Portfolio Optimization

By: Andrew Mendez, Jaxon Ham, Medhansh Sankaran

Markowitz Mean-Variance Model

Data Collected

We sourced our entire historical price panel by programmatically downloading weekly adjusted closing prices for each stock via Python's yfinance API and saved the results into a single Excel file (`sp500_data.xlsx`). Once imported into pandas, we parsed and chronologically sorted the date column, then eliminated any tickers with missing weeks so that every remaining symbol had a complete, gap-free record over the entire back-test horizon. From this clean panel, we computed simple week-over-week returns for each asset, which form the basis of our mean–variance input parameters: the expected return vector μ is calculated as the sample mean of the most recent L weekly returns for each stock, and the covariance matrix Σ is estimated as the sample covariance of those same returns over the same look-back window.

$$r_{i,t} = \frac{P_{i,t}}{P_{i,t-1}} - 1$$

Syntax

Symbol	Description
w_i	Portfolio weight allocated to asset i
μ_i	Expected return of asset i
Σ_{ij}	Covariance between returns of assets i and j
λ	Risk-aversion coefficient controlling the trade-off between return and risk
$P_{i,t}$	Price of asset i at time t
$r_{i,t}$	Return of asset i at time t
L	Number of weeks in the look-back window for estimating returns/covariance
n	Total number of assets (stocks) in the portfolio universe
w_{max}	Maximum allowed weight per asset (e.g., 0.05 or 5%)
C	Maximum allowed turnover in a given week
$R_{p,t}$	Portfolio return at time t
$R_{b,t}$	Benchmark return (e.g., S&P 500) at time t

Parameters - Decision - Objective - Constraint

Parameters

We have three important parameters that can be adjusted to your liking. Further analysis will need to be done to see how performance change as these parameters change.

1. λ
2. L
3. w_{max}

Decision

w_i is the variable were trying to optimize. Weight is specifically defined as the percent of the portfolio you give to stock i.

Objective

In the Markowitz mean–variance model, we are trying in whole to maximize the return of the portfolio. This is done through maximizing the weight toward the stock we expect to perform the best. We define the performance of a stock based on two parts. We have the first term that captures μ_i, which represents the historical mean return of stock i. Then we have the second term, which incorporates the covariance between stock i and stock j, denoted as Σ_{ij}. This measures how the returns of different assets move together over time.

The portfolio weights w_i determine how much capital we allocate to each stock. In the objective function, the weights directly influence both expected return (through $\sum \mu_i w_i$) and risk (through the quadratic form $\sum w_i \Sigma_{ij} w_j$). A higher weight on a stock with a high expected return increases potential gain, but if that stock also has high variance or is strongly correlated with other risky assets, it could raise overall portfolio risk significantly.

This is where diversification naturally arises. Since the covariance term penalizes concentrated positions in assets that move together, the optimizer tends to spread weights across assets that are less correlated, even if some have slightly lower expected returns. This leads to a more stable and robust portfolio. For example, if stock i and stock j are highly correlated, allocating a large weight to both would increase total variance; the optimizer avoids this by reducing exposure to one or both and instead favors assets that diversify risk. In this way, diversification is not something explicitly programmed it is a natural outcome of solving the optimization problem with the covariance matrix included in the objective.

$$\max_{\mathbf{w}} \sum_{i=1}^{n} \mu_i w_i - \lambda \sum_{i=1}^{n} \sum_{j=1}^{n} w_i \Sigma_{ij} w_j$$

$$\mu_i = \frac{1}{L} \sum_{k=1}^{L} r_{i,t-k}$$

$$\Sigma_{ij} = \frac{1}{L} \sum_{k=1}^{L} \left(r_{i,t-k} - \mu_i\right) \left(r_{j,t-k} - \mu_j\right)$$

Constraint

The constraints in the Markowitz mean–variance model play a crucial role in shaping a realistic and well-structured portfolio. One key constraint ensures that the sum of all portfolio weights w_i equals 1, meaning the model allocates 100% of the available capital with no borrowing or uninvested cash. This is known as the budget constraint, and it guarantees that the entire portfolio is fully invested.

Another important constraint places bounds on each individual weight. The lower bound $w_i \geq 0$ enforces a long-only policy, meaning short selling is not allowed. This is common in institutional and retail investment settings, where negative positions are restricted. The upper bound, often set as $w_i \leq w_{\text{max}}$, limits the maximum proportion of capital that can be allocated to any single asset. For example, if w_{max} is 0.05, no more than 5% of the total portfolio can be invested in any one stock.

These constraints work together to enforce diversification and manage risk. Without them, the optimizer might allocate all capital to a single high-return stock, which could expose the portfolio to excessive volatility. By capping the weights and ensuring full investment, the model produces portfolios that are not only mathematically optimal but also practically sound and aligned with common investment guidelines.

$$\sum_{i=1}^{n} w_i = 1, \quad 0 \leq w_i \leq w_{\text{max}} \quad \text{for all } i$$

Results

Inital Set

All Results are ran on a initial investment of \$10,000, $\lambda = 1$, $w_{\max} = 0.05$, and $L = 28$ weeks. Side note, we set the max weight to 0.05 since whenever you let the model diversity its self naturally it would pick only a small handfull of stocks at a given period. Most modern firms still have to set hard limits like this, so we followed foot. Even with the 0.05 the model would still diversify naturally within those margins.

Summary of Results

The back-test demonstrates that a \$10,000 initial investment grew to \$14,657.95 by April 2025, representing a 46.58% total return over the roughly two-and-a-half-year period (see Table 2 for final results). In Figure 1, the portfolio's cumulative wealth curve consistently outpaces the S&P 500 benchmark, especially during market rallies in mid-2023 and late 2024, indicating that the optimizer captured upward trends more effectively than a passive buy-and-hold strategy.

On a risk-adjusted basis, the strategy achieved an annualized return of 19.57% against an annualized volatility of 23.72%, yielding a Sharpe ratio of 0.64 and a Sortino ratio of 1.05 (see Table 1 for performance metrics). Figure 2 plots the weekly return distribution, showing more frequent moderate gains than losses, while Figure 3's rolling 52-week returns highlight sustained outperformance during key intervals throughout the back-test horizon.

Drawdown analysis reveals that the deepest peak-to-trough decline reached 29.34% and persisted for up to 24 weeks (Table 1). The drawdown timeline in Figure 5 shows the most severe retrenchment in early 2024, underscoring that, despite diversification, the model remains exposed to extended market corrections. Notably, the subsequent rebound illustrates the advantages of weekly dynamic rebalancing in recapturing lost ground.

Trading activity was substantial: average weekly turnover stood at 42.80% (median 43.53%), reflecting the model's responsiveness to changing return and covariance estimates (Table 1). Figure 4 charts weekly turnover as a percentage of portfolio value, while Figure 6 breaks out both the count and notional amounts of buy versus sell transactions, revealing spikes in trading volume around market inflection points. High turnover implies that, in a live setting, transaction costs could materially affect net performance.

Finally, the strategy's hit rate the percentage of weeks with positive returns was 57.81% (Table 1), indicating a moderate skew toward winning periods. Taken together, these results demonstrate that the Markowitz mean–variance optimizer delivered meaningful excess returns at the expense of higher turnover and drawdown risk, highlighting the trade-offs inherent in active, rebalanced portfolio management.

Flaws and Improvements

High Turnover

High turnover (average 42.8% per week; Table 1) means that, without trading frictions, the back-test overstates net returns. Enforcing the explicit turnover constraint

$$\sum_{i=1}^{n} \left| w_{i,t} - w_{i,t-1} \right| \leq C$$

within the optimization would cap week-to-week trading and more realistically account for transaction costs.

Fixed Parameters

Our fixed choices of risk aversion λ, look-back L, and weight cap w_{max} can lead to fragile performance. Embedding these as tunable hyperparameters via a grid search or cross-validation over hold-out weeks would reduce overfitting and improve stability

Chases Noise

Relying on the sample covariance matrix exposes the optimizer to estimation noise when n is large. This relates back to the High Turnover in that if we wanted a unconstrained approach we would have to add in some penalty for chasing noise.

Appendix

Table 1: Performance metrics

Metric	Value
Annualised Return (%)	19.57
Annualised Volatility (%)	23.72
Sharpe Ratio	0.64
Sortino Ratio	1.05
Max Drawdown (%)	29.34
Max DD Duration (weeks)	24.00
Hit Rate (%)	57.81
Avg Weekly Turnover (%)	42.80
Median Weekly Turnover (%)	43.53

Table 2: Final Results

Metric	Value
Final Portfolio Value (USD)	14657.95
Return on Investment (%)	46.58
Total Buy Transactions	692.00
Total Sell Transactions	671.00

Figure 1: Cumulative Portfolio Value

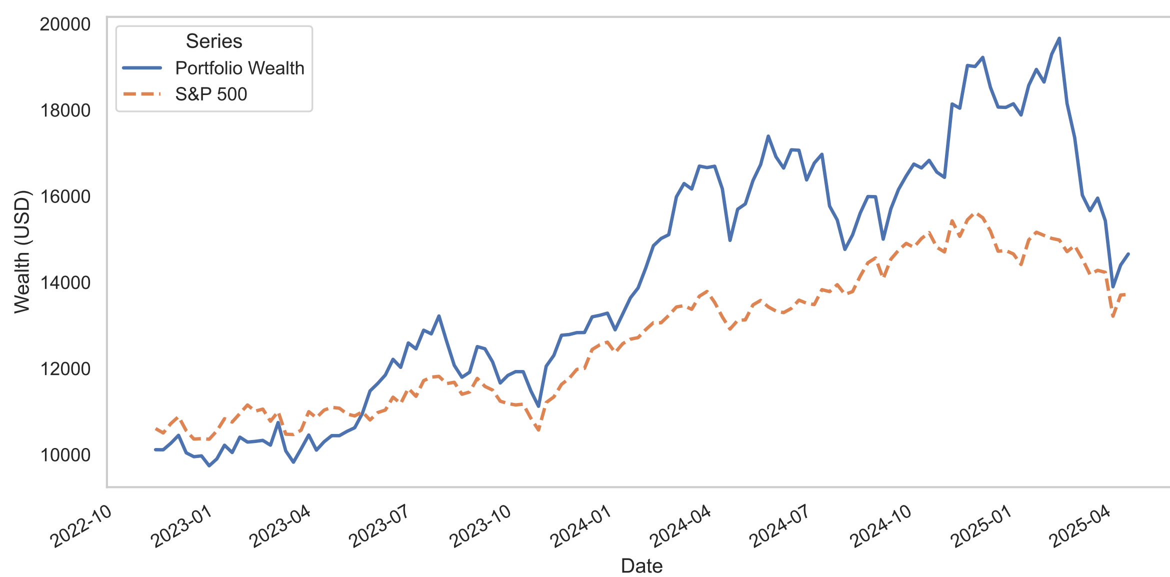


Figure 2: Weekly Returns

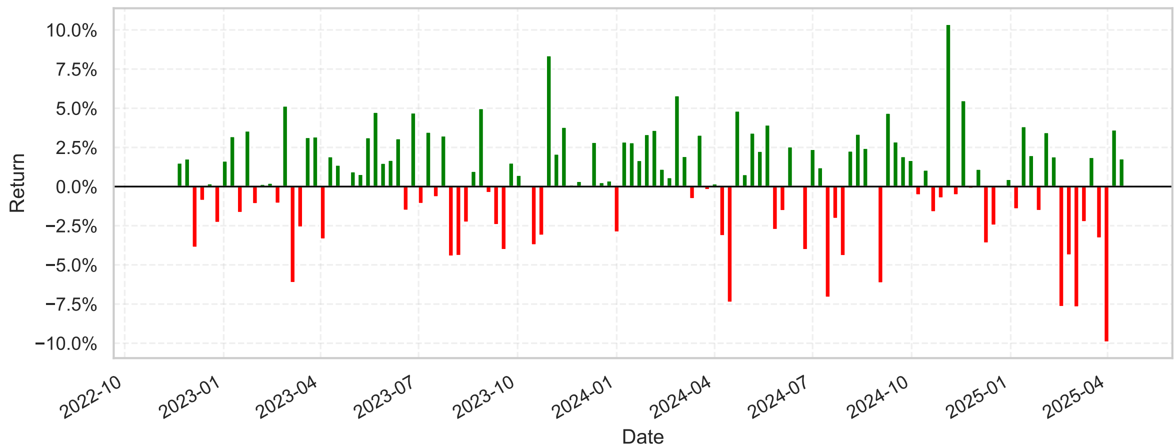


Figure 3: Rolling 52-Week Return (%)

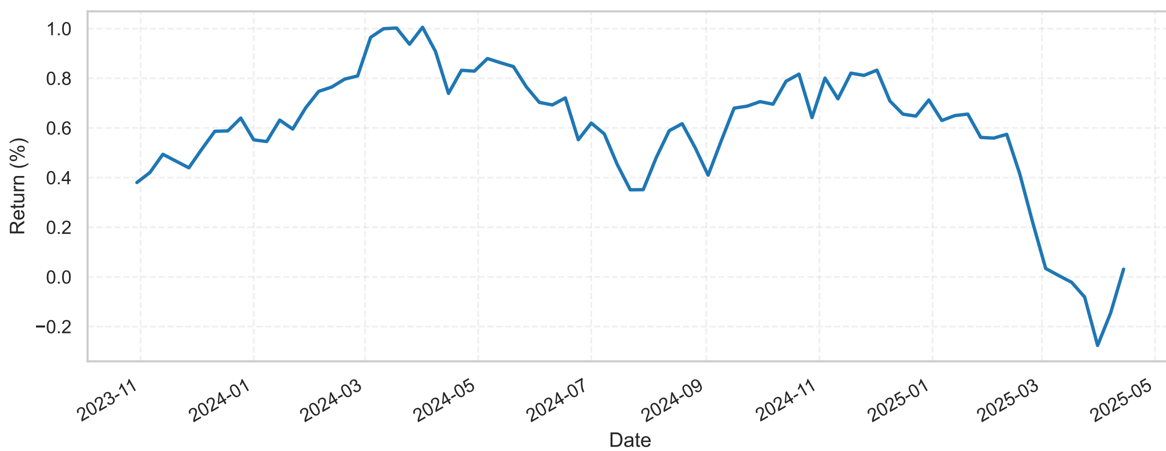


Figure 4: Weekly Portfolio Turnover

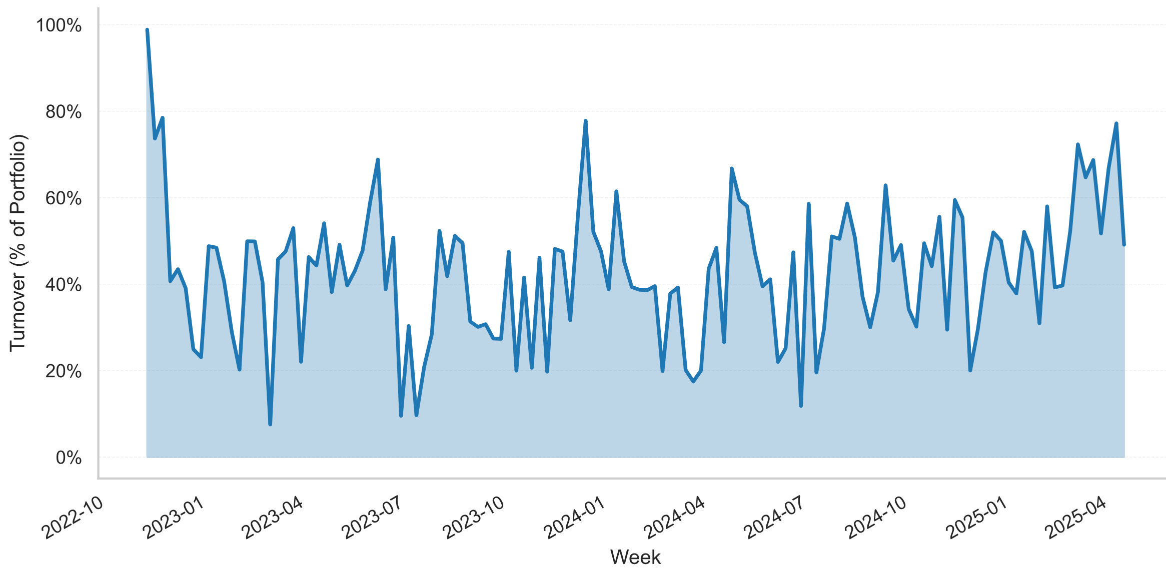


Figure 5: Drawdown Over Time

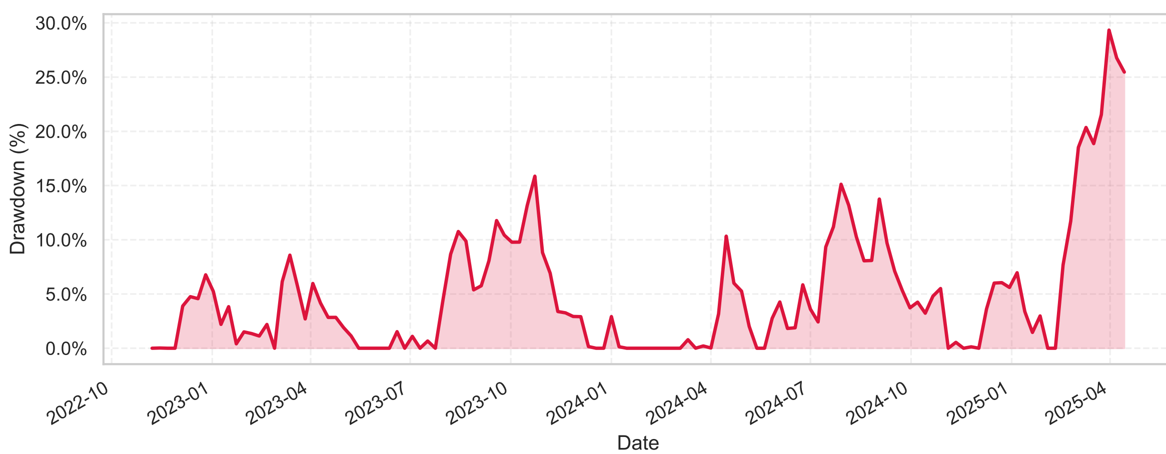
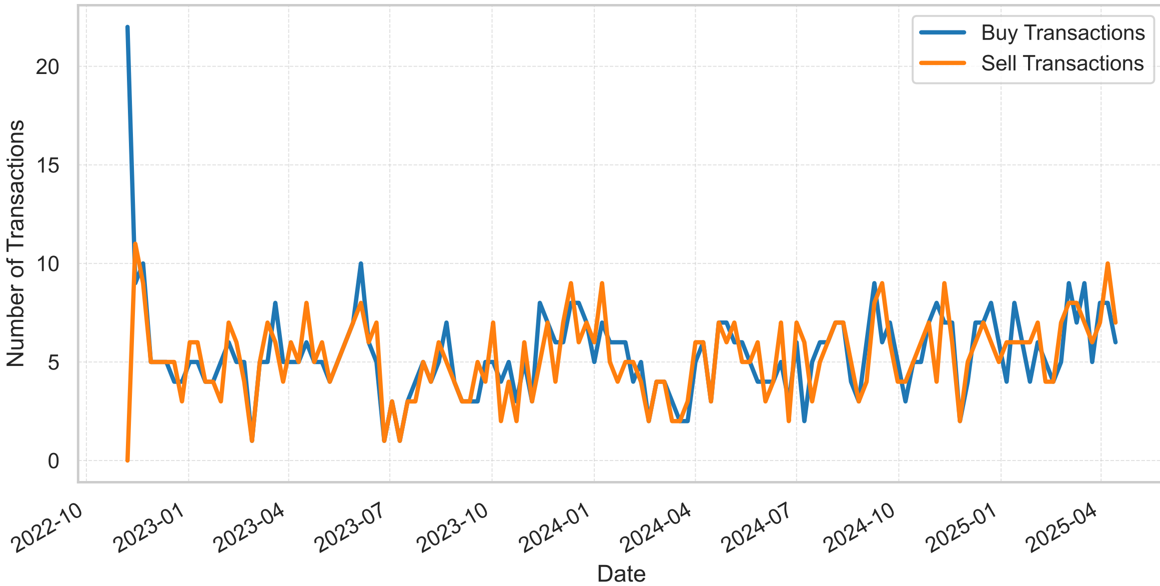
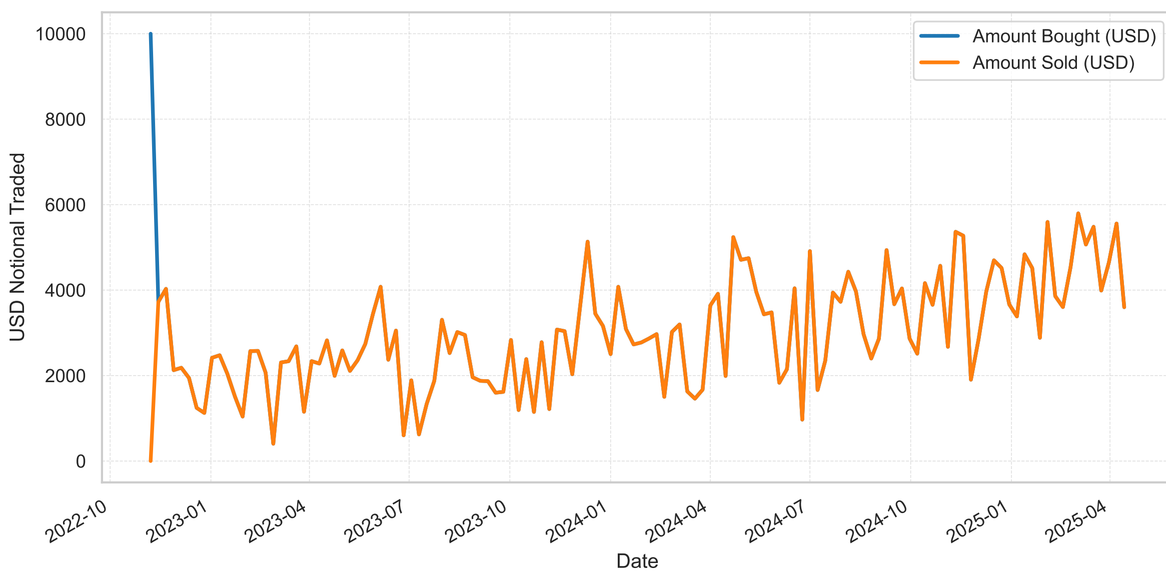


Figure 6: Buy vs Sell Transaction Count/Amount Over Time





Sources

https://en.wikipedia.org/wiki/Harry_Markowitz

https://en.wikipedia.org/wiki/Modern_portfolio_theory

https://en.wikipedia.org/wiki/Covariance_matrix

https://en.wikipedia.org/wiki/Markowitz_model#Demerits_of_the_HM_model

https://en.wikipedia.org/wiki/Portfolio_optimization