High-Prep 1: Quantitative Trading

Idea Summary

Team 42

Inter IIT Tech Meet 14.0

EBULLIENT
SECURITIES

TRADE WALK
BROKING

Contents

1 Problem Setup and Dataset

1.1 Objective and Constraints

The task is to design an intraday strategy on two securities, EBX and EBY, using only limit-order-book–derived features at 1-second frequency. The key constraints are:

- **Capital:** 100 units inventory worth amount at the start of each day.
- **Transaction Cost:** 2 bps (0.02%) per side on traded notional.
- **Max Exposure:** 100 units of inventories on either side of trade(+100 max to -100 min at a time).
- **Square-off:** All positions must be closed by the end of the session; a 1% penalty on open P&L is applied if this fails.
- **Data:** EBY: 279 days; EBX: 510 days of per-second derived features.

1.2 Feature Families

Each file contains per-second engineered features derived from price and volume. We group them into the following families:

- **PB:** Price-band–based features.
- **BB:** Band / Bollinger-style features.
- **V:** Volume-based features.
- **VB:** Volatility-band features.

For consistency across EBX and EBY, we restrict ourselves to the intersection of feature sets (excluding `Time` and `Price`).

2 Feature Engineering and Selection

2.1 Common Features and Spearman Correlation

We first intersect the EBX and EBY schemas to obtain a set of common features. Using only the first N_{train} days as a training block, we compute the Spearman rank correlation between each feature and the mid-price. This provides a robust measure of monotonic association without leaking test-period information.

Table 1 reports the top-ranked features by absolute correlation with price.

Feature	Corr with Price	Abs Corr
BB5_T1	0.999936	0.999936
BB13_T1	0.999936	0.999936
PB9_T9	0.999917	0.999917
PB9_T12	0.999917	0.999917
PB9_T11	0.999917	0.999917
PB9_T1	0.999917	0.999917
PB9_T10	0.999917	0.999917
PB9_T2	0.999917	0.999917
PB9_T3	0.999917	0.999917
PB9_T4	0.999917	0.999917
PB9_T5	0.999917	0.999917
PB9_T6	0.999917	0.999917
PB9_T7	0.999917	0.999917
PB9_T8	0.999917	0.999917
BB4_T1	0.999902	0.999902

Table 1: Top microstructure features ranked by absolute correlation with price.

Key observation: Many PB/BB-derived features are almost deterministic transforms of price. They are useful for regime description but contribute little independent predictive signal unless combined with more robust microstructure structure.

3 Core Strategy Idea

3.1 High-Level Design

The overall design is:

1. Use per-second features to detect trend strength and breakout regimes (via ADX and range-based signals).
2. Within "favourable" regimes, apply a Ridge regression model to forecast short-horizon returns.
3. Convert standardized return forecasts into bounded positions using a two-threshold mapping with cooldown and cost-aware execution.

3.2 Return Target and Standardized Signal

We define the forward return over horizon H seconds as

$$r^{(H)}t = Pt + H - P_t,$$

where P_t is the mid-price at time t. A feature vector X_t is built from microstructure signals (excluding raw `Price` and `Time`). A `StandardScaler` and Ridge regression model produce an estimated return $\hat{\mu}_t$ and residual volatility $\hat{\sigma}$, giving a standardized signal:

$$S_t = \frac{\hat{\mu}_t}{\hat{\sigma}}.$$

Positions are determined via a two-threshold mapping between z_{entry} and z_{full}, capped at $Q_{\max}$, and updated only if the desired position changes and a cooldown C seconds has elapsed. Transaction costs are modeled as $\gamma|q_t - q_{t-1}|P_t$ plus a small end-of-day exposure penalty.

4 Approach 1: ADX + Breakout Regime Features

4.1 ADX Trend Strength

For a 14-period window ($n = 14$), with mid-price P_t:

$$\text{UpMove}t = P_t - Pt - 1, \quad \text{DownMove}t = Pt - 1 - P_t,$$

$$+DM_t = \begin{cases} \text{UpMove}_t, & \text{if UpMove}_t > \text{DownMove}_t \wedge \text{UpMove}_t > 0, \\ 0, & \text{otherwise}, \end{cases}$$

$$-DM_t = \begin{cases} \text{DownMove}_t, & \text{if DownMove}_t > \text{UpMove}_t \wedge \text{DownMove}_t > 0, \\ 0, & \text{otherwise}, \end{cases}$$

$$TR_t = |P_t - P_{t-1}|, \quad ATR_t = \frac{1}{n}\sum_{i=t-n+1}^{t} TR_i,$$

$$+DI_t = 100 \cdot \frac{\sum_{i=t-n+1}^{t} +DM_i}{ATR_t}, \quad -DI_t = 100 \cdot \frac{\sum_{i=t-n+1}^{t} -DM_i}{ATR_t},$$

$$DX_t = 100 \cdot \frac{|(+DI_t) - (-DI_t)|}{(+DI_t) + (-DI_t)}, \quad ADX_t = \frac{1}{n}\sum_{i=t-n+1}^{t} DX_i.$$

ADX_t serves as a continuous feature for trend strength, rather than a hard rule.

4.2 Breakout and Liquidity Sweep

With lookback $L = 50$:

$$\text{RangeHigh}t = \max i = t - L + 1^t P_i, \quad \text{RangeLow}t = \min i = t - L + 1^t P_i,$$

$$\text{RangeWidth}_t = \text{RangeHigh}_t - \text{RangeLow}_t,$$

$$\text{CompressionRatio}_t = \frac{\text{RangeWidth}_t}{\text{RangeHigh}_t},$$

$$\text{BreakoutUp}t = 1[P_t > \text{RangeHigh}_t], \quad \text{BreakoutDown}t = 1[P_t < \text{RangeLow}_t],$$
$$\text{WickUpStrength}_t = \max(0, \text{RangeHigh}_t - P_t), \quad \text{WickDownStrength}_t = \max(0, P_t - \text{RangeLow}_t),$$
$$\text{SweepUp}_t = \text{BreakoutUp}_t \wedge (\text{WickUpStrength}_t > 0.6 \cdot \text{RangeWidth}_t),$$
$$\text{SweepDown}_t = \text{BreakoutDown}_t \wedge (\text{WickDownStrength}_t > 0.6 \cdot \text{RangeWidth}_t),$$
$$\text{SweepStrength}_t = \frac{\text{WickUpStrength}_t + \text{WickDownStrength}_t}{\text{RangeWidth}_t},$$
$$\text{BreakoutEnergy}t = \text{CompressionRatio}t - 1 \times \text{SweepStrength}_t.$$

These become numeric inputs for downstream models, capturing consolidation, fakeouts, and breakout intensity.

5 Approach 2: Ridge Regression

5.1 Training Matrix Construction

For each training day d, we form a feature matrix $X^{(d)}$ and forward return vector $y^{(d)}$ over horizon H, then stack:

$$X = \begin{bmatrix} X^{(1)} \\ X^{(2)} \\ \vdots \\ X^{(D)} \end{bmatrix}, \quad y = \begin{bmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(D)} \end{bmatrix}.$$

Each column of X is standardized using z-score normalization.

5.2 Ridge Objective and Residual Risk

We fit Ridge regression via:

$$\hat{\beta} = \arg\min_{\beta} \left(\|y - X\beta\|^2 + \alpha\|\beta\|^2 \right),$$

with α selected via cross-validated correlation between y and $\hat{y} = X\hat{\beta}$. Residual volatility is:

$$\sigma = \sqrt{\text{Var}(y - \hat{y})},$$

and the standardized forecast used in trading is:

$$S_t = \frac{\hat{\mu}_t}{\sigma}.$$

6 Backtesting Framework

6.1 Why We Trained on 50 Days of EBX

The provided backtesting environment processed data tick-by-tick via callbacks, requiring repeated feature recomputation and preventing vectorized operations. We therefore developed a custom backtester that processes each day at once, enables vectorized feature engineering, and provides complete transparency over execution logic and rule compliance.

To ensure a clean and leak-proof development cycle, we adopted a transparent approach:

Use the first 50 days of EBX exclusively for model training.

The reasons for this choice were:

- EBX provided significantly more data (510 days), enabling more stable estimation.
- We wanted the training set to be long enough to capture multiple microstructure regimes.
- Restricting training to only 50 days ensured that the model does not “see” any patterns from the test horizon.

After preprocessing these 50 days, we fitted:

`StandardScaler` and `Ridge Regression (1-second horizon)`

and saved them as:

`scaler.pkl` and `ridge.pkl`.

These files act as the "frozen" model used to generate out-of-sample predictions for every test day.

6.2 Motivation for Our Custom Backtester

Although the problem statement provided a reference backtester, we encountered several issues:

- It did not properly reset state between days.
- Tick-by-tick processing with per-tick feature recomputation was computationally expensive.
- Trade execution approximations were not transparent.

Hence, we built a *minimal, robust, deterministic backtester of our own*. The philosophy was:

Load one day at a time, process it fully, never leak information across days.

This made the simulation extremely stable, traceable, and perfectly aligned with the constraints of the challenge.

6.3 How Our Day-by-Day Backtester Works

For each trading day d:

$$q_{d,0} = 0, \qquad E_{d,0} = 100{,}000$$

We load that day's CSV file alone and process it *second-by-second*.

At every second t:

1. The full feature row is extracted.
2. We apply `scaler.pkl` to normalize the vector.
3. The normalized features are fed into `ridge.pkl` to obtain an estimated forward return $\hat{\mu}_{d,t}$.
4. We convert the signal:
$$S_{d,t} = \frac{\hat{\mu}_{d,t}}{\hat{\sigma}}$$
into a trading position:
$$q^{\star}_{d,t} \in [-100, +100].$$
5. If:
$$q^{\star}_{d,t} \neq q_{d,t-1} \quad \text{and cooldown has passed}$$
a trade is executed.

Thus, at every second, the strategy makes a fresh decision using only the data available up to that second.

This ensures:

- No look-ahead bias
- No cross-day leakage
- Deterministic and reproducible behaviour
- Perfect adherence to rules: max exposure, cost, flattening, cooldown

6.4 Transaction Costs and Square-Off

We apply the official rules exactly:

$$\text{TCost}d,t = 0.0002 \times |\Delta qd,t| \times P_{d,t}.$$

At the end of day:

$$q_{d,T} \to 0$$

If the model is unable to close the position (which normally never happens in our custom backtester), a penalty of 1% of the residual notional is applied.

6.5 Daily and Cumulative Metrics

After the close:

$$\text{P\&L}^{net}d = Ed, T + 1 - 100{,}000$$

Daily returns R_d, annualised returns, Sharpe ratio, maximum drawdown, and Calmar ratio are computed using standard definitions.

The entire pipeline — from model training to per-second simulation — is *fully reproducible* simply by loading:

$$\texttt{scaler.pkl}, \quad \texttt{ridge.pkl}, \quad \text{and each daily CSV file.}$$

7 Results and Discussion

7.1 Summary Metrics

Symbol	Annual Return	Max DD	Sharpe	Calmar
EBX	35.51%	-1.07%	4.1969	33.2446
EBY	36.35%	-6.62%	3.2368	5.4771

Table 2: Summary performance metrics for EBX and EBY under the Ridge-based strategy.

7.2 Interpretation

Ratio-based metrics (Sharpe, Calmar) appear acceptable due to low volatility and small drawdowns. However, absolute annualized returns are modest, and simulated performance is highly sensitive to transaction cost assumptions. This is consistent with the underlying challenge: at a 1-second horizon, most price variation is noise, and simple linear models struggle to extract stable edge.

8 Limitations and Future Work

8.1 Main Limitations

- The 1-second forward returns are almost unpredictable, limiting the capacity of the model.
- Many features are quasi-deterministic functions of the price, offering little incremental information.
- Execution is modeled simplistically; real-world slippage and order book dynamics would likely reduce returns.
- Fixed thresholds, cooldowns, and caps are not adaptive to volatility, liquidity, or regime shifts.

8.2 Potential Improvements

- Incorporate richer order-flow signals (e.g., imbalance, queue depth, limit/cancel dynamics).
- Use regime-switching or nonlinear models (e.g., tree-based, shallow neural nets) with strong regularization.
- Introduce cost-aware, adaptive position sizing based on expected edge and local volatility.
- Simulate execution more realistically using order-book snapshots to estimate slippage.