Robust Meta-Labeling for Trend Following Strategies

A Comparative Analysis of Failed Heuristics vs. Statistical Rigor

Quantitative Research Report

February 10, 2026

Abstract

This report documents the iterative development of a quantitative trading system. We detail the failure of initial heuristic approaches—specifically the Supertrend "New Trend Engine" and directional LightGBM models (Model A). We then present the final, robust architecture (Model B), which utilizes a Donchian Channel signal generator filtered by a Random Forest Meta-Labeler. Additionally, we introduce a Statistical Arbitrage module for identifying cointegrated pairs to diversify the strategy's risk profile.

1 Data Preprocessing and Universe Selection

Before signal generation, raw OHLCV data undergoes rigorous cleaning to ensure data integrity.

1.1 Implemented Filters

1.1.1 Hampel Filter

To prevent spurious signals caused by data spikes, we apply a Hampel Filter. This filter is preferred over standard deviation filters because it uses robust statistics (Median and MAD) that are not themselves skewed by the outliers they are trying to detect.

For a sliding window W_t of radius k (total length $2k+1$):

$$W_t = \{x_{t-k}, \dots, x_t, \dots, x_{t+k}\}$$

We calculate the local median m_t and the Median Absolute Deviation (MAD):

$$m_t = \text{median}(W_t)$$

$$\text{MAD}_t = \text{median}\Big(\big\{|x_i - m_t| : x_i \in W_t\big\}\Big)$$

To estimate the scale consistent with a Gaussian distribution, we define the robust standard deviation $\hat{\sigma}_t$:

$$\hat{\sigma}_t = 1.4826 \cdot \text{MAD}_t$$

The filter replaces x_t if it deviates by more than $n_\sigma = 3$ standard deviations:

$$y_t = \begin{cases} m_t & \text{if } |x_t - m_t| > n_\sigma \cdot \hat{\sigma}_t \\ x_t & \text{otherwise} \end{cases}$$

1.1.2 Other Filters

- **Forward Filling:** Missing data is forward-filled to preserve causality.
- **Liquidity Filter:** Dynamic filtering based on Dollar Volume to remove illiquid assets.

1.2 Discarded Filters

- **Fat Finger Filter:** A logic rejecting single-day returns $> 50\%$ was discarded as it incorrectly removed legitimate corporate action events (e.g., reverse splits).
- **Penny Stock Filter:** Static price thresholds ($P < \$5$) proved too aggressive and were replaced by the volume filter.

—

2 Signal Generation: Evolution of the Engine

2.1 Attempt 1: The "New Trend Engine" (Supertrend) – *Failed*

Our initial signal generator relied on the Supertrend indicator with complex dual-mode entry logic.

Logic:

- **Supertrend Calculation:** Period $n = 10$, Multiplier $m = 3.0$.
- **Regime:** Bull (Green) if $P_t >$ Band, Bear (Red) if $P_t <$ Band.

Entry Triggers (Dual-Mode):

1. **Mode A (Reversal):** Buy immediately when Regime flips Red $\rightarrow$ Green.
2. **Mode B (Continuation):** Buy if Regime is Green AND $RSI > 50$ AND $MFI > 35$.

Reason for Failure: While conceptually appealing, this engine was prone to "whipsaws" in ranging markets. The specific thresholds (RSI 50, MFI 35) introduced excessive parameters that were brittle to regime changes, leading to curve-fitting.

2.2 Final Solution: Donchian Trend Model – *Successful*

We replaced the complex Supertrend logic with a robust, parameter-light Donchian Breakout system.

Logic: Let $H_{n,t} = \max(P_{t-1}, \ldots, P_{t-n})$ and $L_{n,t} = \min(P_{t-1}, \ldots, P_{t-n})$. The signal s_t is:

$$s_t = \begin{cases} +1 & \text{if } P_t > H_{n,t} \quad \text{(Long Breakout)} \\ -1 & \text{if } P_t < L_{n,t} \quad \text{(Short Breakout)} \\ 0 & \text{otherwise} \end{cases}$$

This generator accepts higher false positive rates in exchange for capturing every major trend, delegating the filtering task to the Meta-Model.

—

3 Feature Engineering and Meta-Modeling

3.1 Model A: Directional Prediction (LightGBM) – *Failed*

The first meta-model attempted to filter signals by predicting the raw direction of the next price bar.

- **Target:** $y_{t+1} = \mathbb{1}(P_{t+1} > P_t)$.
- **Features:** Raw RSI, MFI, and Rolling Mean distances.
- **Model:** LightGBM (Gradient Boosting).

Failure Analysis:

1. **Non-Stationarity:** Raw technical indicators have shifting means and variances, violating the i.i.d. assumption of the learner.
2. **Optimization Difficulty:** LightGBM hyperparameters were unstable on this dataset. The model struggled to converge, oscillating between overfitting noise and underfitting signal.

3.2 Model B: Meta-Labeling (Random Forest) – *Successful*

The final architecture shifts the goal from predicting *price* to predicting *signal quality*.

3.2.1 Meta-Labeling Target

Given a Donchian signal s_t, the target is binary trade profitability:

$$y_t = \begin{cases} 1 & \text{if Trade Profit } > 0 \\ 0 & \text{otherwise} \end{cases}$$

3.2.2 Feature Engineering: Fractional Differentiation

To solve the stationarity issue without losing trend memory (as integer differencing does), we applied **Fractional Differentiation**. We optimize the order d via the Augmented Dickey-Fuller (ADF) test:

$$d^* = \min\{d \mid \text{p-value}(\text{ADF}(\nabla^d x_t)) < 0.05\}$$

This ensures inputs to the model are stationary while retaining the maximum possible history.

3.2.3 Classifier: Random Forest

We selected a **Random Forest Classifier** over LightGBM for the final implementation.

- **Small Data Robustness:** The Donchian generator produces sparse events. Random Forests (bagging) are superior to Boosting in low-sample regimes, reducing the variance of the prediction.
- **Decision Rule:** Execute trade if $\hat{P}(y_t = 1|X_t) >$ Threshold.

3.3 Validation Scheme: CPCV

We utilized **Combinatorial Purged Cross-Validation (CPCV)**.

- **Purging:** Removing training samples that overlap with test labels.
- **Embargoing:** Enforcing a buffer period after test sets to prevent correlation leakage.

—

4 Backtesting and Validation

4.1 Portfolio Construction: Inverse Volatility

To manage risk, capital allocation is inversely proportional to asset volatility:

$$w_i = \frac{1/\sigma_i}{\sum_j (1/\sigma_j)}$$

This prevents high-volatility assets from dominating the portfolio variance.

4.2 Transaction Costs

The backtest incorporates realistic friction modeled via the **Square Root Law of Market Impact**. This law states that market impact is proportional to the square root of the ratio of order size to daily volume.

$$\text{Cost}_{\text{slip}} = k \cdot \sigma_t \cdot \sqrt{\frac{Q}{V_t}}$$

where:

- $k \approx 0.1$ is the market impact coefficient.
- σ_t is the daily volatility (standard deviation of returns).
- Q is the order size (shares or dollars).
- V_t is the daily average volume.

We also apply a fixed fee component per trade to account for exchange and commission fees.

—

5 Statistical Arbitrage Integration

To complement the directional Trend Following strategy, we implemented a mean-reversion module based on Statistical Arbitrage (Pairs Trading). This module identifies pairs of assets that move together in the long run but diverge temporarily.

5.1 Correlation Screening

The universe of assets is first screened to identify pairs with high co-movement. We utilize the **Spearman Rank Correlation** coefficient (ρ_s) rather than Pearson correlation, as it captures monotonic relationships and is less sensitive to outliers.

For two asset price series X and Y, converted to ranks $R(X)$ and $R(Y)$:

$$\rho_s = 1 - \frac{6 \sum d_i^2}{n(n^2 - 1)}$$

where $d_i = R(X_i) - R(Y_i)$. We select the top $k = 30$ pairs with the highest ρ_s for cointegration testing.

5.2 Cointegration Testing (Engle-Granger)

Correlation does not imply mean reversion. To ensure the spread between two assets is stationary, we apply the **Engle-Granger Two-Step Method**.

Step 1: Regression We estimate the hedge ratio β using Ordinary Least Squares (OLS) without an intercept (forcing the spread to mean 0):

$$\beta = \frac{\sum P_t^A P_t^B}{\sum (P_t^B)^2} \quad \text{(approximate OLS slope)}$$

The spread is defined as:

$$S_t = P_t^A - \beta P_t^B$$

Step 2: Stationarity Test We test the residuals S_t for a unit root using the Augmented Dickey-Fuller (ADF) test.

$$\Delta S_t = \gamma S_{t-1} + \sum_{i=1}^{p} \delta_i \Delta S_{t-i} + \epsilon_t$$

A pair is confirmed as a cointegration candidate if the ADF p-value < 0.05. *Example Result:* Asset_010 vs Asset_016 showed a strong cointegration relationship ($p = 0.0017$) with $\beta \approx 0.35$.

5.3 Signal Generation: Z-Score Spread

Once a valid pair is identified, we normalize the spread using a rolling Z-Score to generate entry and exit signals.

$$z_t = \frac{S_t - \mu_t}{\sigma_t}$$

where μ_t and σ_t are the rolling mean and standard deviation over a window $W = 60$.

Trading Logic:

- **Long Spread:** If $z_t < -2.0$ (Spread is too low, expect reversion).
- **Short Spread:** If $z_t > +2.0$ (Spread is too high, expect reversion).
- **Exit:** If $|z_t| < 0.5$ (Spread has reverted to mean).

6 Conclusion

The transition from Model A to Model B highlights the importance of statistical rigor. By abandoning the parameter-heavy Supertrend/LightGBM approach in favor of a Donchian/Random Forest Meta-Labeler with fractionally differentiated features, we achieved a stable, regime-aware trading system. Furthermore, the integration of Statistical Arbitrage provides a non-correlated revenue stream, smoothing the overall portfolio equity curve.