Distributed Stock Price Prediction Using Mean Reversion Signals

November 9, 2025

Overview

This project implements a two-phase, high-performance mean reversion trading workflow using Google's serverless analytics stack (BigQuery) and a streaming backbone (Kafka or RabbitMQ). Phase 1 performs large-scale batch screening over historical data; Phase 2 generates low-latency real-time signals for execution.

Objectives

- Identify mean-reverting equities using scalable statistical tests across the full universe.
- Produce real-time trading signals by combining technical indicators with minimal latency.
- Build a reproducible, cloud-native pipeline that is cost-aware and easy to operate.

System Architecture (Two-Phase)

Phase 1: Batch Screening (BigQuery, MPP). A single analytical query executes in parallel across historical OHLCV data using window functions and CTEs. It computes:

- Augmented Dickey–Fuller (ADF) test for stationarity (mean reversion candidacy)
- Variance Ratio (VR) test to assess random-walk deviations
- Hurst exponent to characterize mean-reverting vs. trending behavior

The output is a ranked list of tickers with associated scores and thresholds for downstream selection.

Phase 2: Real-Time Signal Generation (Kafka/RabbitMQ). A streaming pipeline ingests live price updates for the screened tickers. Concurrent workers compute indicators such as MACD, RSI, and ADX in near real-time and combine them into actionable long/flat/short signals suitable for automated execution.

Methodology

1. **Feature Engineering (Batch):** Compute rolling means/variances, z-scores, and cointegration proxies, alongside ADF/VR/Hurst, per ticker and time horizon.
2. **Candidate Selection:** Filter and rank by a composite mean-reversion score with liquidity and spread constraints.

3. **Signal Logic (Streaming):** Fuse MACD, RSI, ADX, and reversion bands (e.g., z-score thresholds) with debouncing to reduce churn; include basic risk guards (max position, max loss).

4. **Execution Interface:** Emit normalized signals (enter/exit, side, confidence) for an external broker/execution simulator.

Data

- **Historical:** Daily/intraday OHLCV stored/queryable in BigQuery for scalable batch analytics.
- **Real-time:** Price ticks or bars streamed via Kafka/RabbitMQ from a market data source (websocket/feed handler).
- **Metadata:** Universe definitions, corporate actions, and trading calendars.

Evaluation

Backtests (out-of-sample) and live paper trading will be evaluated by:

- Risk-adjusted return (Sharpe), max drawdown, hit ratio, turnover, and capacity.
- Transaction-cost and slippage sensitivity.
- End-to-end latency (ingest $\rightarrow$ signal) and system reliability (uptime, backlog).

Timeline (Indicative)

- **Week 1–2:** Data plumbing; BigQuery schemas; batch SQL prototype for ADF/VR/Hurst; initial ranking.
- **Week 3–4:** Streaming pipeline (Kafka/RabbitMQ) and indicator workers (MACD/RSI/ADX); signal fusion logic.
- **Week 5:** Backtesting, transaction-cost modeling, and parameter sweeps.
- **Week 6:** Paper trading, monitoring dashboards, and documentation.

Risks and Mitigations

- **Data Quality/Latency:** Validate feeds; cache fallbacks; guard against look-ahead bias.
- **Overfitting:** Use walk-forward splits and simple, robust rules; hold-out validation.
- **Costs/Slippage:** Include realistic cost models; stress-test at different liquidity tiers.
- **Cloud Cost Control:** Partitioned tables, scheduled queries, and autoscaling limits.

Deliverables

- BigQuery SQL for batch screening and scoring.
- Streaming services for live indicator computation and signal emission.
- Configuration and runbooks (deployment, monitoring, cost controls).
- Final report with performance results and recommendations.