Bachelor Thesis in Finance at SSE

Erik Jin and Michael Brusis

October 2025

1 Introduction

1.1 Background

It has long been understood that you should not put all your eggs in one basket. This metaphor extends naturally to investing: rather than allocating all your capital to a single stock, you can reduce risk by diversifying across multiple assets. This idea forms the foundation of **Modern Portfolio Theory (MPT)**, introduced by economist *Harry Markowitz* in his seminal paper *"Portfolio Selection"* [2].

The objective of MPT is to maximize expected return for a given level of risk, or equivalently, minimize risk for a given expected return. By plotting expected return against risk (measured by the standard deviation of portfolio returns) for all possible portfolios, one obtains the **efficient frontier**, which is the set of portfolios offering the optimal trade-off between risk and return.

When a **risk-free asset** is introduced, the line that is tangent to the efficient frontier defines the **Capital Market Line (CML)**. The point of tangency represents the **optimal risky portfolio** (the tangency portfolio), which under the assumptions of the **Capital Asset Pricing Model (CAPM)** corresponds to the **market portfolio**. Portfolios along the CML dominate all others, offering the highest expected return for any given level of risk, they have the highest Sharpe-ratio which is the risk-adjusted return.

While Modern Portfolio Theory provides a theoretical framework for constructing efficient portfolios, its practical implementation requires accurate estimates of expected returns, variances, and covariances. These parameters are difficult to measure.

To address these challenges, empirical asset pricing research has developed alternative, data-driven methods for portfolio formation. One of the most influential approaches is the construction of characteristic-sorted portfolios, where stocks are grouped based on observable firm characteristics such as size, value, or momentum. Characteristic-sorted portfolios are designed to capture how expected returns vary with firm characteristics. By ranking stocks based on an observable variable such as size, book-to-market ratio, or momentum, and grouping them into quantiles (for example, deciles or quintiles), researchers can examine how average returns differ across these groups. The difference in returns between high- and low-characteristic portfolios reflects the return premium associated with that characteristic (e.g., the value premium or size effect). In the current finance literature characteristic-sorted portfolios are used as benchmarks for the performance of portfolios. This is because they are simple and transparent. This makes them

- easy to interpret and replicate
- empirically grounded as it has been historically shown that systemic return patterns have been linked to well-known factors like size, value, or momentum.
- meaningful as they capture real risk factors

- stable and robust since the characteristics of stocks are usually stable over time

In recent years, machine learning techniques have been increasingly applied to portfolio optimization. Unlike traditional methods, which often rely on linear assumptions and parametric estimates, machine learning algorithms can capture complex, non-linear relationships between asset returns and risk factors. These models are particularly well-suited for high-dimensional datasets, allowing the integration of multiple sources of information, including firm fundamentals, market indicators, and alternative data. By learning patterns directly from historical data, machine learning approaches can potentially improve the accuracy of expected return and risk estimates, adapt more effectively to changing market conditions, and enhance risk management through better identification of extreme events. Additionally, once trained, these models can automate portfolio re-balancing and scale to large asset universes, providing both flexibility and operational efficiency in modern investment strategies. One notable approach is the Panel Tree (P-Tree) algorithm, which extends traditional tree-based methods to panel data structures [1]. In this context, portfolio optimization begins with the entire asset universe at a root node. The algorithm then systematically splits assets into child nodes based on firm characteristics (such as earnings surprises, trading volume, or book-to-market ratios), forming distinct portfolios at each leaf node. Unlike standard decision trees that optimize prediction accuracy locally, P-Tree uses a global optimization criterion of maximizing the Sharpe ratio of the mean-variance efficient portfolio constructed from all leaf portfolios. This approach maintains a time-invariant tree structure, ensuring consistency across periods while capturing asymmetric interactions between characteristics. For example, liquidity measures may predict returns differently for high-earnings versus low-earnings stocks. By iteratively splitting the cross-section and re-estimating the tangency portfolio, P-Tree generates diversified test assets that span a more efficient frontier than conventional sorted portfolios, while remaining interpretable through its graphical tree structure.

However, machine learning approaches typically require large amounts of data and are predominantly developed and tested in highly liquid markets such as U.S. equities, where extensive information about individual stocks is readily available. This raises an important question: do these sophisticated models maintain their strong performance in less liquid markets with sparser information and fewer observable characteristics? Testing machine learning methods in such environments provides critical insights into their robustness and external validity beyond the well-documented, data-rich settings where they were originally designed.

1.2 Research Question

Can panel-tree portfolio construction methods maintain their advantages over traditional characteristic-sorted portfolios in the Swedish stock market?

1.3 Methodology

1.3.1 Data Gathering

Data from stocks in the Swedish stock market were used. In total there were data for 1176 firms between 1997 and 2020, and on average there were data points for around 300 firms per month. Finbas was used for market data, LSEG and Refinitiv were used for firm fundamentals, the Swedish House of Finance's data center was used for the Swedish Fama-French factors, Official Statistics of Sweden's statistical database was used for the inflation rate.

1.3.2 Boosted Panel Tree Algorithm

This thesis employs the **Boosted Panel Tree (BPTree)** methodology developed by Cong et al. (2024) [1] to construct portfolios and evaluate asset pricing performance in the Swedish equity market. BPTree extends the original Panel Tree (P-Tree) framework by combining multiple weak learners (individual Panel Trees) in a boosting ensemble to enhance out-of-sample performance and reduce estimation noise. Each tree in the sequence focuses on the residual structure left unexplained by previous trees, allowing the model to capture more complex, non-linear relationships between firm characteristics and expected returns.

The algorithm, available in the authors' GitHub repository linked in their article [1], was implemented in R. The data were pre-processed in Python using the `pandas` library before being fed into the BPTree model.

Structure of the Boosted Ensemble. In this implementation, the ensemble consists of three trees, following the setup:

- **Tree 1 (Base P-Tree):** The first tree in the ensemble, referred to as **Tree 1**, produces the initial latent factor $f_t^{(1)}$. This factor represents the return on the value-weighted tangency portfolio formed from the leaf portfolios generated by the optimal tree structure. Because Tree 1 is trained directly on raw excess stock returns without reference to any benchmark factor, $f_t^{(1)}$ captures the dominant dimension of systematic variation in the cross-section of returns that is explainable by observable firm characteristics. In essence, it serves as the *primary latent factor* extracted from the data—a data-driven portfolio that maximizes mean–variance efficiency within the given characteristic space. Subsequent trees in the boosting sequence (Tree 2 and Tree 3) then model the residual variation left unexplained by $f_t^{(1)}$, progressively refining the representation of the efficient frontier.
- **Tree 2 (First Boosted Tree):** The second tree is trained on the residual returns from the first tree—i.e., the portion of returns not captured by $f_t^{(1)}$. By learning from these residuals, Tree 2 refines the model and produces an additional factor $f_t^{(2)}$ that captures new variation in expected returns.

- **Tree 3 (Second Boosted Tree):** The third tree is trained on the residuals after removing the effects of the first two factors, $f_t^{(1)}$ and $f_t^{(2)}$. This final tree captures any remaining non-linear relationships and produces the third factor $f_t^{(3)}$. Together, the three trees form a **three-stage boosted ensemble** that progressively improves portfolio efficiency.

The combined model, often referred to as a **three-layer Boosted Panel Tree (BPTree-3)**, achieves a balance between flexibility and stability—further boosting typically provides diminishing returns while increasing computational complexity.

Algorithmic Procedure. The BPTree algorithm proceeds iteratively through the following steps:

1. **Initialization**: All stocks start in a single root node representing the market portfolio. Firm characteristics are normalized cross-sectionally to $[-1, 1]$ within each period. The initial portfolio factor $f_t^{(0)}$ is computed as the value-weighted market return.

2. **Sequential Tree Fitting (Boosting Process)**: At each boosting iteration b, a new P-Tree is trained on the residual returns—i.e., the portion of returns not yet explained by the ensemble up to iteration $b-1$. Each tree aims to find the optimal split (characteristic and threshold) that maximizes the marginal improvement in the Sharpe ratio of the combined portfolio.

3. **Split Optimization within Each Tree**: Similar to the base P-Tree, each tree evaluates pre-specified thresholds $c_m \in [-0.6, -0.2, 0.2, 0.6]$ for each firm characteristic and selects the split that maximizes the Sharpe ratio of the resulting leaf portfolios after re-optimizing their value-weighted tangency portfolio weights:

$$f_t^{(j)} = \mathbf{w}^{(j)'} \mathbf{R}_t^{(j)}, \quad \mathbf{w}^{(j)} \propto \left(\hat{\mathbf{E}}[\mathbf{R}_t^{(j)} \mathbf{R}_t^{(j)'}] + \gamma \mathbf{I} \right)^{-1} \hat{\mathbf{E}}[\mathbf{R}_t^{(j)}], \tag{1}$$

   where $\mathbf{R}_t^{(j)}$ contains returns of all leaf portfolios after j splits and $\gamma = 10^{-4}$ is a shrinkage parameter that ensures numerical stability.

4. **Ensemble Aggregation**: After each boosting iteration, the new tree's portfolio factor $f_t^{(b)}$ is added to the ensemble with a learning rate $\eta \in (0, 1)$:

$$F_t^{(b)} = F_t^{(b-1)} + \eta f_t^{(b)}, \tag{2}$$

   where $F_t^{(b)}$ denotes the cumulative boosted portfolio factor after b iterations. The learning rate controls the contribution of each tree and helps prevent overfitting. In this implementation, η is set to 1, implying full incorporation of each tree's factor into the ensemble.

5. **Termination**: The algorithm stops when either the maximum number of boosting iterations (three in this study) is reached or when the marginal improvement in the Sharpe ratio falls below a predefined threshold. The final boosted portfolio combines information from all trees, yielding a smoother and more robust estimate of the efficient frontier.

Compared to the original P-Tree, the Boosted Panel Tree algorithm improves generalization by sequentially correcting errors from previous iterations. This ensemble approach mitigates overfitting and enhances out-of-sample portfolio efficiency, producing portfolios that more closely approximate the true efficient frontier.

Hyperparameter Configuration. The Boosted Panel Tree (BPTree) algorithm contains several hyperparameters that control the structure of the trees, the level of regularization, and the stability of the mean–variance optimization. The following baseline parameter values were used, consistent with *Cong et al. (2025)*:

- **Covariance shrinkage parameter $\boldsymbol{\lambda_{\text{cov}} = 10^{-4}}$**: This regularization term is added to the diagonal of the covariance matrix to ensure numerical stability when inverting it to compute portfolio weights. A larger value of λ_{cov} produces smoother, more conservative portfolios by shrinking the covariance matrix toward the identity matrix, thereby limiting extreme weights. Conversely, a smaller value makes the model more responsive to the estimated covariance structure but increases the risk of overfitting and instability.

- **Covariance factor shrinkage parameter $\boldsymbol{\lambda_{\text{cov factor}} = 10^{-5}}$**: Applied at the ensemble (boosting) level, this parameter stabilizes the covariance matrix of the P-Tree factors (i.e., the portfolios generated by each tree). It ensures that the combination of boosted factors remains well-conditioned and prevents any single tree from dominating the final ensemble. Smaller values allow more aggressive exploitation of factor correlations, while larger values yield a more balanced, robust ensemble.

- **Split thresholds $\boldsymbol{c_m \in [-0.6, -0.2, 0.2, 0.6]}$**: These thresholds define the candidate cut points for splitting firm characteristics at each node. During training, the algorithm evaluates each possible split and selects the one that maximizes the Sharpe ratio of the resulting portfolios. The chosen thresholds determine the granularity of partitioning in the cross-section: narrower thresholds allow finer differentiation between firms but may increase noise sensitivity.

- **Minimum leaf size = 3**: This hyperparameter specifies the smallest number of firms that must remain in a terminal node (leaf). It constrains how far the tree can split, preventing overfitting by ensuring that each leaf portfolio is based on sufficient data. Smaller leaf sizes allow deeper,

more flexible trees, whereas larger leaf sizes yield smoother, more stable portfolios. This is derived from rescaling the minimum leaf size for the US market with the ratio of monthly stocks available in the Swedish and US market.

- **Cross-sectional winsorization at 1% and 99%**: At each time period, firm characteristics and returns are winsorized across the cross-section to limit the influence of extreme outliers. Values below the 1st percentile are replaced with the 1st percentile value, and values above the 99th percentile are replaced with the 99th percentile value. This reduces the impact of data errors or transient price spikes on tree splits and portfolio estimation.

- **Maximum tree depth = 10**: This sets the maximum number of consecutive splits allowed per tree, thereby controlling the model's complexity. Deeper trees can capture more intricate, non-linear relationships between firm characteristics and returns, but they also risk overfitting. Limiting tree depth balances model flexibility with generalization performance.

Overall, these hyperparameters jointly determine the trade-off between flexibility and robustness in the BPTree algorithm. The shrinkage parameters (λ_{cov} and $\lambda_{\text{cov factor}}$) primarily control numerical stability and regularization, while the structural parameters (split thresholds, leaf size, and depth) regulate the model's complexity and cross-sectional granularity. The three-tree boosted structure ensures that each subsequent tree incrementally refines the model, capturing residual patterns and improving the approximation of the mean–variance efficient frontier.

1.4 Preliminary Results

To evaluate BPTree performance in the Swedish equity market, we implement three distinct testing scenarios. This multi-scenario approach allows us to assess both in-sample fit and out-of-sample predictive power, while testing for robustness across different time periods.

1.4.1 Scenario A: Full Sample Analysis

Scenario A trains and evaluates the BPTree algorithm on the complete sample period from January 1997 to December 2020. This provides a benchmark for the model's maximum achievable performance when all available data is utilized. The full-sample analysis establishes an upper bound on what P-Tree can achieve in the Swedish market and identifies which characteristics are most important over the entire period. However, this scenario does not test out-of-sample predictive power, as the model is both trained and evaluated on the same data.

1.4.2 Scenario B: Time-Split (Past-Predicting-Future)

Scenario B implements a traditional out-of-sample test by dividing the sample chronologically at January 2010. The model is trained on data from January 1997 to December 2009 and evaluated on the subsequent period from January 2010 to December 2020. This forward validation tests whether patterns learned from historical data can predict future returns, which is the most practically relevant question for investors. By construction, the model has no access to future information during training, making this the primary test of real-world applicability.

1.4.3 Scenario C: Reverse Split (Future-Predicting-Past)

Scenario C provides a robustness check by reversing the train-test split: training on data from January 2010 to December 2020 and testing on the earlier period from January 1997 to December 2009. This approach serves two purposes. First, it tests for look-ahead bias by evaluating whether the model's performance in Scenario B could be attributed to fortuitous timing or cherry-picking of a favorable test period. Second, it assesses whether the relationships between characteristics and returns are persistent across different market regimes. If both Scenarios B and C yield similar out-of-sample performance, this provides strong evidence that the BPTree captures fundamental, time-invariant patterns rather than period-specific anomalies.

1.4.4 The Results for the Scenarios

Table 1: Performance Across All Scenarios

Scenario	Sharpe	CAPM		FF3	
	Ratio	Alpha	t-stat	Alpha	t-stat
A: Full Sample	2.74	21.84%	9.92	21.78%	9.82
B: Time-Split	4.21	21.70%	11.29	21.48%	11.51
C: Reverse Split	4.27	26.58%	15.00	26.55%	14.90

Notes: This table reports the in-sample performance of the first P-Tree factor across three testing scenarios. All t-stats > 9.5 indicate $p < 0.001$ (extremely statistically significant) Alphas represent monthly excess returns (in percent) from time-series regressions on the market factor (CAPM) and Fama-French three-factor model (FF3). t-statistics are based on Newey-West standard errors with 3 lags.

2 Literature Review

2.1 Existing Literature

This thesis is an extension of Cong et al. (2025) [1] paper that tested the performance of Panel Trees on the US stock market.

The core contribution of **Cong et al. (2025)** is the introduction of the **Panel Tree (P-Tree)** framework. The methodology addresses longstanding criticisms of ad-hoc test asset construction through the following key innovations:

1. **Advancing the Efficient Frontier:** P-Trees construct basis portfolios (leaf basis portfolios) that, under the Mean-Variance Efficient (MVE) framework, **significantly advance the efficient frontier** compared to traditional univariate or bivariate-sorted portfolios. Their P-Tree tangency portfolios achieved **high annualized Sharpe ratios**, exceeding 6.37 for a single P-Tree in the U.S. market (1981–2020), which substantially outperformed commonly used factor models.

2. **A Unified Framework for Factor Generation:** The P-Tree model generates traded factors that recover the Stochastic Discount Factor (SDF) and **outperform popular observable and latent factor models** for both investment and cross-sectional pricing.

3. **New Benchmarks for Model Evaluation:** The model's resulting leaf basis portfolios—which capture **nonlinearity and asymmetric interactions** among high-dimensional characteristics—pose a **significant challenge** to existing factor models, with most leaf portfolio alphas remaining unexplained by benchmarks like the Fama-French five-factor model.

4. **Integrating Economics and Machine Learning (ML):** P-Trees employ a **time-invariant tree structure** and a **global split criterion** that is explicitly guided by the economic objective of **maximizing the Sharpe ratio** of the latent factor.

2.2 Contribution to Literature

Our thesis extends this literature by implementing the Boosted P-Tree methodology in the Swedish equity market. The Swedish market is not only smaller and less liquid compared to the US market, but most importantly, for a machine learning algorithm, the amount of data is much more limited. Thus, the results obtained from the previous literature cannot be used to infer the performance of the algorithm in the Swedish market. Our thesis adopts the methodology and obtains a more robust result that can be used as a benchmark for the performance of the algorithm in markets or scenarios when data is limited.

3 Data Description

The data used contained 102,823 observations on a monthly basis. It can be separated into two categories, firm-specific data and macro data. The firm-specific data consists of 19 stock-level features.

Table 2: Firm Characteristics by Category

Category	Features
Size	Market capitalization
Value	Book-to-market, E/P, CF/P, S/P, Price-to-assets
Momentum	12-month momentum, 1-month lagged return
Volatility	12-month volatility
Profitability	ROA, Gross profitability, CFO-to-assets
Growth	Sales growth, Asset growth
Investment	Capex-to-assets, Asset turnover
Leverage	Debt-to-equity
Quality	Asset quality
Trading	Share turnover

This table lists the firm characteristics used in the BPTree analysis, organized by economic category.

There were data on 1176 stocks, and on average there were data for 300 stocks per month.

4 Preliminary Empirical Analysis

The results indicate that the Boosted Panel Tree algorithm is robust even in less-liquid markets with limited information. The algorithm's annualized Sharpe Ratio, the risk-adjusted return, ranges from 2.74 to 4.27 across the three different scenarios tested. In comparison, the annualized Sharpe Ratio calculated from the Swedish market's excess return for the full 1997–2020 period is only 0.53. This demonstrates that a trading strategy following the Boosted Panel Tree algorithm provides a risk-adjusted return that is 5 to 8 times higher than a passive buy-and-hold strategy of the market portfolio, representing significant outperformance on a risk-adjusted basis. The alphas represent the risk-adjusted abnormal return, the returns not explained by risk exposures. Thus, a positive alpha indicates the skill or performance of the model. To calculate the alphas two factor models were used, the Capital Asset Pricing Model and the Fama-French 3-factor model. For both the factor models and across all three scenarios the alphas obtained ranged from 21.48-26.58%, all with a significance level at 1%.

References

[1] Jingyu He Xin He Lin William Cong, Guanhao Feng. Growing the efficient frontier on panel trees. *Journal of Financial Economics*, 2024.

[2] Harry Markowitz. Portfolio selection. *The Journal of Finance*, 7(1):77–91, 1952.