Project: Automated Google Sheets Portfolio Tracker

Intent: The intent of the portfolio tracker was originally for personal use using the built-in Google Finance functions, and manually entering other data. Progressively, it was decided to expand on what was being provided and automating these processes.

Functionality: All data on the sheet is fully automated apart from the ‘Ticker’, ‘Shares’, and ‘Purchase’ columns on the ‘Portfolio Data’ sheet. This is where you would enter your own data.

* The same functionality (in terms of the Python Code) can be applied to other industries outside of finance, to store information into Google Sheets.

Process

Packages:

- Finance: empyrial, quantstats, yfinance

-Google: google.colab, , gspread, google.auth

-General: numpy, pandas, datetime

You may have seen the ‘%%capture’ line, which simply disabled any output. We do this for packages to avoid seeing the !pip download code, as it doesn’t provide any value.

General Process:

The first steps go through authentication regarding your Google Account. Here you will sign in and begin linking your sheet (via URL) to the script. We also define the main data sheet using ‘.get\_worksheet(1)’. Remember, the indexing starts at 0, so this is technically the second sheet.

Next, we pulled the ticker data and the weights each ticker makes up of the portfolio and stored them in the two variables. We used these two lists to run our empyrial back test, store these quantitative statistics in a Pandas DataFrame, and transition them to be stored onto the Google Sheet on the display sheet.

Following this, we download all ticker data from yahoo finance, reindex by column as it becomes resorted after the download, and calculate the percent returns. From here, it was decided to use quantstats to manually calculate each statistic. To limit the computing power, we simplified it to a single function after multiple reworks. This function allows us to add new functions onto the sheet with minimal changes in the existing code.

After, we then implement the dividends, and a dividend calendar. This calendar allows you to see your payout structure throughout the year to easily calculate your yearly and monthly income(s). To adjust the time period, in the ‘month\_names’ column, change the start and end to your desired months.

We then implemented the Sectors for our plotting which will be implemented in the next section. One thing to note here is that in our experience, whenever getting ‘N/A’ it was due to the asset being an ETF. If an asset does not have a proper assignment for Sector, it could falsely be flagged as an ETF.

In the final block of code, we calculate portfolio returns, benchmark returns, and individual asset returns. We then use these for the visualizations on the main display page. More specifically, for both of the charts on the right hand side (Individual Asset Returns, and Portfolio & Benchmark).