFGS warmup, LBFGS with L1 plus entropy sparsity regularisation), in which only the third phase applies sparsity pressure. Step 2 prunes edges below an importance threshold; the number of edges removed depends on how concentrated the trained spline weights were during Step 1. On folds where the sparsity regularisation in the final LBFGS phase concentrated weights into a subset of edges, the architecture compresses; on folds where all edges retain comparable importance, no compression occurs. Step 3 replaces each surviving spline edge with the best-fitting symbolic function from a 14-element library covering polynomials, transcendentals, non-smooth primitives, and the zero function, retaining the spline form where no candidate clears the R^2 threshold of 0.30. Step 4 fine-tunes the affine parameters a, b, c, d in $a \cdot f(bx+c)+d$ for each symbolic edge by a short LBFGS pass. Algebraic simplification then converts the decision function $\text{logit}_{\text{bullish}} - \text{logit}_{\text{bearish}}$ into a closed-form expression in the surviving features, with implied probability $P(\text{Up}) = \sigma(\text{decision})$. The per-step parameters, the symbolic candidate library, the pruning thresholds, the known fragilities, and the workarounds are given in subsection A.8. The extracted formula, the per-edge R^2 values, and the partial-derivative sensitivity analysis appear in Chapter 4.

Three limitations of the output deserve acknowledgement and are discussed further in Section 5.2. The formula is fold-specific by construction, and different folds may produce different formulas with different surviving features. The training sample of 540 events is small, which limits the complexity of the representations that the data can identify and motivates the conservative R^2 threshold. The post-symbolic accuracy can fall below the pre-symbolic accuracy when the symbolic library fits the trained spline less faithfully than the spline fitted the data, in which case the formula approximates the model imperfectly even when the symbolification rate is high.

4 RESULTS

Each model is trained with five seeds per CPCV split, producing 840 prediction entries (28 splits $\times$ 6 models $\times$ 5 seeds), with the path-level Sharpe matrix averaged over the five seeds per (model, split) cell before entering the DSR and PBO computations of Section 3.10.3. Interpretation against the research questions is held to Chapter 5. Three supporting figures (the path-Sharpe boxplot, the confusion matrices, and the bet-size histograms) are provided in subsection A.9.

4.1 Model Comparison

Table 4.1 reports the six trained models plus buy-and-hold as a naive baseline, ranked by median path Sharpe over the seven CPCV backtest paths, with the standard deviation of the path Sharpe distribution as a tiebreaker in ascending order.

Model	Med Sharpe	Sharpe CI	Std SR	DSR	Med Sortino	Mean F1	Mean AUC	Med Max DD	Med Cum Ret
Buy-and-Hold	2.2722	$]1.8903, 2.4685[$	0.5768	n/a	2.3390	n/a	n/a	-0.9998	45,896.39
KAN	0.5588	$]-0.4677, 1.1607[$	0.7539	0.2470	0.3211	0.4070	0.5103	-0.2466	0.0841
AR Logistic	0.3806	$]-0.0544, 1.2101[$	0.7355	0.1896	0.3894	0.4666	0.5204	-0.2739	0.0852
LSTM	0.1541	$]-0.3902, 0.7032[$	0.6388	0.1291	0.1189	0.4238	0.5105	-0.2986	0.0102
XGBoost	0.0493	$]-0.0638, 0.2514[$	0.5830	0.1065	0.0259	0.4271	0.5008	-0.2696	-0.0027
Random Forest	0.0287	$]-0.6165, 0.8834[$	0.7533	0.1023	0.0257	0.4390	0.5166	-0.3775	-0.0579
Logistic Regression	-0.2173	$]-0.7321, 2.0594[$	1.5969	0.0618	-0.1872	0.4355	0.5156	-0.3874	-0.0797

TABLE 4.1: Model comparison ranked by median path Sharpe. DSR is computed against $n_{\text{trials}} = 6$ and is undefined for buy-and-hold.

Three patterns stand out. The top three DSRs (KAN 0.2470, AR Logistic 0.1896, LSTM 0.1291) span 0.12, with smaller gaps to XGBoost (0.1065), Random Forest (0.1023), and Logistic Regression (0.0618).

All mean AUCs sit in the 0.5008 to 0.5204 band. Models are nearly indistinguishable at the classification level, and bet sizing and the dual-weighted loss translate the small classification edges into the wider Sharpe spread.

Six of the seven bootstrap 95% CIs cross zero, and only buy-and-hold stays strictly positive (Table 4.1). The full path-level Sharpe distributions are visualised in Figure A.9.

Buy-and-hold posts the highest median Sharpe (2.2722) and the highest cumulative return ($45{,}896\times$ from compounding the +1 bet across all 1,168 TBL events; the simple HODL return over the labelling window is roughly $306\times$, with BTC going from approximately \$258 on 2015-08-08, peaking near \$120,000 in October 2025, and finishing at roughly \$79,000 on 2026-05-02). The compounded-TBL figure exceeds the simple-HODL figure because TBL event windows overlap, so the same calendar return is counted across multiple events. Trained models trade return for downside protection: KAN's cumulative return of 0.0841 is roughly six orders of magnitude smaller than buy-and-hold's, but its maximum drawdown of -0.2466 is four times shallower.

4.2 Classification Metrics

Per-split F1 ranges narrowly from 0.4070 (KAN) to 0.4666 (AR Logistic). Pooled AUC across all 28 splits sits in the 0.4942 to 0.5149 band. This pooled-across-splits AUC differs slightly from the per-path Mean AUC reported in Table 4.1 because the two metrics aggregate predictions differently.

XGBoost's pooled AUC (0.4942) sits below 0.5. The confusion matrices pooled across all splits and seed-averaged at the predicted-probability stage are provided in Figure A.10.

Two pairs clear $\alpha = 0.05$. Random Forest versus XGBoost at $z = 2.1466$ ($p = 0.0318$) and AR Logistic versus XGBoost at $z = 2.0263$ ($p = 0.0427$). Both are significant in the direction of XGBoost being the lower-AUC half. The full 15-pair table sits in Appendix A.9, Table A.5.

The thirteen non-significant pairs include all KAN comparisons, with the closest non-significant pair on the KAN side being KAN versus XGBoost at $z = -1.6872$ ($p = 0.0916$). The entire AUC range across all six models is 0.021, comparable to the seed-to-seed variation, and both significant pairs lean on the XGBoost lower tail. The practical reading is that XGBoost's AUC is significantly lower than the top two trained models, not that arbitrary pairs of trained models differ at the classification level.

4.3 Financial Performance

The equity curves of the six trained models, with each model's seven paths shown in light colours and the median path highlighted in black, are presented in Figure A.8 of Appendix A.9. The vertical axis is logarithmic so that the relative behaviour of paths in the $0.6\times$ to $3\times$ band of cumulative return is readable.

The path-level Sharpe distributions underlying Table 4.1 produce the bootstrap CIs reported there. Sortino reorders the trained models slightly: AR Logistic (0.3894) leads, followed by KAN (0.3211), LSTM (0.1189), XGBoost (0.0259), Random Forest (0.0257), and Logistic Regression (-0.1872). AR Logistic's Sortino edge over KAN despite a lower Sharpe suggests its path-Sharpe distribution loads more variance on the upside than on the downside. Calmar (Sharpe over maximum drawdown) compresses the trained models into a narrow -0.04 to $+0.08$ band, with KAN at 0.0820 leading among trained models.

A calibration audit on the test-fold predictions checks each model's mean $P(\text{Up})$ against the empirical base rate of 0.5685. Five of six trained models sit within the ± 0.03 tolerance set on the holdout; XGBoost narrowly misses ($\Delta = -0.0371$). KAN's mean $P(\text{Up})$ is 0.5421 ($\Delta = -0.0263$), well within tolerance. The full per-model audit is in Table A.6 of Appendix A.9. The per-fold calibrator of Section 3.9 therefore delivers well-calibrated probabilities, on which the bet-sizing mechanism below operates.

The bet-sizing mechanism of Section 3.10.1 turns calibrated probabilities into discretised bets on $\{-0.75, -0.5, -0.25, 0, 0.25, 0.5, 0.75\}$, with probabilities near 0.5 producing zero bets (structural abstention). KAN and XGBoost abstain on roughly 70% of events (70.1% and 73.1% respectively), against the 50% to 54% band for the other four trained models. KAN also carries the most pronounced long bias on non-abstained bets at 89.6%, against Logistic Regression's milder 73.3% long, 26.7% short, the least directionally biased among trained models. The combination of high abstention and strong long bias for KAN is consistent with the bet-sizing-on-calibrated-probabilities reading: KAN takes few bets but takes them with directional confidence, and this discipline aligns with KAN's small median maximum drawdown (-0.2466, the shallowest among trained models, with XGBoost's -0.2696 the second-shallowest, Table 4.1). The full per-model bet-size distribution is in Table A.7 of Appendix A.9, with the per-model histograms in Figure A.11.

The Probability of Backtest Overfitting evaluates to PBO $= 0.657$ ($23/35$ IS/OOS partitions),

placing the cross-section in the adversarial regime: in roughly two of three partitions, the model that wins on three IS paths underperforms the OOS median on the complementary four paths. The IS ranking is anti-informative.

Re-running PBO on the 5×7 Sharpe matrix with one model dropped at a time identifies Random Forest as the anchor (its exclusion leaves PBO unchanged at 0.657) and Logistic Regression as the largest destabiliser (its exclusion drops PBO to 0.371, $\Delta = -0.286$). The IS/OOS rotation is therefore concentrated in the top five trained models. The full leave-one-out PBO table is in Table A.8 of Appendix A.9.

The joint reading is consistent. No trained model achieves predictive ability that survives multiple-testing correction (top DSR 0.2470 well below 0.95), the in-sample ranking of the six-model cross-section is anti-informative (PBO $= 0.657 > 0.5$), six of seven bootstrap CIs on the median Sharpe cross zero, and only buy-and-hold posts a strict-positive interval over the labelling window.

4.4 Feature Selection Stability and FFD Stability

Across the 28 CPCV folds, no feature achieves stable selection (defined as more than 80% of folds). Two features reach the moderate-stability bucket of 50% to 80%: the ETH/BTC ratio at 57.1% (16 of 28 folds) and log returns at lag 6 at 53.6% (15 of 28 folds). The third-highest is the 30-day natural gas return at 39%, the third feature used in symbolic extraction (Section 4.6). No on-chain feature reaches 50% selection frequency, so the Q1 on-chain free-lunch hypothesis fails at the cross-fold stability level under the daily-data design of this thesis. The intraday case, where on-chain indicators may carry more signal, is left to future work (Section 5.3, L1).

ATR is the only feature subject to fractional differentiation, the only column flagged non-stationary by the cross-fold ADF audit at $\alpha = 0.05$. Across 140 per-fold d^* estimates (28 folds $\times$ 5 seeds), the mean is 0.198 and the standard deviation is 0.085, with $d^* \in [0.050, 0.400]$. The std$(d^*) < 0.1$ result indicates a consistent stationarity structure across time periods, validating the FFD-only-on-ATR design choice of Section 3.6.1. The contrast between high feature-selection turnover and low FFD-d^* turnover indicates that the noise-to-signal ratio for individual features changes across regimes while the underlying time-series property motivating fractional differentiation is stable.

4.5 Symbolic Extraction Results

The symbolic-extraction pipeline runs on split 27, whose test set covers the most recent CPCV partition and is therefore closest to a deployment scenario. The KAN F1 on this fold is 0.3232 averaged over five seeds, below KAN's cross-split mean F1 of 0.4070. Input dimensionality is reduced to the top three features by 28-fold MDA stability selection: the ETH/BTC ratio (57%), log returns at lag 6 (54%), and the 30-day natural gas return (39%). The training sample is 540 events (the 80% portion of the 675 events that survive split 27's preprocessing) and the validation sample is 135 events. The data-aware fallback determines the architecture (the symbolic-extraction stage receives only the CPCV predictions and split metadata, not the per-fold tuned hyperparameters), producing the architecture $[3, 3, 2]$ (three input features, three hidden units, two output classes) with 15 active edges (samples-per-parameter ratio $6.0\times$). The samples-per-parameter floor of 5 sets the hidden width at 3 for this fold's 540 training events, clamped against the CPCV-side default hidden width of 5.

Both Phase 1 (validation accuracy 0.4815 against threshold 0.5693) and the pre-prune gate (0.4667 against 0.5359) fail. The pre-prune edge analysis at importance threshold 0.01 finds 15 of 15 edges

active, so no edges are pruned. All 15 surviving edges clear the R^2 threshold of 0.30 under symbolification, giving a 100% symbolification rate. The R^2 distribution has minimum 0.734, median 0.990, and maximum 1.000, with the top five edges all posting $R^2 > 0.99$ under the $\sin$ primitive.

Algebraic simplification converts the post-fine-tune logits into a closed-form decision function with rational coefficients in the three surviving features. The full expression and its inner-argument decomposition are in Equation 6 of Appendix A.10. The implied probability of an upward move is $P(\text{Up} \mid x) = \sigma(\text{decision}(x))$ where σ is the standard logistic.

Six primitives appear: $\sin$, $\cos$, $\tanh$, x^2, x^3, and x^4, distributed across four outer wrappers (squared, $\sin$, $\cos$, $\tanh$). The ETH/BTC ratio and the 30-day natural gas return appear in all four outer terms, log returns at lag 6 in three. With $|x|$ absent, the formula is everywhere-differentiable at the dataset median.

Pre-symbolic accuracy on the test fold is 46.67% and post-symbolic is 51.85%, a 5.18 percentage-point gain, but neither figure exceeds the 56.85% majority-class baseline, and the post-symbolic figure only marginally clears the 50% baseline.

The marginal effect of each surviving feature on $P(\text{Up})$ is plotted in Figure A.12 of Appendix A.10. A marginal-effect curve traces how the predicted upward-move probability changes as one feature varies across its observed range, with the other two held fixed at their dataset median, isolating the per-feature contribution to the decision function. The three curves show distinct functional shapes: a near-flat curve for the ETH/BTC ratio, indicating low marginal sensitivity at the median despite the feature appearing in all four outer terms of Equation 6; a cyclic shape for log returns at lag 6, produced by the $\sin$ and $\cos$ primitives in the formula and yielding $P(\text{Up})$ crossings of the 0.5 line as the feature sweeps its range; and a monotonic sigmoid-like curve for the 30-day natural gas return, dominated by the bounded $\tanh$ primitive that saturates at the extremes. Per-sigma effects at the median are largest for log returns at lag 6 (-0.4608 on the logit, -0.098 linearised) via the quartic primitive in the first outer term, then the ETH/BTC ratio (-0.2805, -0.060), and smallest for the 30-day natural gas return ($+0.0838$, $+0.018$) despite its appearance in all four outer terms, since the bounded $\tanh$ and x^3 primitives saturate near the median (full sensitivity in Table A.9).

5 DISCUSSION

5.1 Interpreting the Results

The results converge on a coherent reading. Under leakage-free CPCV evaluation with the AFML statistical-correction stack of Section 3.10.3, no architecture in the six-model comparison demonstrates a statistically significant predictive edge on Bitcoin price direction. The top trained-model DSR (KAN at 0.2470) sits well below the 0.95 significance threshold, and PBO $= 0.657$ places the cross-section in the adversarial regime: the in-sample best model underperforms the out-of-sample median in 23 of 35 partitions. Six of seven bootstrap 95% confidence intervals on the median path Sharpe cross zero, with only the buy-and-hold reference (median Sharpe 2.2722) holding a strict-positive interval. The reading is consistent with the semi-strong-form Efficient Market Hypothesis (EMH) (Fama, 1970) for the 2015 to 2026 BTC labelling window.

The leave-one-out PBO refines the picture. Random Forest is the cross-section's anchor: its removal leaves PBO unchanged at 0.657, while every other model's exclusion reduces PBO, with Logistic Regression the largest destabiliser, whose exclusion drops the 5-model PBO to 0.371 ($\Delta = -0.286$). The IS/OOS rotation is therefore concentrated in the top five trained models.

The DeLong 2-of-15 result is a detectable classification-side signal. AR Logistic versus XGBoost ($p = 0.0427$) and Random Forest versus XGBoost ($p = 0.0318$) both clear $\alpha = 0.05$ in the direction of XGBoost being the lower-AUC half. The classification-side significance does not propagate to a financial-side significance claim, since no significant pairwise difference is accompanied by a strict-positive bootstrap CI on the corresponding path-Sharpe difference.

The calibration audit of Section 4.4 supports the EMH reading. KAN's mean $P(\text{Up})$ sits within the ± 0.03 tolerance of the empirical base rate, so the bet-sizing rule of Section 3.10.1 operates on well-calibrated probabilities, and the negative DSR result is a statement about signal strength rather than about probability quality. This is the AFML pay-off. Closing both leakage channels (overlapping labels via purging and embargo, regime non-stationarity via sample weights) does not move the cross-section above the significance threshold; the residual signal is too weak for any compared architecture to clear DSR $= 0.95$, and the IS rankings are anti-informative under PBO. The finding is consistent with Chassot and Audrino (2026): properly fitted baselines are hard to beat, and prior ML-superiority results in the literature have come from suboptimal baseline-fitting schemes that handicapped the comparison.

Inside this null-result reading, KAN posts the rank-1 trained-model median Sharpe (0.5588) and DSR (0.2470) without a dominant classification edge: its mean AUC (0.5103) is mid-pack, and DeLong pairs involving KAN return $p > 0.09$ against every other trained model. The rank-1 comes from the bet-sizing translation: KAN's 70.1% abstention combined with an 89.6% long bias on non-abstained bets produces sparse high-conviction bets, with rank-2 median Sortino (0.3211) and the shallowest median maximum drawdown (-0.2466) among trained models.

5.2 Symbolic Extraction as Contribution

Symbolic extraction is a contribution separable from predictive accuracy. The extracted formula has a 100% symbolification rate (15 of 15 edges) and a closed-form expression on three features. The minimum per-edge R^2 is 0.734, the median is 0.990, and the top five edges all post $R^2 > 0.99$. The de-

cision function uses six primitives ($\sin$, $\cos$, $\tanh$, x^2, x^3, x^4), all smooth everywhere, and the formula is everywhere-differentiable at the dataset median. The closed form provides four artefacts a black-box model cannot: per-feature symbolic derivatives (Table A.9), per-feature marginal-effect curves (Figure A.12) showing the near-flat, cyclic, and monotonic functional shapes of the three surviving features, term-structure decomposition (each feature appears in at least three of four outer terms), and an audit trail from input to probability that a domain expert can inspect. Three caveats forward-referenced from Section 3.11 attach to this output: (i) the formula is fold-specific by construction; (ii) the 540-event training sample limits the representational complexity the data can identify (a constraint addressed in L2 of Section 5.3); and (iii) the post-symbolic accuracy can fall below the pre-symbolic accuracy under unfavourable spline-library mismatch, though on this fold it rose by 5.18 percentage points.

The post-symbolic accuracy gain of 5.18 percentage points (pre 46.67%, post 51.85%) follows from symbolic substitution acting as a smoothing regulariser on the per-edge B-spline shapes, with the affine fine-tuning step then tuning the per-edge scale and bias to the validation distribution. The absolute level does not survive: the post-symbolic 51.85% sits below the 56.85% majority baseline. Two dynamic validation-accuracy gates in the symbolic-extraction pipeline (a Phase 1 gate after the initial Adam training with threshold 56.93%, and a pre-prune gate before edge pruning with threshold 53.59%; see Appendix A.8) also fail on this fold. The symbolic representation faithfully captures a KAN model that has not extracted directional signal on the deployment fold.

5.3 Limitations

Four limitations of the analysis deserve explicit acknowledgement.

L1 (daily resolution discards microstructure): the pipeline operates on daily bars because the macroeconomic and on-chain feature families are only available at daily timestamps, and no free 1-hour or 4-hour source covers all 73 features. Daily was therefore the only common horizon across the four feature families. The natural extension is to upgrade to hourly bars once intraday feeds become available, which would also unlock microstructure features such as order-book imbalance, trade-sign autocorrelation, and intraday volume profiles.

L2 (sample size capped by daily CUSUM events): the 1,168 events that survive the labelling pipeline are small by ML standards and drive the wide bootstrap CIs reported in Chapter 4. The hourly extension noted in L1 would multiply the event count and tighten the bootstrap CIs.

L3 (architectural simplifications for symbolifiability): the KAN is trained with a single hidden layer, a grid size of 3, and three input features in the symbolic-extraction stage of Section 3.11. These choices keep the closed-form formula humanly readable but cap the model's representational capacity. The natural extension is to test whether richer KANs (wider, deeper, or higher grid) improve training performance, accepting that a more complex KAN produces a less interpretable symbolic formula.

L4 (computational cost): Optuna was run with 40 trials per tuned model per fold and simplified hyperparameter dictionaries to keep training tractable across the 28 CPCV splits and the six models. Larger search budgets and richer hyperparameter spaces, especially for KAN and LSTM, would let the tuning explore configurations beyond those covered here.

6 CONCLUSION AND FUTURE WORK

This thesis applies the complete AFML pipeline (CUSUM event filtering, triple-barrier labelling, fractional differentiation, sample weighting, CPCV with purging and embargo, the Deflated Sharpe Ratio, the Probability of Backtest Overfitting, and the DeLong pairwise AUC test) to BTC direction prediction on a 73-feature universe spanning four families: technical, mathematical, external (macroeconomic, crypto-macro, on-chain), and autoregressive lag. It benchmarks KAN against five comparison models (AR Logistic, Logistic Regression, Random Forest, XGBoost, LSTM) under identical CPCV splits ($N = 8$, $k = 2$, 28 splits, 7 backtest paths, 5 seeds per model), producing 840 prediction entries. It then extracts the first closed-form classification formula from a CPCV-trained KAN under the full AFML evaluation. The symbolic formula is the novel deliverable. The six-model benchmark is the methodological scaffolding that validates it.

Under leakage-free evaluation with multiple-testing correction, the literature's 80% to 90% accuracy claims do not survive. The top trained-model DSR (KAN at 0.2470) sits far below the 0.95 significance threshold, and PBO $= 0.657$ places the cross-section in the adversarial regime: 23 of 35 IS/OOS partitions see the in-sample best model underperform the out-of-sample median. The leave-one-out PBO identifies Random Forest as the anchor model whose removal leaves PBO unchanged. Six of seven bootstrap 95% confidence intervals on median path Sharpe cross zero, and only buy-and-hold holds a strict-positive interval of $]1.8903, 2.4685[$. Buy-and-hold's median Sharpe (2.2722) exceeds every trained model on the same labelling window, at the cost of a -99.98% maximum drawdown. The honest answer to Q2 is that no architecture demonstrates a statistically significant predictive edge after correcting for selection bias. KAN posts the strongest single-model evidence (rank-1 trained median Sharpe and rank-1 trained DSR), but its bootstrap CI on the median Sharpe crosses zero. The DeLong test reaches 2 of 15 significant pairs at $\alpha = 0.05$ (AR Logistic versus XGBoost and Random Forest versus XGBoost, both with XGBoost as the lower-AUC half).

What survives is the symbolic-extraction contribution. The pipeline produces a humanly auditable closed-form decision function on three features (the ETH/BTC ratio, log returns at lag 6, and the 30-day natural gas return) with a 100% symbolification rate (15 of 15 edges) and a post-symbolic test-fold accuracy of 51.85%, a 5.18 percentage-point gain over the spline KAN's 46.67%. The formula uses six smooth primitives ($\sin$, $\cos$, $\tanh$, x^2, x^3, x^4) and four outer terms with structurally distinct outer wrappers. All three surviving features have finite, well-defined gradients at the dataset median. Two structural findings from the cross-fold analysis support the symbolic-extraction reading. A lag feature (log returns at lag 6) survives extraction, validating the lag-features-in-MDA-pool design choice of Section 3.4.1. No on-chain feature survives extraction or reaches 50% cross-fold selection frequency, so the Q1 "on-chain free lunch" hypothesis fails at the cross-fold stability level. The absolute post-symbolic accuracy of 51.85% remains below the 56.85% majority baseline, meaning the symbolic representation faithfully captures a model that did not extract directional signal on the deployment fold. Interpretability is therefore a separable contribution from predictive accuracy: closed-form transparency, smooth primitives, finite gradients, and three economically interpretable features are themselves the deliverable, and they do not wait on predictive significance to be valuable.

Beyond the current design, several methodological extensions are available. KAN 2.0 on the deployment fold (Liu et al., 2024a) would let multiplication nodes discover multiplicative feature

interactions that the additive single-hidden-layer KAN of this thesis cannot represent, with the same four-step symbolic pipeline of Section 3.11 applied unchanged. Per-fold symbolic extraction across all 28 CPCV folds would let the analysis count which features and primitives recur and which are one-off, and is the highest-leverage methodological extension available. Higher-frequency (hourly) data would multiply CUSUM event counts by approximately 24 and unlock microstructure-feature spaces such as order-book imbalance and intraday volume profiles. Further extensions worth pursuing include regime-conditional analysis across BTC's distinct historical regimes, a meta-labelling layer (Singh and Joubert, 2019) above the bet-sizing stage, alternative assets such as ETH and gold, and a walk-forward versus CPCV comparison to isolate the contribution of the combinatorial purged design.

Three limitations of the current design themselves motivate future work. First, the results are conditional on a set of reasonable but untested labelling choices: the CUSUM threshold $h = 1.0\bar{\sigma}$, the $\pm 1.5\sigma_t$ profit and stop barriers with σ_t from the span-50 EWMA, the 10-day vertical barrier, and the 2% zero-class removal. A sensitivity analysis over this grid would quantify how much of the observed cross-model rank order is a property of the pipeline itself rather than of these specific constants. Second, the 73-feature universe carries natural redundancy across the technical, mathematical, and lag families; MDA selects the top-scoring features per fold but does not decorrelate them, so a hierarchical-clustering or variance-inflation pre-pass could sharpen the selected sets and clarify whether collinearity distorts the ranking. Third, the feature universe omits several families that recent crypto-quant work treats as first-class signals: investor attention (Google Trends, Wikipedia views), derivatives-market variables (perpetual funding rates, open interest, options-implied volatility), stablecoin supply dynamics, and BTC dominance. Adding these families under the same CPCV protocol would test whether the null result on trained models is a property of the feature universe or of BTC direction prediction more broadly.

Methodological honesty is not a constraint on financial machine learning. It is the prerequisite for treating interpretability as a contribution in its own right, and for reading buy-and-hold's dominance over every trained model as a result rather than a failure.

REFERENCES

Akiba, T., Sano, S., Yanase, T., Ohta, T., and Koyama, M. (2019). Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 2623–2631. ACM.

Arnold, V. I. (1958). On the representation of functions of several variables as a superposition of functions of a smaller number of variables. *Matematicheskii Sbornik*, 46(88):3–74.

Bariviera, A. F. and Merediz-Solà, I. (2021). Where do we stand in cryptocurrencies economic research? a survey based on hybrid analysis. *Journal of Economic Surveys*, 35(2):377–407.

Bourday, R., Aatouchi, I., Kerroum, M. A., and Zaaouat, A. (2024). Cryptocurrency forecasting using deep learning models: A comparative analysis. *HighTech and Innovation Journal*, 5(4).

Breiman, L. (2001). Random Forests. *Machine Learning*, 45(1):5–32.

Brock, W., Lakonishok, J., and LeBaron, B. (1992). Simple technical trading rules and the stochastic properties of stock returns. *The Journal of Finance*, 47(5):1731–1764.

Chassot, J. and Audrino, F. (2026). HARd to beat: The overlooked impact of rolling windows in the era of machine learning. *International Journal of Forecasting*, 42:330–343.

Chen, T. and Guestrin, C. (2016). 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.

Cho, S.-Y., Lee, S., and Kim, H.-G. (2025). Forecasting VIX using interpretable Kolmogorov-Arnold networks. *arXiv preprint arXiv:2502.00980*.

Corsi, F. (2009). A simple approximate long-memory model of realized volatility. *Journal of Financial Econometrics*, 7(2):174–196.

DeLong, E. R., DeLong, D. M., and Clarke-Pearson, D. L. (1988). Comparing the areas under two or more correlated Receiver Operating Characteristic curves: A nonparametric approach. *Biometrics*, 44(3):837–845.

Fama, E. F. (1970). Efficient capital markets: A review of theory and empirical work. *The Journal of Finance*, 25(2):383–417.

Gao, Y., Dong, Z., Wang, X., Wang, Z., Zhang, Y., and Wang, S. (2025). DecoKAN: Interpretable decomposition for forecasting cryptocurrency market dynamics. *arXiv preprint arXiv:2512.20028*.

Garman, M. B. and Klass, M. J. (1980). On the estimation of security price volatilities from historical data. *The Journal of Business*, 53(1):67–78.

Genet, R. and Inzirillo, H. (2024). TKAN: Temporal Kolmogorov-Arnold networks. *arXiv preprint arXiv:2405.07344*.

Guo, C., Pleiss, G., Sun, Y., and Weinberger, K. Q. (2017). On calibration of modern neural networks. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, pages 1321–1330.

Hochreiter, S. and Schmidhuber, J. (1997). Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780.

Hudson, R. and Urquhart, A. (2021). Technical trading and cryptocurrencies. *Annals of Operations Research*, 297(1):191–220.

Kolmogorov, A. N. (1957). On the representation of continuous functions of several variables by superposition of continuous functions of one variable and addition. *Doklady Akademii Nauk SSSR*, 114:953–956.

Liu, Z., Ma, P., Wang, Y., Matusik, W., and Tegmark, M. (2024a). KAN 2.0: Kolmogorov-Arnold networks meet science. *arXiv preprint arXiv:2408.10205*.

Liu, Z., Wang, Y., Vaidya, S., Ruehle, F., Halverson, J., Soljačić, M., Hou, T. Y., and Tegmark, M. (2024b). KAN: Kolmogorov-Arnold networks. *arXiv preprint arXiv:2404.19756*.

Lo, A. W. and MacKinlay, A. C. (1988). Stock market prices do not follow random walks: Evidence from a simple specification test. *The Review of Financial Studies*, 1(1):41–66.

López de Prado, M. (2018). *Advances in Financial Machine Learning*. John Wiley & Sons, Hoboken, NJ.

Loshchilov, I. and Hutter, F. (2019). Decoupled weight decay regularization. *Proceedings of the International Conference on Learning Representations (ICLR)*.

Máté, D., Raza, H., Ahmad, I., and Kovács, S. (2024). Next step for Bitcoin: Confluence of technical indicators and machine learning. *Journal of International Studies*, 17(3):68–94.

Nabar, O. and Shroff, G. (2023). Conservative predictions on noisy financial data. In *Proceedings of the 4th ACM International Conference on AI in Finance (ICAIF)*, Brooklyn, NY. ACM.

Noorizadegan, A., Wang, S., Ling, L., and Dominguez-Morales, J. P. (2026). A practitioner's guide to Kolmogorov-Arnold networks. *arXiv preprint arXiv:2510.25781*.

Oad, V., Pathak, P., Innan, N., D, S., and Shafique, M. (2025). KASPER: Kolmogorov Arnold networks for stock prediction and explainable regimes. *arXiv preprint arXiv:2507.18983*.

Omole, O. and Enke, D. (2024). Deep learning for Bitcoin price direction prediction: Models and trading strategies empirically compared. *Financial Innovation*, 10(117).

Platt, J. (1999). Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In *Advances in Large Margin Classifiers*, pages 61–74. MIT Press.

Sharpe, W. F. (1966). Mutual fund performance. *The Journal of Business*, 39(1):119–138.

Singh, A. and Joubert, J. (2019). Does meta-labeling add to signal efficacy? *SSRN Electronic Journal*.

Ślepaczuk, R. and Bieganowski, B. (2024). Supervised autoencoders with fractionally differentiated features and triple barrier labelling enhance predictions on noisy data. In *Proceedings of the 5th ACM International Conference on AI in Finance (ICAIF)*, New York, NY. ACM.

Urquhart, A. (2016). The inefficiency of Bitcoin. *Economics Letters*, 148:80–82.

Wu, J., Zhang, X., Huang, F., Zhou, H., and Chandra, R. (2024). Review of deep learning models for crypto price prediction: Implementation and evaluation. *arXiv preprint arXiv:2405.11431*.

Yamak, P. T., Li, Y., Zhang, T., and Pathan, M. S. (2025). Kolmogorov-Arnold networks for time series forecasting: A comprehensive review. *Cluster Computing*, 28:929.

Yang, D. and Zhang, Q. (2000). Drift-independent volatility estimation based on high, low, open, and close prices. *The Journal of Business*, 73(3):477–492.

A Appendices

A.1 Methodology Pipeline

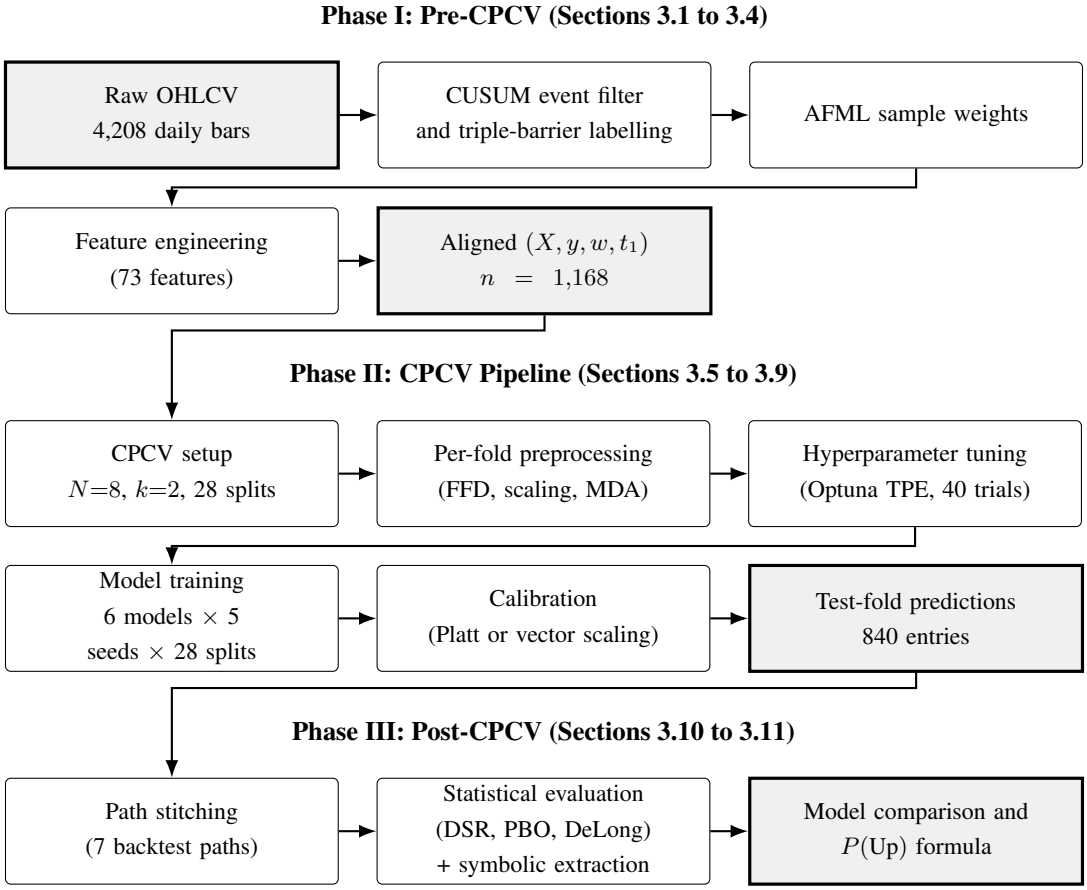


FIGURE A.1: End-to-end methodology pipeline. Light-bordered boxes are processing steps, dark-bordered shaded boxes are data states. ↩

A.2 Data and Labelling Diagnostics

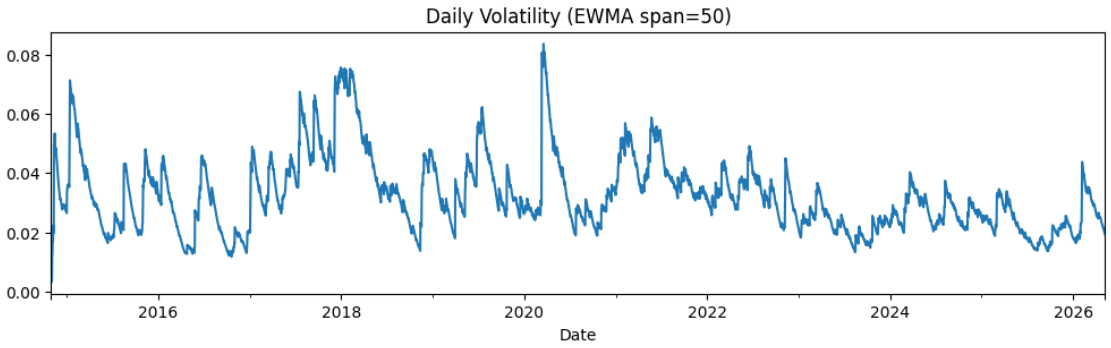


FIGURE A.2: EWMA daily volatility of BTC-USD log returns with span 50. ↩

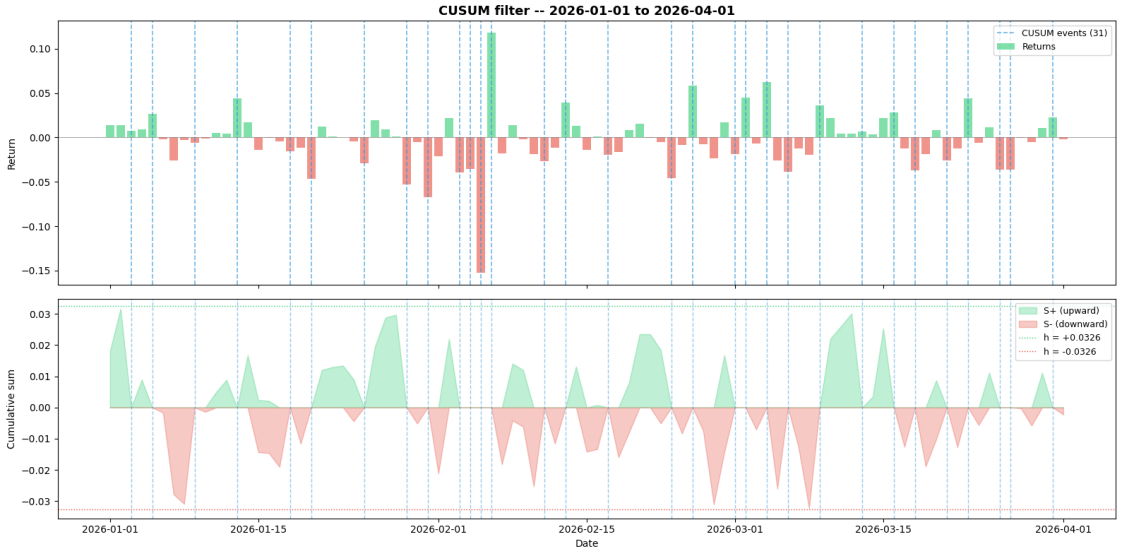


FIGURE A.3: CUSUM event filter applied to BTC daily log returns over a three-month window. ↩

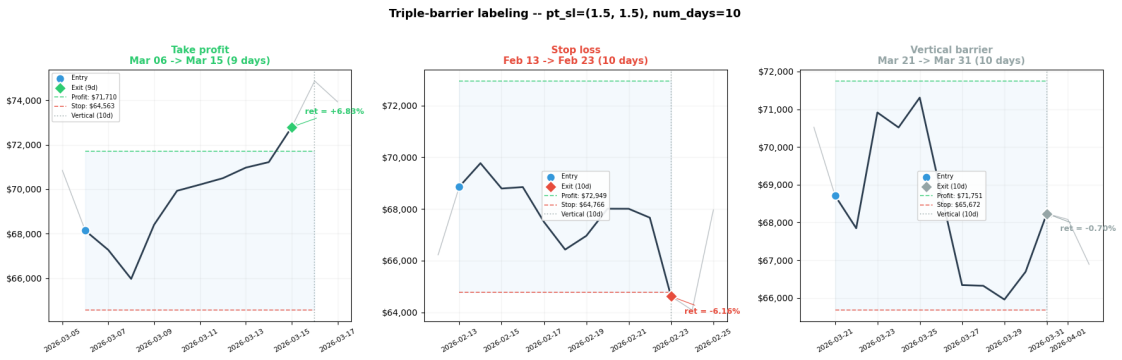


FIGURE A.4: Triple-barrier labelling on three representative events at $pt\_sl = (1.5, 1.5)$ and $num\_days = 10$. Left: profit-take hit at nine days. Middle: stop-loss hit at ten days. Right: vertical-barrier expiry at ten days. ↩

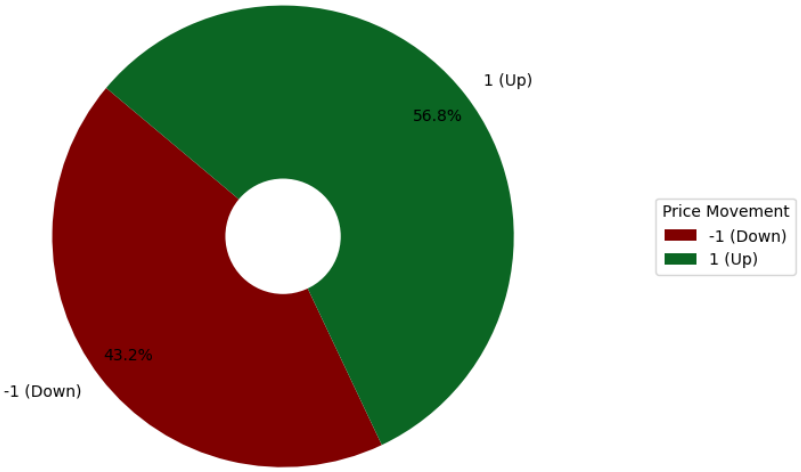


FIGURE A.5: Class distribution of the 1,168 binary-labelled observations after zero-class removal. ↩

A.3 Weighting Diagnostics

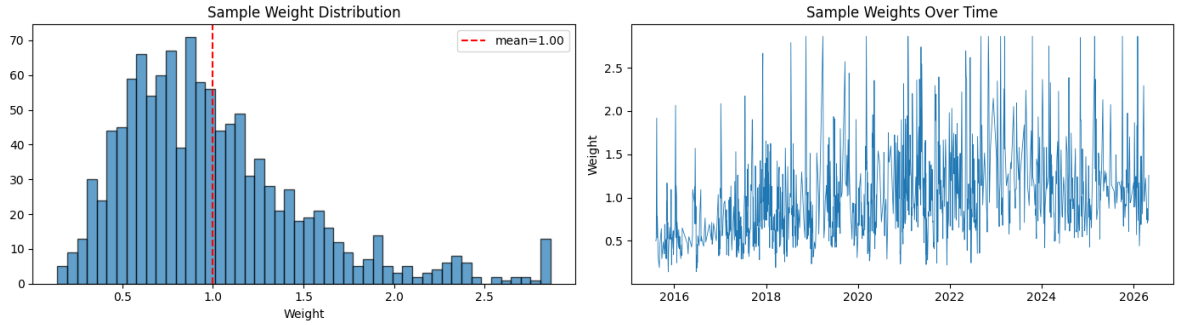


FIGURE A.6: Sample weight diagnostics. Left: distribution of per-event weights after the AFML three-component scheme and 99th-percentile capping. Right: weights over time. ↩

Date	11/18	03/19	03/20	02/21	09/22	11/22	02/23	03/23	10/23	02/25	10/25	02/26
Label	-1	$+1$	-1	$+1$	$+1$	-1	$+1$	$+1$	$+1$	$+1$	-1	-1
Return	-0.151	$+0.191$	-0.387	$+0.233$	$+0.135$	-0.123	$+0.114$	$+0.133$	$+0.115$	$+0.117$	-0.083	-0.171
Holding (d)	5	4	4	5	3	4	6	4	3	4	5	2

TABLE A.1: The twelve events at the 99th-percentile weight cap (2.862). Dates as MM/YY. ↩

A.4 Feature Universe

Family	Count	Examples and parameters	Source
Technical analysis	25	Daily log returns, realised volatility (21d, $\sqrt{365}$), Garman-Klass and Yang-Zhang volatility, ATR (log, span 14), Bollinger width, volatility-term ratios (7/30, 30/90), RSI, MACD (line, signal, histogram), ROC 14, Stochastic %K and %D, Williams %R, CCI, OBV (log), Chaikin oscillator, MFI, rolling skewness and kurtosis, EMA ratios (20/50, 50/200), VWMA ratio (20/50)	BTC OHLCV
Mathematical	9	Shannon entropy (30d), Lempel-Ziv complexity (90d), negentropy (30d), Hurst exponent (180d, sub-windows 14/30/60/90), Lo-MacKinlay variance ratio (90d, lag 7), Jarque-Bera (90d), SADF (min sub-length 90, lag 1), SMT polynomial-1, SMT exponential	BTC log returns
Macroeconomic	20	DXY ROC 30, US 2Y and 10Y yields, 2Y-10Y and 10Y-30Y slopes, VIX, return-over-window pairs (14d, 30d) for S&P 500, Nasdaq, gold, silver, copper, oil, natural gas	Yahoo Finance, FRED
Crypto cross-sectional	1	ETH/BTC ratio (daily ETH-USD over BTC-USD price)	CoinMetrics
On-chain	8	Active-address ROC 14, transaction-count ROC 14, hash-rate ROC 30, MVRV, net exchange flow, fee per transaction, exchange supply percent, native-unit issuance	CoinMetrics (shifted by 1 day)
Autoregressive lag	10	Log returns at lags $\{1, 2, 3, 4, 5, 6, 7, 14, 21, 30\}$	BTC log returns

TABLE A.2: Feature universe (73 features), grouped by family. ↩

A.5 CPCV Groups

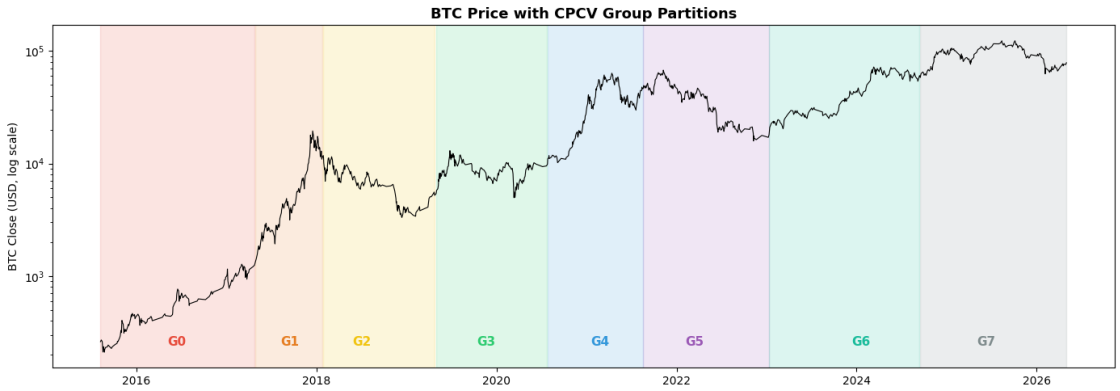


FIGURE A.7: Eight CPCV groups overlaid on BTC closing price, each holding 146 of the 1,168 aligned observations. ↩

A.6 Hyperparameter Search Spaces

Model	Family	Architecture	Hyperparameters	Role
AR Logistic	Econometric	LR on lags 1 to 7, 14, 21, 30	• deterministic: $C = 1.0$, L2, balanced, max_iter=1000	Pure momentum baseline (Q1, Q2)
Logistic Regression	Linear ML	LR on MDA-selected features	• $C \in [10^{-4}, 10^{2}]$ • penalty $\in \{\text{L1}, \text{L2}\}$ (balanced, max_iter=1000)	Linear baseline (Q4)
Random Forest	Ensemble	Bootstrap sampling, balanced_subsample	• n_estimators $\in [100, 250]$ • max_depth $\in [2, 6]$ • min_samples_leaf $\in [15, 40]$ • max_features $\in \{\text{sqrt}, \text{log2}\}$ (n_jobs=-1)	Nonlinear ensemble (Q4)
XGBoost	Ensemble	500 trees, early stopping at 20 rounds	• max_depth $\in [1, 3]$ • lr $\in [10^{-2}, 0.3]$ • min_child_weight $\in [5, 30]$ • subsample, colsample_bytree $\in [0.6, 1.0]$ • gamma $\in [10^{-8}, 1]$ • reg_alpha, reg_lambda $\in [10^{-8}, 10]$ (binary:logistic)	Gradient boosting (Q4)
LSTM	Neural	window=14, single layer, last-hidden pooling	• hidden $\in \{16, 32\}$ • dropout $\in [0.1, 0.5]$ • lr $\in [10^{-4}, 5 \times 10^{-2}]$ ($T_0 = 25$, batch=64, label smoothing 0.1)	Temporal dependencies (Q4)
KAN	Neural	efficient_kan $[n, \text{width}_1, 2]$, single hidden, spline order 3	• $\text{width}_1 \in [2, 6]$ • grid $\in \{3, 5\}$ • lr $\in [5 \times 10^{-4}, 5 \times 10^{-2}]$ • weight_decay $\in [10^{-5}, 5 \times 10^{-3}]$ ($\text{width}_2 = 0$, $T_0 = 30$, label smoothing 0.1)	Interpretable architecture (Q3, Q4)

TABLE A.3: Summary of the six models. Tuned hyperparameter ranges are listed as bullets, with fixed parameters in parentheses. ↩

A.7 Statistical Test Formulae

- **Fractional-difference weight recursion.** For fractional difference of order d: $\omega_0 = 1$, $\omega_k = -\omega_{k-1} \cdot (d - k + 1)/k$. Truncated at the first k with $|\omega_k| < 10^{-4}$, or at $k = 200$. ↩

- **Deflated Sharpe Ratio.** Define

$$E[\max \text{SR}] = \sqrt{2 \ln n}\Big(1 - \frac{\gamma}{2 \ln n}\Big) + \frac{\gamma}{2\sqrt{2 \ln n}}, \qquad \sigma_{\text{SR}} = \sqrt{\frac{1 - \text{skew} \cdot \text{SR} + \frac{\text{kurt}+2}{4} \cdot \text{SR}^2}{n_{\text{obs}} - 1}}.$$

  Then $\text{DSR} = \Phi((\text{SR}_{\text{obs}} - E[\max \text{SR}])/\sigma_{\text{SR}})$, with n the number of candidate strategies and γ the Euler-Mascheroni constant (López de Prado, 2018, Chapter 11). ↩

- **DeLong pairwise AUC test.**

$$z = \frac{\text{AUC}_a - \text{AUC}_b}{\sqrt{\text{Var}(\text{AUC}_a) + \text{Var}(\text{AUC}_b) - 2\,\text{Cov}(\text{AUC}_a, \text{AUC}_b)}},$$

  with Var and Cov from the structural-component method on pooled predicted probabilities. Two-sided p-value from the standard normal CDF (DeLong et al., 1988). ↩

A.8 Symbolic-Extraction Pipeline Specification

The pipeline runs on split 27. The stored d^* and RobustScaler (Section 3.6) are applied to the ATR series and training-fold features, dimensionality is reduced to the top three MDA-selected features, and the data are split chronologically 80/20 into training (540) and validation (135 events), with tanh normalisation $z = \tanh((x - \mu)/(\sigma + 10^{-8}))$ fitted on the training portion. Default architecture is hidden width 5, grid size 5, spline order 3, falling back to hidden width 3 and grid size 3 via the samples-per-parameter envelope of Section 3.11 (Table A.4 details the four steps; Section 4.6 reports the realised split-27 architecture).

Step	Operation	Parameters and detail
1	Three-phase training	Phase 1: 600 Adam steps (learning rate 10^{-3}, weight decay 10^{-3}, Gaussian noise $\sigma = 0.05$ clamped to $[-1, 1]$, early stopping on validation loss). Phase 2a: 20 LBFGS steps (learning rate 10^{-2}, no regularisation). Phase 2b: 20 LBFGS steps with L1 plus entropy regularisation at $\lambda = 0.002$ to encourage sparse activations. Grid extension disabled. Dynamic gates after Phase 1 and pre-prune check validation accuracy against the larger of 0.53 and the validation-set majority-class baseline plus 0.01.
2	Pruning	Validation forward pass populates cached activations; each edge is scored by activation magnitude, and edges below threshold 0.01 are removed. The pruned model reverts to the pre-prune state if any edge removal breaks the forward pass.
3	Symbolification	14 candidates: x, x^2, x^3, x^4, exp, log, $\sqrt{\cdot}$, tanh, sin, cos, arctan, $\lvert x\rvert$, sgn(x), and the zero function. Per-edge winner is the candidate with $R^2 \geq 0.30$; below threshold the edge keeps its spline form. A brute-force fallback iterates through the 13 non-zero candidates when the default symbolic-suggestion routine returns the zero function, fitting each candidate explicitly and recording its R^2.
4	Fine-tuning	30 LBFGS steps at learning rate 4×10^{-4} on each edge's affine parameters $a \cdot f(bx + c) + d$. The model reverts if any step produces a non-finite parameter.

TABLE A.4: Four-step symbolic-extraction pipeline.

After fine-tuning, symbolic algebra simplifies $\text{logit}_{\text{bullish}} - \text{logit}_{\text{bearish}}$ into a closed form on the surviving features (30-second timeout). The $|x|$ primitive is non-differentiable at zero and would break the partial-derivative sensitivity analysis at any feature whose mean falls there; it does not appear in the surviving primitive set.

A.9 Supplementary Results

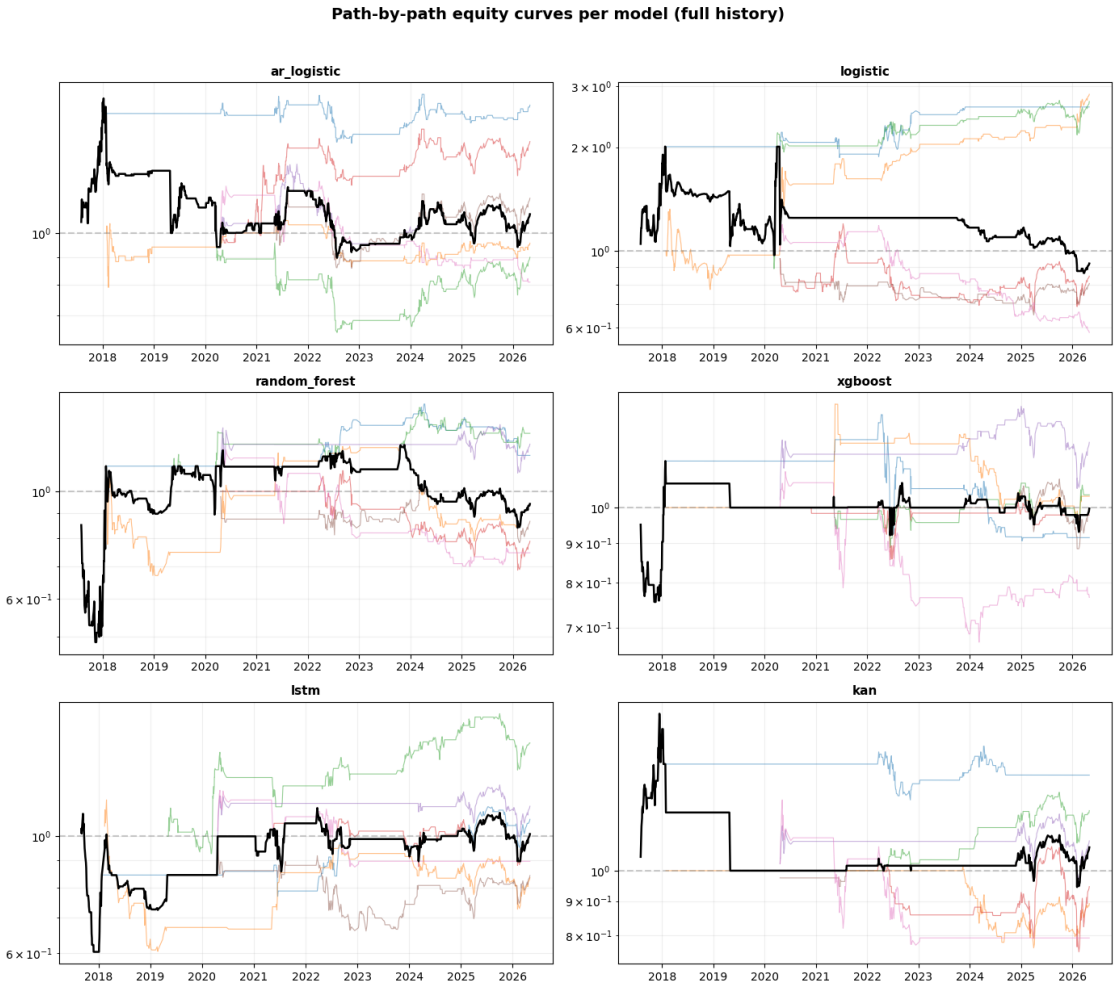


FIGURE A.8: Path-by-path equity curves per model. Seven CPCV paths in muted colours, median path in bold black. Log-scaled vertical axis; dashed line at 1.0 marks breakeven. ↩

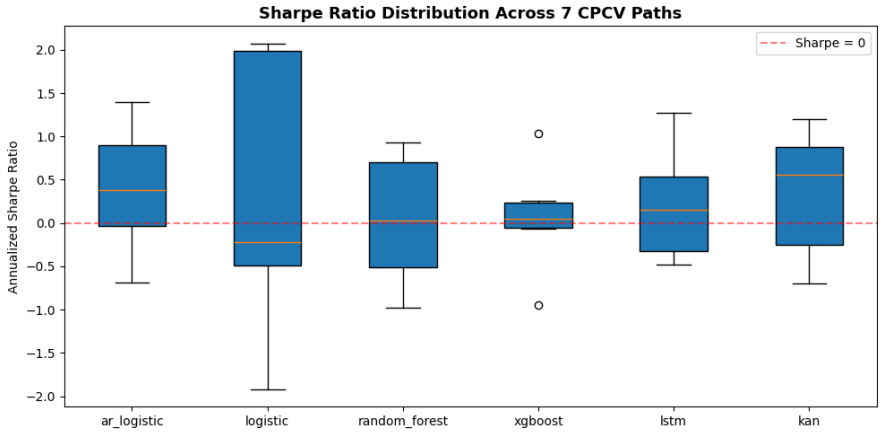


FIGURE A.9: Path Sharpe distribution across the seven CPCV backtest paths, six trained models. Dashed line: SR $= 0$. ↩

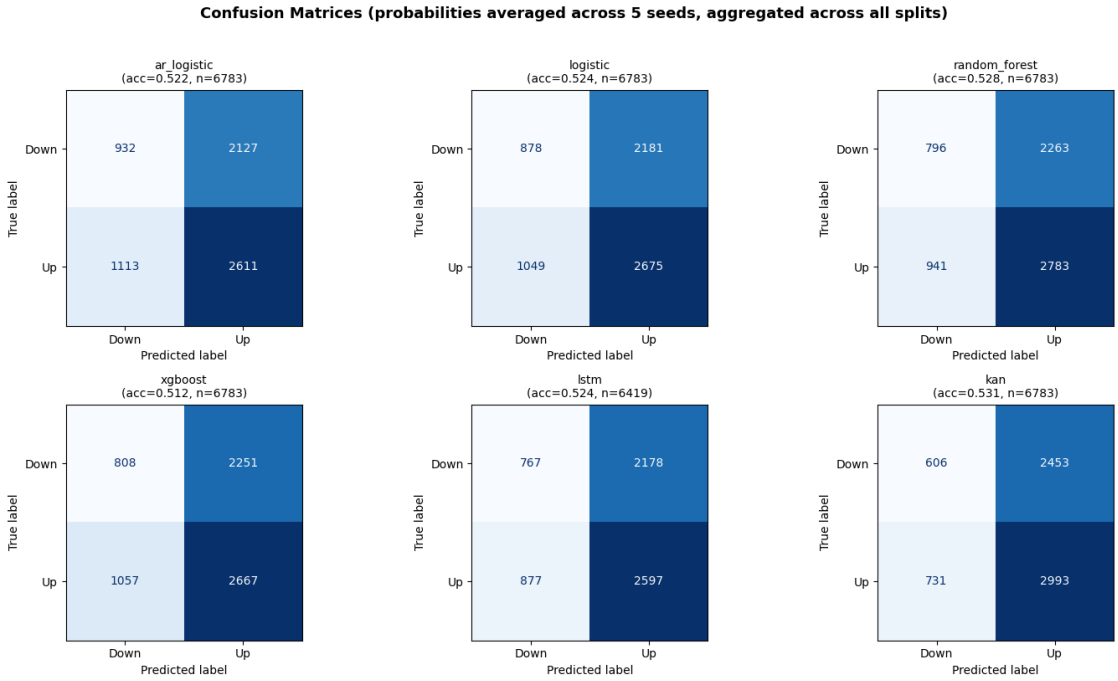


FIGURE A.10: Confusion matrices for the six trained models, seed-averaged per (model, split) and pooled across the 28 splits ($n = 6{,}783$, LSTM $n = 6{,}419$). ↩

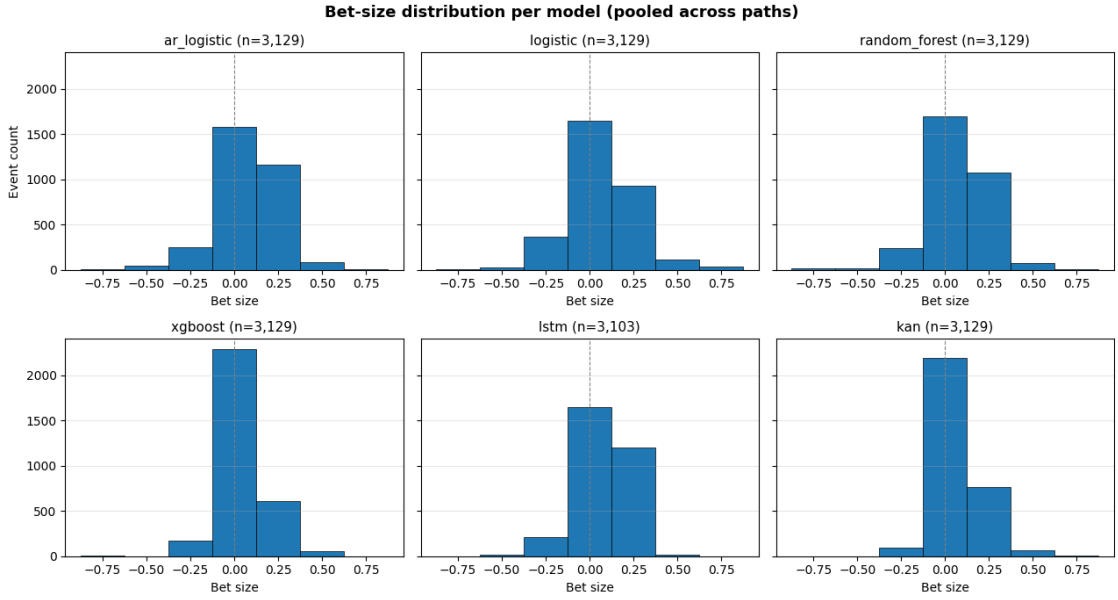


FIGURE A.11: Bet-size distribution per model, pooled across the seven backtest paths. Discrete support $\{-0.75, -0.5, -0.25, 0, 0.25, 0.5, 0.75\}$. ↩

Pair	AUC_a	AUC_b	ΔAUC	z	p	Sig.
AR Logistic vs Logistic	0.5117	0.5084	$+0.0032$	0.3982	0.6905	
AR Logistic vs Random Forest	0.5117	0.5068	$+0.0048$	0.5671	0.5706	
AR Logistic vs XGBoost	0.5117	0.4942	$+0.0175$	2.0263	0.0427	yes
AR Logistic vs LSTM	0.5149	0.5048	$+0.0100$	0.9918	0.3213	
AR Logistic vs KAN	0.5117	0.5062	$+0.0055$	0.7018	0.4828	
Logistic vs Random Forest	0.5084	0.5068	$+0.0016$	0.2528	0.8005	
Logistic vs XGBoost	0.5084	0.4942	$+0.0143$	1.7728	0.0763	
Logistic vs LSTM	0.5100	0.5048	$+0.0052$	0.4964	0.6196	
Logistic vs KAN	0.5084	0.5062	$+0.0022$	0.3596	0.7191	
Random Forest vs XGBoost	0.5068	0.4942	$+0.0127$	2.1466	0.0318	yes
Random Forest vs LSTM	0.5091	0.5048	$+0.0043$	0.4331	0.6650	
Random Forest vs KAN	0.5068	0.5062	$+0.0006$	0.0932	0.9258	
XGBoost vs LSTM	0.4956	0.5048	-0.0092	-0.9208	0.3572	
XGBoost vs KAN	0.4942	0.5062	-0.0121	-1.6872	0.0916	
LSTM vs KAN	0.5067	0.5073	-0.0006	-0.0561	0.9552	

TABLE A.5: Full DeLong pairwise AUC test results, all 15 pairs. Pooled across 28 splits with seed-averaging per (model, split) cell. Significance at $\alpha = 0.05$. ↩

	AR Logistic	Logistic	Random Forest	XGBoost	LSTM	KAN
Mean $P(\text{Up})$	0.5440	0.5506	0.5418	0.5314	0.5497	0.5421
Δ vs base rate	-0.0245	-0.0179	-0.0267	-0.0371	-0.0187	-0.0263
Within ± 0.03	yes	yes	yes	**no**	yes	yes

TABLE A.6: Calibration first-moment audit. Empirical base rate $P(\text{Up}) = 0.5685$; tolerance ± 0.03. Only XGBoost misses ($\Delta = -0.0371$). ↩

Model	n_{events}	Abstention	Mean |bet|	At max (0.75)	Long share	Short share
AR Logistic	3,129	50.5%	0.273	0.6%	81.0%	19.0%
Logistic	3,129	52.6%	0.287	2.8%	73.3%	26.7%
Random Forest	3,129	54.2%	0.274	1.5%	81.2%	18.8%
XGBoost	3,129	73.1%	0.269	0.5%	79.2%	20.8%
LSTM	3,103	53.2%	0.257	0.0%	84.0%	16.0%
KAN	3,129	70.1%	0.272	0.9%	89.6%	10.4%

TABLE A.7: Bet-size distribution per model. Long/short shares exclude abstentions; LSTM count reflects its 14-day warmup (Section 3.7.5). ↩

Model excluded	Logistic Regression	AR Logistic	LSTM	XGBoost	KAN	Random Forest
5-model PBO	0.371	0.457	0.457	0.543	0.543	0.657
Δ vs baseline	-0.286	-0.200	-0.200	-0.114	-0.114	$+0.000$

TABLE A.8: Leave-one-out PBO. Baseline (six-model) PBO is 0.657. Each column drops one model and recomputes PBO on the remaining 5×7 Sharpe matrix. ↩

A.10 Closed-Form Decision Function

The closed-form decision function produced by the four-step symbolic-extraction pipeline of Section 3.11 is given in Equation 6, with coefficients reported as rationals from symbolic-algebra rational

simplification at tolerance 10^{-3}.

$$\begin{aligned}\text{decision}(x) = \tfrac{3}{283}\Bigg(&-\tfrac{284\,\text{ETH/BTC}}{505} - \tfrac{1}{1000}\Big(-\tfrac{1766\,\text{LogRet}_6}{179} - \tfrac{1529}{843}\Big)^4 \\ &-\tfrac{434}{663}\tanh\Big(\tfrac{3217\,\text{NatGas}_{30}}{944} + \tfrac{2141}{935}\Big) + \tfrac{4511}{913}\Big)^2 \\ &-\tfrac{351}{877}\sin\big(A_2(x)\big) + \tfrac{197}{828}\cos\big(A_3(x)\big) + \tfrac{455}{932}\tanh\big(A_4(x)\big) + \tfrac{11}{19}\end{aligned} \tag{6}$$

The inner arguments $A_2(x)$, $A_3(x)$, $A_4(x)$ are themselves multi-feature compositions of the three surviving features under the six surviving primitives ($\sin$, $\cos$, $\tanh$, x^2, x^3, x^4). The implied upward-move probability is $P(\text{Up} \mid x) = \sigma(\text{decision}(x))$ with σ the standard logistic.

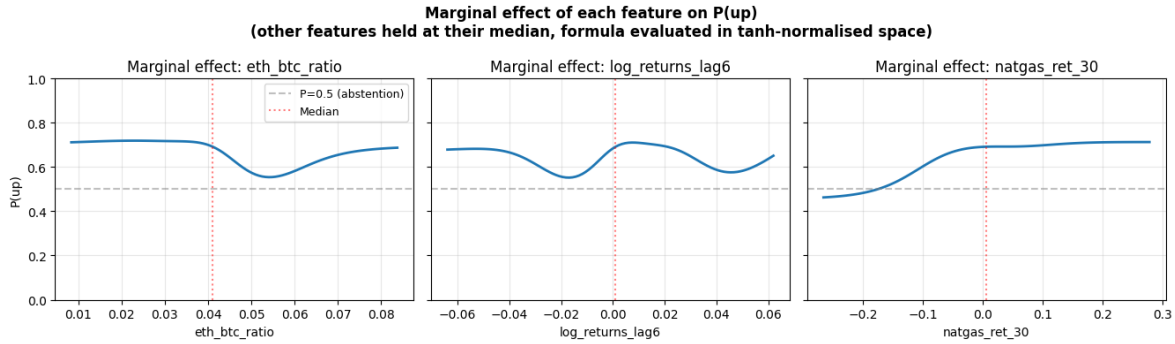


FIGURE A.12: Marginal effect of each surviving feature on $P(\text{Up})$, with the other two features held at their median values, in the tanh-normalised space. Grey dashed line: $P(\text{Up}) = 0.5$. Red dotted line: feature median. ↩

Feature	Median	Std	$\partial/\partial x$ at median	σ-effect on logit	σ-Δp
ETH/BTC ratio	+0.0460	0.0247	−41.1765	−0.2805	−0.0598
Log returns lag 6	−0.0008	0.0394	+30.6748	−0.4608	−0.0983
Natgas ret 30	+0.0025	0.1618	+0.4236	+0.0838	+0.0179

TABLE A.9: Numerical sensitivity at the dataset median. σ-effect on logit is $(\partial\text{decision}/\partial x) \cdot \sigma_x$, and σ-Δp is its linearised probability shift. ↩

A.11 Code and Data Availability

The complete Python pipeline, LaTeX source, and cached preprocessed features supporting this thesis are publicly available at `https://github.com/pterletskiy/BTC_KAN_AFML` under the MIT license. The repository is organised into three phases mirroring the methodology of Chapter 3 (pre-CPCV, CPCV, post-CPCV), with `main.ipynb` as the end-to-end driver. Results reported in this thesis correspond to release tag `v1.0-thesis`. External data are retrieved at runtime from Yahoo Finance (BTC OHLCV, macroeconomic series), FRED (macroeconomic series), and CoinMetrics (crypto-macro and on-chain), all under free-tier access.