Project Report – Portfolio Optimizer

- Ojas Sameer Raverkar

Website - <https://montecarloportop.streamlit.app>

GitHub Repository - <https://github.com/ojasraverkar/portfolio_optimiser>

1. Summary and Foreword

My ulterior aim while doing this project is to improve my quantitative and programming skills. I chose this subject as this has various applications in Quantitative Investment Management. Building this from scratch will strengthen my understanding of foundational finance concepts and the coding challenges will also test my problem-solving skills.

A Portfolio Optimizer is an investment tool. It helps us to define how much weightage we must give to stocks, for a given level of risk. An example for this could be an investor trying to choose the best stocks for his portfolio. He goes through hundreds of stocks, calculates returns over his desirable time period and then creates a priority list in a manner in which his capital will be divided. Let’s assume he goes through NIFTY 50 stocks. He identifies 3 stocks- INFY, TCS and WIPRO who have great returns in past 3 years and he would like to have these in his portfolio. But at the same time, he has to decide that he allocates 25% capital to TCS, 40% to Wipro and the rest to INFY. Why? Maximising returns.

My attempt is to automate this process, with precision, to save time in manual scraping and analysis of securities. The upcoming code has a basic requirement- It must be able to accurately create a portfolio of stocks and appropriately allocate capital to the chosen stocks to maximize returns. This will also incorporate the risk by calculating the risk-reward factor (Sharpe ratio, explained in the later parts.)

An interesting analogy is to fill your suitcase with most useful items while keeping the weight in limit. My attempt will be to create a reader friendly project walk-through with proper comments and logic explanations.

I hope you find this document useful!

1. Problem Statement

For any investor, new or seasoned, a challenging problem arises after picking his winning lineup. The dilemma is how to prioritise stocks and allocate capital amongst them to maximise return while managing risk. Manually estimating this carries a different risk and is a complex, time-consuming process.

1. Objective

Therefore, the objective of this project is to create a simple tool, based and driven by historical data of stocks which solves the capital allocation of problems. Our aim is to automate the process of finding an optimal portfolio based on the fetched historical data. This includes assigning accurate\* weights to each security and allowing users to customize the input.

1. Methodology
   1. Modern Portfolio Theory

The financial model is built on principles of Modern Portfolio Theory, developed by Harry Markowitz. The core idea of this theory is that an assets risk ad return should always be viewed together, as if they are metaphorically married. The theory uses statistical methods to construct diversified portfolios.

* 1. Monte Carlo Simulation

To find the optimal portfolio from a sea of portfolio, a Monte Carlo simulation is employed, which includes generation of thousands of random portfolio weights and calculating the total expected annualized returns and volatility.

* 1. Sharpe Ratio

To identify the best portfolio out of the generated portfolios, we employ a formula which measures the risk adjusted returns of a security. Its states as:

[Sharpe Ratio = (Portfolio Return - Risk-Free Rate) / Portfolio Volatility]

The portfolio with the highest Sharpe ratio is considered to be the most efficient as it provides the highest return per unit of risk taken.

1. Tech Stack

* Programming Language – Python
* Data Handling and Processing – NumPy and Pandas libraries
* Data Acquisition – yfinance (Yahoo finance library)
* Data Visualization – Matplotlib
* Web deployment – Streamlit

1. Implementation

The model fetches historical price data (specifically ‘Close’ column) and calculates logarithmic returns which is a standard for financial time-series analysis due to its additive properties over a time period. Data is then cleaned by dropping missing values or posing invalid symbol errors.

The ‘port\_op’ function is the heart of the financial model, which simulated the Monte Carlo method. It calculates the annualized mean returns and covariance matrix from the logarithmic returns.

The frontend is a simple bit of code done using Streamlit, whose features include a interactive sidebar for user inputs and results are presented in a clean table on the main page.

1. Conclusion and Future plans

The project demonstrates the application of Modern Portfolio Theory through an intuitive web app. It provides a powerful tool for investors to make informed, data-driven decisions about capital allocation.

Future plans and updates that you should look forward to are a back-testing module to compare the model results with major indices in the market, introducing more complex risk metrics algorithms like Sortino ratio and Conditional Value at Risk, adding functions to portfolio rebalancing, improving overall user experience through experimented user flows and improving the model performance through logical updates in general.