2025年【科學探究競賽-這樣教我就懂】

□國中組 ■普高組 □技高組 成果報告格式

題目名稱:『它』會玩股票-利用AI來使投資分析更具效率

一、摘要

本研究旨在探討如何透過AI協助投資分析流程自動化，以解決學生在兼顧課業與投資時所面臨的時間與精力不足問題。傳統投資分析需花費大量時間進行資料搜尋與彙整，對高中生而言效率極低。因此，我們嘗試將此過程自動化。本專題以「轉換公司債資訊蒐集」為核心，分階段設計程式架構，包含下載資料、比對更新、及進一步分析。過程中我們測試了ChatGPT 3.5、Grok 3與Gemini 2.5等AI模型，發現不同AI對於動態爬蟲與邏輯推理的支援程度影響結果成效。最終證實透過適當AI模型輔助，即使不具備完整程式編寫能力，仍可有效建構具備實用功能的投資分析工具，達成節省時間並提升效率的目標。

並且通過此探究，還可一窺AI在將日常瑣事自動化方面的功用，總結出高效使用的技巧。

二、探究題目與動機

我們發現在進行投資時，常常要查詢並梳理大量的資料，而這需要花費大量的時間和精力，但對要準備學測的高中生而言，這兩樣東西卻是彌足珍貴的，因此我們迫切地需要一個解決方案來平衡學習與投資。雖然透過學習並應用程式語言，這枯燥而龐雜的工作似乎可以交由電腦執行，然而，由於程式語言的學習門檻過高，我們依然無法隨心所欲的活用電腦，許多乏味的工作依然只能由自己完成。不過隨著近幾年AI的強勢崛起，讓『它』幫我們寫程式的構想變得可行，因此我們決定通過活用AI來幫助自己跨過學習程式語言這道門檻，以節省自己寶貴的精力和時間。

除了這種較偏向私人領域的願景外，本探究更有一個宏大的普世的願景，那就是探究AI是否有辦法將人類社會各方面的繁複工作自動化，如果有那麼成效如何。而我們之所以會通過以投資分析效率化來以小見大，是因為將其效率化的程式，需要用到動態網頁爬蟲的技術，以及基本的資料分析和比對。故如若AI可以成功幫助我們完成這個程式，那麼只要人們願意，它就可以將生活中那些毫無意義、過度消磨精力的工作給全部省去。

三、探究目的與假設

目的1:製作出一個可以代為查詢並分析投資資料的程式

目的2:探究AI在將繁複工作自動化方面的功用，並且歸納運用方法。

四、探究方法與驗證步驟

步驟1:選定進行投資分析時需要簡化的環節, 並設計相應程式之架構

在進行投資時, 往往需要很多面向的資訊, 例如:公司經營狀況、分析師評價、產品品質等。但是在蒐集資料的時候, 最麻煩的並不是上述提到的那些, 而是轉換公司債, 如果想要以人工的方式去獲得這方面的資訊, 那麼就需要先從網站上下載表單, 跳出即將發行轉換公司債的公司, 接著搜尋該公司的資訊, 據我們估計這個過程將需要投資者每天投入半小時。然而這種簡單的工作如果以程式代替, 最少也可以節省1/2的時間, 並且由於作業過程不需要人類的參與, 這段時間完全可以被活用, 從結果上來看就相當於每天多出了半小時。所以在經過評估後, 我們決定將上述提到的作業流程, 拆分成幾個部分以建立初步的程式架構。

第一部分:利用爬蟲程式從金管會的官網下載114年度的申請案件總表單

第二部分:將下載的檔案與前一天的檔案對比, 觀察有無提到關鍵字的案件, 如無則回報今日無目標, 如有則將新增的案件紀錄於另外一個表格。

第三部分:以爬蟲程式於公開資訊觀察站, 查詢出現在表格上的公司, 揀選出有提到關鍵字「專戶合約」者, 並進行回報。

步驟2:將架構告知AI, 生成程式並進行試錯與修正

1.chatgpt 3.5

作為我們首先嘗試的模型, 坦白說它的表現不甚理想, 它所寫出的程式連第一部分的目標都沒有達成。在經過討論和資料尋找後, 我們認為它之所以失敗, 是因為chatgpt產出的程式, 只能用於靜態網頁, 並且在我們指出這個問題後, 它也並沒有提出修改的建議, 或是推薦動態網頁爬蟲所需的掛件。最後由於該模型並不具備推理模式, 難以理解我們越來越複雜的指示, 所以一時之間, 我們的工作陷入了停滯。

以下是可以在 Google Colab 上執行的 Python 程式碼，它會根據今天的「民國日期」產生正確的 Excel 檔案網址，並下載該檔案：

python 複製 編輯

```python
import datetime
import urllib.parse
import requests

# 計算今天的民國日期
today = datetime.date.today()
roc_year = today.year - 1911  # 民國年
roc_month = today.month
roc_day = today.day

# 組成民國格式的日期字串，例如 1140403
roc_date_str = f"{roc_year:03d}{roc_month:02d}{roc_day:02d}"

# 檔案名稱中的中文部分是這樣的：「申報案件彙總表」
filename_chinese = "申報案件彙總表"
filename_encoded = urllib.parse.quote(filename_chinese)  # 轉成 URL 編碼

# 組出完整網址
url = f"https://www.fsc.gov.tw/userfiles/file/{roc_date_str}{filename_encoded}.xlsx"

# 下載 Excel 檔案
response = requests.get(url)

# 檢查是否成功
if response.status_code == 200:
    with open("下載_申報案件彙總表.xlsx", "wb") as f:
        f.write(response.content)
    print("下載完成 ✅")
else:
    print("下載失敗 ❌，可能今天尚未上傳該檔案")
    print("狀態碼:", response.status_code)
    print("網址:", url)
```

該模型寫出的程式碼

2.Grok 3

在上一個模型失敗後，我們轉為使用xAI推出的Grok 3，該模型具有推理模式，所以能夠給出更實用的意見，比如說:當我們要求它幫助我們產出一個具有動態爬蟲功能的程式時，它就提到可以利用selenium去模擬使用者的動作，再搭配Webdriver，去指定要模擬的瀏覽器。而在我們根據它給的意見，重新修改我們的指示後，我們就成功達成了第一部分的目標，使下載檔案的流程自動化了。

而第二部分的目標，它也順利地達成了。問題是出在第三部分，雖然也和第一部分一樣使用了動態爬蟲的技術，但是由於Grok 3容易產生錯亂、思維較不連貫，且剛剛推出，所以模型較不成熟，因此關於第三部分的指示，它始終無法理解。

Python 程式碼

```python
# 匯入必要的庫
import requests
from bs4 import BeautifulSoup
from urllib.parse import urljoin

# 定義目標網頁的 URL 和基本網址
url = "https://www.sfb.gov.tw/ch/home.jsp?id=1016&parentpath=0,6,52"
base_url = "https://www.sfb.gov.tw"

# 抓取網頁內容
response = requests.get(url)
html_content = response.text

# 使用 BeautifulSoup 解析 HTML
soup = BeautifulSoup(html_content, 'html.parser')

# 尋找包含「檔案下載」的表格
tables = soup.find_all('table')
target_table = None
for table in tables:
    if '檔案下載' in table.text:
        target_table = table
        break

if not target_table:
    print("找不到包含下載連結的表格。")
else:
    # 尋找「檔案下載」欄位的索引
    header_row = target_table.find('tr')
    headers = header_row.find_all('th')
```

該模型寫出的程式碼

3.Gemini 2.5 Pro Preview 03-25

在我們對Grok 3逐漸失望的同時, google於3/25推出了此模型, 由於比賽截止日近在眼前, 因此我們大膽的放棄了之前的進度, 選擇用此模型來將整個程式打掉重練。而這一次就順利的多, 在使用Grok 3時, 我們開了九個聊天室, 才完成第一、二部分的內容, 而透過使用該模型時, 我們只開了兩個聊天室。以上的結果顯示了此模型相對Grok 3更優越的理解能力, 以及更不容易發生錯亂的特性。

好的，沒問題。為了在最終的彙總表格中加入「生效日期」，我們需要做以下調整：

1. 在 **Part 1**（處理 Excel 檔案的部分）：確保從 Excel 篩選出的 `result_df` 包含 '生效日期' 欄位。
2. 在 **Part 2 開始時**：將包含 '生效日期' 的 `result_df` 轉換為 `checked_companies_list_part2` 列表，以便 Part 3 使用。
3. 在 **Part 3 (彙總)**：當從 `checked_companies_list_part2` 創建 DataFrame 時，'生效日期' 就會自然包含在內，並顯示在最終的表格中。

以下是修改後的手動上傳版本完整程式碼：

```
# --- Manual Upload Script with Final Summary including Effective Date ---

# --- 必要函式庫導入 ---
import pandas as pd
import io
import traceback
import os
import time
import random

# 導入 Colab files 功能
try:
    from google.colab import files
    google_colab_available = True
    print("Google Colab 'files' module imported successfully.")
except ImportError:
    google_colab_available = False
    print("警告：未在 Google Colab 環境中，無法使用檔案上傳功能。")

# 導入 Selenium 相關函式庫
try:
    from selenium import webdriver
    from selenium.webdriver.chrome.options import Options
```

```
WebDriver 已關閉所有分頁。

—— Part 3: 最終結果彙總 ——

找到關鍵字 '專戶合約' 的公司：
 證券代號 公司名稱
 1786    科妍

未找到關鍵字 '專戶合約' 的公司 (或查詢過程出錯)：
 證券代號    公司名稱
 3653    健策精密
 8087    麗升能源
 3167    大量科技
 6538      倉和
 3188     鑫龍騰
 4555    台灣氣立
 6771 平和環保-創
 8404    F-百和

—— 整體程式執行完畢 ——
```

上圖為該模型寫出的程式碼

下圖為程式產出的結果

五、結論與生活應用

結論：

在使用程式前，如果利用人工的方式進行工作，那麼至少要耗時半小時，而使用程式後只需要將電

腦開著, 讓程式運行至多六分鐘即可完成。而這次實驗也證實了只要適當的操作, 沒有相關知識的普通人也可以在AI的幫助下寫出一個具備功能的程式。並且經過這次的探究, 我們也總結出了幾個運用AI的技巧:

1.要讓馬跑就必須讓馬吃草

使用AI絕不能只有單方面的給予少量提示, 就要求它交出滿分的答卷, 相反, 一個合格的使用者該要不斷地積極地以大量的文本、資料去向AI闡述自己的構想。同時, 在下指令式的互動失敗後, 要和AI「討論」, 一方面可以它歸納和改進問題, 一方面也可激發我們自己的靈感。例如我們這次能夠克服爬蟲程式這個難關, 就是因為在和AI討論時, 發現了動態網頁爬蟲這個技術和我們元技術的差別, 然後透過這個發現我們改變了製作的思路, 最終克服了難關。

2.博採眾家之長

在這次的探究中我們發現, 依靠單一AI是低效且不可持續的, 每個公司所推出的它, 都各有千秋, 也都各有其難以接受之處, 例如GPT可以良好地記錄對話和適應使用者, 而Gemini則具有良好的推理和理解能力。並且各家廠商都有可能進行更新, 一口氣超過自己的競爭者們, 所以只有不斷的「雨露均霑」, 才能最高效的運用他們。

3.人工智慧和工人智慧

即使發展到現在, 人工腦仍不能代替人腦, 很多人總有個迷思, 認為生成式AI可以包辦整個生成的過程。但根據這次探究的經驗, 我們認為人們依然不能放棄學習, 說穿了AI就像鐵匠手中的槌子, 或許好的槌子確實大有裨益, 但鑄劍的終究是鐵匠。這次的程式雖然表面上是由AI所完成的, 但其實過程中所遇到的難關大多都是由人類解決的, 所以背景知識仍然是不可或缺的。另外在解決的過程中我們也學了很多, 我們對於如何創建一個程式的架構以及網路爬蟲技術有了更深的理解。總結而言, AI不是讓學習遠離我們的工具, 而是讓學習變得可接近的工具, 新時代的到來, 不代表知識變得無用, 相反知識變得更珍貴了, 畢竟我們現在有了一個可以讓我們所有知識和創意都被實現的工具了。

生活應用**:**

坦白來說, 以AI寫程式能達成的應用實在太多了, 不是人力能夠歸納的。在醫院掛號、填寫志願和訂購門票有太多太多的方面可以應用了。除此之外, 在寫程式時, 還可以更深入地學習使用AI, 這個在可預見的未來中, 幾乎是必備的技能。

參考資料

無