Smart Money Futures Signal Bot

Algorithm Description & Validation Report

Generated: October 21, 2025 at 05:46 AM

Project Status	Value
Total Signals Analyzed	397
Validation Period	Oct 19-21, 2025 (2.5 days)
Gross Win Rate	37.5%
Net Win Rate	20.8% - 29.2%
Economic Viability	NOT VIABLE
Recommendation	DO NOT DEPLOY

Table of Contents

1. Executive Summary

After implementing a rigorous validation framework addressing all statistical concerns, **the current signal generation algorithm is economically non-viable**:

■ **Negative expected value** across all execution cost scenarios (-0.146% to -0.256% per trade)

■ **37.5% gross win rate** (before costs) - far below 80% target

■ **-0.036% average gross PnL** - loses money BEFORE execution costs

■■ **Limited data** (2.5 days, 397 signals) introduces variance but trend is clearly negative

RECOMMENDATION: Do NOT deploy to live trading. Continue collecting data while redesigning signal generation logic to address fundamental accuracy issues.

Infrastructure Status: ■ Excellent - All services operational (CVD, Liquidation, Tracker, Watchdog), effectiveness tracking functional, self-learning system working.

2. System Overview

The Smart Money Futures Signal Bot is a cryptocurrency trading signal system that analyzes multiple market indicators across 11 trading symbols and generates real-time BUY/SELL signals sent to Telegram. The system employs a **confluence-based algorithm** requiring alignment across multiple independent indicators before generating signals.

2.1 Trading Symbols

Scalping Strategy (9 symbols): BTCUSDT, ETHUSDT, SOLUSDT, BNBUSDT, LINKUSDT, AVAXUSDT, DOGEUSDT, HYPEUSDT, TRXUSDT, XRPUSDT
Intraday Strategy (3 symbols): YFIUSDT, LUMIAUSDT, ANIMEUSDT

2.2 Signal Frequency

Signals are generated approximately every 5 minutes when market conditions meet the confluence requirements and confidence thresholds. Each signal includes entry price, profit targets, duration (TTL), and market strength indicators.

3. Data Collection Process

The bot continuously collects data from multiple sources every ~5 minutes:

3.1 Primary Data Sources

Data Source	Provider	Collection Method
CVD (Cumulative Volume Delta)	Binance	WebSocket (cvd_service.py)
Open Interest	Coinalyze API	REST API (1-min cache)
Price / VWAP	Binance	Calculated from klines
Liquidations	Binance	WebSocket (liquidation_service.py)
Volume, RSI, EMA	Binance	Calculated from price data
Funding Rate	Binance	REST API (optional)

3.2 Optional Enhancement: 5-Minute Lookback Aggregation

Currently Enabled: Instead of using instant snapshots, the system aggregates the last 5 minutes of data from analysis_log.csv for trend-based decisions. This approach showed 81.4% prediction accuracy in initial testing (vs 14.3% for instant values). Falls back to instant values when less than 2 data points available.

4. Indicator Analysis (9 Technical Indicators)

The bot evaluates 9 technical indicators and creates boolean component signals. Critical thresholds were implemented on Oct 20, 2025 to fix bugs that allowed weak signals to pass.

Indicator	Measures	Component Signals	Threshold
CVD	Buying vs selling pressure	CVD_pos, CVD_neg	Coin-specific (e.g., 20M BTC, 5M ETH)
Open Interest	Futures positioning	OI_up, OI_down	≥0.5% of current OI
VWAP	Mean reversion	Price_below/above_VWAP, VWAP_cross_up/down	Calculated locally
Volume	Trade conviction	Vol_spike, Vol_weak	1.6x median for spike
Liquidations	Forced liquidations	Liq_long, Liq_short	Count comparison
RSI (14-period)	Overbought/oversold	RSI_overbought (>70), RSI_oversold (<30)	Standard levels
EMA (5/20)	Trend direction	EMA_cross_up, EMA_cross_down	Crossover detection
Funding Rate	Long/short bias	Funding_positive, Funding_negative	Direction only
ATR (14-period)	Volatility	Used for target sizing	Calculated for dynamic targets

Critical Bug Fix (Oct 20, 2025): Previously, any positive/negative CVD or OI change triggered component signals. Now CVD must exceed coin-specific thresholds and OI must change by ≥0.5% before components are set. This prevents weak noise from generating false signals.

5. Confluence-Based Signal Generation

Instead of summing all indicator weights, the bot uses **professional confluence logic** that requires strong alignment across multiple independent indicators. This prevents weak signals from passing by requiring structural confirmation.

5.1 Primary Signals (Need 2 of 3 Aligned)

1. CVD - Directional buying/selling pressure
2. Open Interest - New positions opening/closing
3. VWAP - Mean reversion or momentum confirmation

Note: Volume was removed from primary confluence after showing negative -0.48 correlation with wins in backtests. It's still tracked but not required for signal generation.

5.2 Filters (ALL Must Pass)

1. RSI Filter:
- BUY: Accept oversold (RSI < 30) OR neutral (30-70), block overbought
- SELL: Accept any RSI (strict requirement reduced win rate in backtests)

2. EMA Filter: Must not be counter-trend (e.g., no buying into downtrend)

3. Volume Adequacy: No severe weakness (volume must be > 50% of median)

5.3 Directional Blocking (Critical Oct 20 Fix)

This prevents contradictory signals from passing:

BUY signals are BLOCKED if:
- CVD_neg is present (negative buying pressure contradicts BUY)
- OI_down is present (positions closing contradicts new BUY positions)

SELL signals are BLOCKED if:
- CVD_pos is present (positive buying pressure contradicts SELL)
- OI_up is present (positions opening contradicts SELL)

This fix eliminated the majority of losing trades caused by mixed signals.

6. Confidence Calculation (Two-Step Process)

The bot calculates confidence through a sophisticated two-step process that blends formula-based analysis with empirical historical performance.

6.1 Step 1: Formula-Based Confidence

Calculation:

```
score = weighted sum of aligned indicators
confidence_formula = 0.70 + (score - min_score) / (max_score - min_score) × 0.25
```

Range: 70% - 95%
Weights: Coin-specific from config.yaml (e.g., CVD: 0.8, OI: 0.15, VWAP: 1.0)

6.2 Step 2: Empirical Calibration (Self-Learning)

Blends formula confidence with actual historical win rates from effectiveness_log.csv:

Adaptive Weighting Based on Sample Size:

- Low data (<10 samples): 90% formula + 10% empirical
- Medium data (10-30 samples): 70% formula + 30% empirical
- High data (30-50 samples): 50% formula + 50% empirical
- Very high data (50+ samples): 40% formula + 60% empirical

Guardrails:

- No lower floor - allows confidence to drop based on real performance
- Upper clamp at 0.95 - prevents overconfidence

Cached for 5 minutes to reduce computational overhead.

7. Signal Threshold Filtering

Before sending a signal to Telegram, it must pass the minimum confidence threshold:

Current Threshold: 80% for all 11 symbols (raised from 50% on Oct 20, 2025)
Configuration: Per-coin setting in config.yaml as min_score_pct

If calibrated confidence < threshold: Signal = NO_TRADE (suppressed)

8. Target & Duration Calculation

The system uses different strategies for scalping vs intraday symbols:

8.1 Scalping Coins (BTC, ETH, SOL, BNB, LINK, AVAX, DOGE, HYPE, TRX, XRP)

Target Calculation Process:

1. **ATR Baseline** - Calculate 14-period Average True Range as % of price (e.g., 0.53%)
2. **Market Strength Multiplier** - Combines:
   - Volume spike (1.15x-1.3x if volume > 1.3x median)
   - CVD strength (1.2x-1.4x if directional flow strong)
   - OI change (1.08x-1.15x if OI moving significantly)
3. **Magnitude Multiplier** - Predicts HOW FAR price will move based on momentum
4. **Final Target** - ATR_pct × combined_multiplier
   - Min target: 50% of final (e.g., 0.27%)
   - Max target: 100% of final (e.g., 0.53%)

Duration Calculation (Dynamic TTL):

1. **Base Interval** - Derived from ATR volatility (12-45min for scalping)
2. **TTL Multiplier** - Asymmetric adjustment:
   - High volume = shorter TTL (fast moves expected)
   - Aligned OI = longer TTL (structural support)
   - Very strong CVD = shorter TTL
3. **Guardrails** - Floor at 0.75× base, cap at 2.5× base
4. **Final TTL** - base_interval × ttl_multiplier
   - Tier-1 (BTC/ETH): 9-45 minutes typical
   - Mid-cap alts: 34-225 minutes typical

8.2 Intraday Coins (YFI, LUMIA, ANIME)

Strategy: Positional trades, not scalping
Targets: Fixed from config (1.8%-3.0%)
Duration: Dynamic TTL 3-15 hours based on volatility

9. Signal Output & Telegram Formatting

When all checks pass, the bot sends a formatted signal to Telegram via HTML formatting:

Example Signal Format:

```
■ BTCUSDT 15m — BUY
Conf: 85%
■ Signals: CVD+ + OI↑ + VWAP
■ Price: 67,432.00
■ Target: 67,789.25 - 67,895.50 (0.5-0.7%) ■ 12min
■ Market Strength: Strong (1.35x)

CVD: 42,500,000 USDT | OI: 12,450,000,000 (Δ 85,000,000)
VWAP: 67,210.50 | EMA: ■■ 67,380.00/67,190.00
RSI: 48.2 ■■ Neutral
Vol: 1,250,000,000 USDT (med 850,000,000, +47%) ■■
```

Signal Components:

- Verdict (BUY/SELL) with confidence percentage
- Aligned primary indicators for transparency
- Current price with appropriate decimal precision
- Profit targets with dynamic duration (TTL)
- Market strength multiplier showing conditions
- All indicator values (bold if supporting signal)

10. Signal Cancellation & Tracking

The signal_tracker.py service monitors every sent signal in real-time:

Monitoring Process:
• Checks every 30 seconds if current price still supports signal
• Recalculates confidence using current market conditions
• Issues cancellation message (as reply to original) if confidence drops below threshold
• Uses same coin-specific threshold as signal generation (e.g., 80%)

Data Management:
• Active signals stored in active_signals.json with file locking
• Prevents race conditions between multiple processes
• Tracks highest/lowest prices reached during signal lifetime

Result Logging:
• When signal duration expires, result (WIN/LOSS) logged to effectiveness_log.csv
• WIN if target reached before expiration
• LOSS if target not reached or price moved against signal
• Includes full price history for accurate effectiveness measurement

11. Effectiveness Tracking & Self-Learning

Hourly Performance Reports (effectiveness_reporter.py):
• Automated Telegram reports every hour
• Shows win rates across multiple time periods: 1h, 6h, 12h, 24h, 3d, 7d, 14d, 1mo, 3mo, 6mo, 1yr
• Color-coded indicators: ■ ≥60%, ■ 50-60%, ■ <50%
• Only displays periods with actual data
• Example: "■ 24h: 75% (9W-3L)"

Self-Learning Controller (self_learning_controller.py):
• Monitors effectiveness every 6 hours
• Calculates win rates with 95% confidence intervals (Wilson score method)
• Identifies underperforming symbols and verdict imbalances
• Recommends weight optimization when win rate < 80% AND sufficient data exists
• Statistical guardrails prevent overfitting (min 20 samples globally, 5 per symbol+verdict)

Weight Optimizer (weight_optimizer.py):
• Uses logistic regression on matched data
• Merges effectiveness_log.csv with analysis_log.csv (2-minute timestamp window)
• Extracts raw indicator values for model training
• L2 regularization prevents overfitting
• Outputs optimized weights to optimized_weights.json

12. Validation Results (Rigorous Framework)

A rigorous validation framework was implemented to test the algorithm's economic viability using 397 completed signals from Oct 19-21, 2025. The methodology was architect-approved and addresses all statistical concerns including selection bias, lookahead contamination, and realistic execution costs.

12.1 Methodology

■ **No Selection Bias:** Policy chosen from training CV performance only, holdout evaluated exactly once
■ **Walk-Forward Cross-Validation:** 3 folds using TimeSeriesSplit (past→future, no lookahead)
■ **Realistic Execution Costs:** Normal 0.11%, Conservative 0.17%, Stress 0.22%
■ **Statistical Rigor:** Wilson score confidence intervals, minimum sample sizes
■ **Critical Bug Fixes:** Direction-aware target_pct, realized PnL from exit prices

12.2 Data Overview

Dataset	Samples	Period	Win Rate	Avg Gross PnL
Train (70%)	277	Oct 19 05:25 - Oct 20 06:00	19.9%	-0.063%
Holdout (30%)	120	Oct 20 06:09 - Oct 21 04:40	30.0%	-0.032%
Total	397	Oct 19-21, 2025 (2.5 days)	22.9%	-0.050%

12.3 Best Policy Found

Across all three stress scenarios, the optimizer converged to the same policy:

Parameters:
• min_confidence: 0.85
• target_range: 0.0% - 1.5%
• duration_range: 30 - 120 minutes

This policy selected **48 trades** from the 120-sample holdout set (40% selectivity).

12.4 Holdout Performance (Unseen Data)

Scenario	Exec Cost	Trades	Gross WR	Net WR	Avg Net PnL	Total PnL
Normal	0.11%	48	37.5%	29.2%	-0.146%	-7.00%
Conservative	0.17%	48	37.5%	29.2%	-0.201%	-9.65%
Stress	0.22%	48	37.5%	20.8%	-0.256%	-12.29%

Note: Gross WR = Win rate before execution costs, Net WR = Win rate after costs. Wilson 95% CI lower bounds: 18.2%, 18.2%, 11.7% (all far below break-even 50%).

13. Critical Findings & Root Cause Analysis

The validation revealed fundamental issues with the current algorithm:

13.1 Primary Issues

1. ■ Negative Gross PnL (-0.036%)
The algorithm loses money BEFORE accounting for execution costs. This indicates a fundamental lack of edge in signal generation, not just an execution cost problem.

2. ■ Low Win Rate (37.5% gross, 20.8%-29.2% net)
Gross win rate 37.5% is far below the 80% target. After execution costs, net win rate drops to 20.8%-29.2% depending on cost scenario. Gap: 42.5-59.2 percentage points below target.

3. ■ Consistent Losses Across All Scenarios
All three execution cost scenarios show:
• Negative expected value per trade
• Negative total PnL (-7% to -12%)
• Negative Sharpe ratios (-0.37 to -0.62)
• Confidence interval lower bounds far below 50%

4. ■■ Small Sample Size (48 holdout trades)
While variance is present, the direction of effect is consistent across training set, all CV folds, and holdout set - all negative.

13.2 Root Cause Analysis

Why Is the Algorithm Losing Money?

1. Signal Quality Issues
Gross win rate 37.5% suggests poor predictive accuracy. Many signals may be contradictory or based on weak indicators. Confidence calibration may be overestimating signal strength.

2. Target Sizing Problems
Average target 0.391% is modest. Execution costs (0.11%-0.22%) consume 28-56% of target. Even small adverse moves result in losses exceeding targets.

3. Duration Mismatch
Best policy filters for 30-120 minute signals. Market may be moving too fast or too slow for these timeframes. TTL optimization needed based on actual market behavior.

4. Indicator Misalignment
Despite directional blocking fixes, indicators may not be predictive of short-term price movements. Volume showing negative correlation suggests some components hurt rather than help.

14. Recommendations

14.1 Immediate Actions

1. ■ HALT DEPLOYMENT
Current algorithm is economically non-viable and would lose money in live trading. Confidence intervals do not support profitability.

2. ■ COLLECT MORE DATA
Continue running bot in monitoring mode. Collect 7-14 more days of signals (target: 1000+ samples) to reduce variance and improve confidence in findings.

3. ■ PRESERVE CURRENT SYSTEM
Keep all services running (CVD, Liquidation, Signal Tracker, Watchdog). Effectiveness tracking continues to build dataset for future optimization.

14.2 Medium-Term Actions (Redesign Required)

1. Redesign Signal Generation Logic
• **Stricter Confluence:** Require ALL primary indicators aligned (not just 2/3)
• **Enhanced Directional Blocking:** Add more contradiction checks
• **Volume Requirements:** Enforce minimum volume thresholds
• **Regime Filters:** Only trade in favorable market conditions (trending, high volume)

2. Recalibrate Confidence
• Current calibration shows 0.85 confidence produces 37.5% WR
• Implement stronger empirical weighting
• Consider raising confidence threshold to 90%+ or removing weak signals entirely

3. Target Sizing Overhaul
• Current 0.391% average target too small for 0.11%-0.22% costs
• Consider asymmetric targets (wider for uncertain signals)
• Dynamic target scaling based on volatility and confidence

4. TTL Optimization
• 30-120 minute range may not match market behavior
• Test 1-hour, 4-hour, or even longer timeframes
• Previous analysis showed 1-hour durations outperform 15-minute

14.3 Long-Term Actions (Strategy Evolution)

1. Feature Engineering
• Add regime detection (trending vs ranging markets)
• Include funding rate trends when API access restored
• Order book imbalance metrics
• Multi-timeframe confirmation

2. Ensemble Approach
• Separate models for BUY vs SELL signals
• Symbol-specific optimization
• Verdict-specific confidence calibration

3. Ablation Analysis
• Test each indicator individually
• Remove underperforming components
• Identify which signals contribute negative edge

Conclusion

The rigorous validation confirms that **the current signal generation algorithm lacks profitability** even before accounting for execution costs. With a 37.5% gross win rate and -0.036% average gross PnL, the system would lose money in live trading.

Key Takeaway: Focus on signal quality improvements and collect more data before considering deployment. The infrastructure (tracking, monitoring, effectiveness reporting) is solid and should continue operating to build a larger dataset for future optimization.

Validation Status: ■ Architect-Approved, Methodologically Sound
Economic Viability: ■ Not Profitable
Implementation Ready: ■ NO - Redesign Required

This document was automatically generated on October 21, 2025 at 05:46 AM.