QuantPulse: Short-term Portfolio Optimizer

Ayushman Anupam (MDS202411)
Boda Sowmya (MDS202413)

Data Science
Chennai Mathematical Institute

Problem Statement

- **Dynamic and Inconsistent nature of return over time.** In many portfolio methods, the expected return is calculated as the average of past returns. But in reality, returns keep changing with market conditions, so using only the historical average is often wrong.
- **Dynamic nature of Risk over time.** Market volatility is not constant. A single risk value based on past data cannot capture sudden changes, so we need a way to measure risk dynamically.
- **No proper way for short-term "risk-tolerated trade/return."** Existing portfolio methods are made for long-term investing. They do not directly tell us how much return we can aim for in short-term trading while staying within our risk limits.

Example: Why Traditional Methods Fail (NVIDIA)

- **Return behaviour changed completely.**
  NVIDIA's stock price was relatively stable until 2023. But in 2024, the stock entered a strong bullish phase. This means its expected return in 2024 should be much higher than its long-term historical average. Traditional portfolio method fails to capture this.
- **Risk (variance) also increased in 2024.**
  Along with higher returns, NVIDIA's volatility also increased in 2024. However, traditional methods calculate risk using *all past data*, including old stable periods. This hides the recent increase in volatility.
- **Result: Traditional models underperform.**
  Because they use outdated averages for both return and risk, traditional portfolio methods fail to reflect NVIDIA's current behaviour and therefore underperform in short-term decisions.

Our Proposed Solution

- **Better way to estimate expected return.**
  Instead of assuming the expected return is the average of past returns, we use the return predicted by our model. So the expected return becomes the *average of the predicted future returns*, not the historical average.
- **Making both return and risk dynamic.**
  To capture how the market keeps changing, we do not use the entire historical dataset. We either look only at the most recent k data points or apply a *decay factor* so that recent data is given more importance.
- **Method for short-term "risk-tolerated return."**
  Since no existing approach directly handles short-term reward under a risk limit, we propose three methods:
  - a simple finance-based heuristic,
  - a Bayesian approach, and
  - a machine-learning–based method.

Solution Approach

- **Stage 1: Predicting Short-Term Price Movements** using a rolling XGBoost model that updates as new data arrives in realtime.
- **Stage 2: Dynamic Portfolio Construction**
  - **Instead of using historical average return, we take the expected return from the model's predicted prices.** This produces a forward-looking, realistic estimate of short-term returns.
  - Risk and return values are recomputed at every step. **To capture dynamic market behaviour, all calculations use only the most recent k points** (lookback window), rather than old historical data.
  - Constructed portfolios using:
    - Efficient Frontier — focuses on return,
    - Bayesian Method — balances return and risk,
    - HERC — focuses on risk reduction.
  - Portfolio weights are updated continuously through rolling optimisation.

This gives us short-term portfolios that significantly **outperform the market** baseline and **Model future returns and risk**

Abstract

- Most industry portfolio methods use long-term historical averages to estimate return and risk.
- However, these approaches have clear gaps:
  - They do not use model-predicted returns for short-term allocation.
  - They assume return and risk stay stable, which is not true in fast-moving markets.
  - Traditional models cannot handle non-stationary or regime-changing behaviour.
- These limitations make existing methods less effective for short-term trading, where both return and risk can change quickly.
- In our project, we address these issues by:
  - using predicted prices to estimate expected return,
  - capturing dynamic behaviour through recent data or decay factors,
  - proposing three ways to compute short-term "risk-tolerated return"
- Overall, our aim is to create a more adaptive and realistic approach for short-term portfolio optimisation compared to traditional static methods.

Introduction

In this project, we build an end–to–end system for short-term portfolio optimisation using predicted 1-hour price movements of Indian stocks and indices. Traditional portfolio methods rely on long-term historical averages, which fail to capture the fast-changing nature of short-term markets.

Our goal is to create a more adaptive framework by predicting future prices, modelling dynamic risk–return behaviour, and constructing portfolios using multiple optimisation techniques.

Project Workflow:

1. **Data Description (generation and updation)** – collecting and updating stock and index data regularly.
2. **EDA and Data analysis** – cleaning data, exploring patterns, studying returns and volatility.

Introduction - continued

3. **Feature engineering and file creation** – building useful features and preparing storage files for predicted prices.
4. **Predicting future prices** – generating next 1-hour predicted prices for each stock and index.
5. **Portfolio construction** – creating portfolios using our three methods:
   - Heuristic Method
   - Bayesian Method
   - Machine-learning Based Approach
6. **Return calculation and visualization** – computing returns and comparing performance across methods.

Data Description

Collection:

- Data downloaded using Yahoo Finance API at 1-hour frequency (about 7 data points per day).
- To download data for each stock and index special symbol **called ticker** is used, its like a unique ID for each one.
- Each stock and index is saved in a separate file and updated in real time.
- This data is later used for feature engineering and prediction.

Downloaded Fields:
Datetime, Price, Close, High, Low, Open, Volume
Only **Datetime** and **Close** are used in our project.

Stocks and Indices Used

Indices:

1. ^NSEI : NIFTY_50
2. ^NSEBANK : NIFTY_BANK
3. ^CNXIT : NIFTY_IT
4. ^CNXPHARMA : NIFTY_PHARMA
5. ^CNXFMCG : NIFTY_FMCG
6. ^CNXAUTO : NIFTY_AUTO
7. ^CNXMETAL : NIFTY_METAL
8. ^CNXREALTY : NIFTY_REALTY
9. ^CNXENERGY : NIFTY_ENERGY
10. NIFTY_FIN_SERVICE.NS : NIFTY_FIN_SERVICE

Stocks:

11. RELIANCE.NS : RELIANCE_INDUSTRIES_LTD
12. TCS.NS : TATA_CONSULTANCY_SERV_LT
13. SUNPHARMA.NS : SUN_PHARMACEUTICAL_SERV_LTD
14. ICICIBANK.NS : ICICI_BANK_LTD.
15. INFY.NS : INFOSYS_LIMITED
16. SBIN.NS : STATE_BANK_OF_INDIA
17. BHARTIARTL.NS : BHARTI_AIRTEL_LIMITED
18. ITC.NS : ITC_LTD
19. LT.NS : LARSEN_&_TOUBRO_LTD.
20. HINDUNILVR.NS : HINDUSTAN_UNILEVER_LTD.

Data Analysis

Normality and Stationarity Tests on Returns

1. **Shapiro–Wilk and Jarque–Bera Test**: Checks normality of return.
2. **ADF(Augmented Dickey–Fuller) and KPSS Test**: Checks stationarity of return.

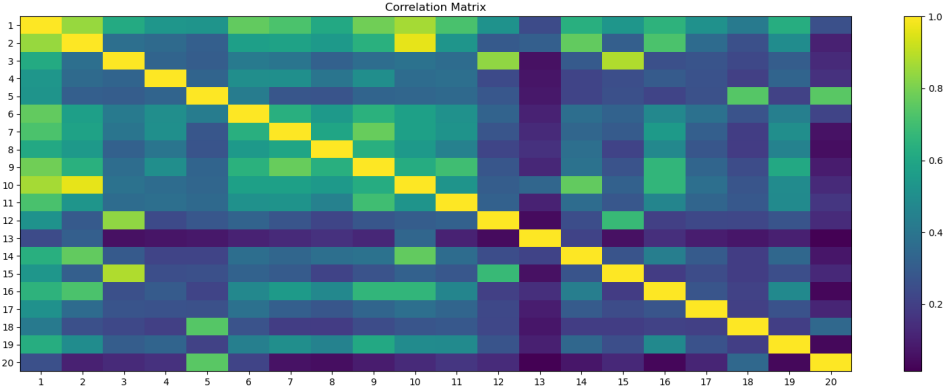
Summary of Five Sample Assets:

Asset	Shapiro p	JB p	ADF p	KPSS p	Conclusion
NIFTY_50	5.3×10^{-50}	0.0	1.2×10^{-29}	0.1	Not Normal, Stationary
NIFTY_BANK	2.3×10^{-47}	0.0	5.4×10^{-30}	0.1	Not Normal, Stationary
NIFTY_IT	9.1×10^{-49}	0.0	0.0	0.1	Not Normal, Stationary
NIFTY_PHARMA	4.7×10^{-47}	0.0	0.0	0.1	Not Normal, Stationary
RELIANCE	1.0×10^{-45}	0.0	0.0	0.1	Not Normal, Stationary

Data Analysis - continued

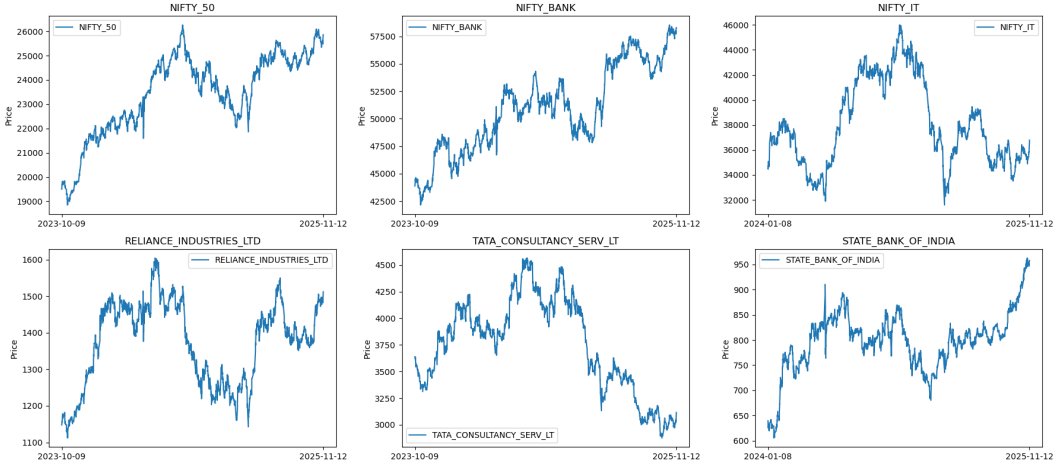
Correlation Heat-Map

This heatmap shows the correlation between stocks. Most stocks are positively correlated, which highlights the need for diversification so that portfolio risk is reduced.



Data Analysis - continued

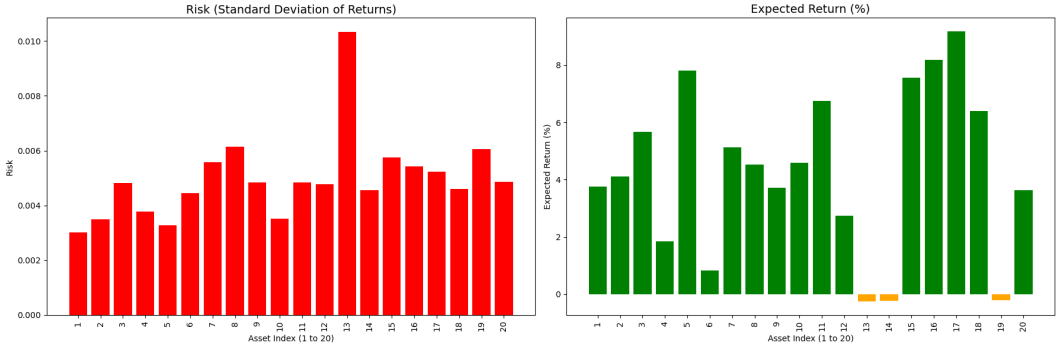
Prices of Some Stocks and Indices



Data Analysis - continued

Risk and Return plot

We can clearly see that different stocks have different risk–return profiles, and note these values change over time. Stocks with higher volatility generally have higher expected return, but the relationship is not stable.



Feature Engineering and Data File Creation

Till this point, we only had the **time** and **close price** for each asset. To build a useful prediction model, we needed more information, so we created several features. Most of them are **lag features**, such as:

- past 8 hours of price data (captures short-term behaviour),
- price at the same hour over the past 7 days (weekly seasonality),
- price on the same date over the past month (monthly pattern),
- price on the same date from the previous year (yearly seasonality).

These features help the model learn both recent movements and recurring seasonal patterns.

We also created a second file for each stock called `[stock_name]_prediction`. It contains four fields: **Time, Actual Price, Predicted Price and Return**.
The first two fields are filled immediately, and the predicted price and return are added later after the model makes its prediction.

Predicting Future Prices for Each Stock

we use a rolling prediction system that generates the next 1-hour price for every stock and index. The modelling notebook performs the following steps:

- **Load the stock's data and prediction file** The model reads the historical lag-feature data and checks which timestamps still need predictions.
- **Identify missing prediction points** A custom function finds all rows where the predicted price is missing. The earliest missing timestamp becomes the starting point for the prediction loop.
- **Rolling-window training** For each step, the model trains on the most recent window (up to the last 2000 data points) and learns the relationship between the engineered features and the actual price.
- **One-step-ahead prediction loop** Using XGBoost, the system predicts the next 1-hour price for each timestamp. After every 20 predictions, results are saved back into the `[stock_name]_prediction` file.

Predicting Future Prices for Each Stock - continued

XGBoost Model and Hyper-parameters Used for prediction

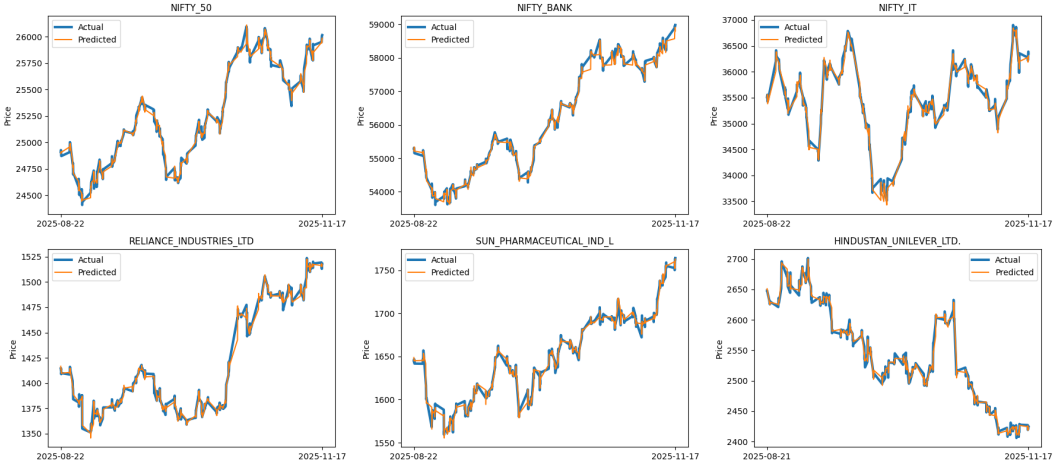
```
model = XGBRegressor(
    n_estimators=650,
    random_state=42,
    colsample_bytree=0.8,
    learning_rate=0.06,
    max_depth=6,
    n_jobs=-1
)
```

For evaluation we used two error metrics:

- **MSE (%)** – measures squared percentage error
- **MAPE (%)** – measures average percentage deviation from actual price

Assets price Prediction Result

Actual vs Predicted Prices of some assets



Assets price Prediction Result - continued

Plot for MSE(%) and MAPE(%) for evaluating result for all assets

Assets price Prediction Result - continued

Observation:

- Across all 20 assets (stocks + indices), the MSE values are extremely low **(mostly between 0.0007 and 0.0025)**.
- MAPE stays in the range of **0.18% to 0.37%**, which indicates that the predicted prices are very close to the actual prices.
- Even volatile sectors like **Realty, Metal, Pharma** maintain acceptable error.
- Index predictions (NIFTY 50, FMCG, IT) show the lowest errors, confirming consistent patterns.

So, **model is accurately capturing short-term price movements.** This gives us confidence that the predicted returns used in portfolio optimisation are **trustworthy, timely, and mathematically stable**.

Portfolio Construction

We constructed portfolios using three different methods, each offering a unique way of balancing risk and reward:

1. **Efficient Frontier (Mean–Variance Method)**
   Uses predicted returns and dynamic covariance matrices to compute the optimal weights that maximise return for a given risk level. This method directly incorporates our predicted expected returns.

2. **Bayesian Portfolio Method**
   Combines prior beliefs with observed data using Bayesian updating. Produces smoother, more stable weights and reduces overfitting to noisy short-term data. Helps in handling uncertainty in predicted returns.

3. **Hierarchical Risk Parity (HERC)**
   Builds a risk-balanced portfolio using hierarchical clustering of assets. Allocates weights based on correlation structure rather than expected return. Works well with non-normal, unstable short-term covariance and reduces concentration risk.

01. Efficient Frontier Method

The Efficient Frontier is a classic mean–variance portfolio optimisation technique introduced by Markowitz. It finds the portfolio that offers the highest return for a given level of risk, or the lowest risk for a given level of return.

- **Uses predicted returns**
  In our project, the expected return is not taken from historical averages. Instead, we use the model–predicted short-term returns, which makes the Efficient Frontier more suitable for intraday forecasting.
- **Dynamic covariance matrix**
  Instead of using a full historical covariance matrix, we compute a dynamic covariance using only recent data (or decaying weights). This captures the latest volatility behaviour.
- **Optimisation objective**
  The method solves a quadratic optimisation problem that balances risk and return, producing optimal weights for the predicted market conditions.

01. Efficient Frontier Method – continued

- **Sharpe Ratio optimisation**
  The Efficient Frontier maximises the Sharpe Ratio, which measures how much excess return we earn per unit of portfolio risk.

$$\text{Sharpe}(\mathbf{w}) = \frac{\mathbf{w}^\top \boldsymbol{\mu} - r_f}{\sqrt{\mathbf{w}^\top \Sigma \mathbf{w}}}$$

  - Numerator: expected excess return over risk-free rate
  - Denominator: portfolio volatility (standard deviation)
  - Higher Sharpe Ratio = better risk-adjusted performance

  This is why our optimisation maximises the Sharpe Ratio: it finds the most efficient short-term portfolio.

- **Output**
  The Efficient Frontier curve and the specific optimal weight vector chosen for our risk–return preference.

Efficient Frontier: Mathematical Formulation

Let predicted returns be $\boldsymbol{\mu} = (\mu_1, \ldots, \mu_n)^\top$ and the covariance matrix Σ.

Portfolio return:

$$R_p = \mathbf{w}^\top \boldsymbol{\mu}$$

Portfolio volatility:

$$\sigma_p = \sqrt{\mathbf{w}^\top \Sigma \mathbf{w}}$$

Sharpe ratio:

$$S(\mathbf{w}) = \frac{\mathbf{w}^\top \boldsymbol{\mu} - r_f}{\sqrt{\mathbf{w}^\top \Sigma \mathbf{w}}}$$

Objective (used in code):

$$\min_{\mathbf{w}} \; -S(\mathbf{w})$$

Constraints:

$$\sum_i w_i = 1, \qquad 0 \le w_i \le 1$$

02. Bayesian Portfolio Method

The Bayesian Portfolio Method incorporates uncertainty in expected returns by combining prior beliefs with observed data. Unlike the classical Efficient Frontier, which uses point estimates, the Bayesian approach produces more stable and robust weight allocations.

- **Uses predicted returns + prior belief**
  We treat the model–predicted return as observed data and combine it with a prior mean (historical trend or neutral belief).
- **Posterior expected return**
  Bayesian updating produces a smoothed, less noisy estimate of expected return, which is helpful for short-term predictions.
- **Reduces overfitting**
  Extreme or noisy predicted returns are shrunk toward the prior, creating more stable portfolios.

02. Bayesian Portfolio Method – continued

- **Short-term predictions are noisy**
  Hourly predicted returns fluctuate a lot. Bayesian updating smooths this noise using prior information.
- **Posterior mean gives better stability**
  The weighted combination of prior and observed return prevents extreme weights driven by prediction spikes.
- **Improves risk–adjusted performance**
  Bayesian shrinkage typically leads to lower variance in weight allocation, improving the short-term Sharpe ratio.
- **Output**
  A set of portfolio weights that balance predicted information with prior belief, resulting in a more stable short-term allocation.

Bayesian Portfolio: Mathematical Formulation

Let the predicted return be $\hat{\mu}$ and the prior mean be μ_0. Let Σ be the covariance matrix and τ the prior confidence.

Posterior expected return:

$$\mu_{\text{post}} = \left(\Sigma^{-1} + \tau I\right)^{-1} \left(\Sigma^{-1}\hat{\mu} + \tau\mu_0\right)$$

Posterior covariance:

$$\Sigma_{\text{post}} = \left(\Sigma^{-1} + \tau I\right)^{-1}$$

Portfolio optimisation:

$$\max_{\mathbf{w}} \ \mathbf{w}^\top \mu_{\text{post}} \quad \text{s.t.} \quad \mathbf{w}^\top \Sigma_{\text{post}} \mathbf{w} \leq \sigma^2, \quad \sum_i w_i = 1, \quad 0 \leq w_i \leq 1$$

03. Hierarchical equal Risk Contribution (HERC)

Hierarchical equal Risk Contribution (HERC) is a portfolio construction method that uses clustering to group similar assets and allocate risk across these clusters. It does not require expected returns, making it more robust when returns are noisy or unstable.

- **Cluster assets based on correlations**
  Assets with similar behaviour are grouped together using hierarchical clustering (dendrogram based).
- **Risk assigned at the cluster level**
  Risk is distributed among clusters before assigning weights to individual assets inside each cluster.
- **No dependence on predicted returns**
  HERC relies only on correlations and volatility, which is useful when short-term return predictions are uncertain.

03. HERC – continued

- **Handles unstable covariance matrices**
  Short-term data often makes the covariance matrix noisy. HERC avoids matrix inversion entirely.
- **Reduces concentration risk**
  By grouping correlated assets, HERC prevents overweighting stocks that move together.
- **Works well with short-term trading**
  Since it does not use expected returns, the method is robust even when predictions fluctuate.
- **Output**
  A diversified set of weights assigned by cluster-level risk allocation rather than pure optimisation.

Comparison of Methods (Risk vs Return)

Efficient Frontier

- **Best for Return**
- Uses predicted returns to try to earn higher profit.
- Works well when the model predictions are strong.
- **Trade-off:** Can take more risk to chase higher returns.

Bayesian Method

- **Balanced: Return + Risk**
- Gives a safer version of the predicted return.
- Good when you want reasonable return without taking too much risk.
- **Trade-off:** Returns may be slightly lower, but risk is reduced.

HERC (Risk Parity)

- **Best for Lower Risk**
- Focuses on spreading risk evenly across assets.
- Useful when market is uncertain or predictions are unreliable.
- **Trade-off:** Doesn't chase high return; aims for safety.

Results Overview

Each method is evaluated using three inputs:

- **Investment date (apply_date):** When the portfolio is created. If the market is closed, the system selects the next available date.
- **Withdrawal date (withdraw_date):** When the position is closed.
- **Lookback period (k):** Number of past days used for rolling optimisation.

Below are some results for the three portfolio methods:

Example 01.

Parameter	Efficient Frontier	Bayesian Method	HERC (Risk-Based)
Apply Date	2025-04-06	2025-04-06	2025-04-06
Withdraw Date	2025-11-06	2025-11-06	2025-11-06
Lookback Period	40 days	40 days	40 days
Return	**49.26%**	**50.86%**	**20.56%**
Avg Volatility	0.004140	0.004012	0.004191
Parametric VaR (95%)	0.6189%	0.6128%	0.5872%

Results Overview - continued

Example 02.

Parameter	Efficient Frontier	Bayesian Method	HERC (Risk-Based)
Apply Date	2025-10-06	2025-10-06	2025-10-06
Withdraw Date	2025-11-06	2025-11-06	2025-11-06
Lookback Period	10 days	10 days	10 days
Return	**15.05%**	**11.15%**	**4.10%**
Avg Volatility	0.002709	0.002590	0.003273
Parametric VaR (95%)	0.3664%	0.3521%	0.5112%

Monthly Return Comparison

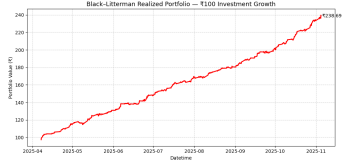
Time	Efficient Frontier	Bayesian	HERC(Risk-Based)	Market
7 Months (Eg 1)	5.87%	6.05%	2.71%	1.02%
1 Month (Eg 2)	15.05%	11.15%	4.10%	1.02%

Results Overview - continued

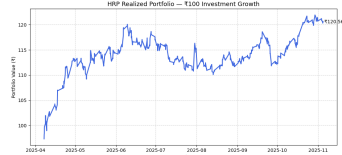
Return plot for example 01.

Efficient Frontier return

Bayesian Method return

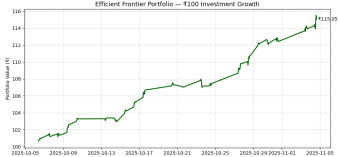


HERC (Risk-Based) return

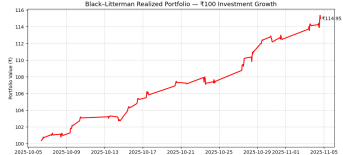


Return plot for example 02.

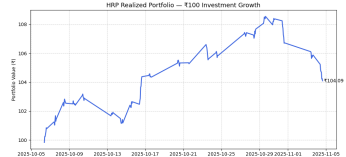
Efficient Frontier return



Bayesian Method return



HERC (Risk-Based) return



Conclusion

- **Our approach captures short-term market behaviour**
  Traditional portfolio methods rely on long historical averages, which fail when markets move quickly but our system uses **predicted 1-hour returns**, allowing portfolios to adjust to real-time market conditions.
- **Dynamic risk estimation instead of static assumptions**
  Industry models often assume risk is constant. We measure risk using only the most recent data, giving a **more accurate, time-sensitive view of volatility**.
- **Different allocation methods for different users**
  - Efficient Frontier $\rightarrow$ for users targeting higher returns
  - Bayesian $\rightarrow$ for balanced, stable portfolios
  - HERC $\rightarrow$ for lowest-risk, defensive positioning
- **Consistent outperformance over market CAGR**
  Across all examples, our methods delivered returns far above the market's baseline CAGR of 13%, proving their effectiveness in short-term horizons.

Industrial Applications

- **Short-term trading and intraday strategies**
  Firms can use predicted 1-hour returns to place fast trades and improve decision-making during volatile market hours.
- **Automated trading systems (Algo Trading)**
  The model can be integrated into automated trading bots to generate buy/sell signals and dynamically rebalance portfolios.
- **Risk management and hedging**
  Dynamic risk estimation helps institutions hedge their exposure quickly based on recent volatility and predicted moves.
- **Portfolio advisory products**
  Robo-advisors and investment platforms can offer personalised portfolios for users depending on whether they prefer high return, balanced, or low-risk strategies.
- **Market surveillance and stress testing**
  Understanding sudden shifts in return and risk helps analysts detect market regime changes and evaluate market stress in real time.

Limitations

Note: For the above model, we assume the following conditions, which may not always be true in real-world markets:

- **Instant trade execution**
  We assume trades execute immediately at the predicted price. In reality, there is delay, slippage, and order-execution time.
- **No transaction costs**
  Brokerage fees, taxes, and other charges are ignored. These reduce actual profit, especially for short-term trades.
- **Trading does not move market prices**
  We assume buying or selling does not affect the price. This holds only for small trade sizes—large trades can move markets.
- **No liquidity constraints**
  We assume the asset can always be bought or sold in any quantity. In low-volume periods, this may not be possible.

References

- "yahoo-finance Python API" — ReadTheDocs, python-yahoofinance.readthedocs.io
- "XGBoost: Extreme Gradient Boosting" — GeeksforGeeks, geeksforgeeks.org/machine-learning/xgboost/
- "Efficient Frontier — Modern Portfolio Theory" — Corporate Finance Institute (CFI), corporatefinanceinstitute.com/.../efficient-frontier
- "Bayesian Machine Learning and Portfolio Applications" — DataRobot Blog, datarobot.com/blog/bayesian-machine-learning/
- "Hierarchical Clustering Portfolio using Riskfolio-Lib" — ReadTheDocs, riskfolio-lib.readthedocs.io