Homework 1 FIN42110 - Financial Data Science

Evan Mc Garry

February 2023

1 The Goldmansachs Group Candlestick Plot

Firstly we will take a look at the price history of The Goldmansachs Group Inc. which was extracted from Yahoo Finance using the "yfinance" repository in Python.

The below figure tracks the price changes in the GS share price which, while useful, needs to be combined with the Relative Strength Index (RSI) to understand the behaviour of traders with regards to this stock. The RSI is a momentum oscillator of the security which measures how quickly traders bid the price of the security *up* or *down* We see that in mid-September there was a significant decrease in share price and the RSI dropped below 30. Once the RSI drops below 30 this indicates that the stock is currently **oversold** and can act as a signal to buy. This was likely the case because over the next 2 months the share price rose rapidly and peaked around mid-November. Surprisingly, there was only a slight increase in shares traded during this period.

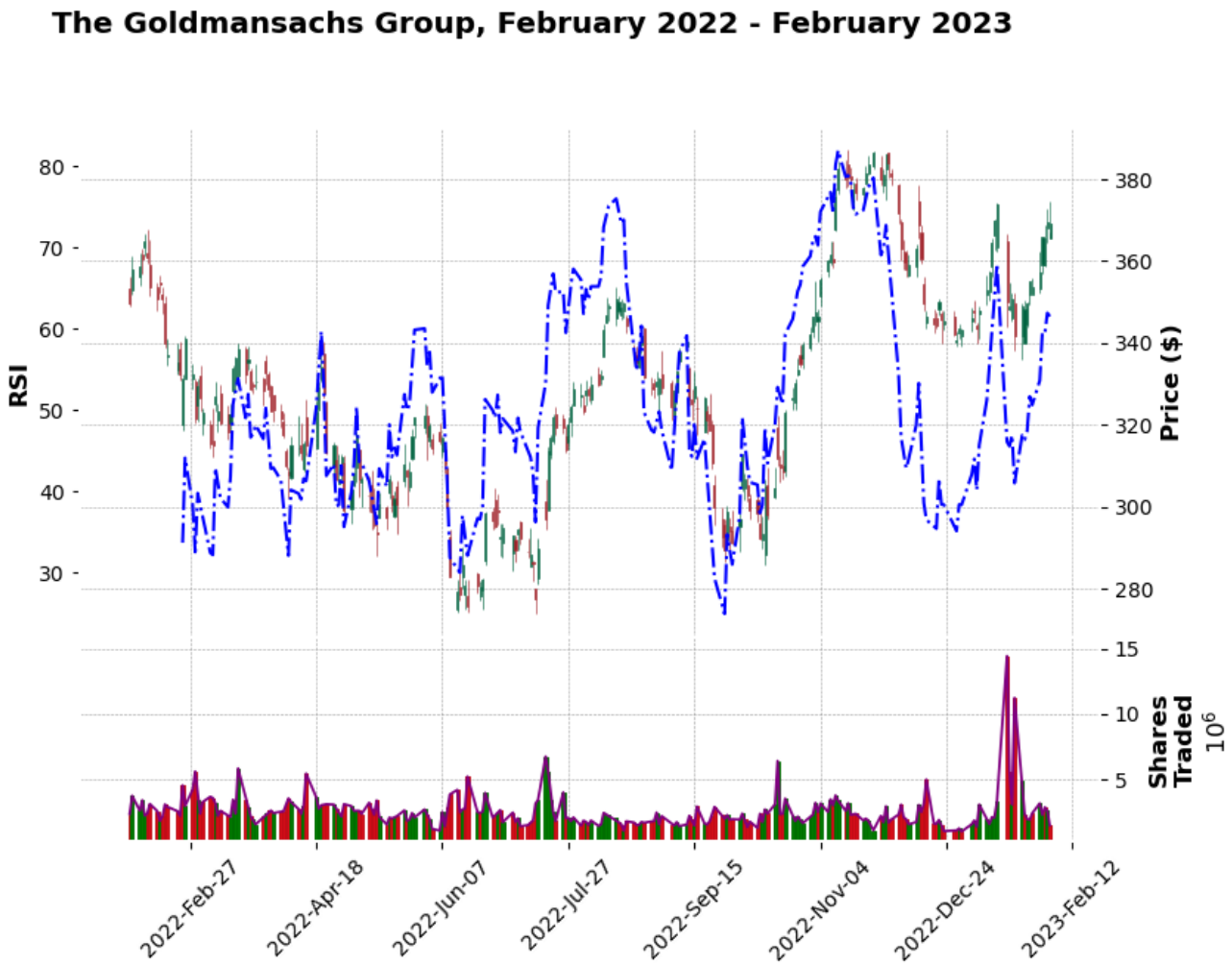


Figure 1: "Candlestick Plot" which includes the Relative Strength Index (RSI) - the blue dotted line - and a subplot of the volume of trades

However at the peak, the RSI was 80 which is very high and signifies that the stock is **overbought** and can act as a signal to sell for traders. The following 2-month period resulted in a less severe price drop but a massive increase in shares traded (likely shares sold as share price is decreasing). The RSI accurately captures

the shifts in momentum across the 12-month period, aiding in the explanation of trader's behaviour towards the stock. You can find the code used to generate this plot in Appendix A.1

2 PDF Scraping

Table 1: Gross Domestic Product 2020

	Ranking	Economy	(millions of US Dollars)
USA	1	United States	20,936,600
CHN	2	China	14,722,731
JPN	3	Japan	5,064,873
DEU	4	Germany	3,806,060
GBR	5	United Kingdom	2,707,744
IND	6	India	2,622,984
FRA	7	France	2,603,004
ITA	8	Italy	1,886,445
CAN	9	Canada	1,643,408
KOR	10	Korea, Rep.	1,630,525
RUS	11	Russian Federation	1,483,498
BRA	12	Brazil	1,444,733
AUS	13	Australia	1,330,901
ESP	14	Spain	1,281,199
MEX	15	Mexico	1,076,163
IDN	16	Indonesia	1,058,424
NLD	17	Netherlands	912,242
CHE	18	Switzerland	747,969
TUR	19	Turkey	720,101
SAU	20	Saudi Arabia	700,118
POL	21	Poland	594,165
SWE	22	Sweden	537,610
BEL	23	Belgium	515,332
THA	24	Thailand	501,795
NGA	25	Nigeria	432,294
AUT	26	Austria	428,965
ARE	27	United Arab Emirates	421,142
IRL	28	Ireland	418,622
ISR	29	Israel	401,954
ARG	30	Argentina	383,067
EGY	31	Egypt, Arab Rep.	363,069
NOR	32	Norway	362,009
PHL	33	Philippines	361,489
DNK	34	Denmark	355,184
HKG	35	Hong Kong SAR, China	346,586
SGP	36	Singapore	339,998
MYS	37	Malaysia	336,664
BGD	38	Bangladesh	324,239
ZAF	39	South Africa	301,924
COL	40	Colombia	271,347
FIN	41	Finland	271,234
VNM	42	Vietnam	271,158
PAK	43	Pakistan	263,687
CHL	44	Chile	252,940
ROU	45	Romania	248,716
CZE	46	Czech Republic	243,530
PRT	47	Portugal	231,256
NZL	48	New Zealand	212,482
PER	49	Peru	202,014
IRN	50	Iran, Islamic Rep.	191,718
GRC	51	Greece	189,410
KAZ	52	Kazakhstan	169,835
IRQ	53	Iraq	167,224
UKR	54	Ukraine	155,582
HUN	55	Hungary	155,013
QAT	56	Qatar	146,374
DZA	57	Algeria	145,164
KWT	58	Kuwait	136,197
MAR	59	Morocco	112,871
ETH	60	Ethiopia	107,645
SVK	61	Slovak Republic	104,574
PRI	62	Puerto Rico	103,138
CUB	63	Cuba	103,131
KEN	64	Kenya	98,843
ECU	65	Ecuador	98,808

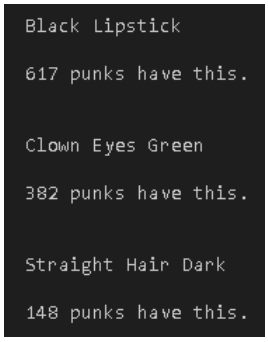
The above table represents a ranking of world economies by their Gross Domestic Product (GDP) in 2020, measured in millions of US Dollars. This was successfully scraped from an external PDF using the open-source "tabula" library of Python to return a "pandas" DataFrame. After this the DataFrame was transformed into a LaTex table using "tabula" resulting in the table above. The code used to achieve this can be found in Appendix A.2.

3 Web Scraping - CryptoPunks

Finally, the code necessary for web-scraping the attributes of a designated CryptoPunk will be provided. Firstly, the data for "CryptoPunk 11" will be scraped in an efficient way for producing the necessary text in the Python terminal. After this, there will be code provided to scrape the data in a way that directly converts the scraped data to a ".csv" file - this is useful because it allows for the data to be easily exported to an SQLite DataBase. This report will include a LaTex table below to give an understanding of how it would appear in the database.

3.1 Python Code - Basic

When the code contained in Appendix A.3 is used, the attributes are provided as an output in the terminal (IDE used = VSCode). A screenshot of this is provided below:



3.2 Python Code - Produce Table

The code provided in Appendix A.4 will produce a ".csv" Table (under the designated filename) which can be imported into an SQLite DataBase. However, as discussed, the ".csv" file is read into a LaTex table for the purposes of this report:

Title	**Description**	**Attribute No.**	**Attributes**
CryptoPunk 11	One of 3840 Female punks.	This punk has 3 attributes, one of 4501 with that many.	*Black Lipstick* **617** punks have this. *Clown Eyes Green* **382** punks have this. *Straight Hair Dark* **148** punks have this.

A Python Code

A.1 Python Code - Q1

This is the code necessary to generate the candlestick plot:

```
import matplotlib.pyplot as plt
import yfinance as yf
import pandas as pd
import matplotlib.dates as mpl_dates
import talib
import mplfinance as fplt

# Q1: Yahoo Finance - Extracting 'GS' Data
ticker_name = 'GS'
yticker = yf.Ticker(ticker_name)
GS = yticker.history(period="1y")

# Isolating & Downloading Price Information History
GS['Return'] = np.log(GS['Close']/GS['Close'].shift(1))
GS_PriceHistory = GS.iloc[:,0:5]
GS_PriceHistory

df = pd.DataFrame(GS_PriceHistory)
df.to_csv(r'C:\Users\Evan Mc Garry\Desktop\FDS\GSPrice_history.csv')
    # Removed 'Time' with Excel Formatting

# Prepare Data for Plotting
GS_df = pd.read_csv(r"C:\Users\Evan Mc Garry\Desktop\FDS\GSPrice_history.csv",
index_col=0, parse_dates=True)
dt_range = pd.date_range(start="2022-02-03", end="2023-02-03") # Full Range of Data
GS_df = GS_df[GS_df.index.isin(dt_range)]
GS_df.head()

# Prepare Relative Strength Index (RSI) for Plotting
rsi = talib.RSI(GS_df.Close)
print(rsi)

# Include RSI in Plot
RSI = fplt.make_addplot(rsi, color="blue", width=1.5, ylabel="RSI", secondary_y=True,
linestyle='dashdot')

# Include Volume of Trades in Subplot
volume = fplt.make_addplot(GS_df["Volume"], color="purple",
    panel=1
    )

# Create Candlestick Plot
fplt.plot(
    GS_df,
    type='candle',
    addplot = [RSI, volume],
    style='charles',
    title='The Goldmansachs Group, February 2022 - February 2023',
    ylabel ='Price ($)',
    volume=True,
    ylabel_lower='Shares\nTraded',
    show_nontrading=True,
    figscale=1.2,
    savefig='GS_Csp'
)
```

*

Note: This code includes text-wrapping to allow full visibility. Remove wrapping if running this code.

A.2 Python Code - Q2

This is the code necessary to successfully scrape and present the GDP Table:

```
import pandas as pd
import tabula as tb

df_new = tb.read_pdf(r"C:\Users\Evan Mc Garry\Desktop\FDS\Homework1\GDP11.pdf", area =
(89, 0, 650, 365), pages = '1', stream=True)
df_new = df_new[0]
df_new.rename(columns={'USA':'', '1':'Ranking', 'United States':'Economy', '20,936,600':
'(millions of US Dollars)'}, inplace=True)
print(df_new.columns)

new_row = pd.DataFrame({'':'USA', 'Ranking':1, 'Economy':'United States',
'(millions of US Dollars)':'20,936,600'}, index=[0])
df = pd.concat([new_row,df_new.loc[:]]).reset_index(drop=True)
[print(df)]

print(df.to_latex(index=False))
```

---

*

Note: This code includes text-wrapping to allow full visibility. Remove wrapping if running this code.

A.3 Python Code - Q3 (Basic)

This is the code that produces the necessary data in text form within the Python terminal.

```
import requests
from bs4 import BeautifulSoup
from csv import writer

# Scraping HTML Content from Webpage
Base_URL = "https://cryptopunks.app/cryptopunks/details/"
PunkNo_URL = '11'
page = requests.get(Base_URL + PunkNo_URL)

soup = BeautifulSoup(page.content, "html.parser")

# Locating Attributes
results = soup.find(id="punkDetails")
print(results.prettify)

# Isolating and Printing Attributes Data as Text
attributes = results.find("div", class_="col-md-10 col-md-offset-1").text
print(attributes)
```

A.4 Python Code - Q3 (Table)

The code below is much more useful as it allows us to tabulate the data within the IDE and import this table into SQLite.

```
import requests
from bs4 import BeautifulSoup
from csv import writer

# Scraping HTML Content from Webpage
Base_URL = "https://cryptopunks.app/cryptopunks/details/"
PunkNo_URL = '11'
page = requests.get(Base_URL + PunkNo_URL)

soup = BeautifulSoup(page.content, "html.parser")

# Locating Attributes
results = soup.find(id="punkDetails")
print(results.prettify)
attrbs = soup.find("div", class_="col-md-10 col-md-offset-1")
print(attrbs.prettify)

# Isolating 'Attributes', 'CryptoPunk Title' and 'CryptoPunk Description'
with open(r'C:\Users\Evan Mc Garry\Desktop\FDS\Homework1\CryptoPunk.csv', 'w',
encoding='utf8', newline='') as f:
    cp_writer = writer(f)
    header = ['Title', 'Description', 'Attribute No.', 'Attributes']
    cp_writer.writerow(header)
    title = results.find('h1', style="margin-top: 0px; margin-bottom: 5px;").text
    desc = results.find('h4', style="margin-top: 0px;").text
    attr_no = results.find('p', style="margin-top: 0px;").text
    attributes_3 = attrbs.find('div', class_="row").text

    data = [title, desc, attr_no, attributes_3]
    cp_writer.writerow(data)

df = pd.read_csv(r"C:\Users\Evan Mc Garry\Desktop\FDS\Homework1\CryptoPunk.csv")
print(df.to_latex(index=False))
```

---

*

Note: The "with open" line of code above includes text-wrapping to allow the line of code to be visible. Remove wrapping if running code.