

Explainable Machine Learning for BTCIRT Limit Order Book Dynamics

Oveys Sayad

Email: oveyssayad@gmail.com

GitHub: <https://github.com/OveysSayad>

Repository: <https://github.com/OveysSayad/explainable-btcirt-lob>

Nobitex · BTCIRT · Sparse LOB Snapshots · Multi-Study Redesign (v2.0)

August 6, 2026

Abstract

LaTeX report generated from the project README for `explainable-btcirt-lob`. Research redesign of explainable ML on sparse Nobitex BTCIRT LOB snapshots (Studies A/B/C/D), with leakage controls and SHAP explainability figures.

Contents

1	1. Why this project matters	4
2	2. Executive summary of results	4
2.1	Data	5
2.2	Study A — Next observed mid movement (3-class) — primary	5
2.3	Study B — Next mid-change direction (binary)	5
2.4	Study C — Strict fixed horizons — pilot / underpowered	5
2.5	Incremental value (development_test Macro F1)	6
3	3. Research questions	6
3.1	Study A — Next observed mid-price movement	6
3.2	Study B — Next mid-price <i>change</i> direction	6
3.3	Study C — Strict 10 / 30 / 60 second forecasting	6
3.4	Study D — Dense data-collection design	6
4	4. End-to-end project flow	7
5	5. Repository map & code logic	8
5.1	Logic highlights by module	9

6	6. Dataset & filtering	9
7	7. Data-quality & observation-gap audit	9
7.1	Quality checks	9
7.2	Gap reality (this run)	10
7.3	Visual evidence	10
8	8. Mathematical core	11
8.1	Mid-price, spread, relative spread	11
8.2	Log return in basis points	11
8.3	Order-book imbalance (depth k)	11
8.4	Weighted OBI	12
8.5	Snapshot OFI proxy (not event-level OFI)	12
8.6	Tick size	12
9	9. Label construction	12
9.1	Study A	12
9.2	Study B	13
9.3	Study C	13
10	10. Feature engineering	14
10.1	Families	14
10.2	Mandatory feature sets	14
11	11. Temporal validation & leakage control	14
12	12. Models & hyperparameters	14
12.1	Baselines	14
12.2	Primary: XGBoost	15
12.3	Challenger: CatBoost	15
13	13. Full results (this run)	15
13.1	Study A — model comparison (development_test)	15
13.2	Feature-set comparison	17
13.3	Study B	18
13.4	Study C	18
13.5	Financial sanity (exploratory — not a strategy)	18
14	14. Explainability	19

14.1	How to read this section	19
14.2	Top TreeSHAP features (Study A)	19
14.3	Top feature families (sum of mean $\backslash$ SHAP $\backslash$)	20
14.4	Permutation importance (top)	21
14.5	Beeswarm plots (class-conditional)	21
14.6	Dependence plots	22
14.7	Local waterfalls (illustrative cases)	23
14.8	SHAP interactions	25
15	15. Challenges we hit	25
16	16. What we do <i>not</i> claim	26
17	17. Reproduction	26
18	18. Outputs catalog	27
19	19. Limitations & future work	27
20	20. Research integrity	27
21	Citation	28
22	Disclaimer	28
22.1	Bottom line for readers & recruiters	28

Nobitex · BTCIRT · Sparse LOB Snapshots · Multi-Study Redesign (v2.0)

Is this a good project to share? Yes — especially as a research / methodology portfolio piece. It is strong because it documents *failed designs*, sparse-data honesty, leakage controls, and explainability limits — not because it claims a trading edge. Share it as **explainable microstructure ML under realistic data constraints**, not as a profitable bot.

1 1. Why this project matters

Most “LOB → next price” demos quietly assume:

- dense, regular sampling (e.g. every 100 ms),
- exact fixed horizons (10s / 30s / 60s),
- event-level order-flow imbalance (OFI).

This dataset is not like that.

Nobitex BTCIRT snapshots arrive with a **median gap of ~69 seconds** (development-test median ~189 seconds). Claiming “30-second forecasting” when the next observation is often minutes away is scientifically wrong.

This repository’s contribution is therefore:

Contribution	Why it matters
Multi-study redesign	Separates next-observation, next-change, and strict-horizon questions
Actual-delay tracking	Every label stores the real future delay
Target-timestamp purging	Prevents label leakage across train/val/test
Trade deduplication	Fixes repeated <code>last_trade_*</code> inflation
Explainability triad	SHAP + permutation + ablation (not SHAP alone)
Honest failure logs	Study C underpower & interaction status logged, not hidden

2 2. Executive summary of results

Run: redesign v2.0 · seed 42 · pipeline completed **Environment:** Python 3.14 · pandas 3.0.5 · sklearn 1.9 · XGBoost 3.4 · CatBoost 1.2.10 · SHAP 0.52 · macOS arm64

2.1 Data

Metric	Value
Raw rows (all symbols)	835,000
BTCIRT retained	39,693 (4.75%)
Date range	2026-01-04 → 2026-02-18
Unique dates	44
Median observation gap	69.35 s (test $\approx$ 188.9 s)
Primary features	80 (full_no_trade)

2.2 Study A — Next observed mid movement (3-class) — primary

Model	Development-test Macro F1	Notes
Majority	0.211	Always STABLE
Previous direction	0.377	
OBI rule	0.385	
XGBoost (primary)	0.470	Frozen specification
CatBoost (challenger)	0.484	Competitive; not declared superior without multi-fold proof

Day-level bootstrap (1000 resamples of dates): Macro F1 mean **0.470**, 95% CI [**0.455**, **0.481**].

Hybrid ε (train-only) = **3.664 bps**. Purged sizes: train **29,568** / val **5,630** / development_test **4,492**.

2.3 Study B — Next mid-change direction (binary)

Item	Value
Primary samples (1 per price run)	25,704
Median time to next change	~69.6 s
XGBoost Macro F1 (dev-test)	0.614

2.4 Study C — Strict fixed horizons — pilot / underpowered

Horizon	Window	Eligible	Modeling
10 s	[5, 15]	0	Underpowered pilot
30 s	[20, 40]	0	Underpowered pilot
60 s	[40, 80]	29,318	Eligible overall, but 0 development_test rows after target purge → no robust claim

2.5 Incremental value (development_test Macro F1)

Quantity	Δ Macro F1
LOB vs price-only (full_no_trade - price_only)	+0.046
Corrected trades (full_with_trade - full_no_trade)	-0.001
Time features (full_no_trade - full_no_time)	+0.004

Interpretation: LOB state adds measurable lift over returns/volatility alone. Corrected trade intensity does **not** help. Time features add a tiny lift and must be audited for collection artifacts.

3 3. Research questions

3.1 Study A — Next observed mid-price movement

Can the current and recent LOB state predict whether the **next observed** mid-price moves **DOWN** / **STABLE** / **UP**?

Uses consecutive snapshots. Stores `actual_delay_seconds`. Does **not** claim an exact clock horizon.

3.2 Study B — Next mid-price *change* direction

Conditional on the mid eventually changing, can the book predict **UP** vs **DOWN**?

Skips unchanged mids; primary sample = one observation per flat price-run.

3.3 Study C — Strict 10 / 30 / 60 second forecasting

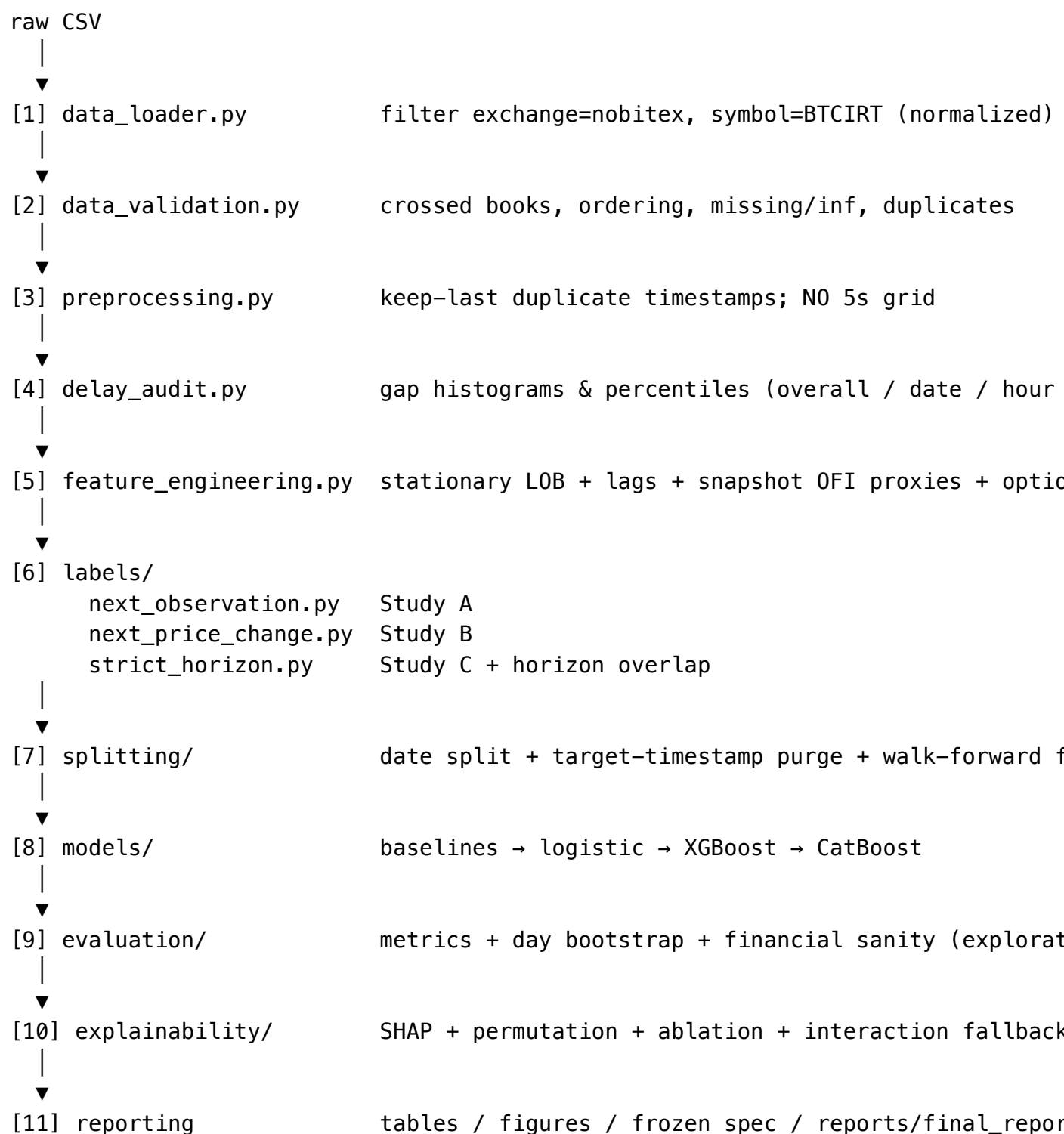
On samples whose future observation delay truly falls in a narrow window, is fixed-horizon prediction feasible?

Defaults: 10 $\square$ [5,15], 30 $\square$ [20,40], 60 $\square$ [40,80]. Under sparse sampling this is mostly a **pilot**.

3.4 Study D — Dense data-collection design

Documentation only (`docs/DENSE_DATA_COLLECTION_DESIGN.md`). No fabricated WebSocket data.

4 4. End-to-end project flow



One command for the full path:

```

./scripts/run_pipeline.sh
# equivalent:
python -m src.pipeline --config configs/project_config.yaml

```

5 5. Repository map & code logic

```

explainable-btcirt-lob/
├── configs/                YAML: project + per-study overrides
├── src/
│   ├── pipeline.py         Orchestrator (Studies A/B/C)
│   ├── data_loader.py      Chunked CSV load + BTCIRT filter
│   ├── data_validation.py   Book integrity audits
│   ├── preprocessing.py    Cleaning; refuses dense grid when gaps are large
│   ├── delay_audit.py       Observation-gap statistics
│   ├── feature_engineering.py
│   ├── trade_deduplication.py
│   ├── labels/             Study A / B / C label builders
│   ├── splitting/          Chronological + purge + walk-forward
│   ├── models/             Baselines, logistic, XGB, CatBoost, calibration, tw
│   ├── evaluation/         Metrics, bootstrap, financial sanity
│   ├── explainability/     SHAP, grouped SHAP, permutation, ablation, interact
│   ├── visualization.py    Figures
│   └── reporting.py        Report helpers (run summaries → reports/)
├── docs/                  Design, formulas, validation, limitations
├── tests/                 46 unit tests (units, labels, purge, no-leakage)
├── scripts/               run_pipeline / study_a|b|c / tests
├── reports/
│   ├── archive/pre_research_redesign/  frozen v1 outputs
│   ├── tables/               CSV results
│   ├── metrics/             JSON metrics
│   ├── figures/             PNG figures
│   ├── models/              frozen_model_specification.json
│   └── final_report.md
└── notebooks/             Thin wrappers that import from src/

```

5.1 Logic highlights by module

Module	Core logic
<code>data_loader</code>	Normalize <code>exchange.lower()</code> , <code>symbol.upper()</code> ; keep only <code>nobitex + BTCIRT</code> ; raise if empty
<code>preprocessing</code>	Sort time; keep last duplicate timestamp; skip 5s alignment when median gap $\square$ 5s
<code>labels/next_observation</code>	<code>target = t+1</code> ; return in bps; hybrid ε from train only
<code>labels/next_price_change</code>	Forward scan to first mid $\neq$ current; optional one-sample-per-run
<code>labels/strict_horizon</code>	Match only inside <code>[lower, upper]</code> ; minimize ‘
<code>splitting/purged_split</code>	Drop rows whose <code>target_timestamp</code> crosses the next split boundary
<code>trade_deduplication</code>	Signature <code>(price, qty) → is_new_trade</code> ; repeated snapshots count as 0
<code>feature_engineering</code>	Depth / OBI / WOBI / microprice / snapshot OFI proxy / 120–1200s windows / event lags
<code>models/xgboost_model</code>	Randomized search on validation Macro F1 ; never on <code>development_test</code>
<code>explainability/*</code>	TreeSHAP; on interaction failure $\rightarrow$ dependence fallbacks + JSON status

Notebooks **must not** re-implement formulas; they import `src/`.

6 6. Dataset & filtering

- **File:** `data/raw/market_data_clean_nobitex.csv` (**not committed** — place locally)
- **Exchange / symbol:** `nobitex / BTCIRT`
- **LOB:** 8 ask + 8 bid price/qty levels
- **Trades:** `last_trade_price`, `last_trade_qty` (often repeated across snapshots)
- **JSON columns** (`data`, `asks`, `bids`, `last_trade`): unused when flattened columns exist

```
df["exchange"] = df["exchange"].astype(str).str.strip().str.lower()
df["symbol"] = df["symbol"].astype(str).str.strip().str.upper()
df = df[(df["exchange"] == "nobitex") & (df["symbol"] == "BTCIRT")]
```

7 7. Data-quality & observation-gap audit

7.1 Quality checks

Missing / infinite values · zero/negative prices & quantities · crossed books ($a1 \leq b1$) · locked books · invalid ask/bid monotonicity · abnormal spreads · timestamp parse failures · exact & timestamp dupli-

cates.

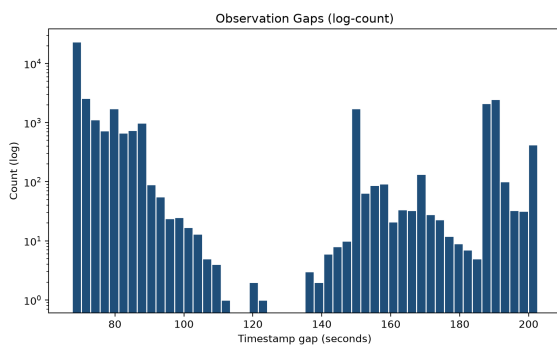
7.2 Gap reality (this run)

Segment	Median gap (s)	% gaps > 60s	% gaps > 120s
Overall	69.35	100%	18.9%
Train	68.88	100%	1.6%
Validation	71.10	100%	45.3%
Development test	188.88	100%	100%

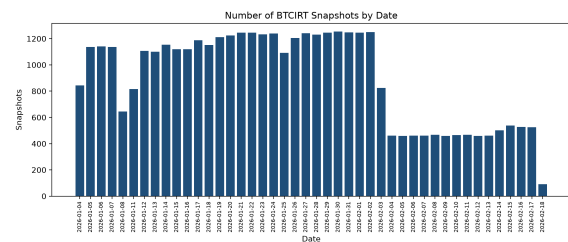
Consequence: every gap exceeds 10s and 30s → Study C 10s/30s eligibility = **0**. A 60s window (40–80s) matches many train/val snapshots whose next print is ~70s, but the late-period test gaps (~3 minutes) fall **outside** the 80s upper bound after purging.

Tables: reports/tables/observation_gap_summary.csv, observation_gap_by_date.csv, observation_gap_by_hour.csv.

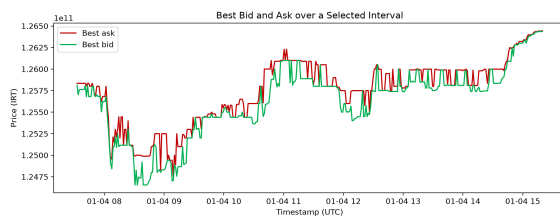
7.3 Visual evidence



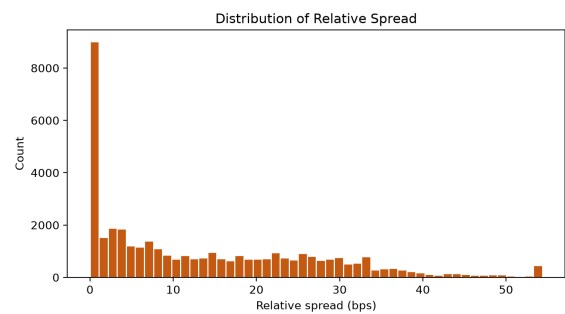
Observation gaps (log scale)



Snapshots by date



Best bid / ask window



Relative spread distribution

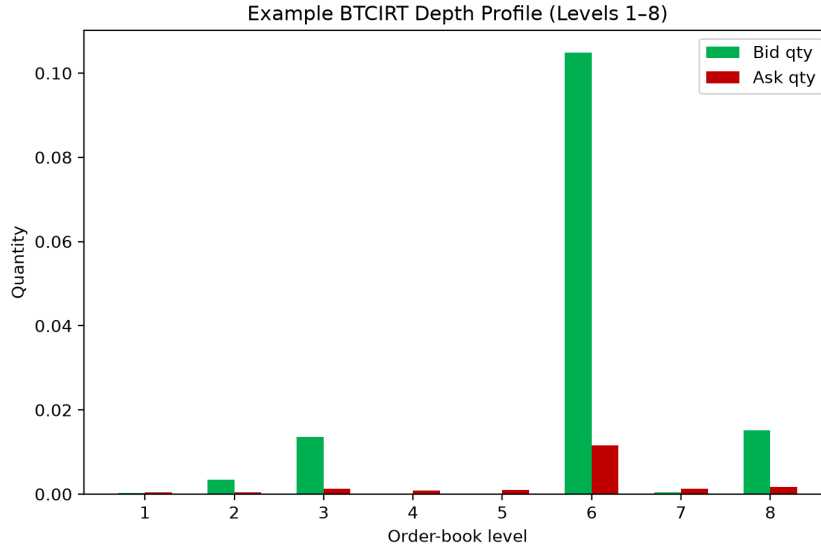


Figure 3: LOB depth profile example

Example depth profile — liquidity is concentrated near the touch.

8 Mathematical core

Formulas use GitHub-flavored LaTeX (\$...\$). Full catalog: [docs/FORMULAS.md](#).

8.1 Mid-price, spread, relative spread

$$m_t = \frac{a_{1,t} + b_{1,t}}{2}, \quad S_t = a_{1,t} - b_{1,t}, \quad \text{RelSpreadBps}_t = 10^4 \frac{S_t}{m_t}$$

8.2 Log return in basis points

Unit-tested so $10^4 \log(100.01/100) \approx 1$:

$$r = 10^4 \log \left(\frac{m_{\text{future}}}{m_{\text{current}}} \right)$$

8.3 Order-book imbalance (depth k)

$$D_{k,t}^a = \sum_{i=1}^k q_{i,t}^a, \quad D_{k,t}^b = \sum_{i=1}^k q_{i,t}^b$$

$$\text{OBI}_{k,t} = \frac{D_{k,t}^b - D_{k,t}^a}{D_{k,t}^b + D_{k,t}^a + \delta}$$

where δ is a tiny constant for numerical stability (e.g. 10^{-12}).

8.4 Weighted OBI

Weights $w_i = e^{-\lambda(i-1)}$ with default $\lambda = 0.5$:

$$\text{WOBI}_t = \frac{\sum_i w_i q_{i,t}^b - \sum_i w_i q_{i,t}^a}{\sum_i w_i q_{i,t}^b + \sum_i w_i q_{i,t}^a + \delta}$$

8.5 Snapshot OFI proxy (not event-level OFI)

$$e_t^b = \mathbf{1}_{\{b_t \geq b_{t-1}\}} q_t^b - \mathbf{1}_{\{b_t \leq b_{t-1}\}} q_{t-1}^b$$

(and similarly for the ask side with opposite inequalities)

$$\text{OFIProxy}_t = e_t^b - e_t^a$$

8.6 Tick size

Empirical mode of positive quote diffs = **10 IRT**. In bps vs huge BTCIRT mids this is $\sim 10^{-6}$ bps, so hybrid ε relies on return quantiles / spreads, not ticks.

9. Label construction

9.1 Study A

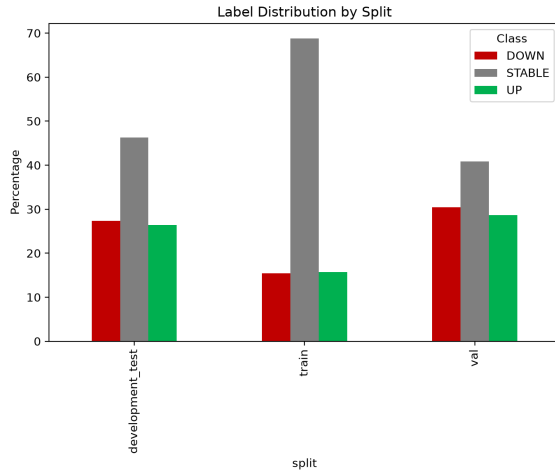
Next snapshot return $\rightarrow$ classes with train-only hybrid ε :

$$\varepsilon = \max(\varepsilon_Q, \varepsilon_{\text{tick}}, 0.5 \times \text{MedianSpreadBps}_{\text{train}})$$

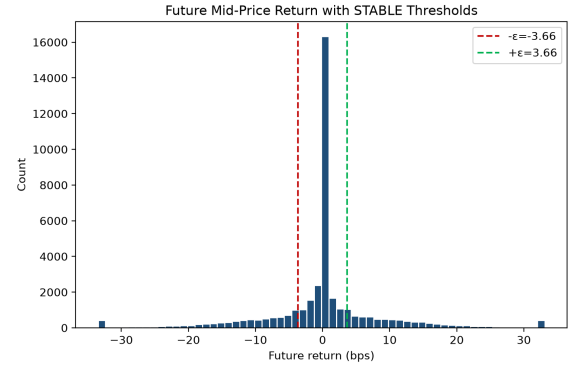
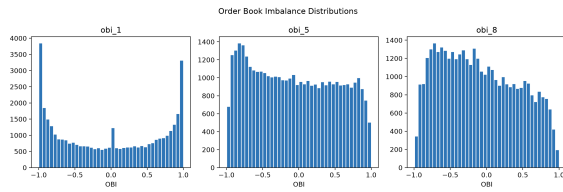
$$y_t = \begin{cases} \text{DOWN} & \text{if } r_t < -\varepsilon \\ \text{STABLE} & \text{if } |r_t| \leq \varepsilon \\ \text{UP} & \text{if } r_t > \varepsilon \end{cases}$$

This run: $\varepsilon = 3.664 \text{ bps} \cdot \text{method} = \text{hybrid} \cdot \text{median delay} \approx 69.35 \text{ s}$.

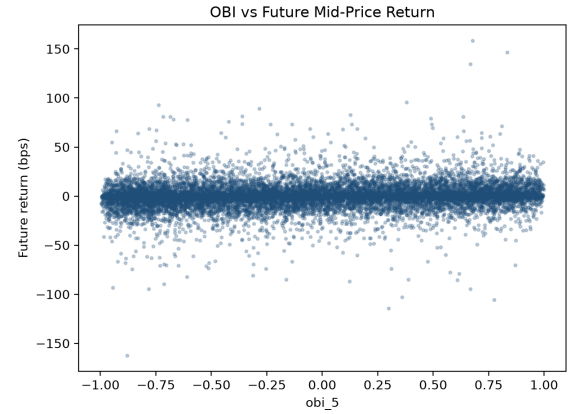
Class mix (development_test): DOWN **27.3%** · STABLE **46.3%** · UP **26.4%**.



Study A class mix by split

Next-observation returns vs ε 

OBI distributions



OBI vs next return

9.2 Study B

First future mid $\neq$ current $\rightarrow$ UP / DOWN. Primary sample = last snapshot of each unchanged price run (avoids flooding one future event).

9.3 Study C

Keep a future snapshot s only when the delay is inside the strict window:

$$\text{Lower}_h \leq t_s - t_t \leq \text{Upper}_h$$

Defaults: 10s $\rightarrow$ [5, 15], 30s $\rightarrow$ [20, 40], 60s $\rightarrow$ [40, 80]. Among matches, minimize $|t_s - t_t - h|$. Store `actual_delay_seconds` and `horizon_error_seconds`.

10 10. Feature engineering

10.1 Families

Price history · Static liquidity / depth / imbalance · Dynamic liquidity / imbalance · Snapshot OFI proxy · Volatility · Trade activity (corrected) · Time · Data-collection metadata (`observation_gap_seconds`, **excluded** from primary).

10.2 Mandatory feature sets

Set	Contents
<code>price_only</code>	Returns + volatility
<code>static_lob /</code> <code>dynamic_lob / lob_full</code>	Book structure & dynamics
<code>full_no_trade</code>	Primary model
<code>full_with_trade</code>	+ corrected trades
<code>full_no_time</code>	Drop cyclical hour/dow

Dictionary: `reports/tables/feature_dictionary.csv` · `docs/FEATURE_DICTIONARY.md`.

11 11. Temporal validation & leakage control

1. Split by **complete calendar dates** (60% / 20% / 20%).
2. Latest block = **development_test** — already inspected in prior analysis → **not a pristine holdout**.
3. **Target-timestamp purging**: a training row is kept only if `target_timestamp < validation_start` (same idea for val vs test).
4. Scalers, ϵ , class weights, and hyperparameter search fit on **training / validation only**.
5. Frozen spec written to `reports/models/frozen_model_specification.json`.
6. **Final independent future holdout (≥ 7 –14 new days) is still pending**.

Tests: `tests/test_purging.py`, `tests/test_no_leakage.py`.

12 12. Models & hyperparameters

12.1 Baselines

Majority · stratified random · previous mid-direction · OBI-5 threshold rule (tuned on train).

12.2 Primary: XGBoost

- Study A: `multi:softprob` (3-class)
- Study B: `binary:logistic`
- Search objective: **validation Macro F1**
- Early stopping on validation
- CPU `tree_method=hist`

Best Study A params (this run):

```
max_depth=7, learning_rate=0.02, n_estimators=1500,
min_child_weight=3, subsample=0.8, colsample_bytree=0.8,
gamma=0, reg_alpha=1, reg_lambda=5, max_delta_step=5,
random_state=42
```

Best Study B params:

```
max_depth=6, learning_rate=0.02, n_estimators=500,
min_child_weight=5, subsample=0.8, colsample_bytree=0.5,
gamma=1, reg_alpha=0.1, reg_lambda=20, max_delta_step=5
```

12.3 Challenger: CatBoost

Study A best (this run): `depth=7, learning_rate=0.1, iterations=750, l2_leaf_reg=10, border_count=64, subsample=1.0` (Bernoulli bootstrap; `bagging_temperature` removed — incompatible with Bernoulli).

Details: [docs/MODEL_SPECIFICATION.md](#).

13. Full results (this run)

13.1 Study A — model comparison (development_test)

Model	Macro F1	Balanced Acc	Log Loss	MCC
Majority	0.211	0.333	—	0.000
Previous direction	0.377	0.377	—	0.083
OBI rule	0.385	0.384	—	0.070
XGBoost	0.470	0.484	1.029	0.218
CatBoost	0.484	0.500	1.021	0.239

XGBoost still used as the **frozen primary** for SHAP / bootstrap continuity; CatBoost is a strong challenger.

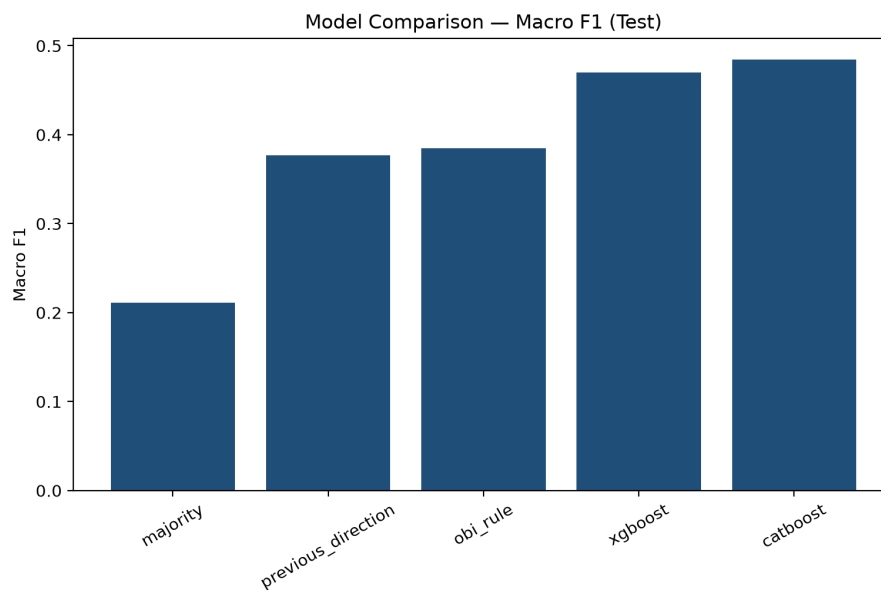
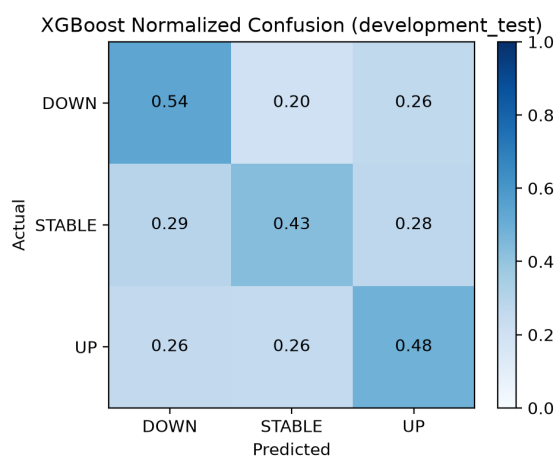
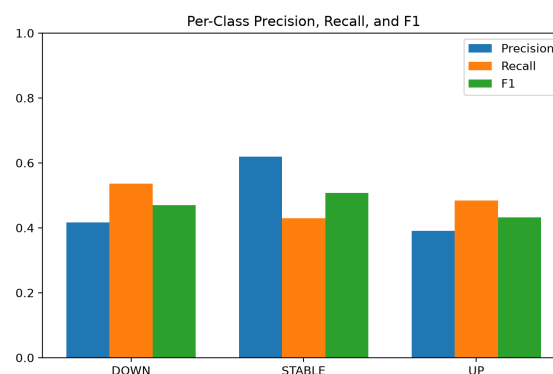


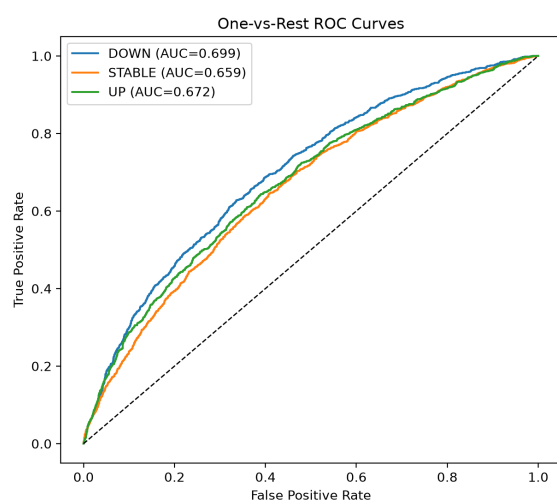
Figure 6: Model comparison Macro F1



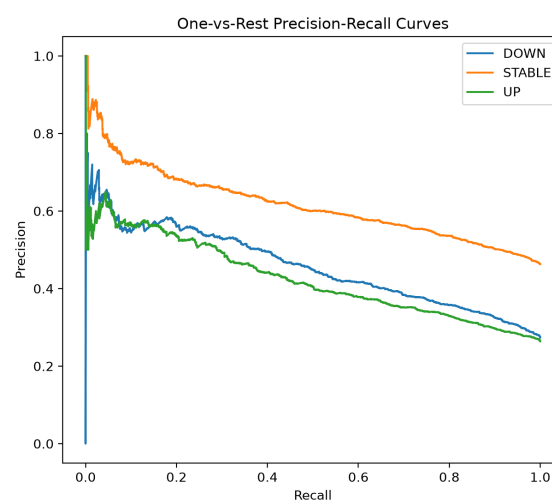
Normalized confusion (XGBoost)



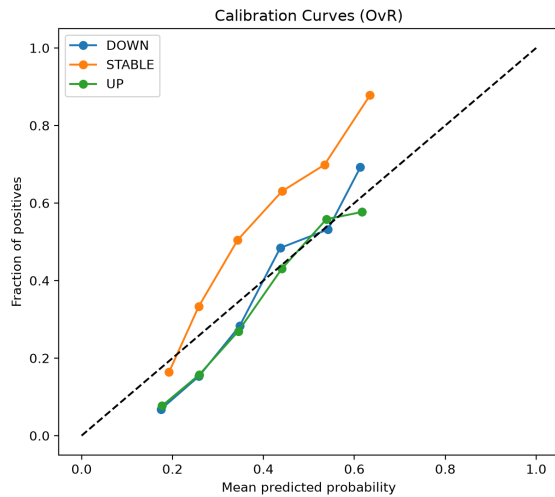
Per-class precision / recall / F1



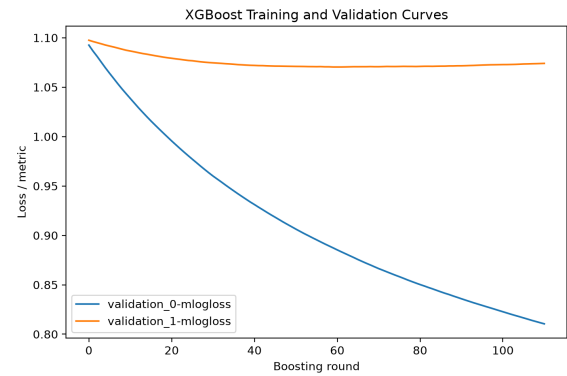
ROC (one-vs-rest)



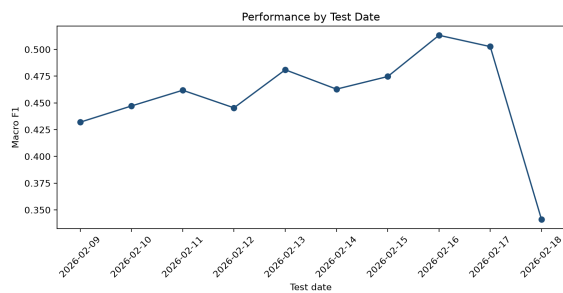
Precision-recall (one-vs-rest)



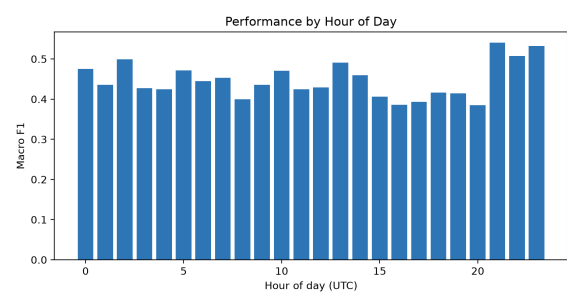
Calibration



Train / validation loss



Performance by date



Performance by hour

13.2 Feature-set comparison

Experiment	#feats	Val Macro F1	Test Macro F1
price_only	12	0.367	0.417
static_lob	34	0.376	0.404
dynamic_lob	30	0.349	0.420
lob_full	64	0.374	0.456
full_no_trade	80	0.384	0.464
full_with_trade	89	0.397	0.463
full_no_time	76	0.379	0.459

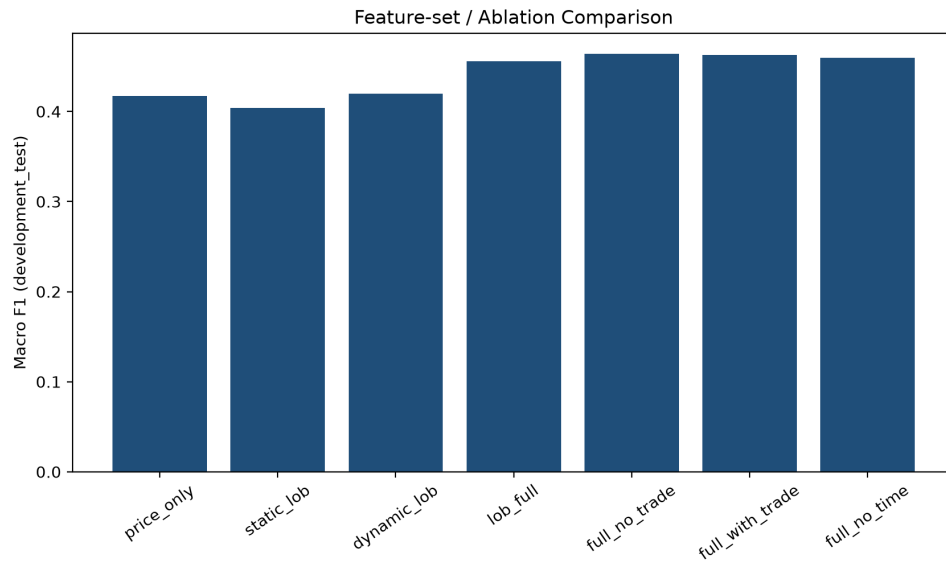


Figure 11: Feature set comparison

LOB features lift Macro F1 over price-only; corrected trades add essentially nothing.

13.3 Study B

Binary next-change direction Macro F1 $\approx$ **0.614** on development_test (val $\approx$ 0.616). Easier task (no STABLE class; nearly balanced UP/DOWN).

13.4 Study C

Honest outcome: **10s/30s impossible** under observed gaps; **60s** has many eligible rows historically concentrated before the sparse test regime, so purged development_test is empty $\rightarrow$ pilot descriptive stats only.

13.5 Financial sanity (exploratory — not a strategy)

Long-only if predict UP (next mid return as proxy):

Scenario	Fees+slip (bps)	Mean net (bps)
Zero cost	0	+0.92
Low cost	10	-2.35
Moderate cost	25	-7.25

Sparse snapshots **do not** support a realistic HFT execution backtest. Prediction $\neq$ profitability.

14 14. Explainability

We require **agreement** across methods before calling a feature “strongly supported”. SHAP answers *how the trained model uses features for a prediction*, **not** whether a feature causes the mid-price to move in the market.

14.1 How to read this section

Plot / table category	What it answers	What it does <i>not</i> say
Global importance bar	Which features move predictions the most <i>on average</i>	Direction of effect; causality
Feature families	Which microstructure <i>families</i> dominate attributions	That every member of the family is useful
Permutation importance	Which features hurt Macro F1 when shuffled on held-out data	Local story for one snapshot
Beeswarm (per class)	Full distribution of attributions for DOWN / STABLE / UP	A single “typical” row
Dependence	How SHAP for feature X changes as X varies (often colored by a second feature)	Pure partial effect free of collinearity
Local waterfalls	Step-by-step attribution for one illustrative prediction	Population-level ranking
Interactions	Whether features co-contribute in TreeSHAP (or dependence fallbacks)	Causal interaction in the market

14.2 Top TreeSHAP features (Study A)

What this table / bar chart says: mean absolute SHAP over the development-test SHAP sample. Larger mean $|\text{SHAP}| \Rightarrow$ the feature tends to push the model’s class scores more strongly across snapshots. Rankings are class-averaged for the 3-class model.

Rank	Feature	Mean \	SHAP\	Family
1	bid_distance_30bps	0.0595	Static imbalance	
2	ask_distance_20bps	0.0495	Static imbalance	
3	ask_distance_30bps	0.0445	Static imbalance	
4	volatility_120s	0.0316	Volatility	
5	volatility_300s	0.0304	Volatility	
6	bid_distance_20bps	0.0295	Static imbalance	
7	spread_std_300s	0.0203	Dynamic liquidity	
8	log_ask_depth_1	0.0191	Static depth	
9	obi_1	0.0179	Static imbalance	
10	hour_cos	0.0163	Time	

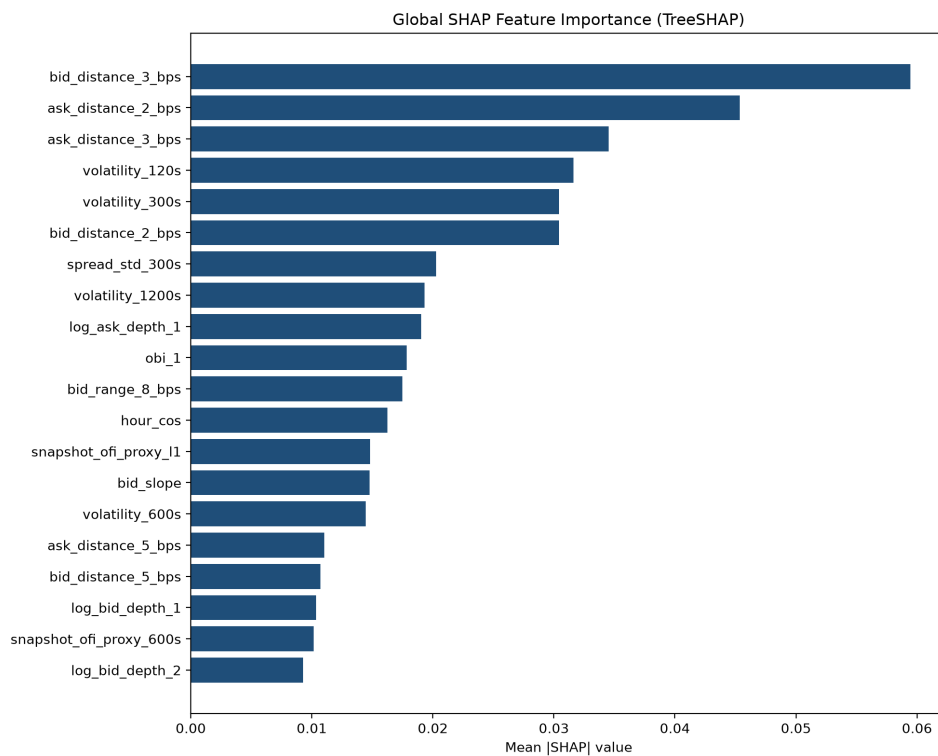


Figure 12: Global SHAP importance

Global bar plot — average magnitude of attribution; does not encode sign (UP vs DOWN).

14.3 Top feature families (sum of mean $\backslash|SHAP|\backslash$)

What this says: we sum mean $\backslash|SHAP|\backslash$ within each engineered family (static imbalance, volatility, depth, ...). It answers *which family of LOB ideas the model leans on*, not which single column is “best.”

1. Static imbalance

2. Volatility
3. Static depth
4. Snapshot OFI proxy
5. Static liquidity

14.4 Permutation importance (top)

What this says: features are randomly shuffled within the development-test set (respecting day blocks where configured); the drop in Macro F1 measures *predictive usefulness under the evaluation metric*. Agreement with TreeSHAP is stronger evidence than either method alone.

1. bid_distance_3_bps
2. ask_distance_2_bps
3. relative_spread_bps
4. ask_distance_3_bps
5. hour_cos

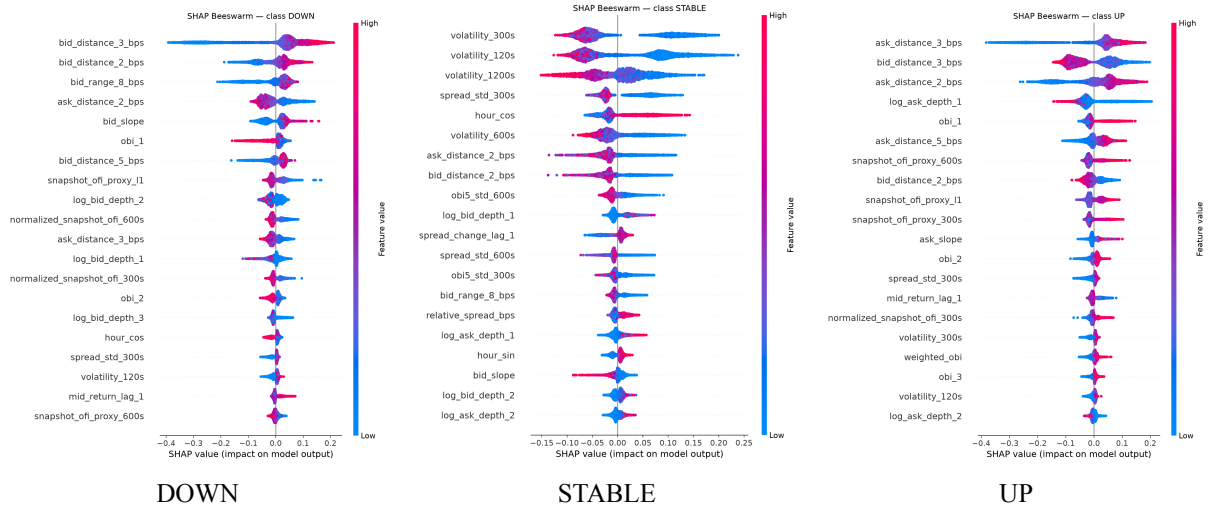
Distance / imbalance features appear in **both** SHAP and permutation — stronger evidence than `hour_cos` alone.

14.5 Beeswarm plots (class-conditional)

What beeswarm plots say: for each class (DOWN / STABLE / UP), every point is one snapshot.

- **Vertical axis:** features ranked by mean $|\text{SHAP}|$ for that class
- **Horizontal axis:** SHAP value (left = pushes *against* that class; right = pushes *for* it)
- **Color:** raw feature value (typically blue = low, red = high)

Use them to see *patterns*: e.g. high imbalance features clustering on the positive side for UP, or volatility spreading attributions for STABLE. Because Study A is multiclass, we plot **one beeswarm per class** so attributions are not mixed across DOWN/STABLE/UP.

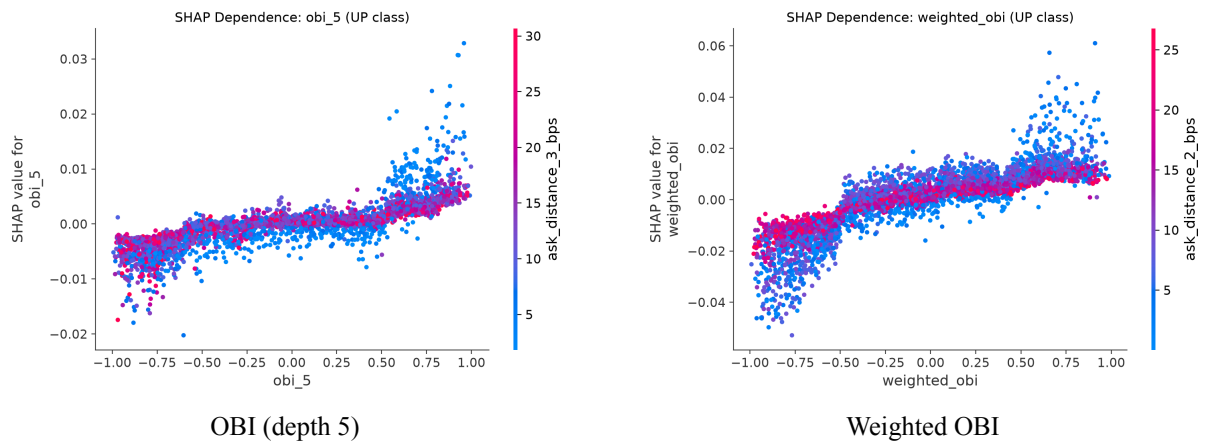


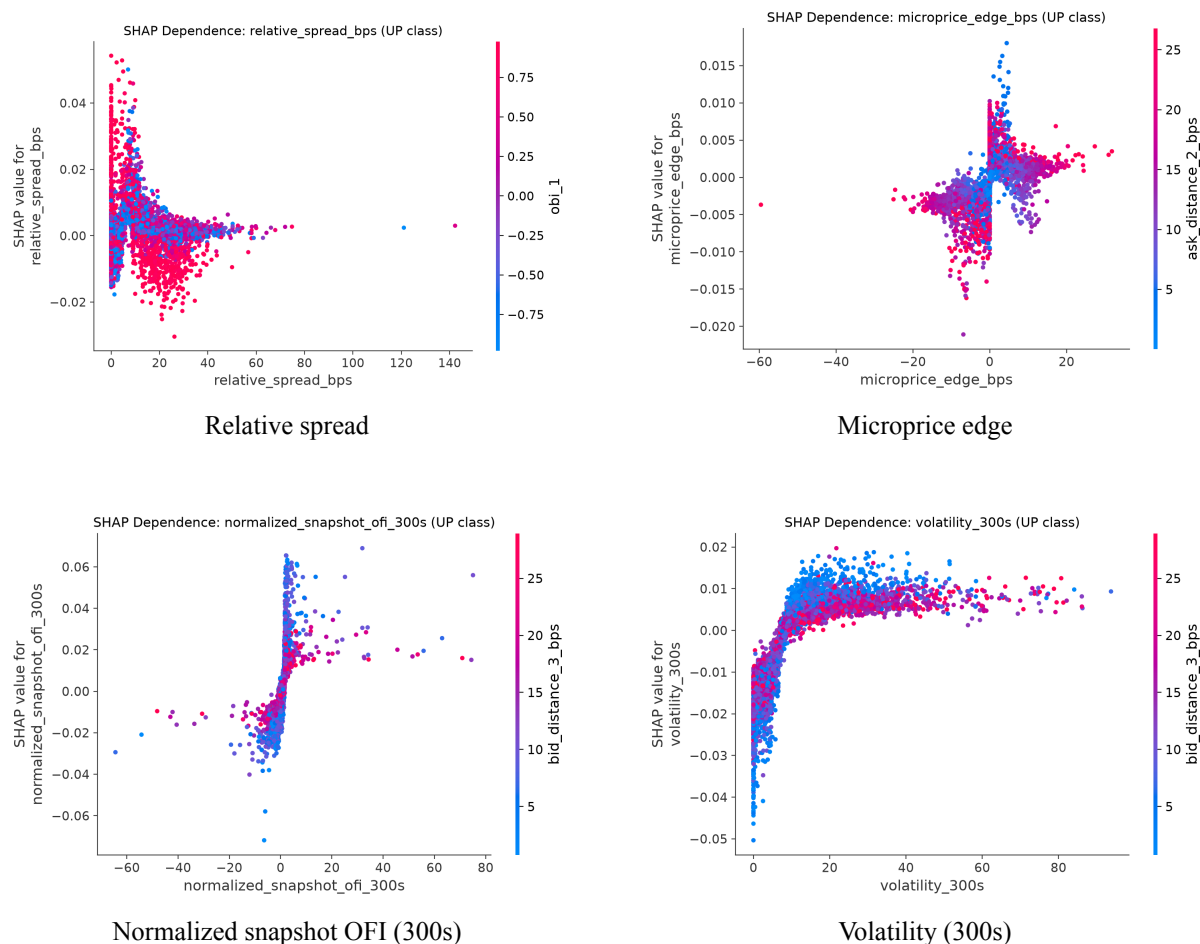
Class-conditional beeswarms — distribution of attributions, not a single summary number.

14.6 Dependence plots

What dependence plots say: for a chosen feature X , each point is a snapshot plotted as $(X, SHAP_X)$ for the **UP** class contribution (when multiclass). Color usually marks an automatically chosen interacting feature. They reveal *how* the model's attribution changes as X moves (nonlinear / threshold behavior) and hint at pairwise co-dependence.

They do **not** isolate a causal partial effect: correlated LOB features can share credit.





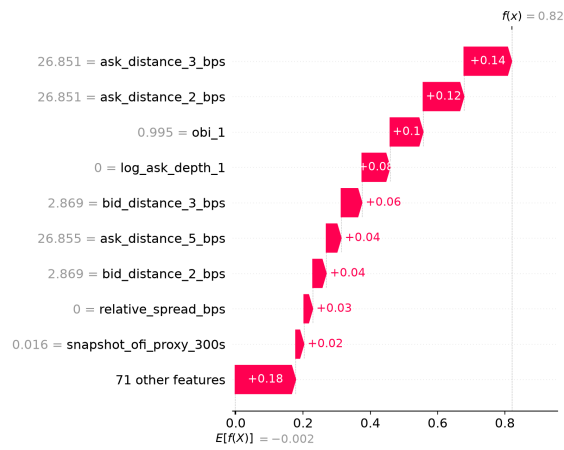
Dependence plots — functional shape of attribution vs feature value (UP-class SHAP).

14.7 Local waterfalls (illustrative cases)

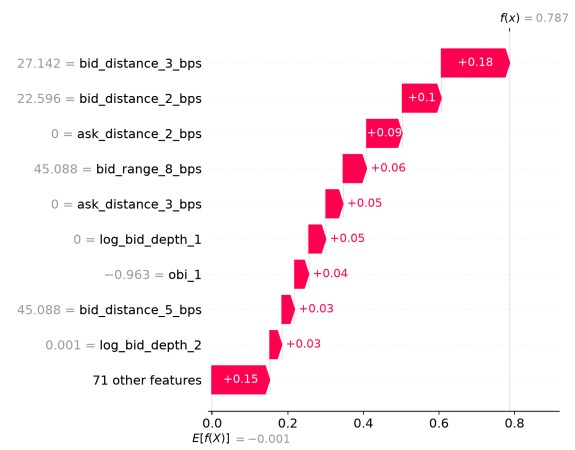
What waterfall plots say: for **one** snapshot, starting from the model’s expected score (base value), each bar shows how a feature pushes the score up or down until the final prediction for the highlighted class. They are case studies, not rankings.

We pick five illustrative development-test cases:

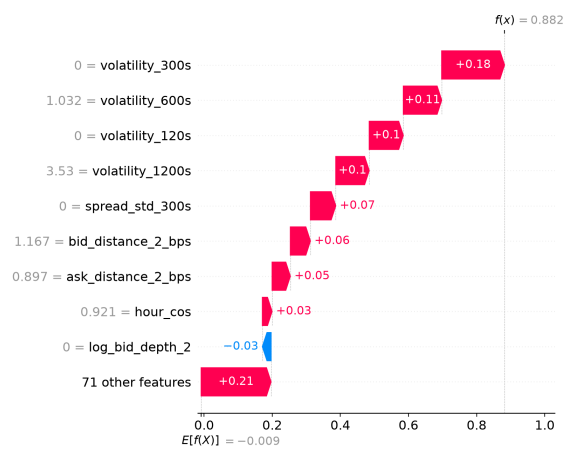
Case	What it is meant to show
Correct high-conf UP	Confident correct UP — which LOB features drove the UP score
Correct high-conf DOWN	Same for DOWN
Correct high-conf STABLE	Same for STABLE (often spread / low-move cues)
Incorrect high-conf	Confident but wrong — where the model “trusted” misleading features
Low-conf borderline	Near-tie probabilities — small opposing attributions



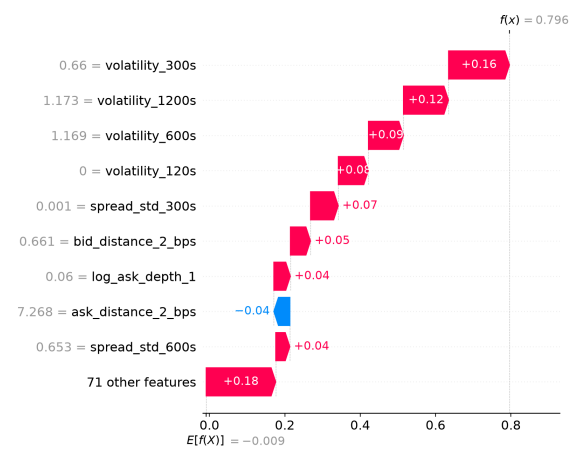
Correct high-conf UP



Correct high-conf DOWN



Correct high-conf STABLE



Incorrect high-conf

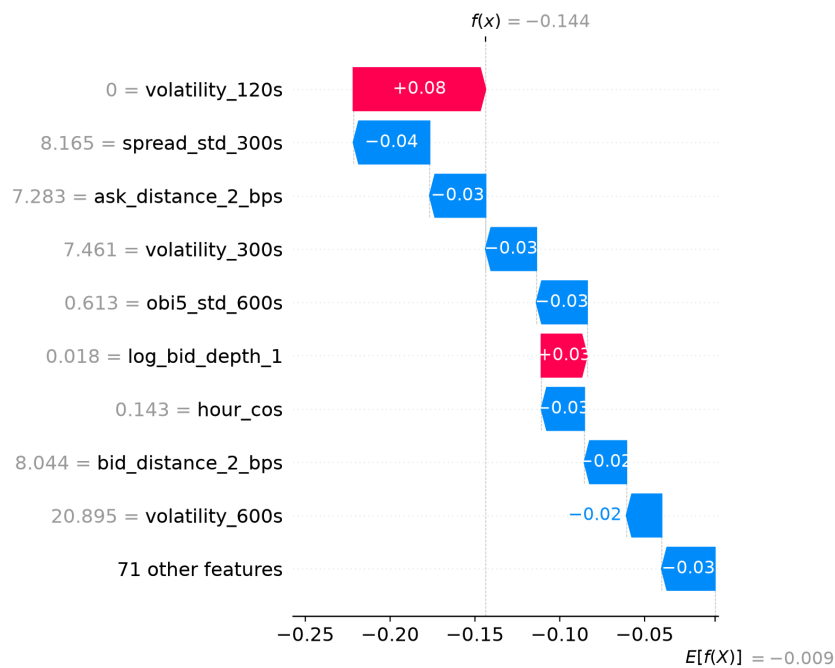


Figure 19: Waterfall borderline

Local waterfalls — *per-snapshot accounting of the model's score; not causal market effects.*

14.8 SHAP interactions

What this category says: TreeSHAP can estimate pairwise *interaction values* (extra attribution when two features act together beyond main effects). We reduce the multiclass tensor to a 2-D mean-absolute interaction matrix when possible and save `reports/tables/shap_interactions_native.csv`. If that fails, we log the failure and fall back to colored dependence plots under `reports/figures/shap/` — **never** fabricated interaction numbers.

Latest run status: see `reports/metrics/shap_interaction_status.json`.

Regenerate without full retuning: `python scripts/regenerate_shap_plots.py`

15 15. Challenges we hit

This section is intentional — sharing failures is what makes the project credible.

Challenge	What happened	Resolution
Misleading fixed horizons	10/30/60s often mapped to the <i>same</i> next snapshot	Redesign into Studies A/B/C; store actual delays
Sparse gaps	Median ~69s; test ~189s; 100% of gaps > 60s	No 5s forward-fill; longer feature windows (120–1200s)
Study C empty horizons	0 eligible rows for 10s & 30s	Label as underpowered pilot; do not invent scores
Study C 60s purge	Eligible overall, but <code>development_test</code> left with 0 after target purge	Descriptive-only; no robust model claim
Repeated last trades	Same <code>last_trade_*</code> across many snapshots	<code>trade_deduplication.py</code> ; exclude from primary
Tiny tick in bps	Tick=10 IRT → ~1e-6 bps vs huge mid	Hybrid ε uses quantiles/spreads
CatBoost API	<code>bagging_temperature</code> invalid with Bernoulli bootstrap	Dropped temperature; keep Bernoulli + subsample
SHAP interactions	Multiclass tensor needs careful 2-D reduction	Reduce over samples/classes or use dependence fallbacks
macOS XGBoost	Needs <code>libomp</code>	<code>.libs/libomp.dylib</code> + <code>DYLD_LIBRARY_PATH</code> in scripts
Development-test contamination	Prior analysis already looked at late dates	Renamed; require future holdout
Class imbalance	STABLE dominates under hybrid ε	Macro F1 primary; balanced sample weights

Archived v1 metrics live in `reports/archive/pre_research_redesign/` and must **not** be mixed with v2 claims.

16 16. What we do *not* claim

- True high-frequency forecasting from these sparse snapshots
- Exact 30-second prediction when the next print is minutes away
- Causal effects from SHAP
- Trading profitability / production alpha
- True event-level OFI
- Generalization to all crypto markets
- That CatBoost “wins” without multi-fold statistical comparison
- That the `development_test` is a final holdout

17 17. Reproduction

```
git clone git@github.com:OveysSayad/explainable-btcirt-lob.git
cd explainable-btcirt-lob
```

```
python -m venv .venv
source .venv/bin/activate
pip install -r requirements.txt
pip install -e .
```

```
# Place the raw CSV:
#   data/raw/market_data_clean_nobitex.csv
```

```
# macOS XGBoost OpenMP:
#   brew install libomp
#   mkdir -p .libs && ln -sf "$(brew --prefix libomp)/lib/libomp.dylib" .libs/libomp.dylib
export DYLD_LIBRARY_PATH="$(pwd)/.libs:${DYLD_LIBRARY_PATH}"
```

```
./scripts/run_pipeline.sh
pytest -q
```

Makefile targets: `make test`, `make all`, `make study-a`, `make study-b`, `make study-c`.

Smoke (fewer trials): `python -m src.pipeline --config configs/smoke_config.yaml`.

18. Outputs catalog

Path	Contents
reports/final_report.md	Paper-style write-up
reports/tables/*.csv	Metrics, gaps, SHAP, ablations, predictions metadata
reports/ARTIFACT_MANIFEST.md	Tampered vs archive artifact map
reports/metrics/*.json	Environment, bootstrap, study metrics, interaction status
reports/figures/	Data quality, labels, models, SHAP, ablation (embedded in this README)
reports/models/frozen_model_specification.json	Model specification
models/study_a/, models/study_b/	Trained model artifacts

Figure layout:

```
reports/figures/
├── data_quality/      # gaps, mid, spread, depth, coverage
├── labels/           # class mix, returns vs ε, OBI
├── models/           # comparison, confusion, ROC/PR, calibration
├── shap/             # beeswarm, dependence, waterfalls
├── ablation/         # feature-set comparison
└── robustness/       # reserved

| reports/archive/pre_research_redesign/ | Pre-redesign freeze |
| docs/ | Design, formulas, validation, dense-collection plan |
```

19. Limitations & future work

See [docs/LIMITATIONS.md](#).

Highest-priority next steps

1. Collect **dense** WebSocket LOB + trades (docs/DENSE_DATA_COLLECTION_DESIGN.md)
2. Evaluate the **frozen** model on $\geq 7-14$ **new days** never used in analysis
3. Complete nested walk-forward *scores* (fold Macro F1 distribution), not only fold calendars
4. Stronger time-feature artifact tests (gap/hour confounds)

20. Research integrity

- No fabricated metrics, figures, API responses, or SHAP interactions

- Failed / underpowered components remain documented
- Prior results archived, not silently overwritten without copy
- Primary metric Macro F1 chosen for class imbalance — not to hide weak DOWN/UP performance
- **Final independent holdout evaluation pending**

21 Citation

```
@software{explainable_btcirt_lob,  
  title = {Explainable ML for BTCIRT Limit Order Book Dynamics},  
  author = {Sayad, Oveys},  
  year = {2026},  
  url = {https://github.com/OveysSayad/explainable-btcirt-lob},  
  note = {Multi-study redesign under sparse Nobitex snapshots}  
}
```

22 Disclaimer

Research and education only. **Not investment advice.** Cryptocurrency trading involves substantial risk of loss.

22.1 Bottom line for readers & recruiters

This repo demonstrates **research engineering**: questioning a bad horizon design, quantifying sparsity, preventing leakage, comparing models fairly, and explaining predictions without overselling. That is rarer — and more valuable — than another notebook that reports a high F1 on shuffled LOB rows.