Leakage-Safe Tree Ensembles for Equity Signal Modeling on OHLCV Data

Yiming Zhang
Department of Mathematics
University of Michigan
Ann Arbor, MI
yimingzh@umich.edu

Qijun Zhang
Department of Mathematics
University of Michigan
Ann Arbor, MI
qijunzh@umich.edu

Felix Widmann
Department of Mathematics
University of Michigan
Ann Arbor, MI
felixwdn@umich.edu

***Abstract*—We present a leakage-safe ML pipeline for U.S. equities using tree ensembles (Random Forest and LightGBM) on WRDS daily OHLCV data. Features are built causally; labels are H-day forward returns; validation uses purged and embargoed time-series CV. Backtests include realistic costs and turnover limits. We report statistical (IC/ICIR, AUC, out-of-sample R-square) and economic metrics (Sharpe, drawdown). Interpretability relies on permutation importance and SHAP.**

***Index Terms*—Random Forest, LightGBM, time-series cross-validation, transaction costs, backtesting**

I. Introduction

A. Literature Review

Predicting cross-sectional stock returns is a core problem in quantitative equity research; given the course prohibition on deep learning, we focus on tree ensembles that capture nonlinearities and interactions while remaining relatively interpretable and robust. On large WRDS/CRSP panels, machine-learning forecasts can improve risk-premium estimation and portfolio performance over linear benchmarks by exploiting interaction effects [1]. Within this family, Random Forest aggregates decorrelated CART trees via bagging and feature subsampling, offering variance reduction, out-of-bag diagnostics, and permutation importance useful for finance pipelines [2]; LightGBM provides a high-capacity boosted alternative with histogram binning, leaf-wise growth, and GOSS/EFB that typically delivers strong accuracy and speed on wide, sparse tabular data [3]. Because features and labels overlap in time, i.i.d. K-fold CV is invalid; instead, purged and embargoed time-series cross-validation is now standard to prevent leakage from overlapping horizons [4], and broader ML work highlights leakage as a key reproducibility risk, underscoring strict backtesting protocols [5].

II. Method

A. Dataset Overview

We use daily OHLCV, and return data from 2013–2023 obtained via Wharton Research Data Service (WRDS) https://wrds-www.wharton.upenn.edu (CRSP US Stock Database: https://www.crsp.org). The sample contains 3459 common stocks, and the full panel has size 3459 tickers $\times$ 2425 trading days $\times$ 45 features.

B. Feature Engineering

The preprocessing features we are using are listed below: 1

Algorithm 1 OHLC Feature Pipeline (compact)

```
Require: Aligned daily $O, H, L, C$ (and Volume; Amount if available) per ticker; window set $\mathcal{W}$; label horizon $H^*$
1: Preprocess: handle splits; align calendar; causal rolls; drop non-trading days
2: for each date $t$ do
3:     Trend/level: $\mathrm{MA}_w(C)$, $\mathrm{EMA}_w(C)$ for $w \in \mathcal{W}$; slopes $\Delta\mathrm{MA}_w$
4:     Momentum: rolling returns $r_{1,5,10,20}$; MACD (DIF, DEA, MACD) and cross flag
5:     Bands/oscillators: Bollinger(20), %B; RSI; Stochastic/$\mathrm{RSV}_{10/20}$; $\mathrm{Aroon}_{10/20}$
6:     Ranges/gaps: $(H-L)/C$, ATR; gaps O−C, C−C; candlestick shadow ratios
7:     Returns (point-in-day): C2C $r_t$, CO $r_t^{co}$, OC $r_t^{oc}$
8:     Volume set: Volume, ADV (rolling mean), Volume z-score/percentile, $\Delta$Volume
9:     Turnover: use turnover (or adjusted turnover if policy requires caps)
10:    VWAP: $\mathrm{VWAP}_t = \frac{\mathrm{Amount}_t}{\mathrm{Volume}_t}$ (guard $\mathrm{Volume}_t > 0$)
11:    Price–Volume Divergence: compute $\rho = \mathrm{corr}\big(\mathrm{VWAP}, \mathrm{Volume}\big)$ over a lookback; feature $= -\rho$
12: end for
13: Cross-sectional winsorize (e.g., 1%/99%), z-score; drop warm-up NaNs; align with labels at $t+H^*$
14: Return feature matrix $X_t$ (OHLC+Returns+Volume), targets $y_{t\to t+H^*}$ =0
```

C. Random Forest

We include Random Forest as a nonparametric baseline on the engineered OHLC+volume features. Training follows the leakage-safe schedule (purged & embargoed time-series CV) and the tuning plan described elsewhere; algorithmic steps are summarized in Alg. 2. Final model selection and evaluation use the project's standard out-of-fold metrics.

D. LightGBM

LightGBM (histogram GBDT with leaf-wise growth) serves as the boosted counterpart on the same feature set. It is trained under the same validation/tuning protocol with early stopping; the procedure is outlined in Alg. 3. Model choice and reporting adhere to the same out-of-fold evaluation used across the project.

Algorithm 2 Random Forest (compact)

```
Require: Data $\{(x_i, y_i)\}_{i=1}^N$, number of trees $T$, max depth $D$, min leaf size $n_{\min}$, feature subsample size $m_{\text{try}}$ =0
 1: for $t = 1$ to $T$ do
 2:     Draw a bootstrap sample $\mathcal{D}_t$ of size $N$ from the training set
 3:     Grow a CART tree on $\mathcal{D}_t$:
 4:     Initialize node set with root containing $\mathcal{D}_t$
 5:     while nodes remain do
 6:         Pop a node with data $S$; if $|S| < n_{\min}$ or depth $= D$, mark as leaf and set prediction
 7:         Randomly select $m_{\text{try}}$ features
 8:         For each candidate split $(j, \tau)$ on those features, compute impurity decrease
                $$\Delta I = I(S) - \frac{|S_L|}{|S|} I(S_L) - \frac{|S_R|}{|S|} I(S_R)$$
            where $I(\cdot)$ is Gini/entropy (classification) or MSE (regression)
 9:         Choose $(j^*, \tau^*)$ with maximal $\Delta I$; if $\Delta I \leq 0$, mark as leaf; else split $S \to (S_L, S_R)$ and push children
10:     end while
11:     Store tree $h_t(x)$
12: end for
13: Aggregate predictions: for classification, majority vote $\hat{y} = \text{mode}\{h_t(x)\}$; for regression, average $\hat{y} = \frac{1}{T} \sum_t h_t(x)$
14: Return ensemble predictor =0
```

Algorithm 3 LightGBM (Histogram GBDT, compact)

```
Require: Data $\{(x_i, y_i)\}_{i=1}^N$, loss $L(y, F)$, trees $M$, learning rate $\eta$, max depth $D$, min leaf size $n_{\min}$, regularization $(\lambda_1, \lambda_2)$; options: GOSS, EFB, feature_fraction, bagging_fraction
 1: (Optional) EFB: bundle mutually exclusive sparse features
 2: Initialize prediction $F^{(0)}(x) = \arg\min_\theta \sum_i L(y_i, \theta)$
 3: for $m = 1$ to $M$ do
 4:     Compute first/second order stats at $F^{(m-1)}$: $g_i = \partial L/\partial F|_i$, $h_i = \partial^2 L/\partial F^2|_i$
 5:     (Optional) GOSS: keep top-$a$ fraction by $|g_i|$, sample remaining with rate $b$ and reweight
 6:     Initialize tree with a single root leaf containing current sample set
 7:     for level $= 1$ to $D$ do
 8:         For each candidate leaf $\ell$, build histograms for features (on binned values); for each split $s$, compute gain:
                $$\text{Gain}(s) = \frac{G_L^2}{H_L + \lambda_2} + \frac{G_R^2}{H_R + \lambda_2} - \frac{(G_L + G_R)^2}{H_L + H_R + \lambda_2} - \lambda_1$$
            where $(G_\cdot, H_\cdot)$ sum $g_i, h_i$ in child nodes
 9:         Select the leaf/split with max positive gain subject to $n_{\min}$ and feature_fraction; split that leaf (leaf-wise growth)
10:         Stop if no positive gain or depth/leaf constraints hit
11:     end for
12:     For each terminal leaf $\ell$, set leaf value $v_\ell = -\frac{\sum_{i \in \ell} g_i}{\sum_{i \in \ell} h_i + \lambda_2}$
13:     Update predictions: $F^{(m)}(x) = F^{(m-1)}(x) + \eta \cdot \text{Tree}_m(x)$
14: end for
15: Return $F^{(M)}(x)$ =0
```

E. Backtesting Metric

Our prior factor backtest used a simple, reproducible decile-sorting protocol on daily close prices with precomputed factor scores: on each rebalance date t (the score index), we aligned tickers with non-missing scores and realized close-to-close returns $r_{t \to t^+} = \frac{C_{t^+}}{C_t} - 1$, ranked by score (descending), formed 10 equal-count deciles, and computed equal-weighted decile returns over $[t, t^+]$; per-period decile series were stacked and compounded to NAVs, and a long–short spread $r^{LS} = r_{\text{D1}} - r_{\text{D10}}$ was tracked. Evaluation reported cross-sectional $\text{IC}(t) = \rho(\text{score}_t, r_{t \to t^+})$ with ICIR (mean/SD across t) and the long–short information ratio $\text{IR} = \overline{r^{LS}}/\sigma(r^{LS})$. This reference implementation did *not* include transaction costs, constraints, or turnover controls.

Algorithm 4 Prior Decile Backtest

```
Require: Factor score panel $\{\text{score}_t(i)\}$, close-price panel $C_t(i)$
 1: Set rebalance dates $\mathcal{T}$ to score index; init decile-return matrices
 2: for each $t \in \mathcal{T}$ except last do
 3:     $t^+ \leftarrow$ next rebalance date; $r_{t \to t^+}(i) = \frac{C_{t^+}(i)}{C_t(i)} - 1$
 4:     Intersect tickers with non-missing scores and returns
 5:     Rank by score (desc); assign deciles $g \in \{1, \ldots, 10\}$ (equal count)
 6:     $r_g(t) = \text{mean}\{r_{t \to t^+}(i) : i \in g\}$                    (equal weight)
 7:     $\text{IC}(t) = \rho(\text{score}_t, r_{t \to t^+})$
 8: end for
 9: Compound decile series to NAVs; $r^{LS}(t) = r_1(t) - r_{10}(t)$
10: Report $\overline{\text{IC}}$, ICIR, $\text{IR} = \frac{\overline{r^{LS}}}{\sigma(r^{LS})}$ and decile NAVs =0
```

III. Pros and Cons

Random Forest is a stable, low-tuning baseline with useful diagnostics and good robustness to noise; drawbacks include possible underfitting of subtle interactions, higher memory with many trees, and potentially higher turnover without constraints. LightGBM is fast and strong on large tabular data, handles missing values well, and offers rich regularization and SHAP-friendly explanations; its trade-offs are greater hyperparameter sensitivity and a higher risk of overfitting without sufficiently large leaves. We omit XGBoost for simplicity and compute efficiency—LightGBM typically matches its accuracy on this workload with faster training, so including both adds redundancy without clear benefit.

IV. Project Deliverables

We will provide a runnable codebase (data prep, feature engineering, model training with leakage-safe CV, and cost-aware backtesting), a short data/feature glossary, and concise result summaries. Evaluation will be strictly out of sample with transaction costs; ideally, we observe positive and stable cross-sectional signal quality (IC above zero with a reasonable ICIR) and either ROC–AUC above random for classification or out-of-sample R^2 above zero for regression. Portfolio summaries will, ideally, show cost-aware Sharpe above a modest baseline with controlled drawdown and turnover, plus brief sensitivity checks.

References

[1] S. Gu, B. Kelly, and D. Xiu, "Empirical Asset Pricing via Machine Learning," *The Review of Financial Studies*, vol. 33, no. 5, pp. 2223–2273, 2020.

[2] L. Breiman, "Random Forests," *Machine Learning*, vol. 45, pp. 5–32, 2001.

[3] G. Ke, Q. Meng, T. Finley, et al., "LightGBM: A Highly Efficient Gradient Boosting Decision Tree," in *Proc. NeurIPS*, 2017, pp. 3146–3154.

[4] M. López de Prado, *Advances in Financial Machine Learning*. Wiley, 2018.

[5] S. Kapoor and A. Narayanan, "Leakage and the Reproducibility Crisis in ML-based Science," *arXiv:2207.07048*, v3, 2023.

[6] R. Arnott, C. R. Harvey, and H. Markowitz, "A Backtesting Protocol in the Era of Machine Learning," Research Affiliates/SSRN Working Paper, 2019.

[7] S. M. Lundberg and S.-I. Lee, "A Unified Approach to Interpreting Model Predictions," in *Proc. NeurIPS*, 2017, pp. 4765–4774.