Factor-ML

1 Overview

Factor-ML uses XGBoost-predicted expected returns to form a long-short equal-weighted portfolio. Expected returns are estimated using the XGBoost model from Chen and Guestrin (2016), following the methodology of Jensen et al. (2025). Stocks are sorted into deciles based on predicted returns, and the portfolio goes long the top decile and short the bottom decile with equal weights within each side.

2 Expected Return Estimation

We estimate expected returns using the XGBoost model (Chen and Guestrin 2016) to predict realized excess returns over the next month as a function of security characteristics today. Security characteristics are percentile-ranked cross-sectionally (between 0 and 1) and then we subtract 0.5 to center it around zero. Missing values are imputed with zeros.

2.1 Training Setup

We use a rolling window of 10 years of training data. For each test month, the training data is split into 5 temporal folds (i.e., 5 consecutive blocks of months in chronological order) for cross-validation. The XGBoost base score (the initial prediction before any trees are added) is set to 0.

The frequency of hyperparameter tuning is controlled by the `test_period_length` parameter, which specifies the number of consecutive test months per chunk. With `test_period_length = 12` (the default), hyperparameters are re-tuned once per year: the model is trained on data up to the first month in the chunk and then used to predict all months in that chunk. Setting `test_period_length = 1` re-tunes every month.

2.2 Hyperparameter Tuning

Our approach to selecting hyperparameters follows a two-stage procedure.

Stage 1. We train 20 different models on the training data based on 20 sets of hyperparameters and a fixed learning rate of 0.15, with a maximum of 1,000 iterations and early stopping after 25 rounds without improvement. For each hyperparameter set, we evaluate the mean squared error across the 5 temporal folds and select the set that leads to the lowest average mean squared error.

Stage 2. We train a new model on the training data using the parameters found in the first stage, except that we now train the model with a learning rate of 0.01 and a maximum of 10,000 iterations (with the same early stopping rule of 25 rounds). We then record the optimal number of iterations for the model (i.e., the number of individual decision trees to include in the ensemble before the model starts to overfit), averaged across the 5 folds.

Finally, we train a model on the full training data (i.e., all data in the rolling window) using a learning rate of 0.01, the non-learning rate parameters from stage 1, and the optimal number of iterations from stage 2, and use this model to estimate expected returns, $\hat{\mu}_t$.

2.3 Hyperparameter Grid

The 20 hyperparameter sets are generated using the `grid_space_filling` function from the `dials` package (https://dials.tidymodels.org/) with the type set to "max_entropy" to cover the parameter space. The tuned hyperparameters and their ranges are:

- **Features** (`colsample_bytree`): fraction of features sampled for each tree, range $[1/K, 1]$ where K is the number of features
- **Tree depth** (`max_depth`): maximum depth of each tree, range $[1, 7]$
- **Sample size** (`subsample`): fraction of observations sampled for each tree, range $[0.2, 1]$
- **Penalty** (`reg_lambda`): L2 ridge penalty, range $[10^{-2}, 10^{2}]$ on a $\log_{10}$ scale

All parameters are drawn from their natural scales, except for penalty, which is drawn from a logarithmic (base 10) scale. Two additional parameters are held fixed: minimum child weight (`min_child_weight`) is set to 1, and minimum split loss (`gamma`) is set to 0.

3 Portfolio Construction

The Factor-ML portfolio uses the expected return estimates $\hat{\mu}_t$ from the XGBoost model and buys the 10% of stocks with the highest expected returns while shorting the 10% of stocks with the lowest expected returns. Stocks are equal-weighted within the long and short sides of the portfolio.

References

Chen, Tianqi, and Carlos Guestrin. 2016. "XGBoost: A Scalable Tree Boosting System." In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 785–94. ACM.

Jensen, Theis Ingerslev, Bryan T Kelly, Semyon Malamud, and Lasse Heje Pedersen. 2025. "Machine Learning and the Implementable Efficient Frontier." *Review of Financial Studies (Forthcoming)*.