

國立高雄大學

課程：基礎程式設計

112學年 第1學期 期末議題專題報告

**題目：B.C. AI 投資大師**

|  |  |  |  |
| --- | --- | --- | --- |
| 系所：  姓名：  學號： | 系所： 資工系  姓名： 林彧頎  學號：A1105505  系所： 生科系  姓名：黃榛柔  學號：A1106135 | 系所： 金管系  姓名：彭一珊  學號：A1103239  系所： 財法系  姓名：郭珈妤  學號：A1092301 | 系所：  姓名：  學號： |

指導教授：曾智義 教授

中　華　民　國 112 年 12月 26 日

目錄

壹、議題與問題說明……………………………………………………… 1

貳、課堂所學及課外自學運用說明……………………………………… 5

參、Python Coding………………………………………………………… 15

肆、執行結果……………………………………………………………… 24

伍、參考資料……………………………………………………………… 25

陸、心　　得……………………………………………………………… 26

**壹、議題與問題說明**

一、議題動機：

在當今快速變化的金融市場中，投資者必須透過大量的金融數據才能對股市進行判斷，而這不僅耗時也費力，另一方面，股市的波動性受多種因素的影響，其中一個關鍵的影響因素是新聞事件。新聞標題有著無可否認的影響力，它們可以觸發市場情緒的劇烈波動，對股市產生重大影響。然而人類雖然可以對新聞標題有一定程度上的敏感性，但如何準確評估仍是一大挑戰。

因此我們想採用Bing Chat生成式AI，以建立一個可以預測新聞標題對隔日股市情況的工具，透過自動化的新聞分析，計算獲利期望值，以提高投資決策的準確性。

而我們的專題也符合了以下幾個SDGS發展目標：

* SDG 12 -負責任消費與生產： 透過生程式AI預測股市，我們能夠更有效地配置資源，降低投資風險，推動金融體系朝向更加穩健可持續的方向發展，體現了對負責任消費與生產的支持。
* SDG 9 -產業、創新、基礎建設： 生程式AI的應用加速了金融科技的發展，促進了投資環境的創新，同時也為更多人提供了參與金融市場的機會。
* SDG10-減少國內及國家間的不平等: 股市波動的不確定性可能導致資本的不均分配，進而加劇社會的不平等現象。透過生程式AI的精準預測，我們可以更有效地避免不必要的不確定性，緩解貧富差距，推動資源的更加均衡分配，實現更加公正的經濟發展。

二、問題構想：

承上述議題動機，我們的系統主要展示以下幾種功能:

1. Research

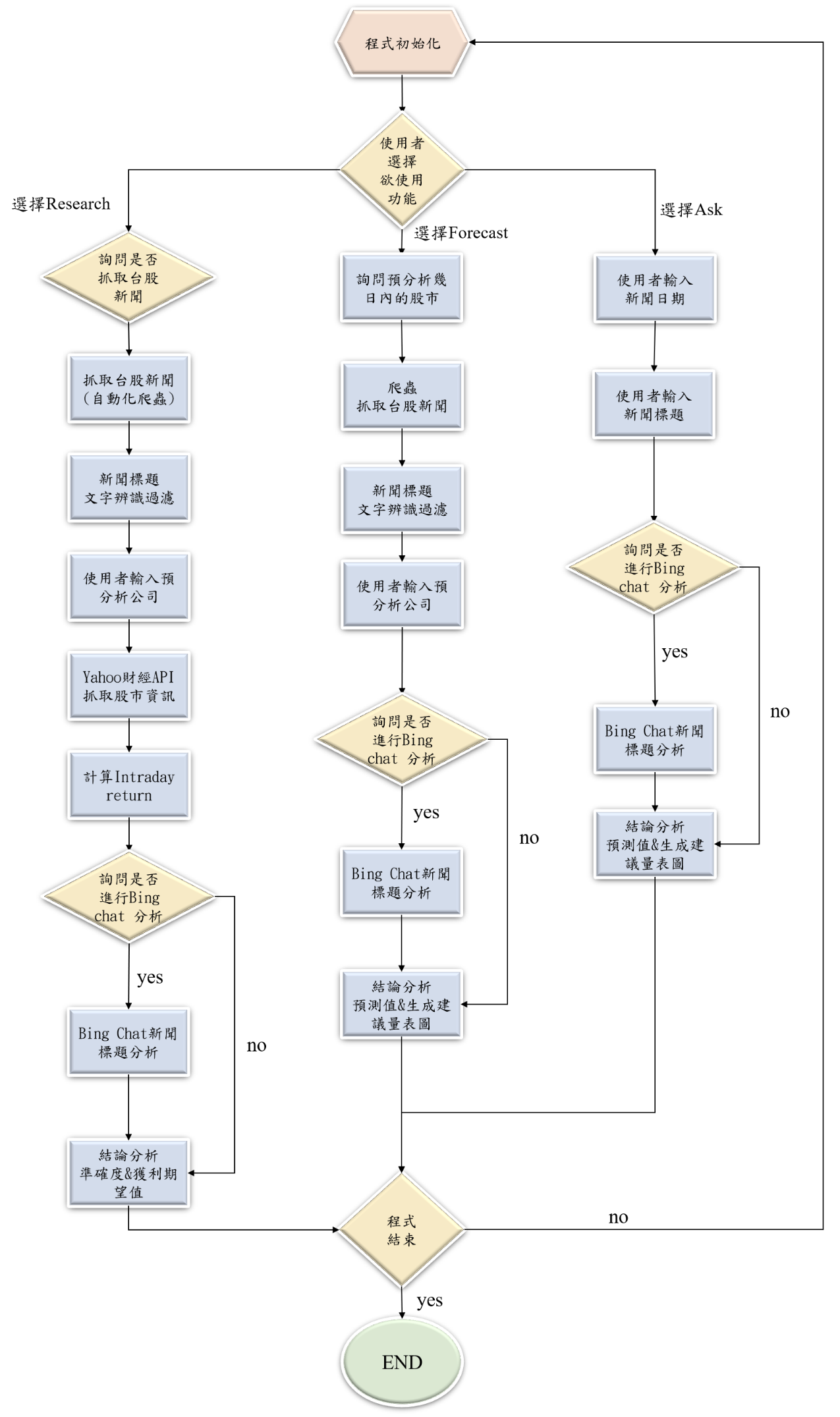
這項功能旨在幫助使用者了解本工具的可行性、幫助研究人員更好的進行研究，與幫助未來prompt的精進與更好的模型訓練。其中，會運用到爬蟲與透過生成式AI分析等功能，希望可提升預測結果的準確度，與高收益的期望值，讓消費者實現可觀的收益｡

1. Forecast

這項功能旨在協助使用者快速獲取網路上的股票新聞標題｡其中，使用者可以輸入欲分析的新聞標題以及指定爬取的日期範圍，以進行自動化爬蟲、新聞標題的細致過濾等功能，並透過Bing Chat進行新聞標題分析，系統生成預測值，希望能為使用者提供對股市趨勢的快速預測。

1. Ask

這項功能旨在提供使用者更彈性的使用方式，允許根據使用者指定的新聞標題進行股市預測。使用者可以輸入單個或多個新聞標題，同時選擇相應的日期。透過Bing Chat進行新聞標題分析，最終呈現預測結果，希望使其能夠根據特定新聞事件做出更靈活的投資分析。

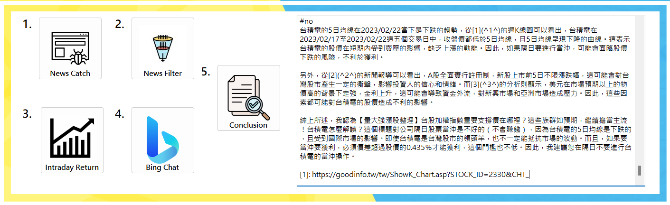
三、方案建立：

根據上述議題，我們希望能夠為金融市場提供創新的想法，於是利用自動化爬蟲抓取財經新聞，結合Bing chat 生程式AI人工智慧進行情感分析，評估市場情緒，研究設計出一個快速預測股市趨勢的系統工具，本系統功能有三項1. Research(研究過程) 2.Forecast(自動化爬蟲新聞預測股市) 3.Ask(使用者指定新聞預測股市)，每一項皆有不同功能，提供使用者更加多元的選擇。系統流程為程式初始化後進入系統主介面(圖1)，由使用者選擇進入Reasearch[1]、Forecast[2]或Ask[3]功能，按照使用者之選擇導向其功能操作。



圖表 1 系統功能主介面

[1]如選擇Research功能，將導向Research介面(圖2)，進行以下步驟：



圖表 2 Research功能介面

A. 詢問是否抓取台股新聞。因爬蟲需要一點時間，我們有提供使用者內建資料可直接使用不用重新爬蟲。如，是，繼續步驟，否，則跳至D

B. 台股新聞自動化爬蟲

C. 新聞標題分析過濾

D. 使用者輸入預分析公司

E. 進行Intraday return抓取(2022~2023上半年)

此步驟我們使用Yahoo財經API獲得的每日股市資訊，利用開盤價和收盤價計算日內回報率，作為預測目標變數。

F. 詢問是否進行Bing chat 分析(此步驟也有提供內建資料)，如是，繼續步驟，否，則跳至H

G. Bing Chat新聞標題分析。

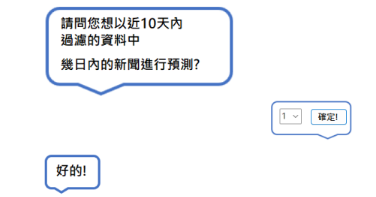
此步驟將獲取的資料提供給Bing Chat分析，其會根據新聞標題的內容，給出一個介於-1到1之間的情感分數，其中-1表示最負面，1表示最正面，0表示中立。我們將其回覆轉換成「Bing Chat分數」，Bing Chat回覆內容中的三種指標「#yes」、「#no」、「#unknown」，分別對應到「Bing Chat分數」中的1分、-1分和0分。

H. 結論分析獲得準確度&獲利期望值。

此步驟我們將「Bing Chat平均分數」與隔日的日內回報率進行相關性分析獲得準確度和獲利期望值。

此功能將我們的研究流程應用在應用程式中，為給予研究人員更精準的數據幫助進行研究，同時也助於未來prompt設計的精進與更好的模型訓練。

[2]如選擇Forecast功能，將導向Forecast介面(圖3)，此功能為幫助一般使用者最快的抓到網路上的股票新聞標題，並依照使用者指定時間範圍進行股票預測，分成以下步驟：



圖表3 Forecast功能介面

A. 詢問預分析幾日內的股市

此步驟由使用者輸入他想於程式自動化網路爬蟲的新聞標題10日內中幾日資料進行新聞預測股市分析。

B. 台股新聞自動化爬蟲

C. 新聞標題分析過濾

D. 使用者輸入預分析公司

E. 詢問是否進行Bing chat 分析(此步驟有提供內建資料)，如是，繼續步驟，否，則跳至G

F. Bing Chat新聞標題分析

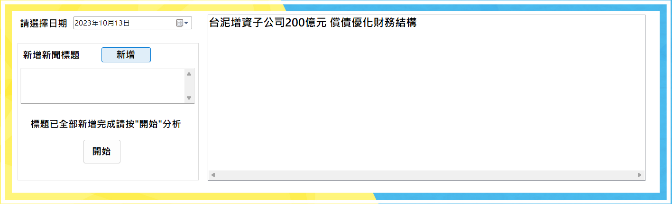
G. 獲得結論分析結果，包含預測值&生成建議量表圖

此步驟將Bing chat分析之「情感分數」轉化成預測值，根據預測值生成建議量表圖，建議量表圖以天氣的形式呈現，分別是晴天，表示股市預測情緒非常正向，可能是一個良好的買進時機；晴時有雲，表示股市預測情緒較為中立，存在一些不確定性，可能是一個觀望的時機；陰天(雲)，表示股市預測情緒有一些負面，可能存在風險；陰天有雨，表示股市預測情緒較為負面，可能不是一個良好的買進時機；雷雨，表示股市預測情緒極為負面，風險極高，投資人應該謹慎避險，這五個指標圖表示不同的建議。



圖表4 Forecast 功能流程

[3]如選擇Ask功能，將導向Ask介面(圖5)，此功能為提供一般使用者更彈性的使用方法，使用者可自由選擇想分析的新聞標題輸入進行分析，更加靈活的運用，分成以下步驟：



圖表5 Ask功能介面

A. 使用者輸入單(多)個新聞標題與選擇日期

B. 詢問是否進行Bing chat 分析(此步驟有提供內建資料)，如是，繼續步驟，否，則跳至D

C. Bing Chat新聞標題分析

D. 獲得結論分析結果，包含預測值&生成建議量表圖

****

圖表6 Ask功能流程

使用者選擇之功能執行完畢後，將跳回系統功能介面(圖1)，重新選擇欲使用之功能，或執行結束程式。

**貳、課堂所學及課外自學運用說明**

**一、第二章之運用**

1. **變數、運算、字串、比較**

我們在本專題中有許多地方都有使用到**變數、運算、字串**與比較的使用，我們的**變數**可能是存取**字串**也可能是存取值等等…，除此之外，於運算中我們也運用到許多的應用中，例如:Bing Chat分析字串、keyword、分數等等...，而這些**變數、運算、字串**的運用，可以幫助我們更好的重複利用會不斷出現的**變數**值，或者是可以透過**運算與比較**，幫助我們有更好的數值分析等等…。

(以下是其中的一個範例程式)

1. # 創建一個空的DataFrame，用於存放每個月份的結果
2. all\_results = pd.DataFrame()
3. keyword=sys.argv[1]
4. j=2
5. # 處理每個月份的原始檔案
6. for month in range(1, 13):
7. file\_name = f'{keyword}ans/{keyword}ans\_{month}.csv'
8. try:
9. df = pd.read\_csv(file\_name)
10. # 將 Y/N 轉換為數字
11. df[' Y/N'] = df[' Y/N'].map({'#yes': 1, '#no': -1, '#unknown': 0})
12. # 進行分組計算並計算平均分數
13. grouped = df.groupby('date')[' Y/N'].mean().reset\_index()
14. grouped.rename(columns={' Y/N': 'point'}, inplace=True)
15. # 計算小數點後第二位的平均分數
16. grouped['point'] = grouped['point'].apply(lambda x: round(x, 2))
17. # 將處理後的結果合併到 all\_results DataFrame 中
18. all\_results = pd.concat([all\_results, grouped])

1.**串列型別(List Type)**

我們在程式中，使用**List**型別來儲存我們的多個關鍵字、數據集合等等，我們主要將List與Dictionary做一個統整的實用，過程中也有利用取list的第幾個數據進行字串比對，數值運算等等…，這些方法可以幫助我們更好的統整資料與數據，並可以於後續的條件式判斷將我們需要的值取出，更可以說是我們本次專題中不可或缺的一部份，因為如果沒有list，則我們的資料會很難處理，可能需要多個變數等等…。

(以下是其中一個程式範例)

1. # 從命令列參數取得股票代號
2. stock\_symbol = sys.argv[1]
3. # Download intraday data for 2330.TW
4. data = yf.download(stock\_symbol, start="2022-01-01", end="2023-06-30", interval="1d")
5. # Calculate intraday returns
6. data["Intraday\_Return"] = (data["Close"] - data["Open"]) / data["Open"] \* 100
7. # Save the intraday return data to a CSV file
8. data.to\_csv(f'Intraday return/{stock\_symbol[0]}{stock\_symbol[1]}{stock\_symbol[2]}{stock\_symbol[3]}\_Intraday\_Return.csv', encoding='utf-8')

2.**字典形別(Dictionary Type)**

我們在程式中，使用Dict型別來儲存我們的多個關鍵字、數據集合等等…，相較於list他可以說主要負責keyword的部分，幫助我們可以快速找到對應的資料，我們主要將List與Dictionary做一個統整的實用，例如於系統中，我們將爬蟲的部分針對list、dictionary將日期和標題對應成字典，然後添加到列表中，過程中也有利用先取Dictionay對應的關鍵字(key方法)，找對應的值(value方法)，並與list搭配，將dictionay中list的第幾個數據進行字串比對，數值運算等等…，這些方法可以幫助我們更好的統整資料與數據，並可以於後續的條件式判斷將我們需要的值取出，更可以說是我們本次專題中不可或缺的一部份，因為如果沒有dict，則我們的資料會很難處理，可能需要多個變數等等…。

(以下是其中一個程式範例)

1. # 將日期和標題對應成字典，然後添加到列表中
2. for date\_element, title\_element in zip(date\_elements, title\_elements):
3. # 獲取現在的日期
4. current\_date = datetime.datetime.today().strftime('%m/%d')
5. try:
6. # 嘗試解析時間，並將其轉換成現在的日期格式
7. time\_obj = datetime.datetime.strptime(date\_element.text, '%H:%M')
8. data = {
9. 'date': current\_date,
10. 'title': title\_element.text
11. }
12. data\_list.append(data)
13. except ValueError:
14. # 如果解析失敗，保留原樣
15. data = {
16. 'date': date\_element.text,
17. 'title': title\_element.text
18. }
19. data\_list.append(data)
20. driver.close()
21. # 輸出所有日期和標題對應的資料
22. for data in data\_list:
23. print(data['date'], data['title'])

**二、第三章之運用**

1.**條件分支(if…elif…else…)**

我們在程式中運用條件分支來達成判斷的功能。例如，在程式結束前與使用者互動，或是讓使用者抉擇要執行的操作時：(以下為其中一個示範)

1. #顯示對應的閾閥圖
2. if average\_point\_percent>50:
3. image=Image.open('股市指標圖/正正.png')
4. image.show()
5. print('股市指標圖/正正.png')
6. elif average\_point\_percent<50 and average\_point\_percent>0:
7. image=Image.open('股市指標圖/正.png')
8. image.show()
9. print('股市指標圖/正.png')
10. elif average\_point\_percent==0:
11. image=Image.open('股市指標圖/普通.png')
12. image.show()
13. print('股市指標圖/普通.png')
14. elif average\_point\_percent<0 and average\_point\_percent>-50:
15. image=Image.open('股市指標圖/負.png')
16. image.show()
17. print('股市指標圖/負.png')
18. elif average\_point\_percent<-50:
19. image=Image.open('股市指標圖/負負.png')
20. image.show()
21. print('股市指標圖/負負.png')

2.**For迴圈**

當我們在進行比對資料時，利用到for迴圈來完成逐筆比對(可能適用於計算、比對等等…)的功能(以下為其中一個示範)。

1. # 比較每一個日期的點數和隔日的Intraday\_Return
2. for index, row in point\_df.iterrows():
3. date = row['date']
4. point = row['point']

**3.While迴圈**

當我們需要不斷執行某事件直到可以做出正確行位或是需要不斷執行某操作時，我們就會使用While: (以下為其中一個示範)

1. # 找到隔日的日期
2. next\_date = date + pd.DateOffset(days=1)
3. while next\_date.strftime('%Y-%m-%d') in holidays:
4. next\_date = next\_date + pd.DateOffset(days=1)

4.**Def 函式**

當我們想將功能劃分開來我們就會使用Def函式，以下是定義的方法:

1. #執行的操作
2. #ex:
3. async def test\_ask() -> None:
4. cookies = json.loads(open(f'{sys.argv[1]}', encoding="utf-8").read())
5. bot = await Chatbot.create(cookies=cookies)
6. response = await bot.ask(
7. prompt=f'(請用中文回答我){sys.argv[2]}',
8. conversation\_style=ConversationStyle.creative,
9. simplify\_response=True,
10. )
11. await bot.close()
12. print(json.dumps(response['text'], ensure\_ascii=False, indent=2)) # 加入 ensure\_ascii=False
13. assert response

**三、第五章之運用**

1.**Access File**

我們在程式中運用讀檔、寫檔等…功能進行資料處理。

1. import csv
2. import sys
3. csv\_file\_path = ["forecast/news\_now.csv"]
4. # 從命令列參數取得關鍵字
5. keyword = sys.argv[1]
6. output\_csv\_file\_path = "forecast/news\_now\_filter.csv"
7. i=0
8. # 儲存含有關鍵字的資料的列表
9. #如果要個別月要放裡面
10. filtered\_data = []
11. for f in csv\_file\_path:
12. # 儲存含有關鍵字的資料的列表
13. #如果要個別月要放裡面
14. #filtered\_data = []
15. # 使用 UTF-8 編碼讀取 CSV 檔案
16. with open(f, mode='r', encoding='utf-8-sig') as csvfile:
17. reader = csv.DictReader(csvfile)
19. for row in reader:
20. if keyword in row['title']:
21. filtered\_data.append({'date': row['date'], 'title': row['title']})
22. # 使用 UTF-8 編碼寫入新的 CSV 檔案
23. #將all改成output\_csv\_file\_path[i]就可以按月輸出.csv
24. with open(output\_csv\_file\_path, mode='w', encoding='utf-8', newline='') as csvfile:
25. # 定義欄位
26. fieldnames = ['date', 'title']
27. # 將 dictionary 寫入 CSV 檔
28. writer = csv.DictWriter(csvfile, fieldnames=fieldnames)
29. # 寫入第一列的欄位名稱
30. writer.writeheader()
31. # 寫入篩選後的資料
32. for data in filtered\_data:
33. writer.writerow({'date': data['date'], 'title': data['title']})
34. i+=1

**四、第六章之運用**

1.**圖片(pillow)**

我們在程式中運用pillow的函式庫進行閾閥值圖的呼叫，並顯示。

1. from PIL import Image, ImageFilter, ImageEnhance
2. if average\_point\_percent>50:
3. image=Image.open('股市指標圖/正正.png')
4. image.show()
5. print('股市指標圖/正正.png')
6. elif average\_point\_percent<50 and average\_point\_percent>0:
7. image=Image.open('股市指標圖/正.png')
8. image.show()
9. print('股市指標圖/正.png')
10. elif average\_point\_percent==0:
11. image=Image.open('股市指標圖/普通.png')
12. image.show()
13. print('股市指標圖/普通.png')
14. elif average\_point\_percent<0 and average\_point\_percent>-50:
15. image=Image.open('股市指標圖/負.png')
16. image.show()
17. print('股市指標圖/負.png')
18. elif average\_point\_percent<-50:
19. image=Image.open('股市指標圖/負負.png')
20. image.show()
21. print('股市指標圖/負負.png')

2.**爬蟲(Request、Beautiful Soup…)**

當我們在抓取分析研究的資料，或是幫助使用者快速抓取網路中的股市新聞時，我們就會透過爬蟲的方法進行抓取，值得一提的是由於我們使用的鉅亨網會比較不好透過Beautiful Soup抓取好幾年前到現在的新聞，所以我們後續有額外使用Selenium動態網路爬蟲的方法(以下是一個爬蟲的片段)。

1. data\_list = []
2. data={}#使用大刮號{}創建字典
3. driver = webdriver.Edge(service=Service(EdgeChromiumDriverManager().install()))
4. url='https://news.cnyes.com/news/cat/tw\_stock\_news?exp=a'
5. driver.get(url)
6. #模擬向下捲動，以確保載入更多內容
7. scrolls = 10
8. for \_ in range(scrolls):
9. driver.find\_element(By.TAG\_NAME,'body').send\_keys(Keys.END)
10. time.sleep(2) # 等待頁面載入
12. #找到當前頁面
13. page\_content=driver.page\_source
14. # 使用 BeautifulSoup 解析頁面内容
15. soup = BeautifulSoup(page\_content, "html.parser")
16. # 抓取日期元素
17. date\_elements = soup.select("div.\_67tN.theme-meta time")
18. # 抓取標題元素
19. title\_elements = soup.select("div.\_1xc2 h3")
20. # 將日期和標題對應成字典，然後添加到列表中
21. for date\_element, title\_element in zip(date\_elements, title\_elements):
22. # 獲取現在的日期
23. current\_date = datetime.datetime.today().strftime('%m/%d')
24. try:
25. # 嘗試解析時間，並將其轉換成現在的日期格式
26. time\_obj = datetime.datetime.strptime(date\_element.text, '%H:%M')
27. data = {
28. 'date': current\_date,
29. 'title': title\_element.text
30. }
31. data\_list.append(data)
32. except ValueError:
33. # 如果解析失敗，保留原樣
34. data = {
35. 'date': date\_element.text,
36. 'title': title\_element.text
37. }
38. data\_list.append(data)
39. driver.close()

**五、課外自學**

**1.Selenium 動態網路爬蟲**

利用Selenium 動態網路爬蟲可以幫助我們更方便的抓取過去到現在的資料，因為每一個網站不一定都能簡單的換網頁等等…或是有可能會有太常且太快的訪問而被檔，因此最好的方法就是搭配動態網路爬蟲與request、Beautiful Soup進行資料爬蟲。(以下是一個片段的範例)

1. import requests
2. from selenium import webdriver
3. from bs4 import BeautifulSoup
4. from selenium.webdriver.common.keys import Keys
5. from selenium.common.exceptions import TimeoutException
6. from selenium.common.exceptions import NoSuchElementException
7. from selenium.webdriver.common.action\_chains import ActionChains
8. from selenium.webdriver.common.by import By
9. from selenium.webdriver.support.ui import Select
10. from selenium.webdriver.support.ui import WebDriverWait
11. from selenium.webdriver.support import expected\_conditions as EC
12. #edge
13. from webdriver\_manager.microsoft import EdgeChromiumDriverManager #edge時(預設瀏覽器)
14. from selenium.webdriver.edge.service import Service
15. data\_list = []
16. data={}#使用大刮號{}創建字典
17. driver = webdriver.Edge(service=Service(EdgeChromiumDriverManager().install()))
18. url='https://news.cnyes.com/news/cat/tw\_stock\_news?exp=a'
19. driver.get(url)
20. #模擬向下捲動，以確保載入更多內容
21. scrolls = 10
22. for \_ in range(scrolls):
23. driver.find\_element(By.TAG\_NAME,'body').send\_keys(Keys.END)
24. time.sleep(2) # 等待頁面載入
26. #找到當前頁面
27. page\_content=driver.page\_source
28. # 使用 BeautifulSoup 解析頁面内容
29. soup = BeautifulSoup(page\_content, "html.parser")
30. # 抓取日期元素
31. date\_elements = soup.select("div.\_67tN.theme-meta time")
32. # 抓取標題元素
33. title\_elements = soup.select("div.\_1xc2 h3")
34. # 將日期和標題對應成字典，然後添加到列表中
35. for date\_element, title\_element in zip(date\_elements, title\_elements):
36. # 獲取現在的日期
37. current\_date = datetime.datetime.today().strftime('%m/%d')
38. try:
39. # 嘗試解析時間，並將其轉換成現在的日期格式
40. time\_obj = datetime.datetime.strptime(date\_element.text, '%H:%M')
41. data = {
42. 'date': current\_date,
43. 'title': title\_element.text
44. }
45. data\_list.append(data)
46. except ValueError:
47. # 如果解析失敗，保留原樣
48. data = {
49. 'date': date\_element.text,
50. 'title': title\_element.text
51. }
52. data\_list.append(data)
53. driver.close()

**2.** **csv、 datetime、time、pandas、yfinance、sys**

以上這幾種方法，分別對應csv:處理csv資料、datetime:可以幫我抓取本機時間、time:可以幫我執行暫停操作、pandas:可以幫我將大量資料轉成dataframe格式、yfinance:透過連API的方法抓取yahoo finance股市資訊、sys: 可以幫我執行參數的操作，以下為範例程式碼:

1. # Save the intraday return data to a CSV file
2. data.to\_csv(f'Intraday return/{stock\_symbol[0]}{stock\_symbol[1]}{stock\_symbol[2]}{stock\_symbol[3]}\_Intraday\_Return.csv', encoding='utf-8')
3. # Download intraday data for 2330.TW(yfinancec、pandas)
4. data = yf.download(stock\_symbol, start="2022-01-01", end="2023-06-30", interval="1d")
5. stock\_symbol = sys.argv[1]# 從命令列參數取得股票代號
6. time.sleep(1)#time(暫停時間1秒)
7. os.system("cls")#(clear 視窗)
8. now = datetime.datetime.now()#(顯示現在時間)

**3.** **Bing Chat API python**

我們結合老師於開課時介紹的生成式AI，Bing Chat幫助我們分析數據，我們透過連結Bing Chat API，並搭配json抓取cookie的方法使用。

1. import asyncio
2. import json
3. from pathlib import Path
4. from re\_edge\_gpt import Chatbot
5. from re\_edge\_gpt import ConversationStyle
6. import sys
7. async def test\_ask() -> None:
8. cookies = json.loads(open(f'{sys.argv[1]}', encoding="utf-8").read())
9. bot = await Chatbot.create(cookies=cookies)
10. response = await bot.ask(
11. prompt=f'(請用中文回答我){sys.argv[2]}',
12. conversation\_style=ConversationStyle.creative,
13. simplify\_response=True,
14. )
15. await bot.close()
16. print(json.dumps(response['text'], ensure\_ascii=False, indent=2)) # 加入 ensure\_ascii=False
17. assert response
18. if \_\_name\_\_ == "\_\_main\_\_":
19. try:
20. loop = asyncio.get\_running\_loop()
21. except RuntimeError:
22. loop = asyncio.get\_event\_loop()
23. loop.run\_until\_complete(test\_ask())

**4.** **C# Windows Form APP製作系統介面**

我們使用C#製作系統介面，負責美觀的呈現。



**參、Python Coding**

一、本程式GitHub

[**https://github.com/Yukimagi/FinanceChatAI\_with\_threshold\_picture**](https://github.com/Yukimagi/FinanceChatAI_with_threshold_picture)

一、函式庫匯入

我們 import 許多需要的功能，例如csv、PIL、requests、Beautiful Soup等等...,這些功能將幫助我們更好的調用本系統需要執行的資料讀取、爬蟲、圖片、運算、API抓取資訊等等…，當然這些功能都會需要對命令提示字元進行相關的匯入，才能更好的使用這些功能，以下為簡單的一個import範例，之後的程式碼將對應功能進行介紹，因此也會有對應的import。

\*參考資料[1~4]

1. Import datetime#(顯示現在時間)

2. import time#time(暫停時間1秒)

二、爬蟲應用

我們在程式中主要有兩大區域會需要進行爬蟲的使用，第一個地方是Research功能，另一個地方則是Forecast功能，不同的地方是，Research功能所使用的爬蟲比較複雜，因為需要抓去過去幾年到現在的資訊，需要做的處理比較多，程式碼也很長(詳細程式碼可以到github news\_catch.py檔看，因為有2574行)，而Forecast的則是抓取近10日的新聞，因此相對處理簡單，因此上面這個程式碼將是Forecast爬蟲的程式碼，我們使用動態網路爬蟲Selenium與request、Beautiful Soup的方法使抓到我們要的資訊。\*參考資料[3]

1. import requests

2. from selenium import webdriver

3. from bs4 import BeautifulSoup

4. from selenium.webdriver.common.keys import Keys

5. from selenium.common.exceptions import TimeoutException

6. from selenium.common.exceptions import NoSuchElementException

7. from selenium.webdriver.common.action\_chains import ActionChains

8.

9. from selenium.webdriver.common.by import By

10. from selenium.webdriver.support.ui import Select

11. from selenium.webdriver.support.ui import WebDriverWait

12. from selenium.webdriver.support import expected\_conditions as EC

13.

14. #edge

15. from webdriver\_manager.microsoft import EdgeChromiumDriverManager #edge時(預設瀏覽器)

16. from selenium.webdriver.edge.service import Service

17.

18. import time

19. import csv

20. import datetime

21.

22. data\_list = []

23. data={}#使用大刮號{}創建字典

24. driver = webdriver.Edge(service=Service(EdgeChromiumDriverManager().install()))

25. url='https://news.cnyes.com/news/cat/tw\_stock\_news?exp=a'

26. driver.get(url)

27.

28. #模擬向下捲動，以確保載入更多內容

29. scrolls = 10

30. for \_ in range(scrolls):

31.     driver.find\_element(By.TAG\_NAME,'body').send\_keys(Keys.END)

32.     time.sleep(2)  # 等待頁面載入

33.

34. #找到當前頁面

35. page\_content=driver.page\_source

36.

37. # 使用 BeautifulSoup 解析頁面内容

38. soup = BeautifulSoup(page\_content, "html.parser")

39. # 抓取日期元素

40. date\_elements = soup.select("div.\_67tN.theme-meta time")

41.

42. # 抓取標題元素

43. title\_elements = soup.select("div.\_1xc2 h3")

44.

45. # 將日期和標題對應成字典，然後添加到列表中

46. for date\_element, title\_element in zip(date\_elements, title\_elements):

47.     # 獲取現在的日期

48.     current\_date = datetime.datetime.today().strftime('%m/%d')

49.     try:

50.         # 嘗試解析時間，並將其轉換成現在的日期格式

51.         time\_obj = datetime.datetime.strptime(date\_element.text, '%H:%M')

52.         data = {

53.             'date': current\_date,

54.             'title': title\_element.text

55.         }

56.         data\_list.append(data)

57.     except ValueError:

58.         # 如果解析失敗，保留原樣

59.         data = {

60.             'date': date\_element.text,

61.             'title': title\_element.text

62.         }

63.         data\_list.append(data)

64.

65. driver.close()

66. # 輸出所有日期和標題對應的資料

67. for data in data\_list:

68.     print(data['date'], data['title'])

69.

70. with open('forecast/news\_now.csv', mode='w', encoding='utf-8', newline='') as csvfile:

71.     # 定義欄位

72.     fieldnames = ['date', 'title']

73.

74.     # 將 dictionary 寫入 CSV 檔

75.     writer = csv.DictWriter(csvfile, fieldnames=fieldnames)

76.

77.     # 寫入第一列的欄位名稱

78.     writer.writeheader()

79.     for data in data\_list:

80.         # 寫入資料

81.         writer.writerow({'date': data['date'], 'title': data['title']})

三、文字過濾新聞標題(對應的股市公司)：

在此步驟我們會透過變數定義關鍵字，並使用字典與list相結合的方法將包含keyword的新聞標題過濾出來，並運用讀檔、寫檔的方法將閾處理和處理後的資料儲存，此步驟會被運用於Research與Forecast功能中。

1. import csv

2. import sys

3.

4. csv\_file\_path = ["forecast/news\_now.csv"]

5. # 從命令列參數取得關鍵字

6. keyword = sys.argv[1]

7. output\_csv\_file\_path = "forecast/news\_now\_filter.csv"

8.

9. i=0

10. # 儲存含有關鍵字的資料的列表

11. #如果要個別月要放裡面

12. filtered\_data = []

13.

14. for f in csv\_file\_path:

15.     # 儲存含有關鍵字的資料的列表

16.     #如果要個別月要放裡面

17.     #filtered\_data = []

18.     # 使用 UTF-8 編碼讀取 CSV 檔案

19.     with open(f, mode='r', encoding='utf-8-sig') as csvfile:

20.         reader = csv.DictReader(csvfile)

21.

22.         for row in reader:

23.             if keyword in row['title']:

24.                 filtered\_data.append({'date': row['date'], 'title': row['title']})

25.

26.     # 使用 UTF-8 編碼寫入新的 CSV 檔案

27.     #將all改成output\_csv\_file\_path[i]就可以按月輸出.csv

28.     with open(output\_csv\_file\_path, mode='w', encoding='utf-8', newline='') as csvfile:

29.         # 定義欄位

30.         fieldnames = ['date', 'title']

31.

32.         # 將 dictionary 寫入 CSV 檔

33.         writer = csv.DictWriter(csvfile, fieldnames=fieldnames)

34.

35.         # 寫入第一列的欄位名稱

36.         writer.writeheader()

37.

38.         # 寫入篩選後的資料

39.         for data in filtered\_data:

40.             writer.writerow({'date': data['date'], 'title': data['title']})

41.     i+=1

四、Yahoo Fiannce API & Intraday Return計算：

由於在Research中需要在最後結論分析中比較我們的預測分數與實際的Return值，是否有一定的相關度(準確度)，因此這步驟我們會抓取Yahoo Finance API並透過計算的方式，將整個dataframe中的資料抓取出我們要的數據並運算出Intraday return。\*參考資料[2]

1. # 第三方套件 yfinance, 可用來串接 Yahoo Finance API 下載股票的價量資訊. 而且拿到的資料就是 Pandas 的 DataFrame

2. import yfinance as yf

3. import pandas as pd

4. import csv

5. import sys

6.

7. # In[2]:

8.

9. # 從命令列參數取得股票代號

10. stock\_symbol = sys.argv[1]

11.

12. # Download intraday data for 2330.TW

13. data = yf.download(stock\_symbol, start="2022-01-01", end="2023-06-30", interval="1d")

14.

15. # In[3]:

16.

17. # Calculate intraday returns

18. data["Intraday\_Return"] = (data["Close"] - data["Open"]) / data["Open"] \* 100

19.

20. # In[4]:

21.

22. # Save the intraday return data to a CSV file

23. data.to\_csv(f'Intraday return/{stock\_symbol[0]}{stock\_symbol[1]}{stock\_symbol[2]}{stock\_symbol[3]}\_Intraday\_Return.csv', encoding='utf-8')

五、Bing Chat分析

本系統最主要的一個功能之一即為Bing Chat分析，在Bing Chat中，我主要透過抓取Bing Chat API來達到自動化的詢問大量的問答，我在使用的過程中，也遇到一些問題，使我必須要分批執行，才可以得出研究的分析結果，但資料量如果小於300則可以一次做完，這與Bing Chat的防範機制有關。\*參考資料[1]

1. import asyncio

2. import json

3. from pathlib import Path

4.

5. from re\_edge\_gpt import Chatbot

6. from re\_edge\_gpt import ConversationStyle

7. import sys

8.

9. async def test\_ask() -> None:

10.     cookies = json.loads(open(f'{sys.argv[1]}', encoding="utf-8").read())

11.     bot = await Chatbot.create(cookies=cookies)

12.     response = await bot.ask(

13.         prompt=f'(請用中文回答我){sys.argv[2]}',

14.         conversation\_style=ConversationStyle.creative,

15.         simplify\_response=True,

16.     )

17.     await bot.close()

18.     print(json.dumps(response['text'], ensure\_ascii=False, indent=2))  # 加入 ensure\_ascii=False

19.     assert response

20.

21. if \_\_name\_\_ == "\_\_main\_\_":

22.     try:

23.         loop = asyncio.get\_running\_loop()

24.     except RuntimeError:

25.         loop = asyncio.get\_event\_loop()

26.     loop.run\_until\_complete(test\_ask())

六、結論分析

我們於結論分析中，主要運用Bing Chat計算出的分數，計算它的平均值(當然Research的結論分析方法會是計算準確度與期望值，而前面架構圖已介紹過，因此這裡不再贅述)。

1. import pandas as pd

2. from PIL import Image, ImageFilter, ImageEnhance

3. # 創建一個空的DataFrame，用於存放每個月份的結果

4. all\_results = pd.DataFrame()

5.

6. file\_name = 'ask/ans\_ask.csv'

7. df = pd.read\_csv(file\_name)

8.

9. # 將 Y/N 轉換為數字

10. df[' Y/N'] = df[' Y/N'].map({'#yes': 1, '#no': -1, '#unknown': 0})

11.

12. # 進行分組計算並計算平均分數

13. grouped = df.groupby('date')[' Y/N'].mean().reset\_index()

14. grouped.rename(columns={' Y/N': 'point'}, inplace=True)

15.

16. # 計算小數點後第二位的平均分數

17. grouped['point'] = grouped['point'].apply(lambda x: round(x, 2))

18. # 將處理後的結果合併到 all\_results DataFrame 中

19. all\_results = pd.concat([all\_results, grouped])

20.

21. # 將最終結果寫入一個結果檔案

22. all\_results.to\_csv('ask/point\_ask.csv', index=False)

23.

24. # In[ ]:

25.

26. # 讀取所有結果

27. all\_results = pd.read\_csv('ask/point\_ask.csv')

28.

29. # 計算所有日期的平均分數

30. average\_point = all\_results['point'].mean()

31.

32. # 將平均分數轉換為百分比

33. average\_point\_percent = round(average\_point \* 100, 2)

34.

35. print(f'平均結論值：{average\_point\_percent}')

36.

37. if average\_point\_percent>50:

38.     image=Image.open('股市指標圖/正正.png')

39.     image.show()

40.     print('股市指標圖/正正.png')

41. elif average\_point\_percent<50 and average\_point\_percent>0:

42.     image=Image.open('股市指標圖/正.png')

43.     image.show()

44.     print('股市指標圖/正.png')

45. elif average\_point\_percent==0:

46.     image=Image.open('股市指標圖/普通.png')

47.     image.show()

48.     print('股市指標圖/普通.png')

49. elif average\_point\_percent<0 and average\_point\_percent>-50:

50.     image=Image.open('股市指標圖/負.png')

51.     image.show()

52.     print('股市指標圖/負.png')

53. elif average\_point\_percent<-50:

54.     image=Image.open('股市指標圖/負負.png')

55.     image.show()

56.     print('股市指標圖/負負.png')

七、C# Windows Form App美觀介面簡介

我們系統因為每個功能都是用一個python檔撰寫，因此為了統整所有的程式碼，我們額外用C#的Windows Form APP撰寫系統的美觀介面，因此主程式碼依舊是python，而以下示範我們是如何於C#程式碼呼叫python檔。

1. string pythonScriptPath = "BingChat.py";

2. //請找自己的python.exe路徑貼上

3. string pythonInterpreter = "C:\\Users\\user\\AppData\\Local\\Programs\\Python\\Python311\\python.exe";

4. //請寫入你的bingchat.json路徑，注意格式(python注重)

5. //D:\Downloads\FinanceChatAI - 複製\FinanceChatAI\bin\Debug\net6.0-windows

6. //string cookiepath = "D:/Downloads/FinanceChatAI - 複製/FinanceChatAI/bin/Debug/net6.0-windows/bingchat.json";

7. string cookiepath = "bingchat.json";

8.

9. var message = "請您拿" + date + "當下5日內" + keyword2 + "的均線漲跌進行分析為漲還是跌，接著對以下新聞標題與其5日均線之漲跌經過綜合分析，並請考慮到當沖要獲利的話，必須價差超過股價的 0.435％才能獲利，那麼這個標題對公司隔日股票當沖是好(會賺錢)還是壞(不會賺錢)，接著請注意回答重點是:如果隔日股票會漲(會賺錢)請只回答#yes，而如果隔日股票會跌(不會賺錢)則只回答#no，如果不確定則只回答#unknown，請盡可能的不要回答#unknown，然後於下一行用簡短的句子進行詳細說明， 而以下為要分析的新聞標題:" + record;

10. ans\_textBox.AppendText(message + Environment.NewLine + Environment.NewLine);

11.

12. ProcessStartInfo psi = new ProcessStartInfo();

13. psi.FileName = pythonInterpreter;

14. psi.Arguments = $"{pythonScriptPath} \"{cookiepath}\" \"{message}\""; //傳參數給python

15. psi.UseShellExecute = false;

16. psi.RedirectStandardOutput = true; // 捕獲輸出

17. psi.CreateNoWindow = true;

18.

19.

20. using (Process process = new Process())

21. {

22. process.StartInfo = psi;

23. process.Start();

24.

25.

26. process.WaitForExit();

27.

28. if (process.ExitCode == 0)

29. {

30. answer[i] = process.StandardOutput.ReadToEnd();//讀所有python輸出

31. ans\_textBox.AppendText($"Answer: {answer[i]}" + Environment.NewLine + Environment.NewLine);

32. //Console.WriteLine($"Answer:\n {answer[i]}");

33.

34. string pattern = @"(#unknown|#no|#yes)";

35.

36. MatchCollection matches = Regex.Matches(answer[i], pattern);

37.

38. foreach (Match match in matches)

39. {

40. string keywordText = match.Value;

41. // 寫入 CSV 資料行

42. using (StreamWriter writer = new StreamWriter(ans\_path, true, Encoding.UTF8)) // 第二個參數為 true，表示追加模式 // 需要更動的地方

43. {

44. writer.WriteLine(date + "," + record + "," + keywordText);

45. }

46. }

47.

48. i++;

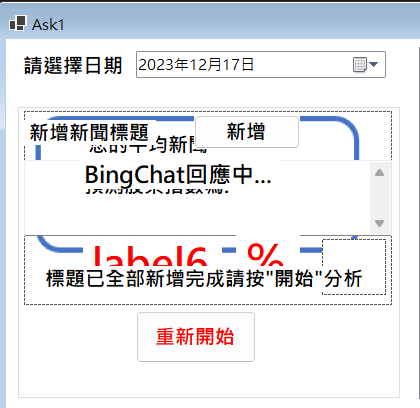
49. }

50. else

51. {

52. Console.WriteLine("Python script execution failed.");

53.  }



圖表 透過物件的方法進行美觀設計

八、全功能程式流程統整

**Research:**

news\_catch.py->news\_filter.py->Intraday\_return.py->BingChat.py->Conclusion.py

(介面BY:FinanceChatAI)

**Forecast:**

Catch\_news\_now.py->news\_filter\_now.py ->BingChat.py->Conclusion\_Forecast.py

(介面BY:FinanceChatAI)

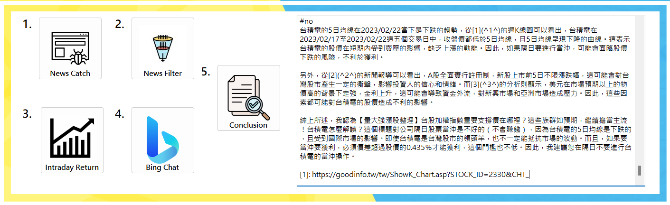
**Ask:**

BingChat.py->Conclusion\_Ask.py

(介面BY:FinanceChatAI)

**肆、執行結果**

模擬一：研究員A正在進行生成式AI與金融股市之間的相關研究，為了優化研究的結果,並分析更好的方法，研究員會透過自動化抓取股市新聞、文字辨識過濾新聞、抓取yahoo finance 抓取股市相關資訊，並計算出Intraday Return、詢問Bing Chat新聞分析結果，並計算出對應的分數，最後進行結果分析，提供準確度與期望值，而研究員可以在需要的地方進行prompt優化、更新爬蟲的時間、並調整模型的訓練，以達到更好的系統效能。因此以下模擬此情境:



模擬二：今天有一位普通消費者，他希望透過最快速的方法，依照他想分 溪的股市時間長短(ex:3天)進行股市預測分析，因此他將選擇他想分析的時間，系統將自動提供他隔日對應股票公司的股市預測分數與閾值圖。因此以下模擬此情境:



模擬三：今天有一位消費者，他想用他的額外消息，或特別的屬性提供系統分析，則他將輸入他想分析的日期與公司，並提供相關數據，讓系統可以自動分析，並提供相對應的股市預測分數與閾值圖，因此以下模擬此情境:

****

**[](https://www.youtube.com/embed/TZ-cEXJ9enc?feature=oembed)**

**伍、參考資料**

1.Bing Chat API:

<https://github.com/Integration-Automation/ReEdgeGPT>

2.Yahoo Finance API:

<https://finance.yahoo.com/?guccounter=1&guce_referrer=aHR0cHM6Ly93d3cuZ29vZ2xlLmNvbS8&guce_referrer_sig=AQAAAHcQwneIMwZswPI1hkPwWV84ybQrjHm6ZktGLJPvCDtj5b3rwDS72Vy8Iw6lPF-YLMS76IxPXhawgR4D5sMTCWKQbtjZnEvNPpJ79GULidKG6y0S-3ZLoInO2jz3OO7EXpehCiMTbZFJBlJB4wEyP3CUR-D_uCqWebLOIvqDSBIl>

3. Selenium 動態網路爬蟲:

<https://hackmd.io/@meebox/SJ7Um4Whs>

4.其他的import 資料

<https://steam.oxxostudio.tw/category/python/library/datetime.html>

<https://adsl52410.pixnet.net/blog/post/188617714-python-idle%E4%B8%AD%E5%AF%A6%E7%8F%BE%E6%B8%85%E9%99%A4%E7%95%AB%E9%9D%A2%E6%8C%87%E4%BB%A4(%E9%81%A9%E7%94%A82.x%E3%80%813.x%E7%89%88%E6%9C%AC>

<https://officeguide.cc/python-time-tutorial-examples/>

**陸、心得**

● A1105505林彧頎

我在這學期的期末專題中感觸很深，首先非常感謝老師讓我可以使用我這學期參加比賽的作品來當我的期末專題，當然過程中我也有多增加老師上課的內容於我的系統之中，下半學期的內容其實比較難，但也相對比較少，卻也讓我學到很多，而本專題可以說是將老師這學期所教導的融會貫通，並也融入生成式AI的功能，我在與組員的配合中，教會組員許多他們沒接觸過的東西，我也很慶幸組員很勇於發問，搞懂這個系統的實作方法，我也很開心可以透過這次的機會重新整理自己的作品，並優化它，整體來說，我透過本次專題從頭參與到尾，基本上每個步驟都有我的身影無論企劃、架構等等…，尤其是程式的部分，我覺得程式永遠都無法學完，因為有太多功能，而且科技不斷地發展，其實這個實作，當初是我從不會python,而且第一次接觸生成式AI與爬蟲，在不斷地做中學，學中做，而製作出來的，我覺得過程中所帶來的成就是非常高的，雖然花了我將近1個月以上的時間，每日每夜研究並做出來的成果，我最終也將這個成果投到演討會，並成功發表論文，因此我很感謝本專題提供我展現此實作的機會，並且很感謝老師與組員，這學期讓我學到很多東西!非常感謝!

● A1103239彭一珊

這次的期末專題我覺得非常實用，因為要盯一支股票的起伏本來就是一件很麻煩的事情，如果你全部人工去盯盤，估計一整天都沒辦法專心做其他事情，就會像我的某些同學一樣，一邊上課一邊盯盤。但是結合了自動化AI就可以方便很多，未來要是可以再進一步結合程式下單，估計就可以達到跟存股一樣，買了就睡覺的輕鬆投資法。然後非常感謝我的組員，每個都很給力，各項分工大家也都提早繳交。

● A1106135黃榛柔

因期中過後時間變得比較緊湊，課程的難度加深許多，我投入的時間也增加許多，但我反而覺得很有收穫。而期末專題有了期中的經驗，讓我們有小組的默契分工明確更有效率，這次也還是要謝謝我的組員，不厭其煩的回答我問題幫助我很多，我才能順利完成我的part。

● A1092301郭珈妤

我是這學期才剛開始學習程式,同組的成員都很厲害,除了讓我覺得自己還有很多要學習的以外,也驚訝於程式的神奇,居然透過手指在鍵盤上敲打就可以創造出一個對自己或人們很有幫助的系統!令人很有成就感的是,我從開學時零基礎到能夠利用這門課所學完成這次的期中專題，我在這堂課的收穫很多!

自從開始學習程式,我越來越覺得在未來，科技勢必會蓬勃發展，每一個人都需要倚靠科技，不如就先學習起來，也許未來有一天會對於我的人生有不一樣的轉捩點!

