勞動部產業新尖兵計畫

人工智慧金融應用與實務培訓班

課程模組： AI 金融科技課程 - 金融大數據分析

標的 2.Yahoo 財經 - 那斯達克

葉建華 (Yeh, Jian-hua)

tdi.jhyeh@tdi.edu.tw
au4290@gmail.com

講次內容

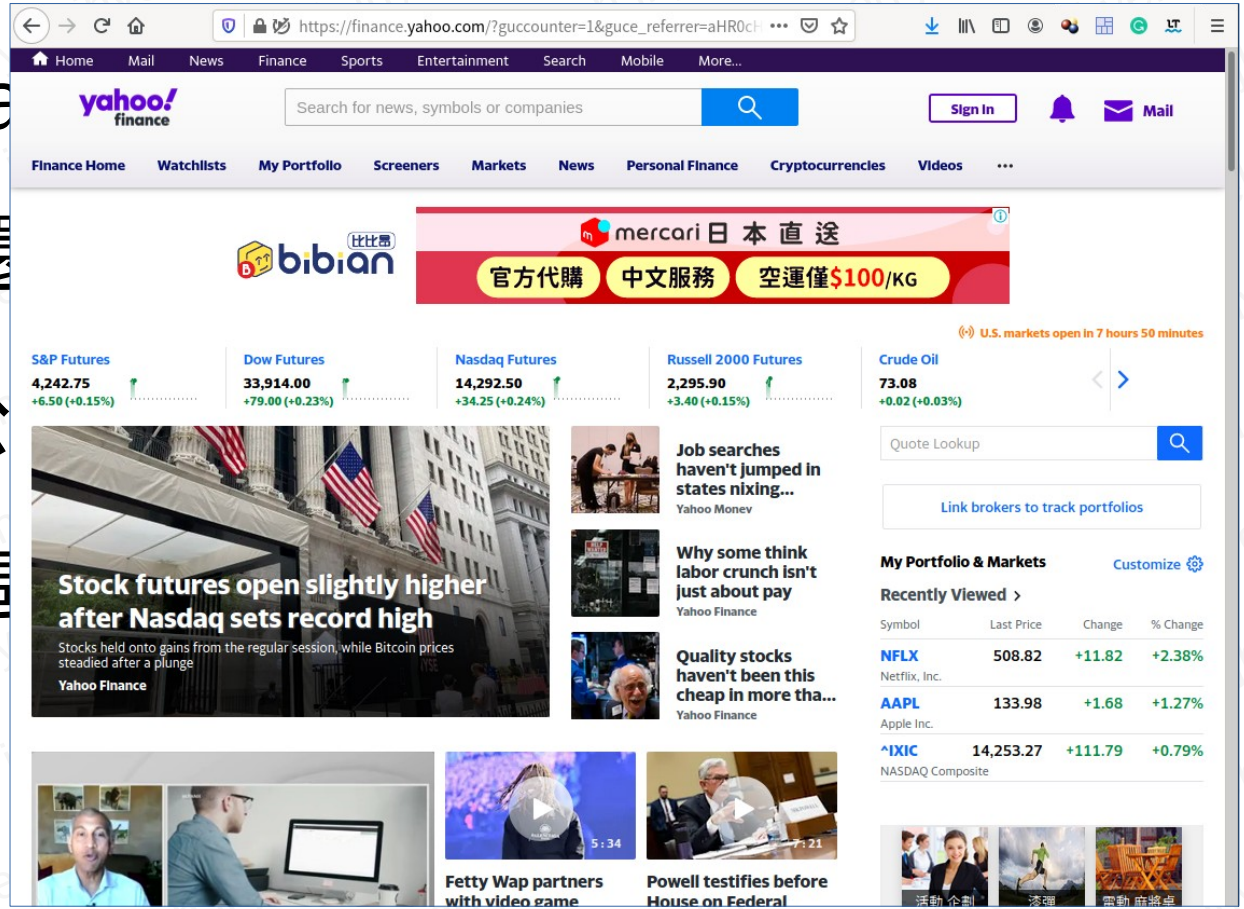
- Yahoo 財經 - 那斯達克每日收盤資訊介紹
- 網路爬蟲與資料庫的規劃與操作
- SQL 收盤價查詢
- 專題：那斯達克收盤價走勢

Yahoo 財經官網

- https://finance.yahoo.com
- 是 https 哦！（還記得嗎？）
- .com 代表商業公司單位
- .tw？沒有了，這是美國公司，只有 .com

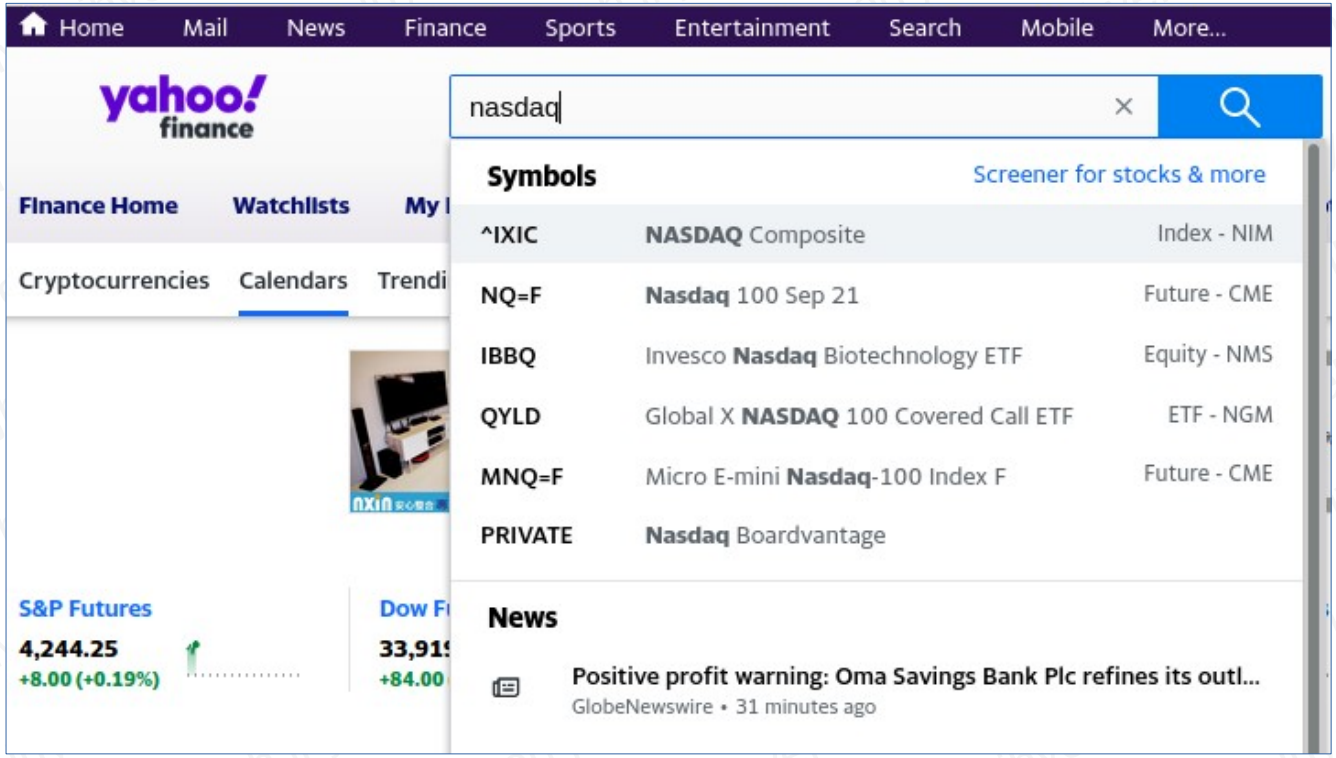
Yahoo 財經官網

- https://finance.ya
- 是 https 哦！（還
- .com 代表商業公
- .tw？ 沒有了，這



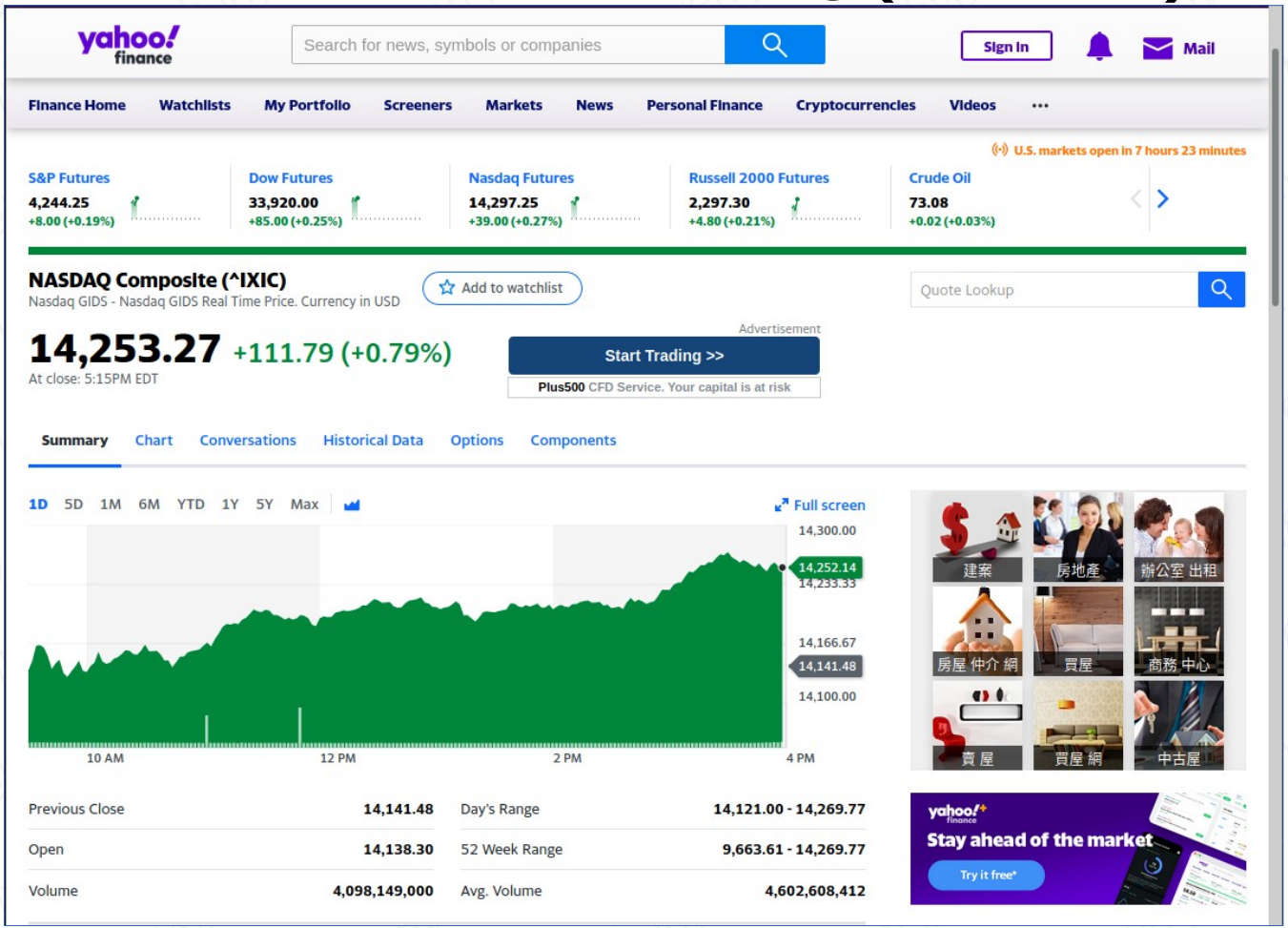
尋找 NASDAQ...

- 正上方搜尋框鍵入「 NASDAQ 」



看到 ^IXIC 了嗎?
那就是 NASDAQ 指數!

NASDAQ(^IXIC)



看到左半部中間的

「 Historical Data 」了嗎?

NASDAQ 歷史資料

https://finance.yahoo.com/quote/^IXIC/history?p=^IXIC

yahoo! finance

Search for news, symbols or companies

Sign In

Mail

Finance Home | Watchlists | My Portfolio | Screeners | Markets | News | Personal Finance | Cryptocurrencies | Videos | ...

Time Period: Jun 23, 2020 - Jun 23, 2021

Show: Historical Prices

Frequency: Daily

Apply

Currency in USD

Download

Date	Open	High	Low	Close*	Adj Close**	Volume
Jun 23, 2021	14,263.40	14,313.00	14,263.38	14,310.59	14,310.59	1,033,890,000
Jun 22, 2021	14,138.29	14,269.77	14,121.00	14,253.27	14,253.27	4,405,260,000
Jun 21, 2021	14,047.42	14,150.78	13,960.04	14,141.48	14,141.48	4,523,120,000
Jun 18, 2021	14,096.93	14,129.22	14,009.04	14,030.38	14,030.38	6,105,960,000
Jun 17, 2021	13,999.13	14,196.21	13,998.93	14,161.35	14,161.35	4,526,780,000
Jun 16, 2021	14,085.55	14,129.69	13,903.73	14,039.68	14,039.68	4,632,480,000
Jun 15, 2021	14,166.64	14,171.02	14,052.16	14,072.86	14,072.86	4,517,770,000
Jun 14, 2021	14,083.47	14,175.45	14,056.67	14,174.14	14,174.14	4,394,380,000
Jun 11, 2021	14,030.85	14,069.42	14,006.59	14,069.42	14,069.42	4,140,190,000
Jun 10, 2021	13,933.88	14,031.19	13,904.40	14,020.33	14,020.33	4,889,500,000
Jun 09, 2021	13,980.23	14,003.50	13,906.45	13,911.75	13,911.75	5,607,720,000
Jun 08, 2021	13,946.32	13,981.72	13,831.98	13,924.91	13,924.91	5,894,140,000

上方有過濾條件設定：
資料期間、資料型態、頻率種類

NASDAQ： 2020 年歷史資料

- 設定資料期間、資料型態、頻率種類
- 按「 Apply 」按鈕
- 你看到了什麼?

NASDAQ： 2020 年歷史資料

Time Period: Jan 01, 2020 - Dec 31, 2020 Show: Historical Prices Frequency: Daily

Apply

Currency in USD

Download

Date	Open	High	Low	Close*	Adj Close**	Volume
Dec 30, 2020	12,906.51	12,924.93	12,857.76	12,870.00	12,870.00	5,292,210,000
Dec 29, 2020	12,965.39	12,973.33	12,821.96	12,850.22	12,850.22	4,680,780,000
Dec 28, 2020	12,914.64	12,930.89	12,827.45	12,899.42	12,899.42	5,076,340,000
Dec 24, 2020	12,791.54	12,833.55	12,767.64	12,804.73	12,804.73	3,305,950,000
Dec 23, 2020	12,834.94	12,841.92	12,758.67	12,771.11	12,771.11	7,028,650,000
Dec 22, 2020	12,785.22	12,840.57	12,695.31	12,807.92	12,807.92	5,700,760,000
Dec 21, 2020	12,596.14	12,751.27	12,525.22	12,742.52	12,742.52	5,156,470,000
Dec 18, 2020	12,804.93	12,809.60	12,654.60	12,755.64	12,755.64	7,088,670,000
Dec 17, 2020	12,730.78	12,765.25	12,696.35	12,764.75	12,764.75	4,994,090,000
Dec 16, 2020	12,611.04	12,687.32	12,566.38	12,658.19	12,658.19	4,561,600,000

種類

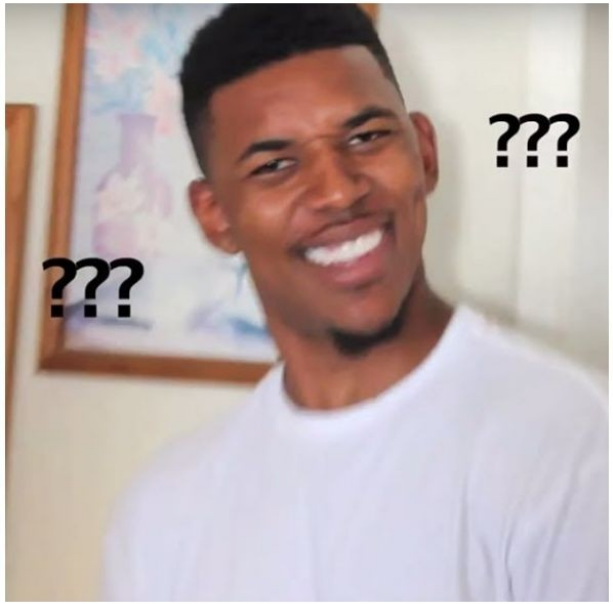
很明顯是 NASDAQ 2020 年的「日線資料」。有看到 Download 連結嗎?

NASDAQ： 2020 年歷史資料

- 2020 年日線資料 Download 連結：
  - https://query1.finance.yahoo.com/v7/finance/download/%5EIXIC?period1=1577836800&period2=1609372800&interval=1d&events=history&includeAdjustedClose=true
  - period1、 period2 是重點！
  - interval=1d 是指日線資料
  - events=history 是指歷史資料
  - includeAdjustedClose=true 是指還原過的收盤價，在此不討論

NASDAQ： 2020 年歷史資料

- 這是什麼？！
  - period1=1577836800
  - period2=1609372800



Epoch Time，或稱 Unix 時間

- 定義： https://zh.wikipedia.org/wiki/UNIX%E6%97%B6%E9%97%B4
  - 是 UNIX 或類 UNIX 系統使用的時間表示方式：從 UTC1970 年 1 月 1 日 0 時 0 分 0 秒起至現在的總秒數（UTC：格林威治標準時間）

Epoch Time，或稱 Unix 時間

- 換算一下這兩個
  - period1=
  - period2=

```python
from datetime import date, datetime

period1 = 1577836800
period2 = 1609372800
dt1 = date.fromtimestamp(period1)
dt2 = date.fromtimestamp(period2)
print(dt1)          # 2020-01-01
print(dt2)          # 2020-12-31

# 那如果要倒過來操作呢?
# 2021-01-01
period3 = int(datetime(2021, 1, 1, 0, 0).timestamp())
# 2021-06-21
period4 = int(datetime(2021, 6, 21, 0, 0).timestamp())
print(period3)
print(period4)
```

NASDAQ 歷史資料

- 自訂期間： 2021-01-01~2021-06-21
- Download 連結為何?

```
from datetime import datetime
import pytz

# 2021-01-01
dt1 = pytz.utc.localize(datetime(2021, 1, 1, 0, 0))
period1 = int(dt1.timestamp())
# 2021-06-21
dt2 = pytz.utc.localize(datetime(2021, 6, 21, 0, 0))
period2 = int(dt2.timestamp())

urlstr = 'https://query1.finance.yahoo.com/v7/finance/download/%5EIXIC?'
urlstr += 'period1='+str(period1)+'&period2='+str(period2)
urlstr += '&interval=1d'
print(urlstr)
```

NASDAQ 歷史資料

- 自訂期間： 2021-01-01~2021-06-21
- Download 連結為何?
  - https://query1.finance.yahoo.com/v7/finance/download/%5EIXIC?period1=1609459200&period2=1624233600&interval=1d
  - 我們甚至把還原過的收盤價參數都拿掉了！

那就抓吧！ 怎麼抓?

NASDAQ 歷史資料

這次用 requests!

- 自訂
- Dov
  - ht ad/ %
    pe l=1d
  - 我

```python
from datetime import datetime
import pytz
import requests

# 2021-01-01
dt1 = pytz.utc.localize(datetime(2021, 1, 1, 0, 0))
period1 = int(dt1.timestamp())
# 2021-06-21
dt2 = pytz.utc.localize(datetime(2021, 6, 21, 0, 0))
period2 = int(dt2.timestamp())

urlstr = 'https://query1.finance.yahoo.com/v7/finance/download/%5EIXIC?'
urlstr += 'period1='+str(period1)+'&period2='+str(period2)
urlstr += '&interval=1d'
print(urlstr)

outfname = 'nasdaq.20210101-20210621.csv'
resp = requests.get(urlstr)
with open(outfname, 'w') as outf:
    outf.write(resp.text)
print('NASDAQ歷史資料下載完成')
```

NASDAQ 歷史資料

這次用 requests!

- 自訂
- Dov
  - ht
    %
    pe
  - 我

```python
from datetime import datetime
import pytz
import request

# 2021-01-01
dt1 = pytz.utc
period1 = int(
# 2021-06-21
dt2 = pytz.utc
period2 = int(

urlstr = 'http
urlstr += 'per
urlstr += '&in
print(urlstr)

outfname = 'na
resp = request
with open(outf
    outf.write
print('NASDAQ
```

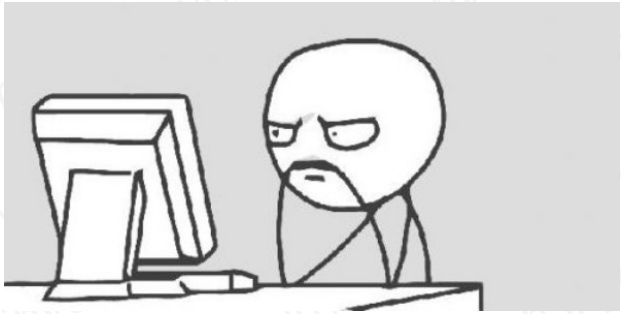
nasdaq.20210101-20210621.csv
~/Desktop/summer.course/python.bda/proj4

```csv
Date,Open,High,Low,Close,Adj Close,Volume
2021-01-04,12958.519531,12958.719727,12543.240234,12698.450195,12698.450195,6546740000
2021-01-05,12665.650391,12828.269531,12665.650391,12818.959961,12818.959961,6904420000
2021-01-06,12666.150391,12909.629883,12649.990234,12740.790039,12740.790039,7648340000
2021-01-07,12867.339844,13090.910156,12867.339844,13067.480469,13067.480469,6777010000
2021-01-08,13160.219727,13208.089844,13036.549805,13201.980469,13201.980469,7223660000
2021-01-11,13048.780273,13138.269531,12999.509766,13036.429688,13036.429688,6876420000
2021-01-12,13062.059570,13105.040039,12963.919922,13072.429688,13072.429688,7181380000
2021-01-13,13088.009766,13171.150391,13051.059570,13128.950195,13128.950195,7072920000
2021-01-14,13174.750000,13220.160156,13098.410156,13112.639648,13112.639648,6671090000
2021-01-15,13099.900391,13139.830078,12949.759766,12998.500000,12998.500000,6402970000
2021-01-19,13132.730469,13206.860352,13078.700195,13197.179688,13197.179688,6229100000
2021-01-20,13342.549805,13486.129883,13329.769531,13457.250000,13457.250000,6771630000
2021-01-21,13521.480469,13560.349609,13454.070313,13530.910156,13530.910156,7183390000
2021-01-22,13474.809570,13567.139648,13463.660156,13543.059570,13543.059570,5931240000
2021-01-25,13681.209961,13728.980469,13368.679688,13635.990234,13635.990234,7139410000
2021-01-26,13681.719727,13702.690430,13603.190430,13626.059570,13626.059570,6781460000
2021-01-27,13486.580078,13538.419922,13192.910156,13270.599609,13270.599609,11102160000
2021-01-28,13323.290039,13507.639648,13316.519531,13337.160156,13337.160156,9823150000
2021-01-29,13284.719727,13322.000000,12985.049805,13070.690430,13070.690430,7809670000
2021-02-01,13226.179688,13431.459961,13132.469727,13403.389648,13403.389648,7014220000
2021-02-02,13543.099609,13652.700195,13535.860352,13612.780273,13612.780273,7240220000
2021-02-03,13718.309570,13723.830078,13585.339844,13610.540039,13610.540039,7465240000
2021-02-04,13674.059570,13778.419922,13631.620117,13777.740234,13777.740234,7218680000
2021-02-05,13824.879883,13878.160156,13761.660156,13856.299805,13856.299805,6697720000
```

NASDAQ 歷史資料

- Yahoo 財經沒有提供
  - 成交值
  - 成交筆數

這就真的沒辦法了 ...



CSV 格式

- 第 1 行:
  - Date,Open,High,Low,Close,Adj Close,Volume
- 從 CSV 找到這行: 「 Date 」
  - 從這以下，都是這種個股收盤資料
- 所以呢?

講次內容

- Yahoo 財經 - 那斯達克每日收盤資訊介紹
- 網路爬蟲與資料庫的規劃與操作
- SQL 收盤價查詢
- 專題：那斯達克收盤價走勢

從網路爬蟲到資料庫

- 分工!

我們設定 2021-01-01~2021-06-21

  - 網路爬蟲： Yahoo 財經下載只一個 CSV！ 已抓!
  - CSV 解析：把 CSV 依據欄位進行拆解
  - 資料庫：把拆解完的欄位對應資料儲存到資料表中

Part 1. 網路爬蟲

2021-01-01~2021-06-21

只有一個 CSV 檔案!

已抓!

不用再忙了!

Part 2. 解析 CSV

- 如何對應?
  - Date,Open,High,Low,Close,Adj Close,Volume
  - 2021-01-04,12958.519531,12958.719727, 12543.240234,12698.450195,12698.450195, 6546740000
  - 目標除了倒數第二欄 Adj Close 的所有欄位
    - 就是「 Date,Open,High,Low,Close,Volume 」
    - 也就是「日期、開、高、低、收、量」

再次提醒，沒有「值」和「筆數」

資料解析框架

- 針對 CSV 格式檔案進行後處理
  - 找到「 Date 」，以下各行開始解析成 Python 資料結構
- 抓取除了倒數第二欄 Adj Close 的所有欄位，第 1 欄文字、最後一欄整數、其他欄浮點數
  - 我稱它們為 date, op, hi, lo, cl, vol

資料解析框架

- 針對
  - 找 資料
    結
- 抓取 數
  - 我

```python
import csv

infname = 'nasdaq.20210101-20210621.csv'
with open(infname) as csvf:
    csvReader = csv.reader(csvf)
    recs = list(csvReader)

inData = False                  # 檔案一開始的內容不是我們要的
count = 0
for rec in recs:
    try:
        if (rec[0]=='Date' and inData==False):  # 我們要找的是"Date"這行以下的資料
            inData = True       # 找到了，設定資料處理旗標為真
            continue            # 跳過，我們不要這行
        if (inData):
            # 使用串列生成式，跳過第5欄Adj Close
            data = [rec[i] for i in range(7) if len(rec[0])==10 and i!=5]
            if len(data)>0:
                print(data)  # 我們先印出來看，下階段我們要存進資料庫裡!
                count += 1
    except:
        pass
print('交易日數', count)
```

資料解析框架

- 針對
  - 找
    結
- 抓取
  - 我

```
import csv

infname = '2020.currency.csv'
```

```
['2021-05-25', '13721.540039', '13751.139648', '13631.799805', '13657.169922', '4084480000']
['2021-05-26', '13693.940430', '13750.160156', '13679.589844', '13738.000000', '4231140000']
['2021-05-27', '13742.589844', '13776.519531', '13701.629883', '13736.280273', '5057550000']
['2021-05-28', '13792.049805', '13820.870117', '13747.610352', '13748.740234', '4435220000']
['2021-06-01', '13829.059570', '13836.169922', '13678.769531', '13736.480469', '4155670000']
['2021-06-02', '13743.240234', '13775.889648', '13689.740234', '13756.330078', '5059810000']
['2021-06-03', '13655.750000', '13684.129883', '13548.929688', '13614.509766', '5367460000']
['2021-06-04', '13697.250000', '13826.820313', '13692.009766', '13814.490234', '4341800000']
['2021-06-07', '13802.820313', '13889.110352', '13784.889648', '13881.719727', '4602940000']
['2021-06-08', '13946.320313', '13981.719727', '13831.980469', '13924.910156', '5894140000']
['2021-06-09', '13980.230469', '14003.500000', '13906.450195', '13911.750000', '5607720000']
['2021-06-10', '13933.879883', '14031.190430', '13904.400391', '14020.330078', '4889500000']
['2021-06-11', '14030.849609', '14069.419922', '14006.589844', '14069.419922', '4140190000']
['2021-06-14', '14083.469727', '14175.450195', '14056.669922', '14174.139648', '4394380000']
['2021-06-15', '14166.639648', '14171.019531', '14052.160156', '14072.860352', '4517770000']
['2021-06-16', '14085.549805', '14129.690430', '13903.730469', '14039.679688', '4632480000']
['2021-06-17', '13999.129883', '14196.209961', '13998.929688', '14161.349609', '4526780000']
['2021-06-18', '14096.929688', '14129.219727', '14009.040039', '14030.379883', '6105960000']
交易日數 116
```

```
    except:
        pass
print('交易日數', count)
```

抓出來了！果然不難！

資料解析框架

- 用串列收起來，接下來要用！

```
import csv

infname = 'nasdaq.20210101-20210621.csv'
with open(infname) as csvf:
    csvReader = csv.reader(csvf)
    recs = list(csvReader)

inData = False                  # 檔案一開始的內容不是我們要的
count = 0
alldata = []
for rec in recs:
    try:
        if (rec[0]=='Date' and inData==False):  # 我們要找的是"Date"這行以下的資料
            inData = True    # 找到了，設定資料處理旗標為真
            continue         # 跳過，我們不要這行
        if (inData):
            # 使用串列生成式，跳過第5欄Adj Close
            data = [rec[i] for i in range(7) if len(rec[0])==10 and i!=5]
            if len(data)>0:
                #print(data)  # 我們先印出來看，下階段我們要存進資料庫裡!
                alldata.append(data)
                count += 1
    except:
        pass
print('交易日數', count)
```

Part 3. 資料庫

- 規劃資料表，建立資料表 如何 CREATE TABLE?
- Python 資料結構 (list) 對應資料表
  - 產出 SQL 字串 如何 INSERT INTO?
- 執行 SQL 字串，資料新增到資料表內

資料表處理框架

- 建立資料表
  - SQLite, CREATE TABLE...
- 將 Python 資料結構 dict 轉換成 SQL 字串
  - SQLite, INSERT INTO...
- 做基本的檢查，看看有多少筆資料進資料表了
  - SQLite, SELECT COUNT(*) FROM...

建立資料表

- 本例建立兩個資料表
  - stkinfo： 存放上市公司名稱資料 (id, name)
  - stktrade： 存放個股交易資料
    - id, date, vol, tra, val, op, hi, lo, cl
    - 名稱放在 stkinfo 作為對照，節省空間，避免不一致

建立資料表

- 本
  - s
  - s

```python
import sqlite3

conn = sqlite3.connect('mystock.db')
sqlstr = 'CREATE TABLE stkinfo (id TEXT, name TEXT, PRIMARY KEY(id))'
conn.execute(sqlstr)
print('建立stkinfo資料表完成：', sqlstr)

sqlstr = 'CREATE TABLE stktrade (id TEXT, date TEXT, vol INTEGER, tra INTEGER, \
          val INTEGER, op REAL, hi REAL, lo REAL, cl REAL, PRIMARY KEY(id, date))'
conn.execute(sqlstr)
print('建立stktrade資料表完成：', sqlstr)

conn.commit()
conn.close()
```

這些你都做過了！

List 轉換成 SQL 存入

- 當日那斯達克收盤資料 list 結構
  - [日期 , 開 , 高 , 低 , 收 , 量]
- 拆解結構
  - 用 list 索引解出當日開高低收量的資訊
  - 轉成單一 SQL INSERT INTO 字串，注意要對值和筆數補 0
  - 執行 SQL

List 轉換成 SQL 存入

```python
import sqlite3

stkinfoSQL = set()
sqlstr = "INSERT INTO stkinfo (id, name) VALUES ('^IXIC', '那斯達克指數');"
stkinfoSQL.add(sqlstr)

stktradeSQL = []
count = 0
for rec in alldata:
    # 注意順序哦! 代號(0),名稱(1),收盤(2),開盤(3),最高(4),最低(5),成交股數(6),成交金額(7),成交筆數(8)
    sqlstr = "INSERT INTO stktrade (id, date, vol, tra, val, op, hi, lo, cl) VALUES "
    sqlstr += "('^IXIC', '"+rec[0]+"', "+rec[4]+", 0, 0, "
    sqlstr += rec[1]+", "+rec[2]+", "+rec[3]+", "+rec[4]+");"
    stktradeSQL.append(sqlstr)
print('stktrade records: ', len(stktradeSQL))

# 把容器裡的SQL字串通通拿來執行!
def dumpSQL(conn, container):
    count = 0
    for sqlstr in container:
        try:
            conn.execute(sqlstr)
            count += 1
            if (count%10000 == 0):    # 一陣子commit()確保寫入
                print(count, 'records')
                conn.commit()
        except:
            pass
    print('Total', count, 'records')
    conn.commit()
```

解出當日美元與人民幣

字串

```python
# 資料表新增資料開始!
print('資料表新增資料中...')
conn = sqlite3.connect('mystock.db')

dumpSQL(conn, stkinfoSQL)
print('資料表stkinfo新增資料完成! ')
dumpSQL(conn, stktradeSQL)
print('資料表stktrade新增資料完成! ')

conn.close()

# 最後把所有trade的SQL寫出來看看...
outfname = 'sqlout.txt'
with open(outfname, 'w') as outf:
    for sqlstr in stktradeSQL:
        outf.write(sqlstr+'\n')
```

List 轉換成 SQL 存入

```
import sqlite3

stkinfoSQL = set()
sqlstr = "INSERT INTO stkinfo (id, name) VALUES ('^IXIC', '那斯達克指數');"
stkinfoSQL.

stktradeSQL
count = 0
for rec in
    # 注意順                                  ,最高(4),最低(5),成交股數(6),成交金額(7),成交筆數(8)
    sqlstr                                    , vol, tra, val, op, hi, lo, cl) VALUES "
    sqlstr                                    [4]+", 0, 0, "
    sqlstr                                    +", "+rec[4]+");"
    stktradeSQL.append(sqlstr)
print('stktrade records: ', len(stktradeSQL))

# 把容器裡的SQL字串通通拿來執行!
def dumpSQL(conn, container):
    count = 0
    for sqlstr in container:
        try:
            conn.execute(sqlstr)
            count += 1
            if (count%10000 == 0):    # 一陣子commit()確保寫入
                print(count, 'records')
                conn.commit()
        except:
            pass
    print('Total', count, 'records')
    conn.commit()
```

```
stktrade records:  116
資料表新增資料中...
Total 1 records
資料表stkinfo新增資料完成!
Total 116 records
資料表stktrade新增資料完成!
```

解出當日美元與人民幣

字串

```
# 資料表新增資料開始!
print('資料表新增資料中...')
conn = sqlite3.connect('mystock.db')

dumpSQL(conn, stkinfoSQL)
print('資料表stkinfo新增資料完成! ')
dumpSQL(conn, stktradeSQL)
print('資料表stktrade新增資料完成! ')

conn.close()

# 最後把所有trade的SQL寫出來看看...
outfname = 'sqlout.txt'
with open(outfname, 'w') as outf:
    for sqlstr in stktradeSQL:
        outf.write(sqlstr+'\n')
```

List 轉換成 SQL 存入

```python
import sqlite3

stkinfoSQL = set()
sqlstr = "INSERT INT
stkinfoSQL.

stktradeSQL
count = 0
for rec in
    # 注意順
    sqlstr
    sqlstr
    sqlstr
    stktradeSQL.appe
print('stktrade reco

# 把容器裡的SQL字串通過
def dumpSQL(conn, co
    count = 0
    for sqlstr in co
        try:
            conn.exe
            count +=
            if (cour
                prin
                conn
        except:
            pass
    print('Total', count, 'records')
    conn.commit()
```

```
stktra
資料表新
Total
資料表s
Total
資料表s
```

sqlout.txt
~/Desktop/summer.course/python.bda/proj4

```
INSERT INTO stktrade (id, date, vol, tra, val, op, hi, lo, cl) VALUES ('^IXIC', '2021-01-04', 12698.450195, 0, 0, 12958.519531, 12958.719727, 12543.240234, 12698.450195);
INSERT INTO stktrade (id, date, vol, tra, val, op, hi, lo, cl) VALUES ('^IXIC', '2021-01-05', 12818.959961, 0, 0, 12665.650391, 12828.269531, 12665.650391, 12818.959961);
INSERT INTO stktrade (id, date, vol, tra, val, op, hi, lo, cl) VALUES ('^IXIC', '2021-01-06', 12740.790039, 0, 0, 12666.150391, 12909.629883, 12649.990234, 12740.790039);
INSERT INTO stktrade (id, date, vol, tra, val, op, hi, lo, cl) VALUES ('^IXIC', '2021-01-07', 13067.480469, 0, 0, 12867.339844, 13090.910156, 12867.339844, 13067.480469);
INSERT INTO stktrade (id, date, vol, tra, val, op, hi, lo, cl) VALUES ('^IXIC', '2021-01-08', 13201.980469, 0, 0, 13160.219727, 13208.089844, 13036.549805, 13201.980469);
INSERT INTO stktrade (id, date, vol, tra, val, op, hi, lo, cl) VALUES ('^IXIC', '2021-01-11', 13036.429688, 0, 0, 13048.780273, 13138.269531, 12999.509766, 13036.429688);
INSERT INTO stktrade (id, date, vol, tra, val, op, hi, lo, cl) VALUES ('^IXIC', '2021-01-12', 13072.429688, 0, 0, 13062.059570, 13105.040039, 12963.919922, 13072.429688);
INSERT INTO stktrade (id, date, vol, tra, val, op, hi, lo, cl) VALUES ('^IXIC', '2021-01-13', 13128.950195, 0, 0, 13088.009766, 13171.150391, 13051.059570, 13128.950195);
INSERT INTO stktrade (id, date, vol, tra, val, op, hi, lo, cl) VALUES ('^IXIC', '2021-01-14', 13112.639648, 0, 0, 13174.750000, 13220.160156, 13098.410156, 13112.639648);
INSERT INTO stktrade (id, date, vol, tra, val, op, hi, lo, cl) VALUES ('^IXIC', '2021-01-15', 12998.500000, 0, 0, 13099.900391, 13139.830078, 12949.759766, 12998.500000);
INSERT INTO stktrade (id, date, vol, tra, val, op, hi, lo, cl) VALUES ('^IXIC', '2021-01-19', 13197.179688, 0, 0, 13132.730469, 13206.860352, 13078.700195, 13197.179688);
INSERT INTO stktrade (id, date, vol, tra, val, op, hi, lo, cl) VALUES ('^IXIC', '2021-01-20', 13457.250000, 0, 0, 13342.549805, 13486.129883, 13329.769531, 13457.250000);
INSERT INTO stktrade (id, date, vol, tra, val, op, hi, lo, cl) VALUES ('^IXIC', '2021-01-21', 13530.910156, 0, 0, 13521.480469, 13560.349609, 13454.070313, 13530.910156);
```

人民幣

字串

講次內容

- Yahoo 財經 - 那斯達克每日收盤資訊介紹
- 網路爬蟲與資料庫的規劃與操作
- SQL 收盤價查詢
- 專題：那斯達克收盤價走勢

SQL 收盤價查詢

- 舉例，查一下 NASDAQ 在 2021 年 2 月的收盤價為何?
- 怎麼做? 如何 SELECT-FROM-WHERE?

SQL 收盤價查詢

- 舉例，查一下 NASDAQ 在 2021 年 2 月的收盤價為何?

```
import sqlite3

sqlstr = "SELECT date, cl FROM stktrade WHERE id='^IXIC' and date>='2021-02-01' and date<'2021-03-01';"
conn = sqlite3.connect('mystock.db')
cur = conn.cursor()
results = cur.execute(sqlstr)
for rec in results:
    print(rec)
conn.commit()
conn.close()
```

SQL 收盤價查詢

- 舉例，查一下 NASDAQ 在 2021
  價為何?

```
import sqlite3

sqlstr = "SELECT date, cl FROM stktrade WHERE id='^IXIC' and date>='2
conn = sqlite3.connect('mystock.db')
cur = conn.cursor()
results = cur.execute(sqlstr)
for rec in results:
    print(rec)
conn.commit()
conn.close()
```

```
('2021-02-01', 13403.389648)
('2021-02-02', 13612.780273)
('2021-02-03', 13610.540039)
('2021-02-04', 13777.740234)
('2021-02-05', 13856.299805)
('2021-02-08', 13987.639648)
('2021-02-09', 14007.700195)
('2021-02-10', 13972.530273)
('2021-02-11', 14025.769531)
('2021-02-12', 14095.469727)
('2021-02-16', 14047.5)
('2021-02-17', 13965.490234)
('2021-02-18', 13865.360352)
('2021-02-19', 13874.459961)
('2021-02-22', 13533.049805)
('2021-02-23', 13465.200195)
('2021-02-24', 13597.969727)
('2021-02-25', 13119.429688)
('2021-02-26', 13192.349609)
```

講次內容

- Yahoo 財經 - 那斯達克每日收盤資訊介紹
- 網路爬蟲與資料庫的規劃與操作
- SQL 收盤價查詢
- 專題：那斯達克收盤價走勢

專題：人民幣匯率走勢

- 請製作那斯達克收盤資料
  - 在 2021 年的走勢
- 使用試算表作為輔助
  - 你要有基本的試算表操作基礎哦！

Google： " 試算表教學 "，趕快看一看！

那斯達克歷史資料 (^IXIC)

- 那斯達克 2021 年走勢

```
import sqlite3

# 那斯達克指數的id=^IXIC
outfname = '^IXIC.2021.csv'
sqlstr = "SELECT date, cl FROM stktrade WHERE id='^IXIC' and date>='2021-01-01' and date<'2022-01-01';"
conn = sqlite3.connect('mystock.db')
cur = conn.cursor()
results = cur.execute(sqlstr)
with open(outfname, 'w') as outf:
    for rec in results:
        outf.write(rec[0]+', '+str(rec[1])+'\n')
    conn.close()

print(outfname, '存檔完畢')
```

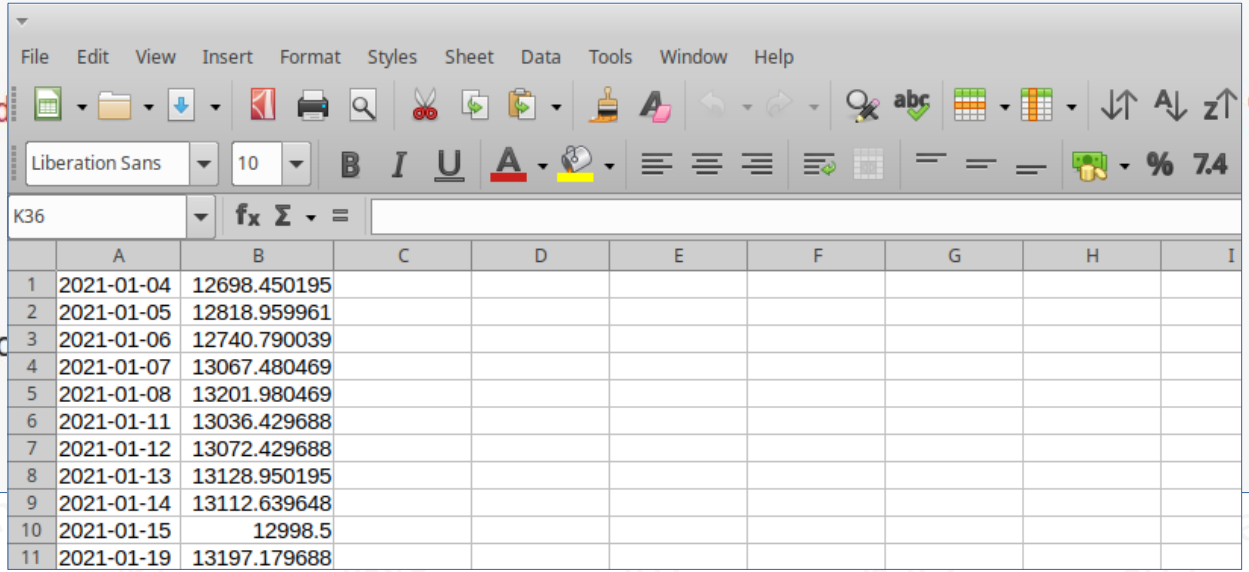
那斯達克歷史資料 (^IXIC)

- 那斯達克 2021 年走勢

```
import sqlite3

# 那斯達克指數的id=^IXIC
outfname = '^IXIC.2021.csv'
sqlstr = "SELECT date, cl FROM stktrac
conn = sqlite3.connect('mystock.db')
cur = conn.cursor()
results = cur.execute(sqlstr)
with open(outfname, 'w') as outf:
    for rec in results:
        outf.write(rec[0]+', '+str(rec
    conn.close()

print(outfname, '存檔完畢')
```



	A	B	C	D	E	F	G	H	I
1	2021-01-04	12698.450195							
2	2021-01-05	12818.959961							
3	2021-01-06	12740.790039							
4	2021-01-07	13067.480469							
5	2021-01-08	13201.980469							
6	2021-01-11	13036.429688							
7	2021-01-12	13072.429688							
8	2021-01-13	13128.950195							
9	2021-01-14	13112.639648							
10	2021-01-15	12998.5							
11	2021-01-19	13197.179688							

那斯達克歷史資料 (^IXIC)

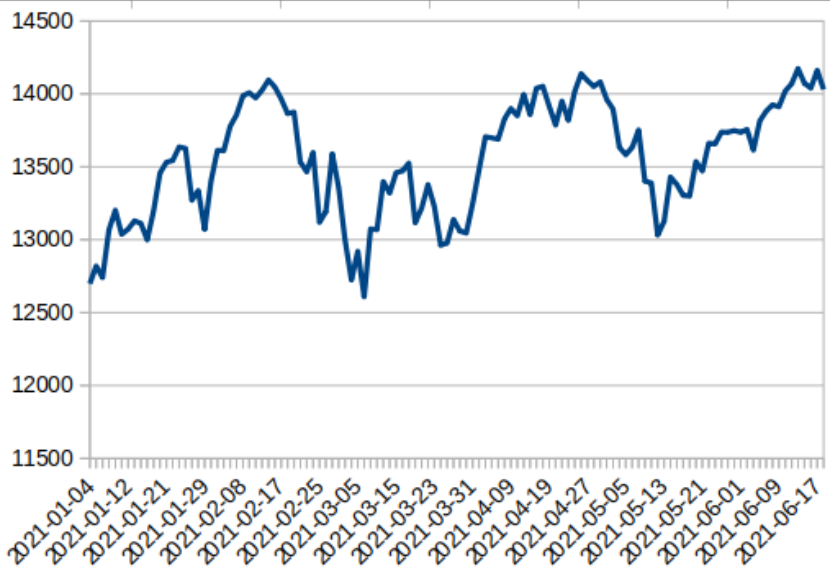
- 那斯達克 2021 年走勢

```
import sqlite

# 那斯達克指數的
outfname = '^
sqlstr = "SEL
conn = sqlite
cur = conn.cu
results = cur
with open(out
    for rec i
        outf.
    conn.clos

print(outfnam
```

	A	B
1	2021-01-04	12698.450195
2	2021-01-05	12818.959961
3	2021-01-06	12740.790039
4	2021-01-07	13067.480469
5	2021-01-08	13201.980469
6	2021-01-11	13036.429688
7	2021-01-12	13072.429688
8	2021-01-13	13128.950195
9	2021-01-14	13112.639648
10	2021-01-15	12998.5
11	2021-01-19	13197.179688
12	2021-01-20	13457.25
13	2021-01-21	13530.910156
14	2021-01-22	13543.05957
15	2021-01-25	13635.990234
16	2021-01-26	13626.05957
17	2021-01-27	13270.599609
18	2021-01-28	13337.160156
19	2021-01-29	13070.69043
20	2021-02-01	13403.389648
21	2021-02-02	13612.780273
22	2021-02-03	13610.540039
23	2021-02-04	13777.740234
24	2021-02-05	13856.299805
25	2021-02-08	13987.639648



這個講次中，你應該學到了 ...

- Yahoo 財經 - 那斯達克每日收盤資訊介紹
- 網路爬蟲與資料庫的規劃與操作
- SQL 匯率資訊查詢
- 製作那斯達克收盤價走勢