**Python應用實作專題:定期定額投資分析與視覺化，從數據獲取到圖表展示**

**Dollar Cost Averaging Investment Analysis and Visualization: From Data Acquisition to Chart Display**

林晉宇

國立中央大學 資訊工程學系

[jinyulin34@gmail.com](mailto:jinyulin34@gmail.com)

【摘要】 本專題報告旨在設計和實現一個[自動化定期定額（Dollar Cost Averaging, DCA）投資比較分析與視覺化的儀表板](https://dca.jinlk.dev/)。本文利用Python中的yfinance進行金融數據的獲取，並使用Streamlit框架構建互動式網頁應用。通過此儀表板，使用者可以輸入投資金額、投資頻率、資產及投資時長，系統會自動計算並顯示投資組合的價值變化。報告詳細介紹了DCA策略的計算邏輯、數據處理與圖表繪製的過程，以及應用的設計與實現。該儀表板可以有效地幫助使用者視覺化其投資策略，提供便捷的投資決策支持。

【關鍵詞】定期定額投資、自動化分析、數據視覺化、Python、Streamlit

# 前言

定期定額投資(Dollar Cost Averaging, DCA)是一種在金融市場中廣泛使用的投資策略。該策略通過在固定時間間隔內，定期投入固定金額的資金來購買目標資產，從而在長期內平滑市場波動對投資的影響。DCA策略主要優勢在於能夠降低市場波動風險，並且不需要深入的技術分析能力或金融背景知識，同時避免投資者因市場情緒波動而進行不理性的投資決策。

有鑑於網路上鮮少有將不同標的資產進行長期DCA回報率比較的網站，本次專題的目的即設計和實現一個自動化定期定額投資分析與視覺化的儀表板，通過此儀表板，使用者可以輸入投資金額、頻率、時間，系統自動計算並顯示歷史回測後的投資組合價值變化。本文詳細介紹DCA策略的計算邏輯、數據處理、圖表繪製的過程、以及應用的設計與實現。

本報告將從選擇資產、數據獲取、系統設計與實現、開發流程與部署、結果與討論等方面詳細闡述該專題的研究過程和成果，介紹生成式語言模型對本次專題的幫助，並對專題未來的研究、實作方向提出建議。

# 專案規劃

在專案開始時，我便確定了要使用Streamlit框架搭配Python進行開發。Streamlit是一個開源的Python框架，專為快速構建數據應用而設計，允許開發者通過少量的程式碼來創建精美的數據呈現網頁，這非常適合本次專題的需求。Python則提供了豐富的數據處理和分析工具，使我們能夠高效地獲取和處理金融數據。

本次專案開發時程，預估約一個月，從一開始的釐清架構，獲取數據並清洗，到最後計算數據、回報率後，將資料呈現在網頁上。預想功能包含以下幾個方面:

1. **數據獲取:** 使用yfinance庫從Yahoo Finance獲取歷史價格數據，支持多種資產類型（如股票、加密貨幣、指數等）。將數據獲取功能封裝在單獨的模塊中，方便後續調用和擴展。
2. **用戶介面輸入:** 提供直觀的設置面板，允許用戶選擇投資資產、輸入投資金額、選擇投資頻率和開始時間。提供多選功能，允許用戶同時比較多個資產的投資表現。
3. **投資回報計算:** 根據用戶的輸入參數，計算定期定額投資的累積投資價值和回報率。支持多種投資頻率，如每週、每兩週、每月或每年。
4. **圖表呈現:** 繪製累積投資價值變化圖，展示從投資起始日期到當前日期的投資價值隨時間的變化。繪製回報率比較圖，展示多個資產在相同投資期間的回報率。
5. **網站部署:** 使用Nginx作為反向代理服務器，將HTTP請求轉發到後端的Streamlit應用。確保網站的穩定運行和安全性。

# 方法與技術

# 選擇資產

我主要以下面五種類型資產進行分析: **比特幣（BTC）、以太坊（ETH）、黃金（Gold）、標準普爾500指數（S&P 500）以及美國20年國債ETF（iShares 20+ Year Treasury Bond ETF, TLT）**。

**比特幣**是全球最具代表性的加密貨幣，因其去中心化和有限供應的特性，被許多投資者視為數字黃金。比特幣的歷史價格波動較大，具有高風險高回報的特性。選擇比特幣作為研究資產，可以幫助分析高波動性資產在定期定額投資策略中的表現。

**以太坊**是僅次於比特幣的第二大加密貨幣，因其支持智能合約和去中心化應用（DApps）的特性，具有廣泛的應用前景。以太坊的價格波動性也相對較高，選擇以太坊有助於比較不同加密貨幣在定期定額投資策略中的表現。

**黃金**是一種歷史悠久的避險資產，具有抗通脹和價值儲存的功能。黃金價格通常在經濟不確定性或市場波動時上升，因而被許多投資者用來分散投資風險。選擇黃金作為研究資產，可以了解其在定期定額投資中的穩定性和避險效果。

**標準普爾500指數**是美國股市中涵蓋500家大公司的綜合指數，被廣泛認為是衡量美國經濟健康狀況的代表。選擇S&P 500作為研究資產，可以分析股票市場投資在定期定額策略中的長期表現和成長潛力。

**美國國債**被視為全球最安全的投資之一，因其由美國政府擔保，具有極低的違約風險。國債收益相對穩定，但回報率較低。這邊選擇美國長期國債的ETF基金作為投資標的，其投資方式便捷，提供投資者參與長期美國國債市場。

這些資產皆為主流資產，涵蓋不同風險和回報特性的投資標的，從而全面評估定期定額投資策略在不同資產類別中的效果。通過比較這些資產在定期定額投資中的表現，我們能夠更好地理解該策略在不同市場環境下的優勢和局限。

# 數據獲取

我使用Python中的yfinance來獲取資產價格的歷史數據。可以從yfinance獲取各種金融數據，包括股票、加密貨幣、指數等等。

數據獲取的函式單獨寫在*fetchPrice.py*檔案，該函式接受兩個參數: 資產的股票代碼(ticker\_symbol)以及需要獲取的天數，回傳為Pandas Dataframe格式。

# 框架選擇

我選擇Streamlit作為主要的開發框架，實現本次專案的儀表板網站。Streamlit是一個開源的Python框架，專為快速建構數據應用而設計，允許開發者通過少量的程式碼來創建精美的數據呈現網頁，適合用於數據科學與機器學習領域的應用。Streamlit也提供許多現成的小組件(widgets)，並支援Matplotlib、Plotly等數據視覺化的模組，同時可以輕鬆的部署到伺服器或雲端服務上。基於他的便捷性、即時反應、良好互動性，我選擇使用Streamlit作為開發這個專題的主要框架。

# 計算邏輯

定期定額投資的計算邏輯主要包含以下步驟:

1. **選擇初始購買日期:** 根據用戶選擇的投資頻率（如每週、每兩週、每月或每年），生成一系列的購買日期。這些日期表示投資者在這些特定日期進行資產購買。
2. **資產價格獲取:** 通過yfinance獲取所選資產在購買日期的歷史價格。為了確保在市場關閉日也能進行投資，若購買日期不在交易日，則使用最近的前一個交易日價格。(程式碼如下)

***for i, date in enumerate(all\_dates):***

***d = date***

***while d not in asset\_data.index:***

***d -= pd.Timedelta(days=1)***

***price = asset\_data.loc[d, 'Close']***

1. **計算購買單位:** 根據每次投資的固定金額，計算在每個購買日期購買的資產單位數量。這些單位數量為用戶每次投入的金額除以當天的資產價格。

***if date in purchase\_dates:***

***total\_purchase\_amount += self.amount / price***

1. **累積投資數量:** 將每個購買日期的資產單位累積起來，表示總的投資數量。

***if date in purchase\_dates:***

***total\_invest\_usd += self.amount***

1. **計算投資價值:** 在每一天，根據累積的資產單位數量和當天的資產價格計算總投資價值。這可以反映資產價值隨時間的變化。(在每次購買日期回圈內，進行計算)

***total\_asset\_value[i] = total\_purchase\_amount \* price***

1. **計算投資回報率:** 在每一天，根據總投資價值和累積投資金額計算投資回報率。

***return\_rate = (total\_asset\_value[i] / total\_invest\_usd - 1)\*100***

1. **比較不同資產:** 用戶可以選擇多種資產進行比較。在每種資產的計算邏輯相同的情況下，系統會生成所有選定資產的投資價值變化圖和回報率圖，以便用戶進行直觀比較。

# 系統設計與實現

# 系統架構

架構部分，主要分成以下幾個部分:

1. **前端:** 使用Streamlit框架建構互動式網頁應用，提供用戶界面（UI）以便用戶輸入投資參數（如資產、投資金額、頻率和起始時間等）。前端負責將用戶的輸入參數發送到後端進行處理，並接收後端返回的計算結果和圖表數據，然後在頁面上進行展示。
2. **後端:** 使用python作為後端開發語言，進行主要的邏輯計算和數據處理工作。後端負責接收前端發送的投資參數，調用數據獲取模塊獲取歷史數據，並按照定期定額投資策略的計算邏輯進行投資模擬計算。計算結果包括每個資產的累積投資價值、回報率和購買細節等，後端將這些結果以結構化數據的形式返回給前端。
3. **Nginx反向代理:** 使用Nginx作為反向代理服務器，將HTTP請求轉發到後端的Streamlit應用。提供負載均衡和安全性功能，確保系統的穩定運行。

圖1、架構圖

# 程式架構

本次專案程式包含五個Python檔案，分別為:

1. ***DCA\_Dcashboard.py***

|  |  |
| --- | --- |
| ***asset\_map*** | 字典，用於映射資產名稱和其相應的股票代碼 |
| ***time\_map*** | 字典，用於映射時間範圍名稱和其相應的起始日期 |
| ***asset*** | 變數，用於存儲用戶選擇的投資資產名稱 |
| ***amount*** | 變數，用於存儲用戶輸入的每次投資金額 |
| ***frequency*** | 變數，用於存儲用戶選擇的投資頻率（如每週、每兩週、每月或每年） |
| ***startFrom*** | 變數，用於存儲用戶選擇的投資開始時間 |
| ***compareWith*** | 列表，用於存儲用戶選擇的用來比較的其他資產名稱 |
| ***calculateBtn*** | 用於觸發投資計算 |
| ***\_\_init\_\_()*** | Constructor，用於初始化class變數 |
| ***display\_settings()*** | 顯示頁面左側輸入框 |
| ***calculate()*** | 當使用者按下計算按鈕後，執行計算以及繪製圖表的邏輯 |
| ***display()*** | 顯示頁面基本元素 |

1. ***fetchPrice.py***

|  |  |
| --- | --- |
| ***fetchPrice(ticker\_symbol, days)*** | 傳入股票 / 資產代碼以及天數，回傳該股票 / 資產的價格二維陣列 |

1. ***config.py***

|  |  |
| --- | --- |
| ***asset\_map*** | 字典，用於映射資產名稱和其相應的股票代碼(預設11種) |
| ***time\_map*** | 字典，用於映射時間範圍名稱和其相應的起始日期(預設7種起始時間) |

1. ***essential.py***

|  |  |
| --- | --- |
| ***displayEssential()*** | 顯示頁面基本元素(footer) |

1. ***info.py***

顯示info頁面的簡介內容。

# 數據處理

1. **數據獲取:** 使用yfinance庫來獲取所選資產的歷史價格數據。這些數據包含資產的每日收盤價、開盤價、最高價、最低價和交易量等。數據獲取函數在*fetchPrice.py*程式中，該函數接受資產代碼（ticker symbol）和需要獲取的天數，返回包含歷史數據 (Pandas DataFrame)。

***def fetchPrice(ticker\_symbol, days):***

***end\_date = datetime.now()***

***start\_date = end\_date - timedelta(days=days)***

***data = yf.download(ticker\_symbol, start=start\_date, end=end\_date)***

***return data***

1. **生成購買日期:** 根據用戶選擇的投資頻率（如每週、每兩週、每月或每年），生成一系列的購買日期。如果購買日期不在交易日，則使用最近的前一個交易日價格。利用Pandas的日期範圍生成函數來生成這些購買日期，並確保每個購買日期對應一個交易日價格。
2. **計算:** 根據每次投資的固定金額，計算在每個購買日期購買的資產單位數量。這些單位數量為用戶每次投入的金額除以當天的資產價格。將每個購買日期的資產單位累積起來，表示總的投資數量。這可以幫助計算在每個時間點的總投資價值。根據總投資價值和累積投資金額計算投資回報率。

# 前端設計、圖表展示

圖表展示部分是本應用的關鍵部分。通過直觀的圖表，用戶可以清晰地看到其投資策略的效果和不同資產之間的比較。

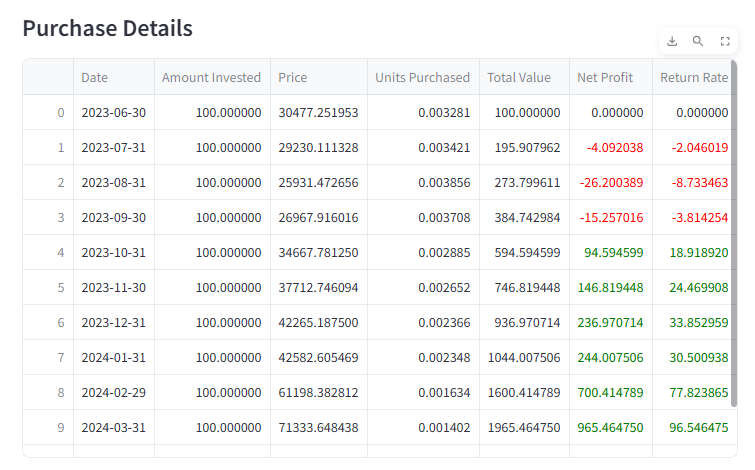
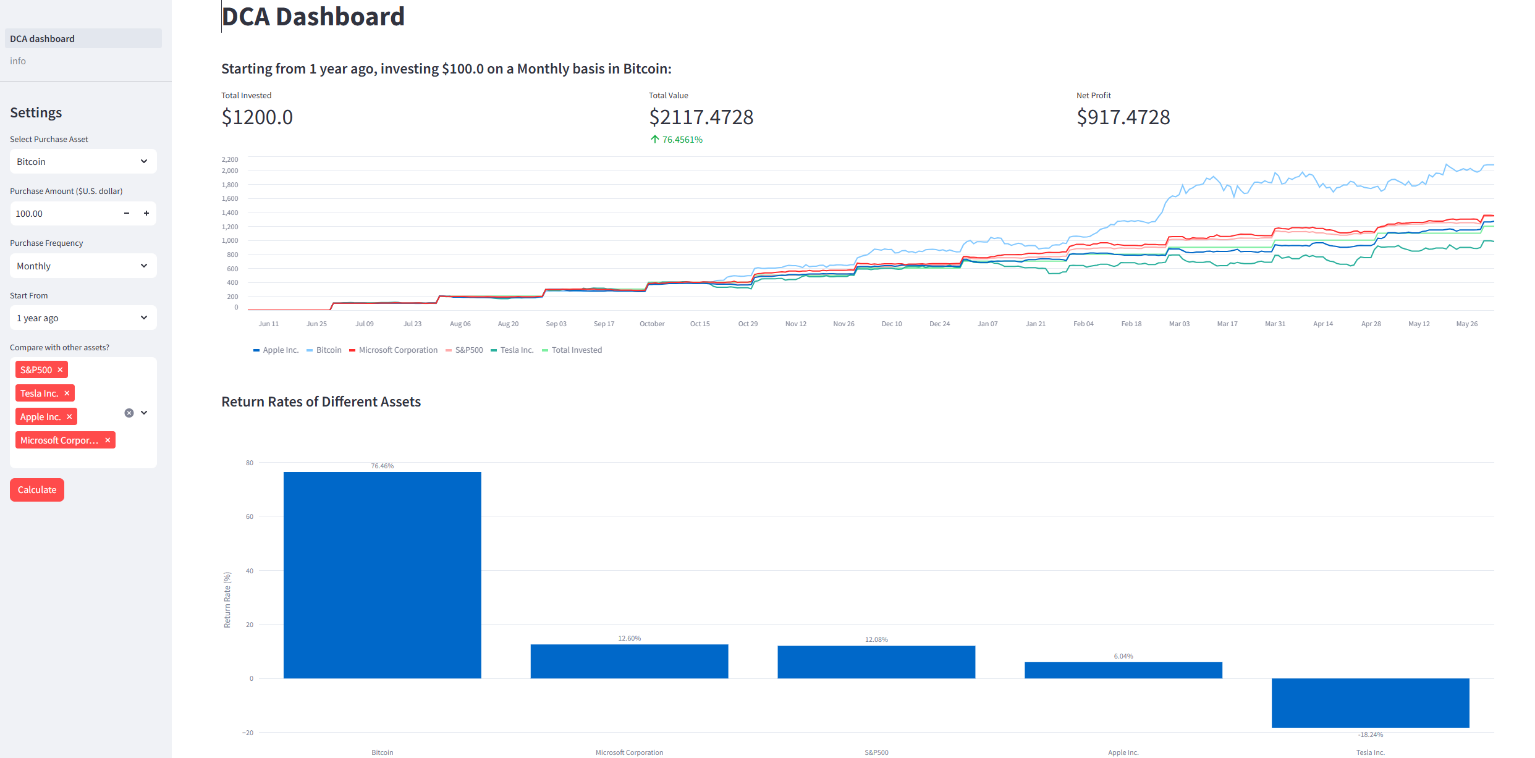
1. **輸入介面:** 用戶可以在左側的設定面板中選擇購買資產、投資金額、投資頻率和開始時間，並可以選擇與其他資產進行比較。使用Streamlit框架提供的選擇框、輸入框和按鈕，保證用戶可以方便地進行各種操作。
2. **回報率呈現:** 在用戶提交參數後，系統會計算總投資金額、總投資價值和凈利潤，並以簡潔的方式顯示在頁面頂部。回報率呈現部分包括總投資金額、總投資價值和凈利潤三個指標，分別顯示用戶的投資總額、當前投資價值和總凈利潤。這個圖表使用Streamlit的*metric*函數展示這些數據。
3. **投資價值變化圖:** 繪製用戶選定資產的累積投資價值變化圖。圖表顯示從投資起始日期到當前日期的投資價值隨時間的變化。在圖表中，同時展示總投資金額和不同資產的價值變化，幫助用戶直觀比較不同投資策略的效果。這個圖表使用Pandas和Plotly庫進行數據處理和可視化。Pandas負責處理和整理數據，而Plotly負責生成交互式圖表。再將圖表嵌入在Streamlit網頁中展示。
4. **回報率比較表:** 繪製不同資產的回報率柱狀圖，展示用戶選定的多個資產在相同投資期間的回報率。在柱狀圖中，顯示每個資產的回報率，並標註具體百分比值，幫助用戶進行回報率的直觀比較。
5. **購買時程表:** 顯示每個購買日期的詳細信息，包括投資金額、購買價格、購買單位數量、總投資價值、凈利潤和回報率。購買細節表使用Streamlit的數據框顯示功能，並通過條件格式高亮顯示凈利潤和回報率，正收益顯示為綠色，負收益顯示為紅色。使用者也可以將其數據匯出成excel表格。

圖2、前端介面(對應第五項)

圖2、前端介面(對應前四項)



# 開發模式

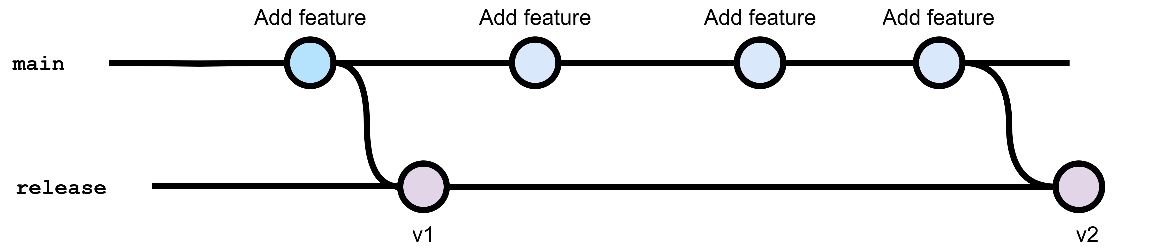
這個專題的開發過程，主要就只有使用git做版本控制的管理，git flow設計上，分為兩條分支: main以及release，main是作為開發用的分支(雖然理論上應該再開一個dev分支)，每次在本地端開發新的功能commit後，push到main分支，當完成到一定程度後，會merge到release分支，伺服器端對應到release分支，但這部分沒有做CI/CD，所以每次更新後，需要ssh進伺服器pull新的版本下來。(圖2)

圖3、git flow

# 項目部署

由於Streamlit開發的網站並不是靜態的，本質上就是一個Python的程式，需要透過與使用者互動將數據變化呈現在網頁上，當Streamlit應用啟動時，會啟動一個本地的Web伺服器來提供服務。

所以本次專題，我將網站架設在自己個人的小伺服器上，將release分支的程式pull下來，使用screen讓streamlit應用在伺服器端持續運行，並設置Nginx反向代理將http請求轉發到streamlit應用的port。(圖3)

圖4、伺服器端系統示例圖

# 生成式語言模型的幫助

在這次的專案中，我大量使用生成語言模型作為我的助手，對我來說幫助十分顯著。在專案的初期，對於整體的架構尚不清晰，對於我要使用的技術框架也一無所知。然而，我開始下prompt和classroom gpt進行溝通，我發現我能夠快速地獲得大量的幫助。這些工具幫助我理解並掌握了專案架構，也幫助我熟悉需要使用的技術框架，提高了我的開發效率，並減少我在探索和研究上的時間成本。我不需要花費大量的時間去尋找和學習新的技術，而是可以專注於實現我的專案目標。 此外，這些工具還提供我在程式撰寫過程中遇到的問題的解決方案，提高了我的效率。總之，透過使用生成語言模型作為我的助手，我能夠更有效率地完成我的專案，並且在過程中獲得大量的學習和成長。

# 結論

通過這次專題，我學到了許多新技術，也包含Python在數據處理和可視化方面的強大之處。我利用Python中的多個模組，包括yfinance、Pandas、Plotly和Streamlit，來實現自動化投資分析和可視化的儀表板。這個過程中，我學會了如何獲取和處理金融數據、如何進行回報率的計算，以及如何通過圖表將數據直觀地展示給用戶。

我也將這個期末專案部署在自己的伺服器上，連結為[dca.jinlk.dev](https://dca.jinlk.dev)。網站提供了一個方便的界面，讓用戶可以輸入自己的投資參數，並查看不同資產的投資表現。

在專題的開發過程中，生成式語言模型對我的幫助非常大。特別是在我不熟悉Python數據處理或Streamlit語法的情況下，通過classroom gpt，我能夠快速獲取所需的知識，並應用到專題中去。這不僅節省了大量的時間，也提高了開發效率。無論是在數據處理的邏輯設計上，還是在前端框架的選擇和實現上，生成式語言模型都提供了有力的支持。

總之，這次專題不僅提高了我的技術能力，讓我更深入地理解了定期定額投資策略的實際應用，更重要的是，我學習到python作為程式語言，透過各種開源模組能做到的事情數不勝數。完成這次期末專案後，未來有機會或有使用者反饋的話，我也會繼續增進功能，完善使用者體驗!