Completed_Task2

December 9, 2025

[2]:
```python
from bs4 import BeautifulSoup
import requests
import pandas as pd

url = "https://en.wikipedia.org/wiki/
 ↪List_of_public_corporations_by_market_capitalization"
headers = {
    "User-Agent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) "
                  "AppleWebKit/537.36 (KHTML, like Gecko) "
                  "Chrome/120.0.0.0 Safari/537.36"
}

response = requests.get(url, headers=headers)
soup = BeautifulSoup(response.text, "lxml")

tables = soup.find_all("table", class_="wikitable")
print("Number of wikitable tables found:", len(tables))

Credit_Card_Fraud = tables[1]
row_s = Credit_Card_Fraud.find_all("tr")
data = []

for row in row_s[1:]:
    cols = row.find_all(["th", "td"])
    cols = [c.get_text(strip=True) for c in cols]
    data.append(cols)

df = pd.DataFrame(data, columns=[
    "Rank", "First quarter", "Second quarter", "Third quarter",
    "Fourth quarter", "", "", "", ""
])

df
```

```
Number of wikitable tables found: 27
```

[2]:
```
  Rank First quarter                     Second quarter Third quarter  \
0    1                              Apple3,337,000[43]
```

```
1    2                         Microsoft2,791,000[45]
2    3                            Nvidia2,644,000[44]
3    4                            Amazon2,016,000[46]
4    5                          Alphabet1,895,000[47]
5    6                              Meta1,460,000[48]
6    7                 Berkshire Hathaway1,140,000[49]
7    8                               Tesla833,529[51]
8    9                            Broadcom787,247[50]
9   10                           Eli Lilly782,950[53]

                    Fourth quarter
0            Nvidia3,850,000[44]                Nvidia4,542,000[44]
1         Microsoft3,700,000[45]             Microsoft3,850,000[45]
2             Apple3,060,000[43]                 Apple3,794,000[43]
3            Amazon2,330,000[46]              Alphabet2,975,000[47]
4          Alphabet2,150,000[47]                Amazon2,341,000[46]
5              Meta1,860,000[48]                  Meta1,845,000[48]
6          Broadcom1,300,000[50]              Broadcom1,589,000[50]
7              TSMC1,170,000[52]                 Tesla1,478,000[51]
8  Berkshire Hathaway1,050,000[49]                TSMC1,448,000[52]
9             Tesla1,020,000[51]    Berkshire Hathaway1,086,000[49]
```

[3]:
```
url = "https://en.wikipedia.org/wiki/
  ↪List_of_public_corporations_by_market_capitalization"
headers = {
    "User-Agent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) "
                  "AppleWebKit/537.36 (KHTML, like Gecko) "
                  "Chrome/120.0.0.0 Safari/537.36"
}

response = requests.get(url, headers=headers)
soup = BeautifulSoup(response.text, "lxml")

tables = soup.find_all("table", class_="wikitable")
print("Number of wikitable tables found:", len(tables))

Credit_Card_Fraud = tables[2]
row_s = Credit_Card_Fraud.find_all("tr")
data = []

for row in row_s[1:]:
    cols = row.find_all(["th", "td"])
    cols = [c.get_text(strip=True) for c in cols]
    data.append(cols)

df = pd.DataFrame(data, columns=[
    "Rank", "First quarter", "Second quarter", "Third quarter",
```

```
    "Fourth quarter", "", "", "", ""
])

df
```

```
Number of wikitable tables found: 27
```

[3]:

```
  Rank First quarter                  Second quarter Third quarter  \
0    1                        Microsoft3,126,000[45]
1    2                            Apple2,648,000[43]
2    3                           Nvidia2,259,000[44]
3    4                         Alphabet1,893,000[47]
4    5                           Amazon1,874,000[46]
5    6                             Meta1,238,000[48]
6    7                Berkshire Hathaway912,130[49]
7    8                         Eli Lilly739,660[53]
8    9                               TSMC705,690[52]
9   10                           Broadcom614,220[50]

                Fourth quarter                                    \
0            Apple3,322,000[45]           Microsoft3,543,000[43]
1        Microsoft3,230,000[43]               Apple3,198,000[45]
2           Nvidia3,182,000[44]              Nvidia2,979,000[44]
3         Alphabet2,267,000[47]            Alphabet2,058,000[47]
4           Amazon2,011,000[46]              Amazon1,956,000[46]
5             Meta1,279,000[48]                Meta1,448,000[48]
6               TSMC901,390[52]    Berkshire Hathaway993,020[49]
7  Berkshire Hathaway879,670[49]                  TSMC900,670[52]
8          Eli Lilly815,210[53]                 Tesla835,810[51]
9           Broadcom747,360[50]              Broadcom805,670[50]


0            Apple3,785,000[43]
1           Nvidia3,289,000[44]
2        Microsoft3,134,000[45]
3         Alphabet2,331,000[47]
4           Amazon2,307,000[46]
5             Meta1,478,000[48]
6            Tesla1,296,000[51]
7         Broadcom1,087,000[50]
8             TSMC1,024,000[52]
9  Berkshire Hathaway978,890[49]
```

[4]:

```
url = "https://en.wikipedia.org/wiki/
 ↪List_of_public_corporations_by_market_capitalization"
headers = {
    "User-Agent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) "
                  "AppleWebKit/537.36 (KHTML, like Gecko) "
```

```python
                  "Chrome/120.0.0.0 Safari/537.36"
}

response = requests.get(url, headers=headers)
soup = BeautifulSoup(response.text, "lxml")

tables = soup.find_all("table", class_="wikitable")
print("Number of wikitable tables found:", len(tables))

Credit_Card_Fraud = tables[3]
row_s = Credit_Card_Fraud.find_all("tr")
data = []

for row in row_s[1:]:
    cols = row.find_all(["th", "td"])
    cols = [c.get_text(strip=True) for c in cols]
    data.append(cols)

df = pd.DataFrame(data, columns=[
    "Rank", "First quarter", "Second quarter", "Third quarter",
    "Fourth quarter", "", "", "", ""
])

df
```

```
Number of wikitable tables found: 27
```

[4]:
```
  Rank First quarter                 Second quarter Third quarter  \
0    1                            Apple2,609,000[43]
1    2                        Microsoft2,146,000[45]
2    3                         Alphabet1,332,000[47]
3    4                           Amazon1,058,000[46]
4    5                             Nvidia686,090[44]
5    6                Berkshire Hathaway677,770[49]
6    7                              Tesla656,420[51]
7    8                               Meta549,480[54]
8    9                               TSMC482,410[52]
9   10                               Visa473,870[55]

                 Fourth quarter                                     \
0            Apple3,050,000[43]              Apple2,677,000[43]
1        Microsoft2,532,000[45]          Microsoft2,346,000[45]
2         Alphabet1,530,000[47]           Alphabet1,662,000[47]
3           Amazon1,337,000[46]             Amazon1,312,000[46]
4           Nvidia1,044,000[44]             Nvidia1,074,000[44]
5              Tesla829,670[51]                Tesla794,200[51]
6  Berkshire Hathaway745,010[49]                 Meta772,490[48]
```

```
7                 Meta735,450[54]    Berkshire Hathaway769,260[49]
8                 TSMC523,410[52]             Eli Lilly509,890[53]
9                 Visa497,370[55]                  Visa480,990[55]
```

```
0              Apple2,994,000[43]
1          Microsoft2,795,000[45]
2           Alphabet1,764,000[47]
3             Amazon1,570,000[46]
4             Nvidia1,223,000[44]
5                 Meta909,000[48]
6                Tesla789,930[51]
7   Berkshire Hathaway783,550[49]
8            Eli Lilly553,370[53]
9                 TSMC539,390[52]
```

[5]:

```
url = "https://en.wikipedia.org/wiki/
 ↪List_of_public_corporations_by_market_capitalization"
headers = {
    "User-Agent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) "
                  "AppleWebKit/537.36 (KHTML, like Gecko) "
                  "Chrome/120.0.0.0 Safari/537.36"
}

response = requests.get(url, headers=headers)
soup = BeautifulSoup(response.text, "lxml")

tables = soup.find_all("table", class_="wikitable")
print("Number of wikitable tables found:", len(tables))

Credit_Card_Fraud = tables[4]
row_s = Credit_Card_Fraud.find_all("tr")
data = []

for row in row_s[1:]:
    cols = row.find_all(["th", "td"])
    cols = [c.get_text(strip=True) for c in cols]
    data.append(cols)

df = pd.DataFrame(data, columns=[
    "Rank", "First quarter", "Second quarter", "Third quarter",
    "Fourth quarter", "", "", "", ""
])

df
```

```
Number of wikitable tables found: 27
```

```
[5]:   Rank First quarter                 Second quarter Third quarter  \
0     1                           Apple2,850,000[43]
1     2                       Microsoft2,311,000[45]
2     3                        Alphabet1,846,000[47]
3     4                          Amazon1,659,000[46]
4     5                           Tesla1,114,000[51]
5     6                Berkshire Hathaway779,150[49]
6     7                            Nvidia684,880[44]
7     8                              Meta605,250[54]
8     9                              TSMC540,670[52]
9    10                      UnitedHealth479,830[56]

                 Fourth quarter                                  \
0            Apple2,212,000[43]            Apple2,221,000[43]
1        Microsoft1,920,000[45]        Microsoft1,737,000[45]
2         Alphabet1,435,000[47]         Alphabet1,254,000[47]
3           Amazon1,080,000[46]           Amazon1,151,000[46]
4              Tesla697,660[51]              Tesla831,150[51]
5  Berkshire Hathaway602,450[49]  Berkshire Hathaway596,410[49]
6       UnitedHealth481,870[56]       UnitedHealth472,410[56]
7    Johnson & Johnson467,090[57]   Johnson & Johnson429,500[57]
8            Tencent445,990[59]              Visa374,380[55]
9               Meta436,390[54]              Meta364,650[60]


0            Apple2,066,000[43]
1        Microsoft1,787,000[45]
2         Alphabet1,145,000[47]
3             Amazon856,940[46]
4  Berkshire Hathaway681,770[49]
5       UnitedHealth495,370[56]
6    Johnson & Johnson461,840[57]
7          ExxonMobil454,240[58]
8               Visa439,950[55]
9            Tencent405,090[59]
```

```
[6]: import numpy as np
N_A = np.nan
#Fixing the data to my liking
#APPLE
Apple_MV = np.array([2850000, 2212000, 2221000, 2066000, 2609000, 3050000,␣
 ↪2677000, 2994000, 2648000, 3322000, 3198000, 3785000, 3337000, 3060000,␣
 ↪3794000])
print(f'Apple Market Value:', Apple_MV)

#Microsoft
```

```python
Microsoft_MV = np.array([2311000, 1920000, 1737000, 1787000, 2146000, 2532000, 2346000, 2795000, 3126000, 3230000, 3543000, 3134000, 2791000, 3700000, 3850000])
print(f'Microsoft Market Value:', Microsoft_MV)

#Alphabet
Alphabet_MV = np.array([1846000, 1435000, 1254000, 1145000, 1332000, 1530000, 1662000, 1764000, 1893000, 2267000, 2058000, 2331000, 1895000, 2150000, 2975000])
print(f'Alphabet Market Value:', Alphabet_MV)

#Amazon
Amazon_MV = np.array([1659000, 1080000, 1151000, 856940, 1058000, 1337000, 1312000, 1570000, 1874000, 2011000, 1956000, 2307000, 2016000, 2330000, 2341000])
print(f'Amazon Market Value:', Amazon_MV)

#Tesla
Tesla_MV = np.array([1114000, 697660, 831150, N_A, 656420, 829670, 794200, 789930, N_A, N_A, 835810, 1296000, 833529, 1020000, 1478000])
print(f'Tesla Market Value:', Tesla_MV)

#Berkshire Hathaway
Berkshire_Hathaway_MV = np.array([779150, 602450, 596410, 681770, 677770, 745010, 769260, 783550, 912130, 879670, 993020, 978890, 1140000, 1050000, 1086000])
print(f'Berkshire Hathaway Market Value:', Berkshire_Hathaway_MV)

#Nvidia
Nvidia_MV = np.array([684880, N_A, N_A, N_A, 686090, 1044000, 1074000, 1223000, 2259000, 3182000, 2979000, 3289000, 2644000, 3850000, 4542000])
print(f'Nvidia Market Value:', Nvidia_MV)

#Meta
Meta_MV = np.array([605250, 436390, 364650, N_A, 549480, 735450, 772490, 909000, 1238000, 1279000, 1448000, 1478000, 1460000, 1860000, 1845000])
print(f'Meta_Market Value:', Meta_MV)

#TSMC
TSMC_MV = np.array([540670, N_A, N_A, N_A, 482410, 523410, N_A, 539390, 705690, 901390, 900670, 1024000, N_A, 1170000, 1448000])
print(f'TSMC Market Value:', TSMC_MV)

#UnitedHealth
UnitedHealth_MV = np.array([479830, 481870, 472410, 495370, N_A, N_A, N_A, N_A, N_A, N_A, N_A, N_A, N_A, N_A, N_A])
```

```
print(f'UnitedHealth Market Value:', UnitedHealth_MV)

#Johnson & Johnson
Johnson_and_Johnson_MV= np.array([N_A, 467090, 429500, 461840, N_A, N_A, N_A,␣
↪N_A, N_A, N_A, N_A, N_A, N_A, N_A, N_A])
print(f'TSMC Market Value:', TSMC_MV)

#Tencent
Tencent_MV = np.array([N_A, 445990, N_A, 405090, N_A, N_A, N_A, N_A, N_A, N_A,␣
↪N_A, N_A, N_A, N_A, N_A])
print(f'Johnson and Johnson Market Value:', Johnson_and_Johnson_MV)

#Visa
Visa_MV = np.array([N_A, N_A, 374380, 439950, 473870, 497370, 480990, N_A, N_A,␣
↪N_A, N_A, N_A, N_A, N_A, N_A])
print(f'Visa Market Value:', Visa_MV)

#ExxonMobil
ExxonMobil_MV = np.array([N_A, N_A, N_A, 454240, N_A, N_A, N_A, N_A, N_A, N_A,␣
↪N_A, N_A, N_A, N_A, N_A])
print(f'ExxonMobil Market Value:', ExxonMobil_MV)

#Eli Lilly
Eli_Lilly_MV = np.array([N_A, N_A, N_A, N_A, N_A, N_A, 509890, 553370, 739660,␣
↪815210, N_A, N_A, 782950, N_A, N_A])
print(f'Eli Lilly Market Value:', Eli_Lilly_MV)

#Broadcom
Broadcom_MV = np.array([N_A, N_A, N_A, N_A, N_A, N_A, N_A, N_A, 61422, 747360,␣
↪805670, 1087000, 787247, 1300000, 1589000])
print(f'Broadcom Market Value:', Broadcom_MV)


Time = np.array([0.25, 0.50, 0.75, 1.0, 1.25, 1.50, 1.75, 2.0, 2.25, 2.50, 2.
↪75, 3.0, 3.25, 3.50, 3.75])
```

```
Apple Market Value: [2850000 2212000 2221000 2066000 2609000 3050000 2677000
2994000 2648000
 3322000 3198000 3785000 3337000 3060000 3794000]
Microsoft Market Value: [2311000 1920000 1737000 1787000 2146000 2532000 2346000
2795000 3126000
 3230000 3543000 3134000 2791000 3700000 3850000]
Alphabet Market Value: [1846000 1435000 1254000 1145000 1332000 1530000 1662000
1764000 1893000
 2267000 2058000 2331000 1895000 2150000 2975000]
Amazon Market Value: [1659000 1080000 1151000  856940 1058000 1337000 1312000
1570000 1874000
```

```
 2011000 1956000 2307000 2016000 2330000 2341000]
Tesla Market Value: [1114000.  697660.  831150.      nan  656420.  829670.
794200.  789930.
      nan      nan  835810. 1296000.  833529. 1020000. 1478000.]
Berkshire Hathaway Market Value: [ 779150  602450  596410  681770  677770
745010  769260  783550  912130
  879670  993020  978890 1140000 1050000 1086000]
Nvidia Market Value: [ 684880.      nan      nan      nan  686090. 1044000.
1074000. 1223000.
 2259000. 3182000. 2979000. 3289000. 2644000. 3850000. 4542000.]
Meta_Market Value: [ 605250.  436390.  364650.      nan  549480.  735450.
772490.  909000.
 1238000. 1279000. 1448000. 1478000. 1460000. 1860000. 1845000.]
TSMC Market Value: [ 540670.      nan      nan      nan  482410.  523410.
nan  539390.
  705690.  901390.  900670. 1024000.      nan 1170000. 1448000.]
UnitedHealth Market Value: [479830. 481870. 472410. 495370.     nan     nan
nan     nan     nan
     nan     nan     nan     nan     nan     nan]
TSMC Market Value: [ 540670.      nan      nan      nan  482410.  523410.
nan  539390.
  705690.  901390.  900670. 1024000.      nan 1170000. 1448000.]
Johnson and Johnson Market Value: [    nan 467090. 429500. 461840.     nan
nan     nan     nan     nan
     nan     nan     nan     nan     nan     nan]
Visa Market Value: [    nan     nan 374380. 439950. 473870. 497370. 480990.
nan     nan
     nan     nan     nan     nan     nan     nan]
ExxonMobil Market Value: [    nan     nan     nan 454240.     nan     nan
nan     nan     nan
     nan     nan     nan     nan     nan     nan]
Eli Lilly Market Value: [    nan     nan     nan     nan     nan     nan 509890.
553370. 739660.
 815210.     nan     nan 782950.     nan     nan]
Broadcom Market Value: [     nan      nan      nan      nan      nan      nan
nan      nan
   61422.  747360.  805670. 1087000.  787247. 1300000. 1589000.]
```

[7]:

```
# Core AI Leaders
Nvidia_MV = np.array([684880, N_A, N_A, N_A, 686090, 1044000, 1074000, 1223000,
↪2259000, 3182000, 2979000, 3289000, 2644000, 3850000, 4542000])

Microsoft_MV = np.array([2311000, 1920000, 1737000, 1787000, 2146000, 2532000,
↪2346000, 2795000, 3126000, 3230000, 3543000, 3134000, 2791000, 3700000,
↪3850000])
```

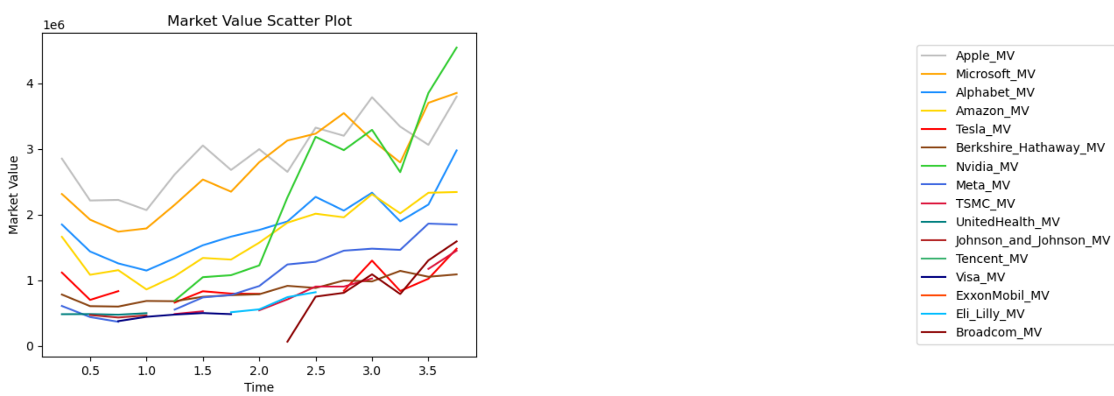
```python
Alphabet_MV = np.array([1846000, 1435000, 1254000, 1145000, 1332000, 1530000,␣
↪1662000, 1764000, 1893000, 2267000, 2058000, 2331000, 1895000, 2150000,␣
↪2975000])

Apple_MV = np.array([2850000, 2212000, 2221000, 2066000, 2609000, 3050000,␣
↪2677000, 2994000, 2648000, 3322000, 3198000, 3785000, 3337000, 3060000,␣
↪3794000])

Meta_MV = np.array([605250, 436390, 364650, N_A, 549480, 735450, 772490,␣
↪909000, 1238000, 1279000, 1448000, 1478000, 1460000, 1860000, 1845000])


# AI investors, and integrators and users
Amazon_MV = np.array([1659000, 1080000, 1151000, 856940, 1058000, 1337000,␣
↪1312000, 1570000, 1874000, 2011000, 1956000, 2307000, 2016000, 2330000,␣
↪2341000])

Tesla_MV = np.array([1114000, 697660, 831150, N_A, 656420, 829670, 794200,␣
↪789930, N_A, N_A, 835810, 1296000, 833529, 1020000, 1478000])

Broadcom_MV = np.array([N_A, N_A, N_A, N_A, N_A, N_A, N_A, N_A, 61422, 747360,␣
↪805670, 1087000, 787247, 1300000, 1589000])

TSMC_MV = np.array([540670, N_A, N_A, N_A, 482410, 523410, N_A, 539390, 705690,␣
↪901390, 900670, 1024000, N_A, 1170000, 1448000])

Tencent_MV = np.array([N_A, 445990, N_A, 405090, N_A, N_A, N_A, N_A, N_A, N_A,␣
↪N_A, N_A, N_A, N_A, N_A])



# Limitted AI Integration
Berkshire_Hathaway_MV = np.array([779150, 602450, 596410, 681770, 677770,␣
↪745010, 769260, 783550, 912130, 879670, 993020, 978890, 1140000, 1050000,␣
↪1086000])

UnitedHealth_MV = np.array([479830, 481870, 472410, 495370, N_A, N_A, N_A, N_A,␣
↪N_A, N_A, N_A, N_A, N_A, N_A, N_A])

Johnson_and_Johnson_MV= np.array([N_A, 467090, 429500, 461840, N_A, N_A, N_A,␣
↪N_A, N_A, N_A, N_A, N_A, N_A, N_A, N_A])

Visa_MV = np.array([N_A, N_A, 374380, 439950, 473870, 497370, 480990, N_A, N_A,␣
↪N_A, N_A, N_A, N_A, N_A, N_A])
```

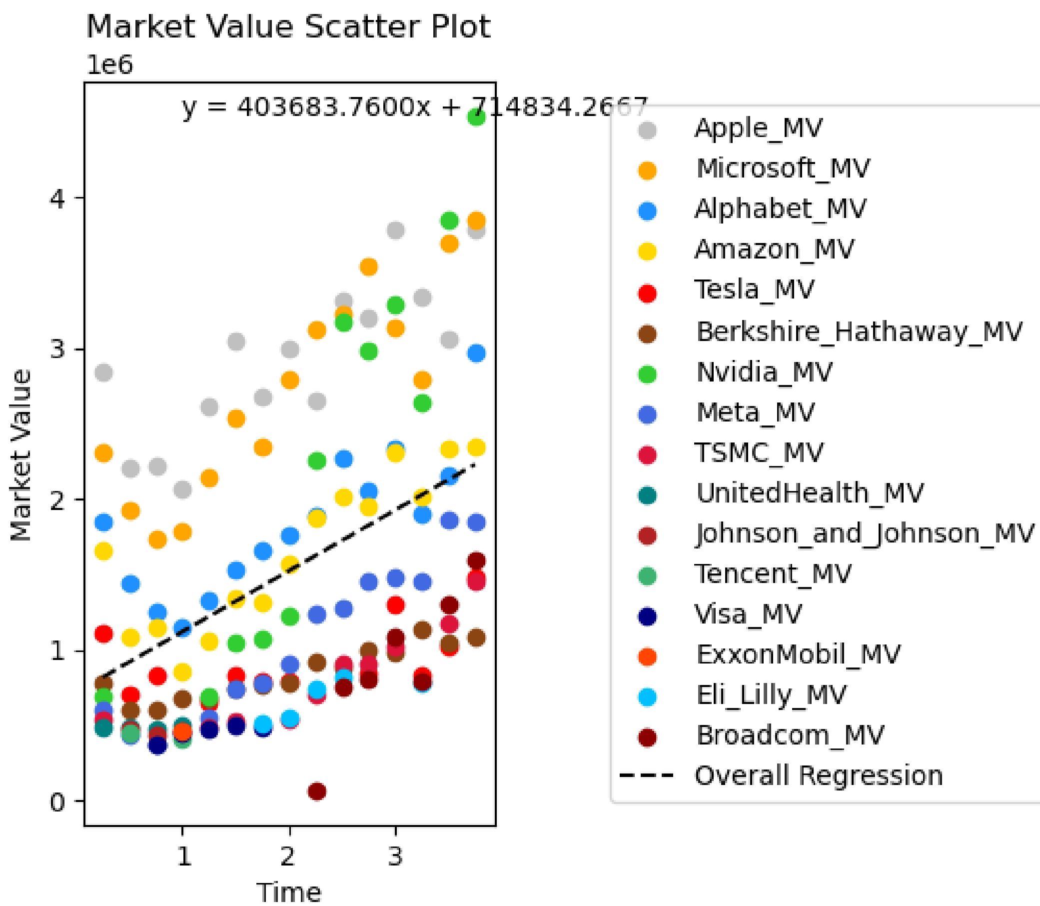
```
ExxonMobil_MV = np.array([N_A, N_A, N_A, 454240, N_A, N_A, N_A, N_A, N_A, N_A,␣
↪N_A, N_A, N_A, N_A, N_A])

Eli_Lilly_MV = np.array([N_A, N_A, N_A, N_A, N_A, N_A, 509890, 553370, 739660,␣
↪815210, N_A, N_A, 782950, N_A, N_A])
```

[22]:
```
import matplotlib.pyplot as plt

Market_Value = [Apple_MV, Microsoft_MV, Alphabet_MV, Amazon_MV, Tesla_MV,␣
↪Berkshire_Hathaway_MV, Nvidia_MV, Meta_MV, TSMC_MV, UnitedHealth_MV,␣
↪Johnson_and_Johnson_MV, Tencent_MV, Visa_MV, ExxonMobil_MV, Eli_Lilly_MV,␣
↪Broadcom_MV]
labels = ["Apple_MV", "Microsoft_MV", "Alphabet_MV", "Amazon_MV", "Tesla_MV",␣
↪"Berkshire_Hathaway_MV", "Nvidia_MV", "Meta_MV", "TSMC_MV",␣
↪"UnitedHealth_MV", "Johnson_and_Johnson_MV", "Tencent_MV", "Visa_MV",␣
↪"ExxonMobil_MV", "Eli_Lilly_MV", "Broadcom_MV"]
colors = ["silver", "orange", "dodgerblue", "gold", "red", "saddlebrown",␣
↪"limegreen", "royalblue", "crimson", "teal", "firebrick", "mediumseagreen",␣
↪"navy", "orangered", "deepskyblue", "darkred"]
Time = np.array([0.25, 0.50, 0.75, 1.0, 1.25, 1.50, 1.75, 2.0, 2.25, 2.50, 2.
↪75, 3.0, 3.25, 3.50, 3.75])



for y, label, c in zip(Market_Value, labels, colors):
    plt.plot(Time, y, label=label, color=c)

plt.xlabel("Time")
plt.ylabel("Market Value")
plt.title("Market Value Scatter Plot")

plt.legend(loc="center left", bbox_to_anchor=(2, 0.5))
plt.tight_layout()
plt.show()


for y, label, c in zip(Market_Value, labels, colors):
    plt.scatter(Time, y, label=label, color=c)


all_y = np.concatenate(Market_Value)
mask = ~np.isnan(all_y)
all_Time = np.tile(Time, len(Market_Value))


coef = np.polyfit(all_Time[mask], all_y[mask], 1)
reg_line = np.poly1d(coef)
```

```python
plt.plot(Time, reg_line(Time), color='black', linestyle='--', label="Overall Regression")

# Regression equation text
plt.text(1.0, max(all_y[mask]), f"y = {coef[0]:.4f}x + {coef[1]:.4f}", fontsize=10, color="black")



plt.xlabel("Time")
plt.ylabel("Market Value")
plt.title("Market Value Scatter Plot")
plt.legend(loc="center left", bbox_to_anchor=(1.25, 0.5))
plt.plot(Time, np.poly1d(np.polyfit(Time, y, 1))(Time), label=f"y={np.polyfit(Time, y, 1)[0]:.4f}x+{np.polyfit(Time, y, 1)[1]:.4f}")

plt.tight_layout()
plt.show()
```

```
/var/folders/9r/bk9mfv2548537j8q9xs0nbtc0000gn/T/ipykernel_605/3745716232.py:18:
UserWarning: Tight layout not applied. The left and right margins cannot be made
large enough to accommodate all axes decorations.
  plt.tight_layout()
```





[23]:

```
Market_Value = [Apple_MV, Microsoft_MV, Alphabet_MV, Amazon_MV, Tesla_MV, Berkshire_Hathaway_MV, Nvidia_MV, Meta_MV, TSMC_MV, UnitedHealth_MV, Johnson_and_Johnson_MV, Tencent_MV, Visa_MV, ExxonMobil_MV, Eli_Lilly_MV, Broadcom_MV]
labels = ["Apple_MV", "Microsoft_MV", "Alphabet_MV", "Amazon_MV", "Tesla_MV", "Berkshire_Hathaway_MV", "Nvidia_MV", "Meta_MV", "TSMC_MV", "UnitedHealth_MV", "Johnson_and_Johnson_MV", "Tencent_MV", "Visa_MV", "ExxonMobil_MV", "Eli_Lilly_MV", "Broadcom_MV"]
colors = ["silver", "orange", "dodgerblue", "gold", "red", "saddlebrown", "limegreen", "royalblue", "crimson", "teal", "firebrick", "mediumseagreen", "navy", "orangered", "deepskyblue", "darkred"]
Time = np.array([0.25, 0.50, 0.75, 1.0, 1.25, 1.50, 1.75, 2.0, 2.25, 2.50, 2.75, 3.0, 3.25, 3.50, 3.75])


plt.figure(figsize=(6, 6))
```

```
Market_Value = [Apple_MV]
labels = ["Apple_MV"]
colors = ["silver"]
Time = np.array([0.25, 0.50, 0.75, 1.0, 1.25, 1.50, 1.75, 2.0, 2.25, 2.50, 2.
↪75, 3.0, 3.25, 3.50, 3.75])

for y, label, c in zip(Market_Value, labels, colors):
    plt.scatter(Time, y, label=label, color=c)

    # Linear regression
    coef = np.polyfit(Time, y, 1)  # slope and intercept
    reg_line = np.poly1d(coef)
    plt.plot(Time, reg_line(Time), color='black', linestyle='--',␣
↪label='Regression Line')

    # Equation next to the legend
    plt.text(2.0, max(y), f"y = {coef[0]:.4f}x + {coef[1]:.4f}", fontsize=10)

plt.xlabel("Time")
plt.ylabel("Market Value")
plt.title("Apple Market Value Scatter Plot")
plt.legend(loc="center left", bbox_to_anchor=(1.25, 0.5))
plt.tight_layout()
plt.show()





Market_Value = [Microsoft_MV]
labels = ["Microsoft_MV"]
colors = ["orange"]
Time = np.array([0.25, 0.50, 0.75, 1.0, 1.25, 1.50, 1.75, 2.0, 2.25, 2.50, 2.
↪75, 3.0, 3.25, 3.50, 3.75])


for y, label, c in zip(Market_Value, labels, colors):
    plt.scatter(Time, y, label=label, color=c)


    # Linear regression
    coef = np.polyfit(Time, y, 1)
    reg_line = np.poly1d(coef)
```

```python
    plt.plot(Time, reg_line(Time), color='black', linestyle='--',␣
↪label='Regression Line')

    # Insert regression equation
    plt.text(2.0, max(y), f"y = {coef[0]:.4f}x + {coef[1]:.4f}", fontsize=10)


plt.xlabel("Time")
plt.ylabel("Market Value")
plt.title("Microsoft Market Value Scatter Plot")
plt.legend(loc="center left", bbox_to_anchor=(1.25, 0.5))
plt.plot(Time, np.poly1d(np.polyfit(Time, y, 1))(Time), label=f"y={np.
↪polyfit(Time, y, 1)[0]:.4f}x+{np.polyfit(Time, y, 1)[1]:.4f}")

plt.tight_layout()
plt.show()





Market_Value = [Alphabet_MV]
labels = ["Alphabet_MV"]
colors = ["dodgerblue"]
Time = np.array([0.25, 0.50, 0.75, 1.0, 1.25, 1.50, 1.75, 2.0, 2.25, 2.50, 2.
↪75, 3.0, 3.25, 3.50, 3.75])

for y, label, c in zip(Market_Value, labels, colors):
    plt.scatter(Time, y, label=label, color=c)



    # Linear regression
    coef = np.polyfit(Time, y, 1)
    reg_line = np.poly1d(coef)
    plt.plot(Time, reg_line(Time), color='black', linestyle='--',␣
↪label='Regression Line')

    # Insert regression equation
    plt.text(2.0, max(y), f"y = {coef[0]:.4f}x + {coef[1]:.4f}", fontsize=10)


plt.xlabel("Time")
plt.ylabel("Market Value")
plt.title("Alphabet Market Value Scatter Plot")
plt.legend(loc="center left", bbox_to_anchor=(1.25, 0.5))
```

```
plt.plot(Time, np.poly1d(np.polyfit(Time, y, 1))(Time), label=f"y={np.
 ↪polyfit(Time, y, 1)[0]:.4f}x+{np.polyfit(Time, y, 1)[1]:.4f}")

plt.tight_layout()
plt.show()




Market_Value = [Amazon_MV]
labels = ["Amazon_MV"]
colors = ["gold"]

for y, label, c in zip(Market_Value, labels, colors):
    plt.scatter(Time, y, label=label, color=c)


    # Linear regression
    coef = np.polyfit(Time, y, 1)
    reg_line = np.poly1d(coef)
    plt.plot(Time, reg_line(Time), color='black', linestyle='--',␣
 ↪label='Regression Line')

    # Insert regression equation
    plt.text(2.0, max(y), f"y = {coef[0]:.4f}x + {coef[1]:.4f}", fontsize=10)


plt.xlabel("Time")
plt.ylabel("Market Value")
plt.title("Amazon Market Value Scatter Plot")
plt.legend(loc="center left", bbox_to_anchor=(1.25, 0.5))
plt.plot(Time, np.poly1d(np.polyfit(Time, y, 1))(Time), label=f"y={np.
 ↪polyfit(Time, y, 1)[0]:.4f}x+{np.polyfit(Time, y, 1)[1]:.4f}")

plt.tight_layout()
plt.show()




Market_Value = [Tesla_MV]
labels = ["Tesla_MV"]
colors = ["red"]

for y, label, c in zip(Market_Value, labels, colors):
    plt.scatter(Time, y, label=label, color=c)
```

```
    # FIX for N_A values
    mask = ~np.isnan(y)

    # Linear regression
    coef = np.polyfit(Time[mask], y[mask], 1)
    reg_line = np.poly1d(coef)
    plt.plot(Time[mask], reg_line(Time[mask]), color='black', linestyle='--',␣
 ↪label='Regression Line')

    # Insert regression equation
    plt.text(2.0, max(y[mask]), f"y = {coef[0]:.4f}x + {coef[1]:.4f}",␣
 ↪fontsize=10)


plt.xlabel("Time")
plt.ylabel("Market Value")
plt.title("Tesla Market Value Scatter Plot")
plt.legend(loc="center left", bbox_to_anchor=(1.25, 0.5))
plt.plot(Time, np.poly1d(np.polyfit(Time, y, 1))(Time), label=f"y={np.
 ↪polyfit(Time, y, 1)[0]:.4f}x+{np.polyfit(Time, y, 1)[1]:.4f}")

plt.tight_layout()
plt.show()



Market_Value = [Berkshire_Hathaway_MV]
labels = ["Berkshire_Hathaway_MV"]
colors = ["saddlebrown"]

for y, label, c in zip(Market_Value, labels, colors):
    plt.scatter(Time, y, label=label, color=c)


    # Linear regression
    coef = np.polyfit(Time, y, 1)
    reg_line = np.poly1d(coef)
    plt.plot(Time, reg_line(Time), color='black', linestyle='--',␣
 ↪label='Regression Line')

    # Insert regression equation
    plt.text(2.0, max(y), f"y = {coef[0]:.4f}x + {coef[1]:.4f}", fontsize=10)


plt.xlabel("Time")
plt.ylabel("Market Value")
plt.title("Berkshire Hathaway Market Value Scatter Plot")
```

```
plt.legend(loc="center left", bbox_to_anchor=(1.25, 0.5))
plt.plot(Time, np.poly1d(np.polyfit(Time, y, 1))(Time), label=f"y={np.
↪polyfit(Time, y, 1)[0]:.4f}x+{np.polyfit(Time, y, 1)[1]:.4f}")

plt.tight_layout()
plt.show()




Market_Value = [Nvidia_MV]
labels = ["Nvidia_MV"]
colors = ["limegreen"]

for y, label, c in zip(Market_Value, labels, colors):
    plt.scatter(Time, y, label=label, color=c)


    # FIX for N_A values
    mask = ~np.isnan(y)

    # Linear regression
    coef = np.polyfit(Time[mask], y[mask], 1)
    reg_line = np.poly1d(coef)
    plt.plot(Time[mask], reg_line(Time[mask]), color='black', linestyle='--',␣
↪label='Regression Line')

    # Insert regression equation
    plt.text(2.0, max(y[mask]), f"y = {coef[0]:.4f}x + {coef[1]:.4f}",␣
↪fontsize=10)


plt.xlabel("Time")
plt.ylabel("Market Value")
plt.title("Nvidia Market Value Scatter Plot")
plt.legend(loc="center left", bbox_to_anchor=(1.25, 0.5))
plt.plot(Time, np.poly1d(np.polyfit(Time, y, 1))(Time), label=f"y={np.
↪polyfit(Time, y, 1)[0]:.4f}x+{np.polyfit(Time, y, 1)[1]:.4f}")

plt.tight_layout()
plt.show()



Market_Value = [Meta_MV]
labels = ["Meta_MV"]
colors = ["royalblue"]
```

```
for y, label, c in zip(Market_Value, labels, colors):
    plt.scatter(Time, y, label=label, color=c)

    # FIX for N_A values
    mask = ~np.isnan(y)

    # Linear regression
    coef = np.polyfit(Time[mask], y[mask], 1)
    reg_line = np.poly1d(coef)
    plt.plot(Time[mask], reg_line(Time[mask]), color='black', linestyle='--',␣
 ↪label='Regression Line')

    # Insert regression equation
    plt.text(2.0, max(y[mask]), f"y = {coef[0]:.4f}x + {coef[1]:.4f}",␣
 ↪fontsize=10)


plt.xlabel("Time")
plt.ylabel("Market Value")
plt.title("Meta Market Value Scatter Plot")
plt.legend(loc="center left", bbox_to_anchor=(1.25, 0.5))
plt.plot(Time, np.poly1d(np.polyfit(Time, y, 1))(Time), label=f"y={np.
 ↪polyfit(Time, y, 1)[0]:.4f}x+{np.polyfit(Time, y, 1)[1]:.4f}")

plt.tight_layout()
plt.show()



Market_Value = [TSMC_MV]
labels = ["TSMC_MV"]
colors = ["crimson"]

for y, label, c in zip(Market_Value, labels, colors):
    plt.scatter(Time, y, label=label, color=c)


    # FIX for N_A values
    mask = ~np.isnan(y)

    # Linear regression
    coef = np.polyfit(Time[mask], y[mask], 1)
    reg_line = np.poly1d(coef)
    plt.plot(Time[mask], reg_line(Time[mask]), color='black', linestyle='--',␣
 ↪label='Regression Line')
```

```
    # Insert regression equation
    plt.text(2.0, max(y[mask]), f"y = {coef[0]:.4f}x + {coef[1]:.4f}",␣
↪fontsize=10)


plt.xlabel("Time")
plt.ylabel("Market Value")
plt.title("TSMC Market Value Scatter Plot")
plt.legend(loc="center left", bbox_to_anchor=(1.25, 0.5))
plt.plot(Time, np.poly1d(np.polyfit(Time, y, 1))(Time), label=f"y={np.
↪polyfit(Time, y, 1)[0]:.4f}x+{np.polyfit(Time, y, 1)[1]:.4f}")

plt.tight_layout()
plt.show()




Market_Value = [UnitedHealth_MV]
labels = ["UnitedHealth_MV"]
colors = ["teal"]

for y, label, c in zip(Market_Value, labels, colors):
    plt.scatter(Time, y, label=label, color=c)


    # FIX for N_A values
    mask = ~np.isnan(y)

    # Linear regression
    coef = np.polyfit(Time[mask], y[mask], 1)
    reg_line = np.poly1d(coef)
    plt.plot(Time[mask], reg_line(Time[mask]), color='black', linestyle='--',␣
↪label='Regression Line')

    # Insert regression equation
    plt.text(2.0, max(y[mask]), f"y = {coef[0]:.4f}x + {coef[1]:.4f}",␣
↪fontsize=10)


plt.xlabel("Time")
plt.ylabel("Market Value")
plt.title("UnitedHealth Market Value Scatter Plot")
plt.legend(loc="center left", bbox_to_anchor=(1.25, 0.5))
plt.plot(Time, np.poly1d(np.polyfit(Time, y, 1))(Time), label=f"y={np.
↪polyfit(Time, y, 1)[0]:.4f}x+{np.polyfit(Time, y, 1)[1]:.4f}")
```

```
plt.tight_layout()
plt.show()



Market_Value = [Johnson_and_Johnson_MV]
labels = ["Johnson_and_Johnson_MV"]
colors = ["firebrick"]

for y, label, c in zip(Market_Value, labels, colors):
    plt.scatter(Time, y, label=label, color=c)


    # FIX for N_A values
    mask = ~np.isnan(y)

    # Linear regression
    coef = np.polyfit(Time[mask], y[mask], 1)
    reg_line = np.poly1d(coef)
    plt.plot(Time[mask], reg_line(Time[mask]), color='black', linestyle='--',␣
 ↪label='Regression Line')

    # Insert regression equation
    plt.text(2.0, max(y[mask]), f"y = {coef[0]:.4f}x + {coef[1]:.4f}",␣
 ↪fontsize=10)


plt.xlabel("Time")
plt.ylabel("Market Value")
plt.title("Johnson and Johnson Market Value Scatter Plot")
plt.legend(loc="center left", bbox_to_anchor=(1.25, 0.5))
plt.plot(Time, np.poly1d(np.polyfit(Time, y, 1))(Time), label=f"y={np.
 ↪polyfit(Time, y, 1)[0]:.4f}x+{np.polyfit(Time, y, 1)[1]:.4f}")

plt.tight_layout()
plt.show()




Market_Value = [Tencent_MV]
labels = ["Tencent_MV"]
colors = ["mediumseagreen"]

for y, label, c in zip(Market_Value, labels, colors):
    plt.scatter(Time, y, label=label, color=c)
```

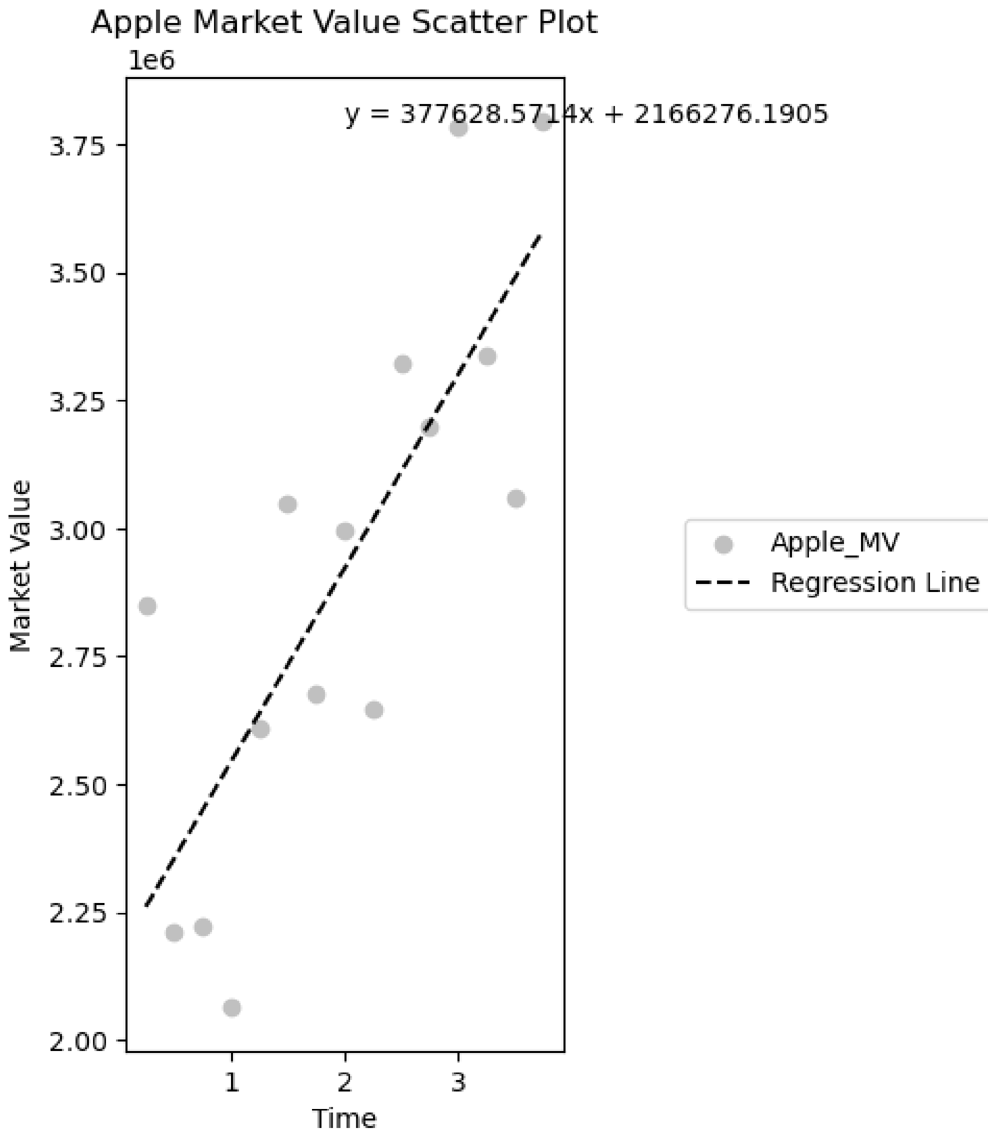
```
    # FIX for N_A values
    mask = ~np.isnan(y)

    # Linear regression
    coef = np.polyfit(Time[mask], y[mask], 1)
    reg_line = np.poly1d(coef)
    plt.plot(Time[mask], reg_line(Time[mask]), color='black', linestyle='--',␣
 ↪label='Regression Line')

    # Insert regression equation
    plt.text(2.0, max(y[mask]), f"y = {coef[0]:.4f}x + {coef[1]:.4f}",␣
 ↪fontsize=10)


plt.xlabel("Time")
plt.ylabel("Market Value")
plt.title("Tencent Market Value Scatter Plot")
plt.legend(loc="center left", bbox_to_anchor=(1.25, 0.5))
plt.plot(Time, np.poly1d(np.polyfit(Time, y, 1))(Time), label=f"y={np.
 ↪polyfit(Time, y, 1)[0]:.4f}x+{np.polyfit(Time, y, 1)[1]:.4f}")

plt.tight_layout()
plt.show()



Market_Value = [Visa_MV]
labels = ["Visa_MV"]
colors = ["navy"]

for y, label, c in zip(Market_Value, labels, colors):
    plt.scatter(Time, y, label=label, color=c)


    # FIX for N_A values
    mask = ~np.isnan(y)

    # Linear regression
    coef = np.polyfit(Time[mask], y[mask], 1)
    reg_line = np.poly1d(coef)
    plt.plot(Time[mask], reg_line(Time[mask]), color='black', linestyle='--',␣
 ↪label='Regression Line')

    # Insert regression equation
    plt.text(2.0, max(y[mask]), f"y = {coef[0]:.4f}x + {coef[1]:.4f}",␣
 ↪fontsize=10)
```

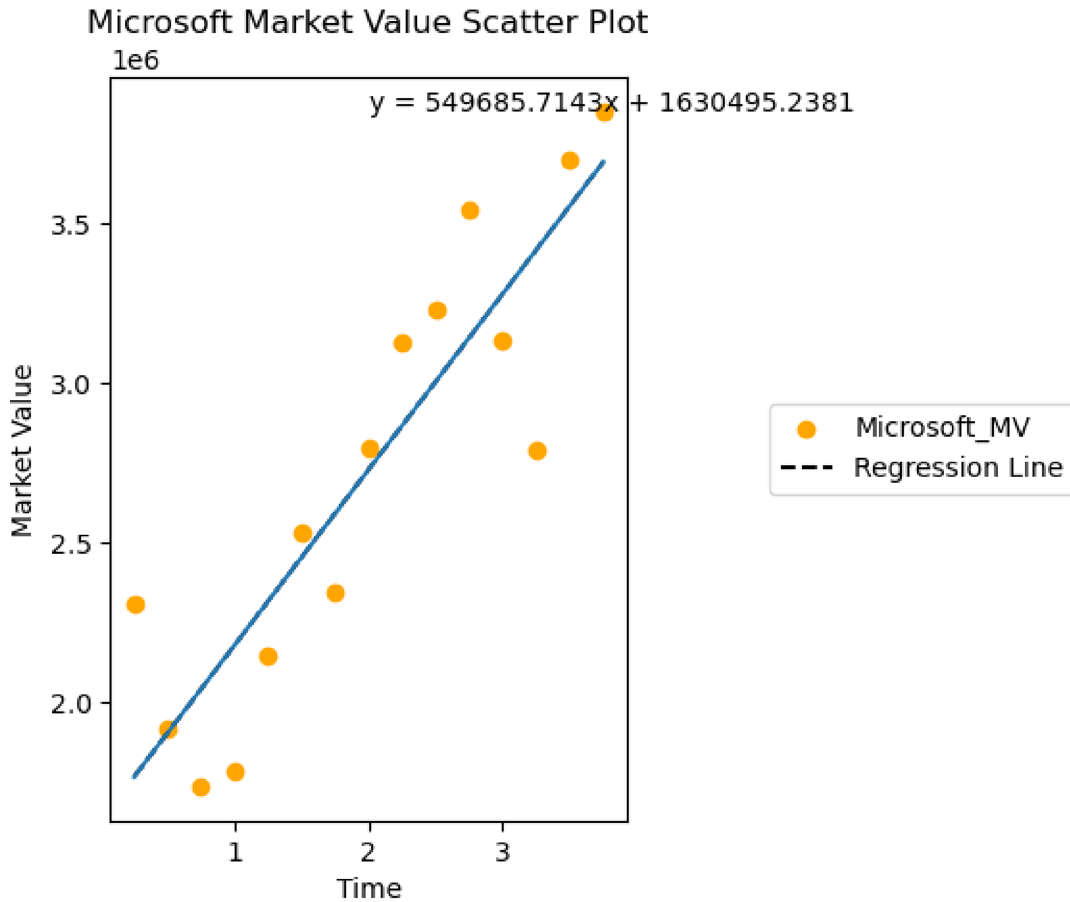
```
plt.xlabel("Time")
plt.ylabel("Market Value")
plt.title("Visa Market Value Scatter Plot")
plt.legend(loc="center left", bbox_to_anchor=(1.25, 0.5))
plt.plot(Time, np.poly1d(np.polyfit(Time, y, 1))(Time), label=f"y={np.
 ↪polyfit(Time, y, 1)[0]:.4f}x+{np.polyfit(Time, y, 1)[1]:.4f}")

plt.tight_layout()
plt.show()
```

```
Market_Value = [ExxonMobil_MV]
labels = ["ExxonMobil_MV"]
colors = ["orangered"]

for y, label, c in zip(Market_Value, labels, colors):
    plt.scatter(Time, y, label=label, color=c)


    # FIX for N_A values
    mask = ~np.isnan(y)

    # Linear regression
    coef = np.polyfit(Time[mask], y[mask], 1)
    reg_line = np.poly1d(coef)
    plt.plot(Time[mask], reg_line(Time[mask]), color='black', linestyle='--',␣
 ↪label='Regression Line')

    # Insert regression equation
    plt.text(2.0, max(y[mask]), f"y = {coef[0]:.4f}x + {coef[1]:.4f}",␣
 ↪fontsize=10)


plt.xlabel("Time")
plt.ylabel("Market Value")
plt.title("ExxonMobil Market Value Scatter Plot")
plt.legend(loc="center left", bbox_to_anchor=(1.25, 0.5))
plt.plot(Time, np.poly1d(np.polyfit(Time, y, 1))(Time), label=f"y={np.
 ↪polyfit(Time, y, 1)[0]:.4f}x+{np.polyfit(Time, y, 1)[1]:.4f}")

plt.tight_layout()
plt.show()
```

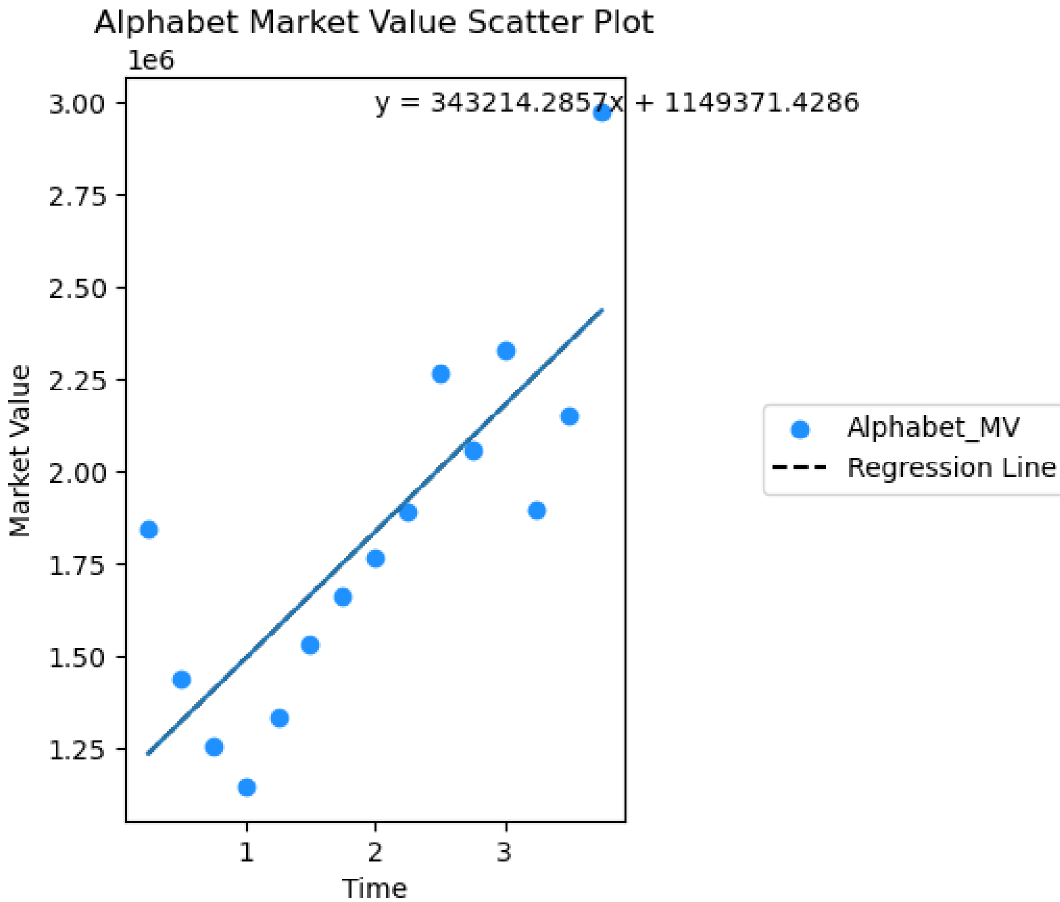
```
Market_Value = [Eli_Lilly_MV]
labels = ["Eli_Lilly_MV"]
colors = ["deepskyblue"]

for y, label, c in zip(Market_Value, labels, colors):
    plt.scatter(Time, y, label=label, color=c)


    # FIX for N_A values
    mask = ~np.isnan(y)

    # Linear regression
    coef = np.polyfit(Time[mask], y[mask], 1)
    reg_line = np.poly1d(coef)
    plt.plot(Time[mask], reg_line(Time[mask]), color='black', linestyle='--',␣
↪label='Regression Line')

    # Insert regression equation
    plt.text(2.0, max(y[mask]), f"y = {coef[0]:.4f}x + {coef[1]:.4f}",␣
↪fontsize=10)


plt.xlabel("Time")
plt.ylabel("Market Value")
plt.title("Eli Lilly Market Value Scatter Plot")
plt.legend(loc="center left", bbox_to_anchor=(1.25, 0.5))
plt.plot(Time, np.poly1d(np.polyfit(Time, y, 1))(Time), label=f"y={np.
↪polyfit(Time, y, 1)[0]:.4f}x+{np.polyfit(Time, y, 1)[1]:.4f}")

plt.tight_layout()
plt.show()



Market_Value = [Broadcom_MV]
labels = ["Broadcom_MV"]
colors = ["darkred"]

for y, label, c in zip(Market_Value, labels, colors):
    plt.scatter(Time, y, label=label, color=c)



    # FIX for N_A values
    mask = ~np.isnan(y)
```

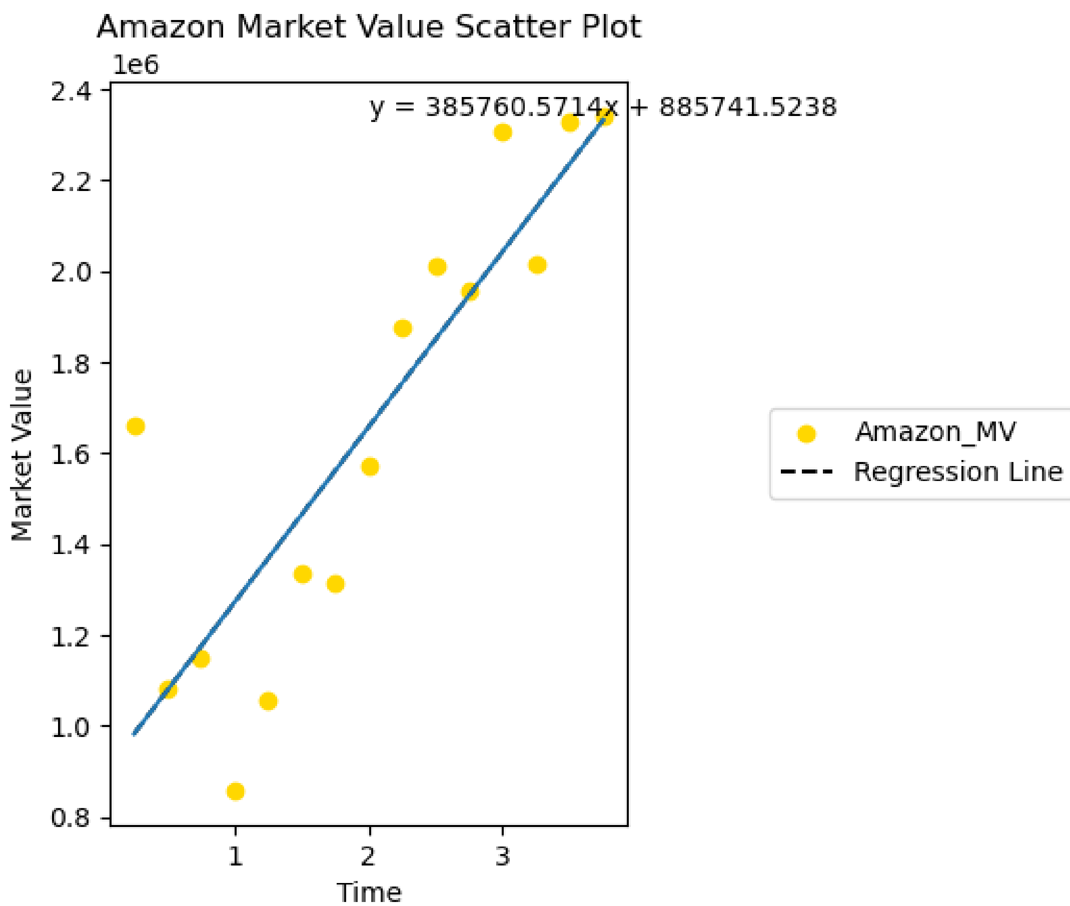
```
    # Linear regression
    coef = np.polyfit(Time[mask], y[mask], 1)
    reg_line = np.poly1d(coef)
    plt.plot(Time[mask], reg_line(Time[mask]), color='black', linestyle='--',␣
 ↪label='Regression Line')

    # Insert regression equation
    plt.text(2.0, max(y[mask]), f"y = {coef[0]:.4f}x + {coef[1]:.4f}",␣
 ↪fontsize=10)


plt.xlabel("Time")
plt.ylabel("Market Value")
plt.title("Broadcom Market Value Scatter Plot")
plt.legend(loc="center left", bbox_to_anchor=(1.25, 0.5))
plt.plot(Time, np.poly1d(np.polyfit(Time, y, 1))(Time), label=f"y={np.
 ↪polyfit(Time, y, 1)[0]:.4f}x+{np.polyfit(Time, y, 1)[1]:.4f}")

plt.tight_layout()
plt.show()
```

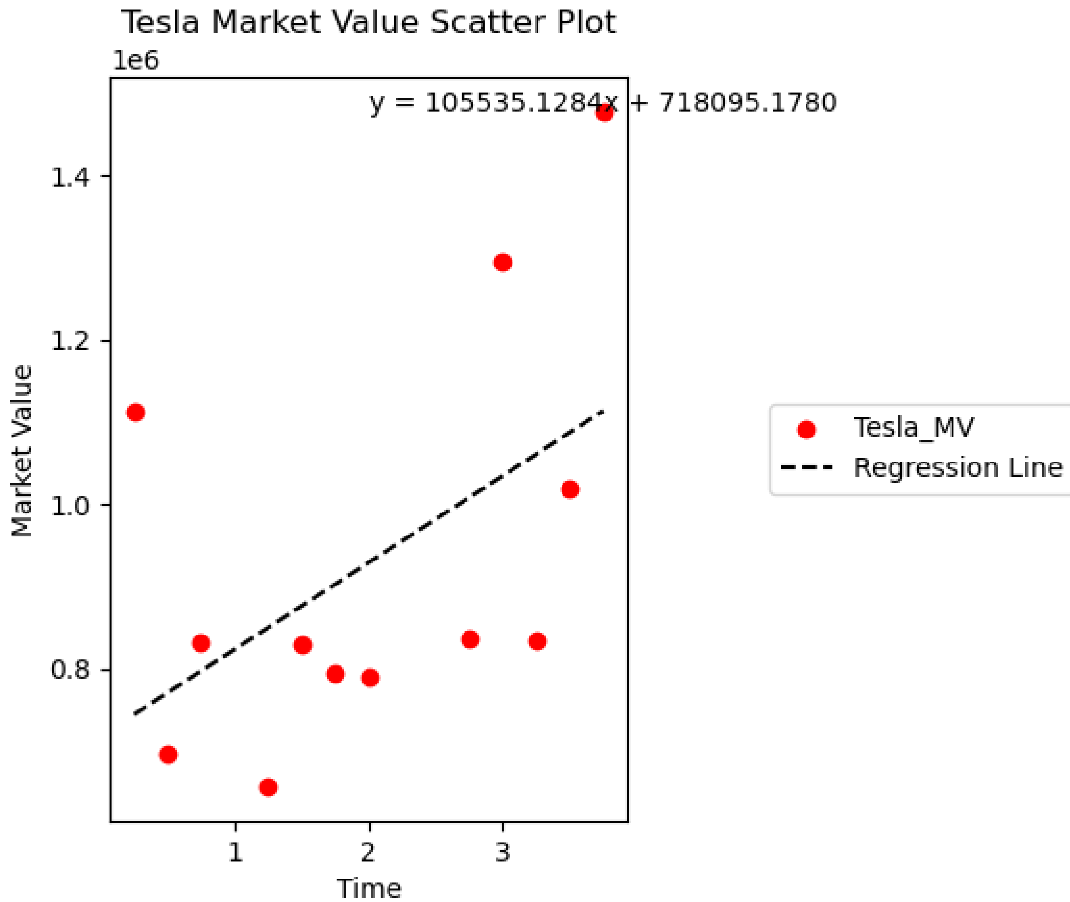

Apple Market Value Scatter Plot
1e6
y = 377628.5714x + 2166276.1905
3.75
3.50
3.25
3.00
2.75
2.50
2.25
2.00
Market Value
1
2
3
Time
Apple_MV
Regression Line

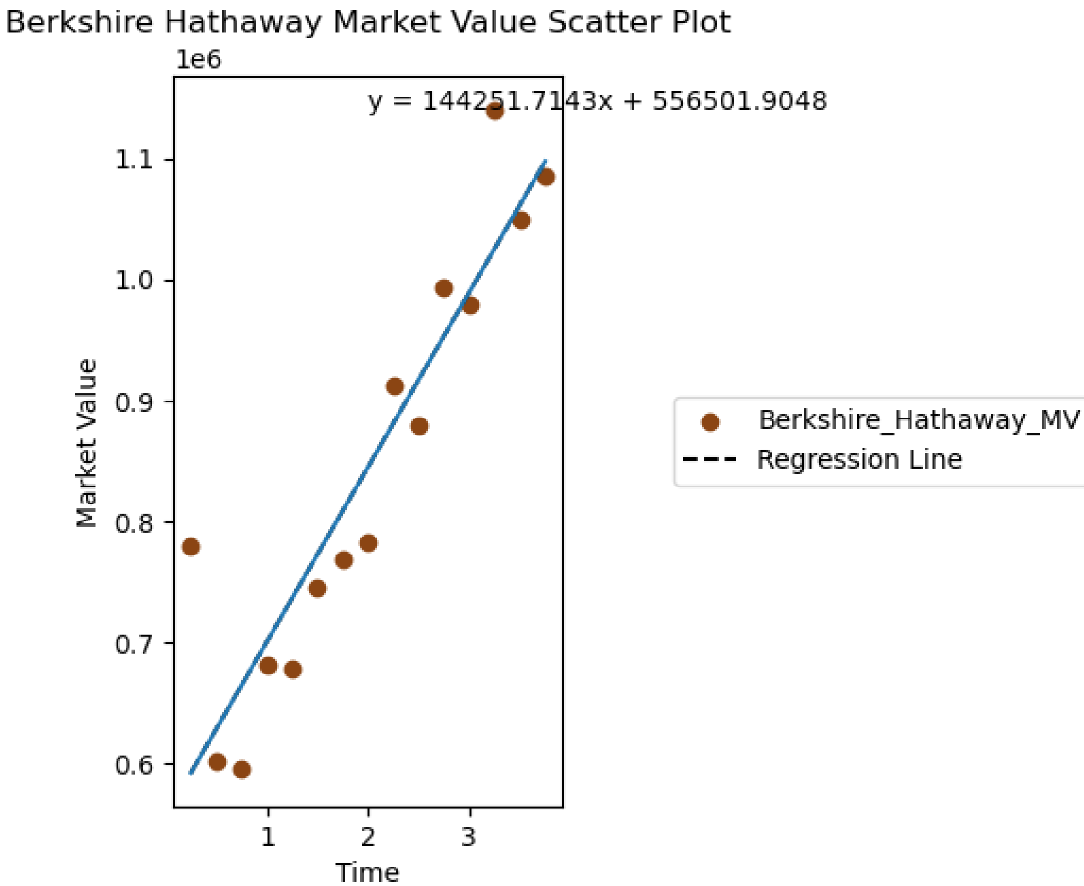


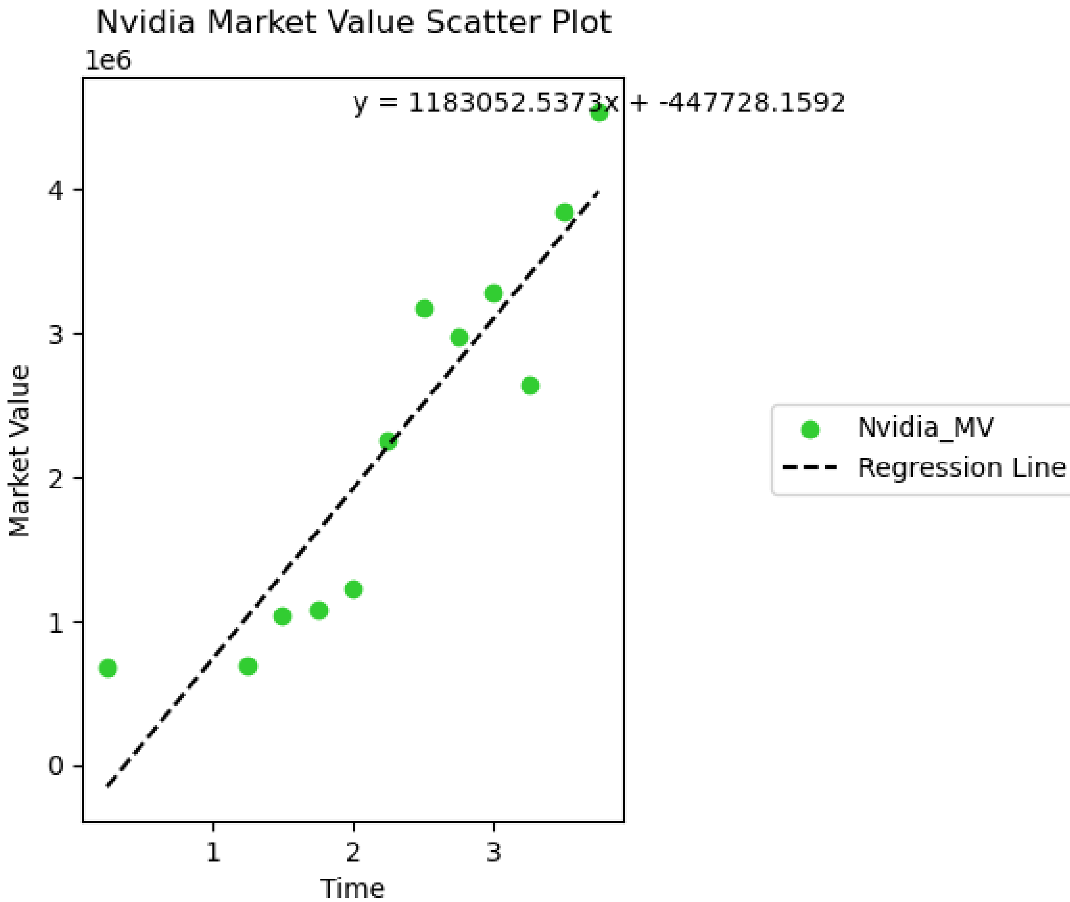
Microsoft Market Value Scatter Plot
1e6
y = 549685.7143x + 1630495.2381
3.5
3.0
2.5
2.0
Market Value
1
2
3
Time
Microsoft_MV
Regression Line

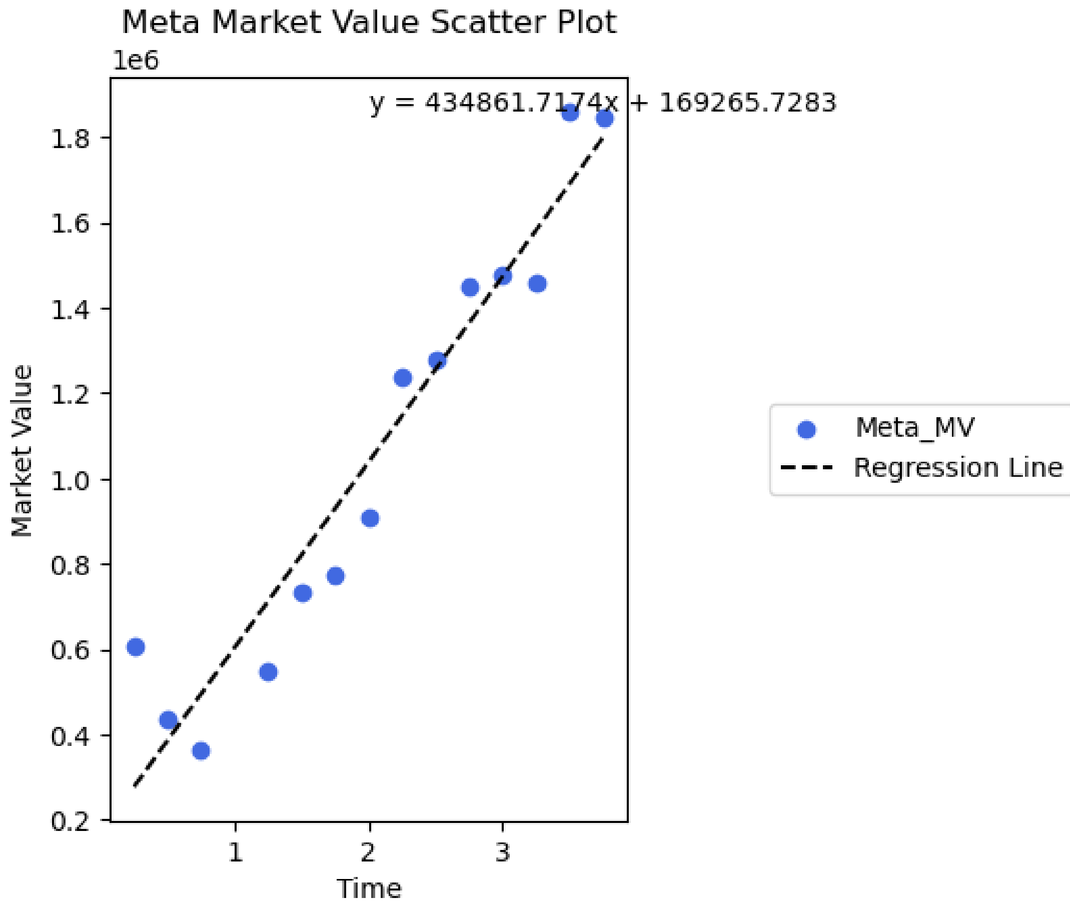

Alphabet Market Value Scatter Plot
1e6
y = 343214.2857x + 1149371.4286
3.00
2.75
2.50
2.25
2.00
1.75
1.50
1.25
Market Value
1
2
3
Time
Alphabet_MV
Regression Line


Amazon Market Value Scatter Plot
1e6
2.4
2.2
2.0
1.8
1.6
1.4
1.2
1.0
0.8
Market Value
y = 385760.5714x + 885741.5238
1
2
3
Time
Amazon_MV
Regression Line

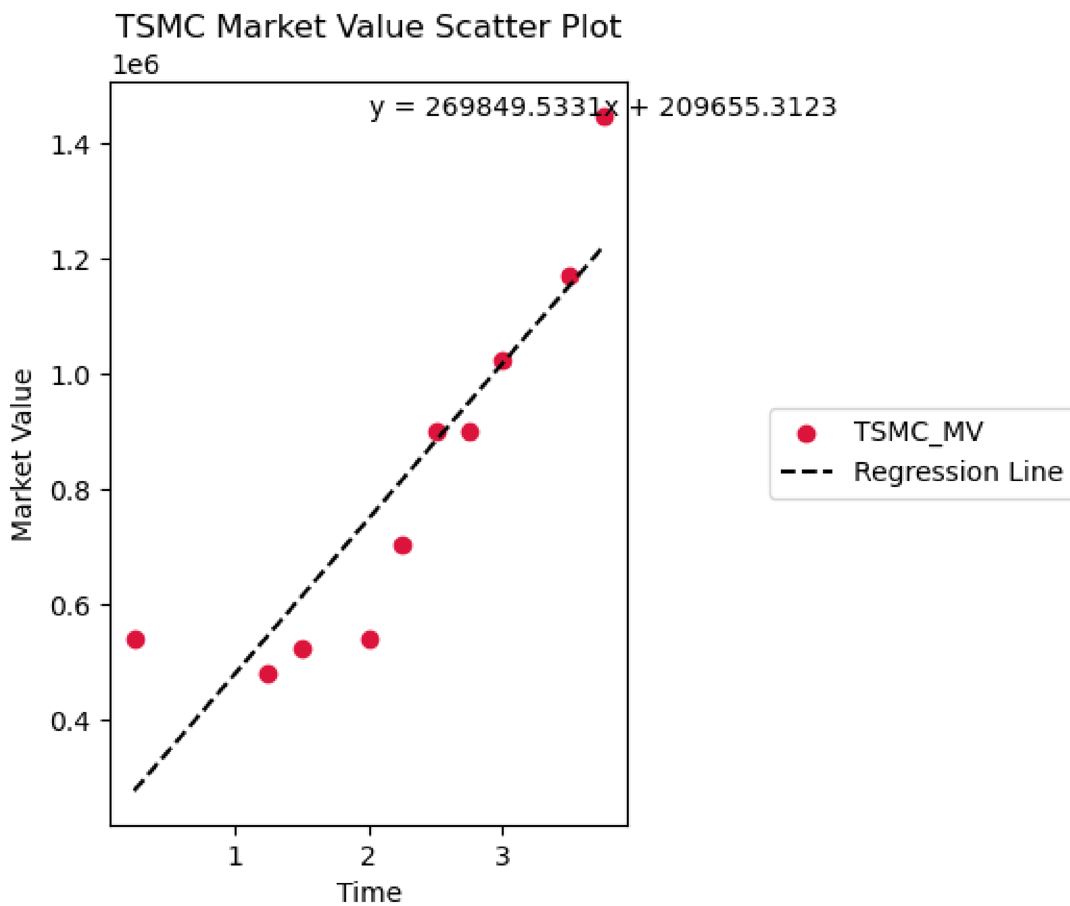

Tesla Market Value Scatter Plot
1e6
y = 105535.1284x + 718095.1780
1.4
1.2
1.0
0.8
Market Value
1
2
3
Time
Tesla_MV
Regression Line


Berkshire Hathaway Market Value Scatter Plot
1e6
y = 144251.7143x + 556501.9048
1.1
1.0
0.9
0.8
0.7
0.6
Market Value
1
2
3
Time
Berkshire_Hathaway_MV
Regression Line

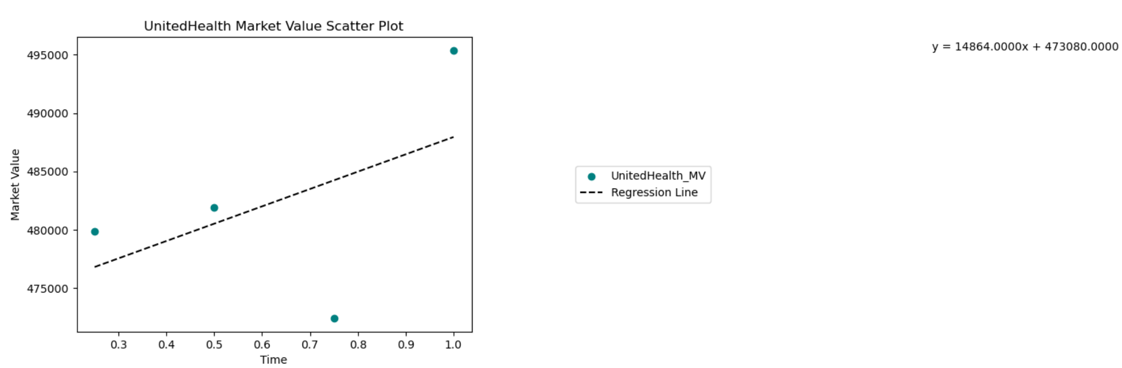

Nvidia Market Value Scatter Plot
1e6
y = 1183052.5373x + -447728.1592
Market Value
Time
Nvidia_MV
Regression Line
0
1
2
3
4
1
2
3



Meta Market Value Scatter Plot
1e6
y = 434861.7174x + 169265.7283
Market Value
Time
Meta_MV
Regression Line



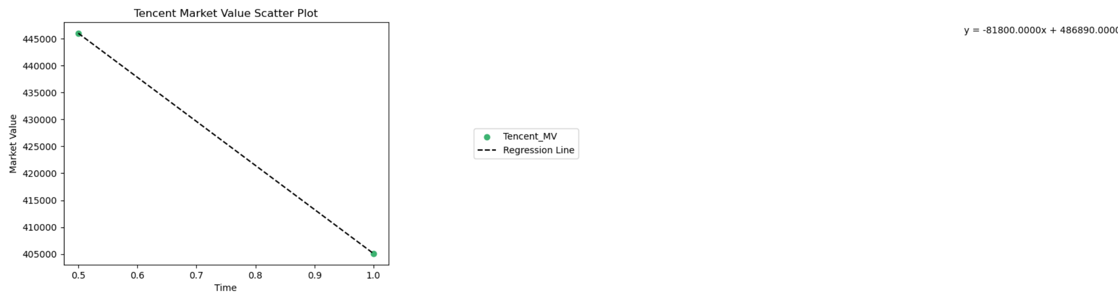

```
/var/folders/9r/bk9mfv2548537j8q9xs0nbtc0000gn/T/ipykernel_605/1863615942.py:310
: UserWarning: Tight layout not applied. The left and right margins cannot be
made large enough to accommodate all axes decorations.
  plt.tight_layout()
```

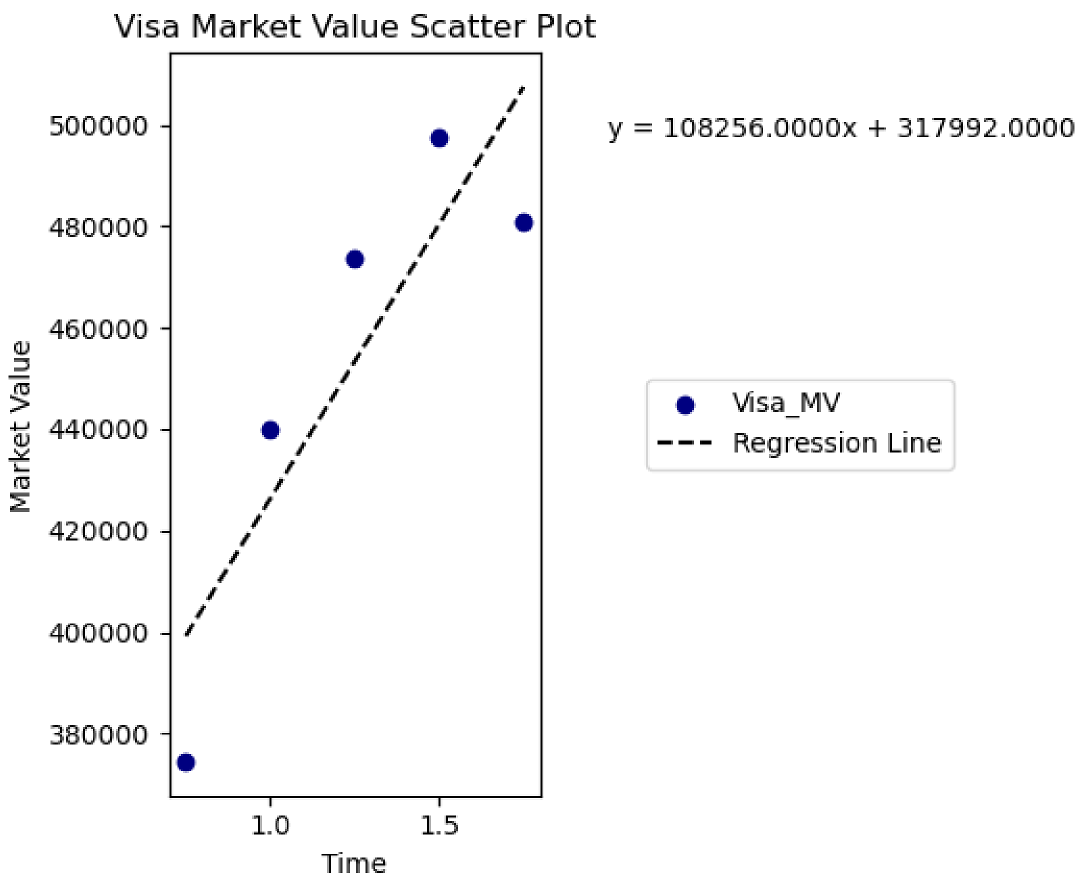



```
/var/folders/9r/bk9mfv2548537j8q9xs0nbtc0000gn/T/ipykernel_605/1863615942.py:341
```

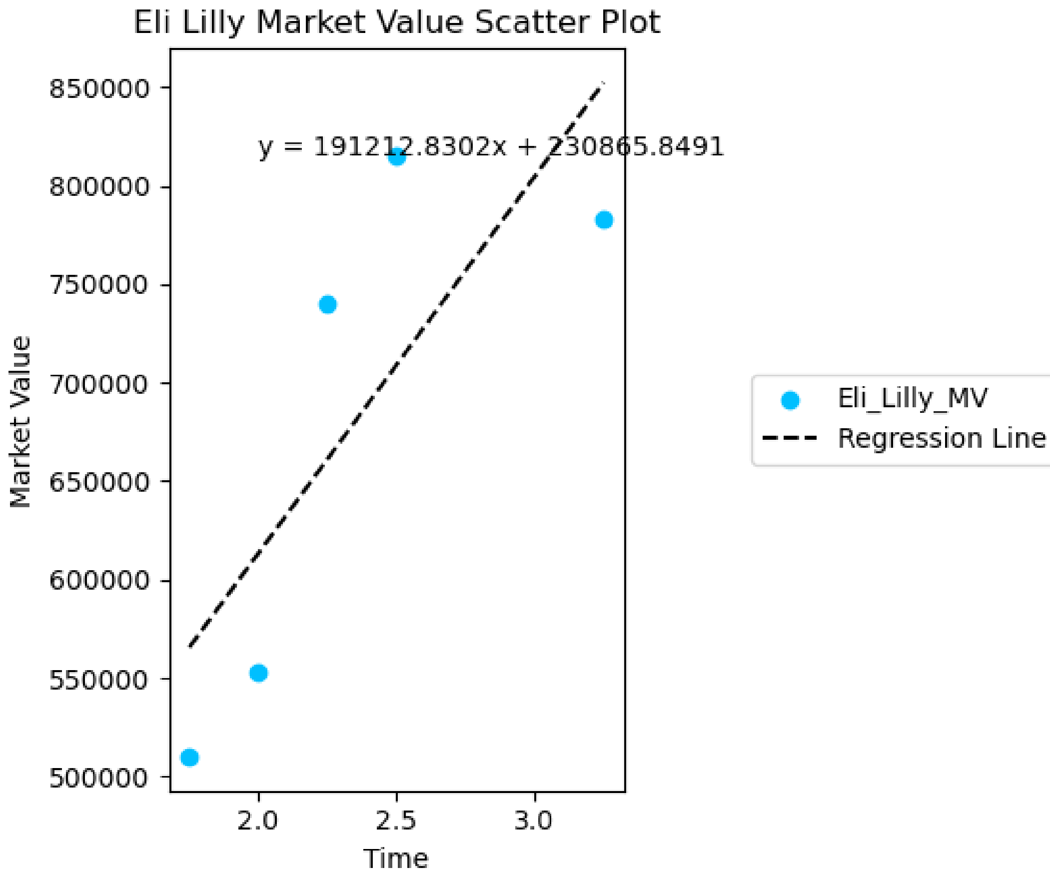
```
: UserWarning: Tight layout not applied. The left and right margins cannot be
made large enough to accommodate all axes decorations.
  plt.tight_layout()
```

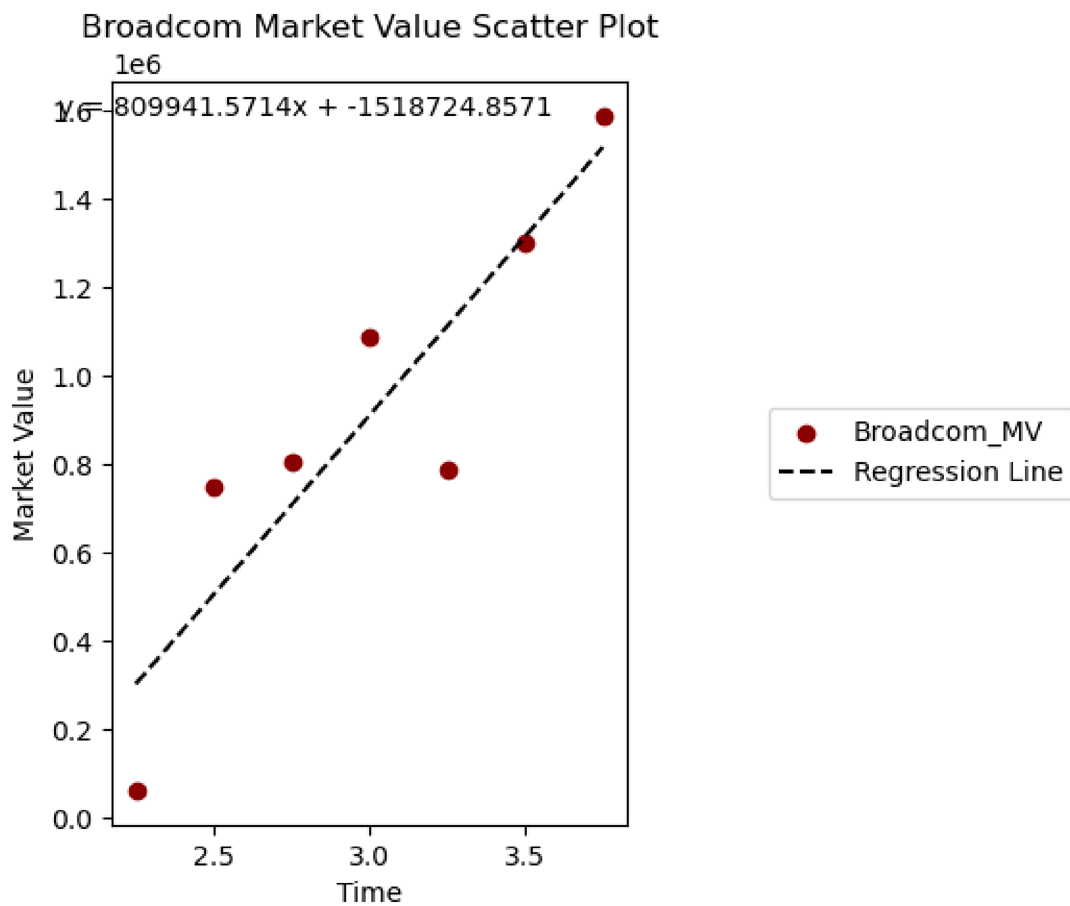

```
/var/folders/9r/bk9mfv2548537j8q9xs0nbtc0000gn/T/ipykernel_605/1863615942.py:373
: UserWarning: Tight layout not applied. The left and right margins cannot be
made large enough to accommodate all axes decorations.
  plt.tight_layout()
```






```
/Users/macbookpro/anaconda3/lib/python3.11/site-
packages/IPython/core/interactiveshell.py:3505: RankWarning: Polyfit may be
poorly conditioned
  exec(code_obj, self.user_global_ns, self.user_ns)
/var/folders/9r/bk9mfv2548537j8q9xs0nbtc0000gn/T/ipykernel_605/1863615942.py:436
: UserWarning: Tight layout not applied. The left and right margins cannot be
made large enough to accommodate all axes decorations.
  plt.tight_layout()
```

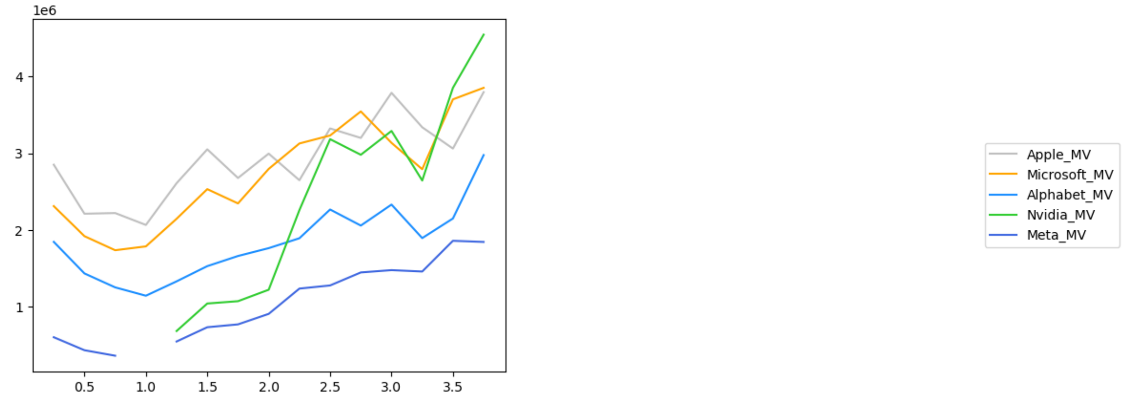

Eli Lilly Market Value Scatter Plot
y = 191212.8302x + 230865.8491
850000
800000
750000
700000
650000
600000
550000
500000
Market Value
2.0
2.5
3.0
Time
Eli_Lilly_MV
Regression Line

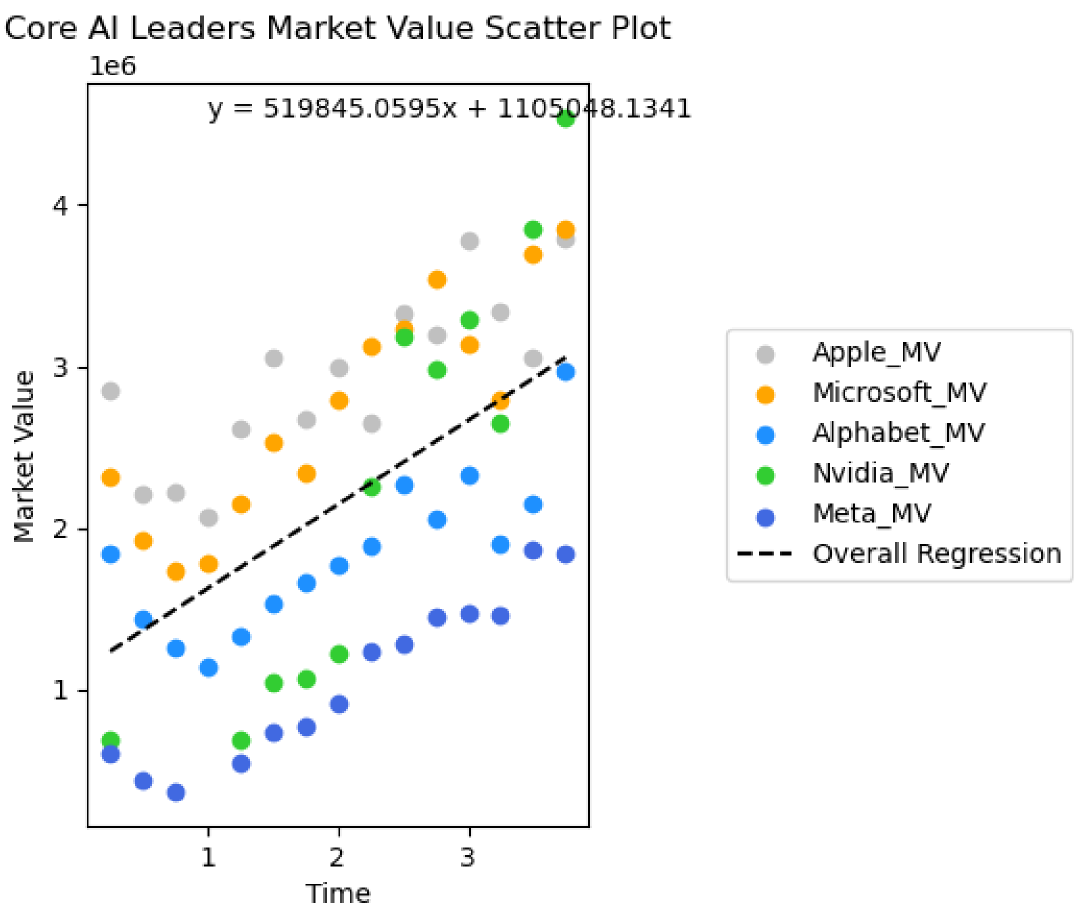


[15]:
```
# Core AI Leaders

Market_Value = [Apple_MV, Microsoft_MV, Alphabet_MV, Nvidia_MV, Meta_MV]
labels = ["Apple_MV", "Microsoft_MV", "Alphabet_MV", "Nvidia_MV", "Meta_MV"]
colors = ["silver", "orange", "dodgerblue", "limegreen", "royalblue"]
Time = np.array([0.25, 0.50, 0.75, 1.0, 1.25, 1.50, 1.75, 2.0, 2.25, 2.50, 2.
↪75, 3.0, 3.25, 3.50, 3.75])



for y, label, c in zip(Market_Value, labels, colors):
    plt.plot(Time, y, label=label, color=c)





plt.legend(loc="center left", bbox_to_anchor=(2, 0.5))
```

```
plt.tight_layout()
plt.show()



for y, label, c in zip(Market_Value, labels, colors):
    plt.scatter(Time, y, label=label, color=c)

all_y = np.concatenate(Market_Value)
mask = ~np.isnan(all_y)
all_Time = np.tile(Time, len(Market_Value))


coef = np.polyfit(all_Time[mask], all_y[mask], 1)
reg_line = np.poly1d(coef)

plt.plot(Time, reg_line(Time), color='black', linestyle='--', label="Overall␣
 ↪Regression")

# Regression equation text
plt.text(1.0, max(all_y[mask]), f"y = {coef[0]:.4f}x + {coef[1]:.4f}",␣
 ↪fontsize=10, color="black")




plt.xlabel("Time")
plt.ylabel("Market Value")
plt.title("Core AI Leaders Market Value Scatter Plot")
plt.legend(loc="center left", bbox_to_anchor=(1.25, 0.5))
plt.tight_layout()
plt.show()
```

```
/var/folders/9r/bk9mfv2548537j8q9xs0nbtc0000gn/T/ipykernel_605/1224520370.py:18:
UserWarning: Tight layout not applied. The left and right margins cannot be made
large enough to accommodate all axes decorations.
  plt.tight_layout()
```

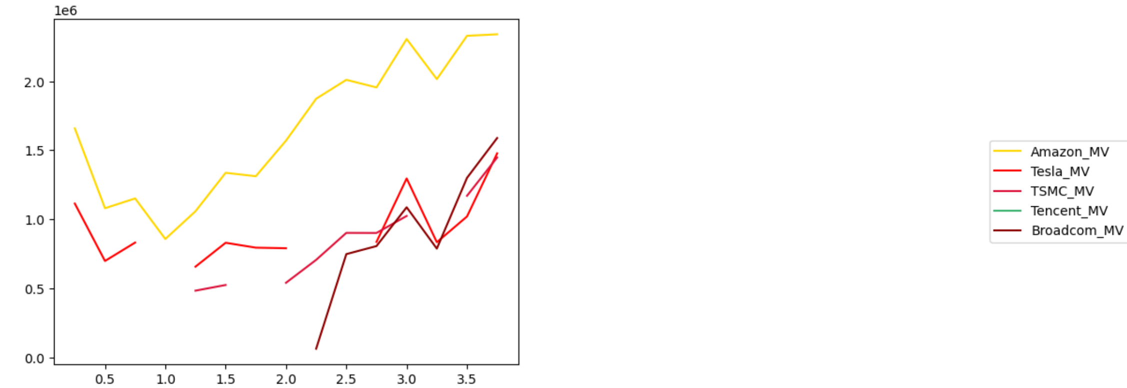


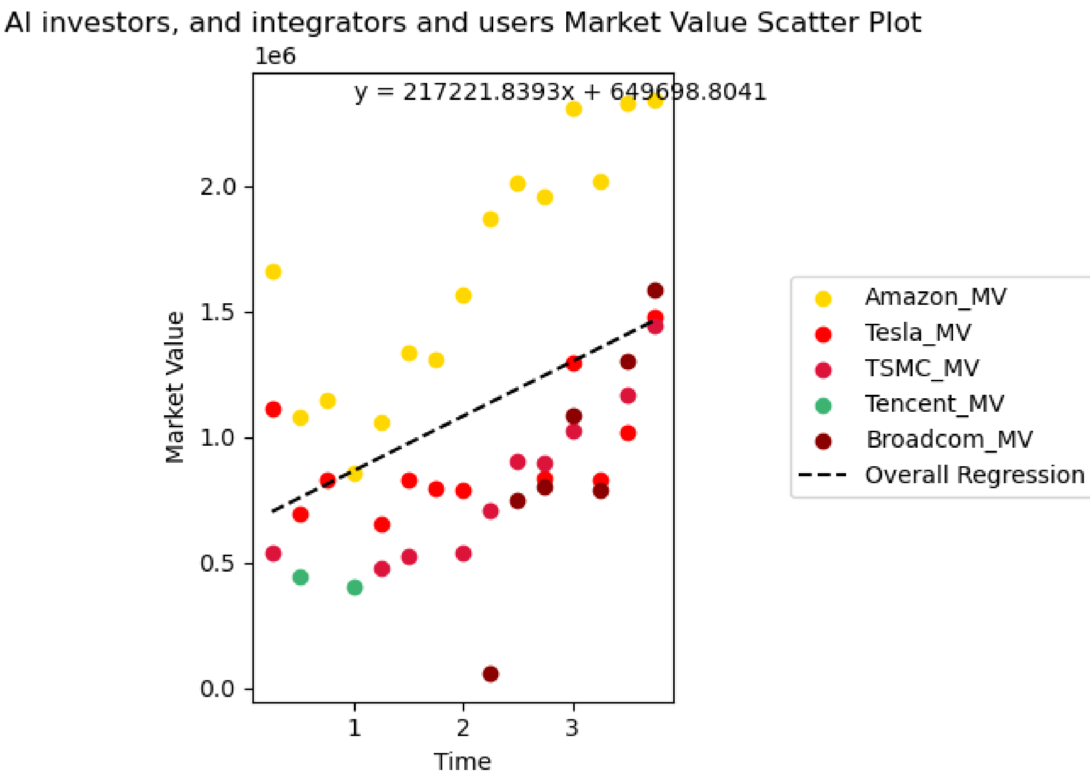


[16]:
```
# AI investors, and integrators and users

Market_Value = [Amazon_MV, Tesla_MV, TSMC_MV, Tencent_MV, Broadcom_MV]
labels = ["Amazon_MV", "Tesla_MV", "TSMC_MV", "Tencent_MV", "Broadcom_MV"]
```

```
colors = ["gold", "red", "crimson", "mediumseagreen", "darkred"]
Time = np.array([0.25, 0.50, 0.75, 1.0, 1.25, 1.50, 1.75, 2.0, 2.25, 2.50, 2.
 ↪75, 3.0, 3.25, 3.50, 3.75])



for y, label, c in zip(Market_Value, labels, colors):
    plt.plot(Time, y, label=label, color=c)

plt.legend(loc="center left", bbox_to_anchor=(2, 0.5))
plt.tight_layout()
plt.show()



for y, label, c in zip(Market_Value, labels, colors):
    plt.scatter(Time, y, label=label, color=c)


all_y = np.concatenate(Market_Value)
mask = ~np.isnan(all_y)
all_Time = np.tile(Time, len(Market_Value))


coef = np.polyfit(all_Time[mask], all_y[mask], 1)
reg_line = np.poly1d(coef)

plt.plot(Time, reg_line(Time), color='black', linestyle='--', label="Overall␣
 ↪Regression")

# Regression equation text
plt.text(1.0, max(all_y[mask]), f"y = {coef[0]:.4f}x + {coef[1]:.4f}",␣
 ↪fontsize=10, color="black")

plt.xlabel("Time")
plt.ylabel("Market Value")
plt.title("AI investors, and integrators and users Market Value Scatter Plot")
plt.legend(loc="center left", bbox_to_anchor=(1.25, 0.5))
plt.tight_layout()
plt.show()
```

```
/var/folders/9r/bk9mfv2548537j8q9xs0nbtc0000gn/T/ipykernel_605/3802754890.py:14:
UserWarning: Tight layout not applied. The left and right margins cannot be made
large enough to accommodate all axes decorations.
  plt.tight_layout()
```

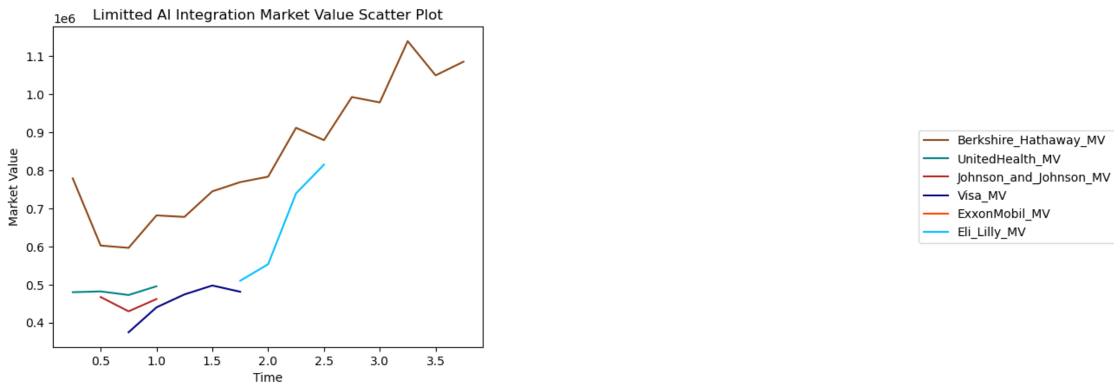


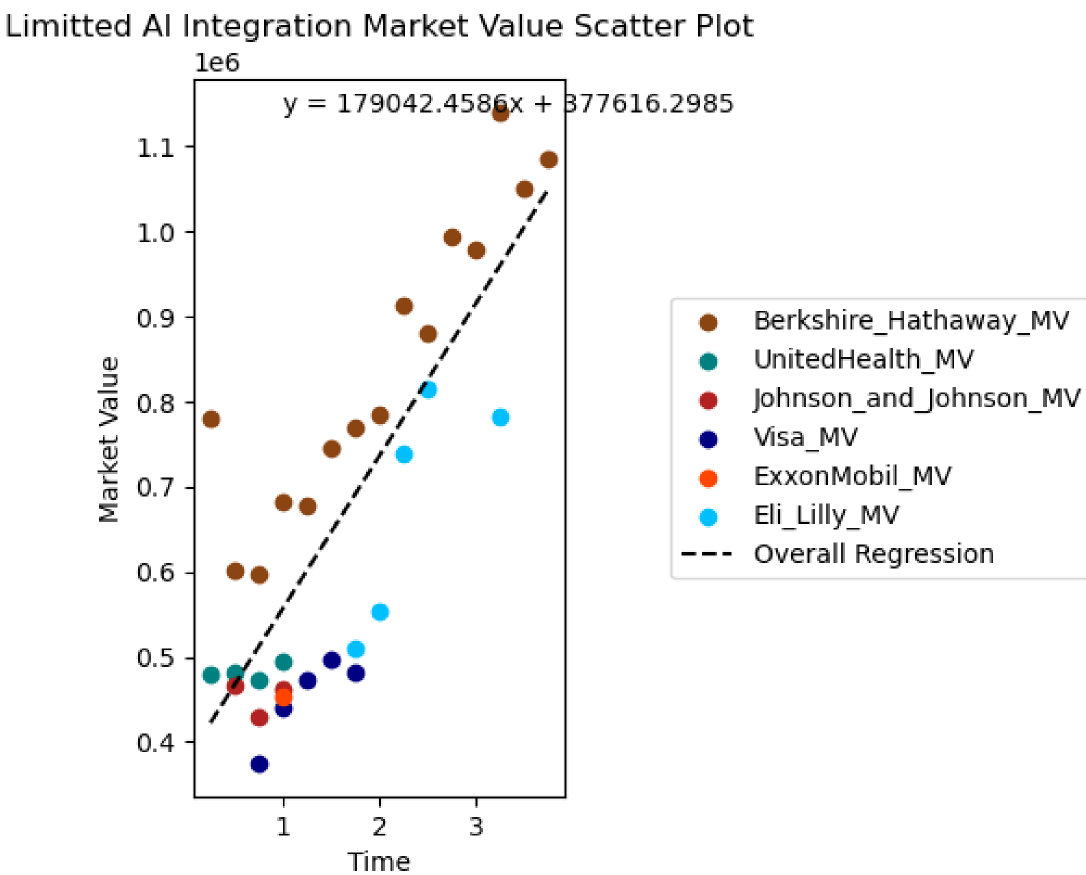


[18]:
```
# Limitted AI Integration

Market_Value = [Berkshire_Hathaway_MV, UnitedHealth_MV, Johnson_and_Johnson_MV,␣
↪Visa_MV, ExxonMobil_MV, Eli_Lilly_MV]
labels = ["Berkshire_Hathaway_MV", "UnitedHealth_MV", "Johnson_and_Johnson_MV",␣
↪"Visa_MV", "ExxonMobil_MV", "Eli_Lilly_MV"]
```

```python
colors = ["saddlebrown", "teal", "firebrick", "navy", "orangered",␣
↪"deepskyblue"]
Time = np.array([0.25, 0.50, 0.75, 1.0, 1.25, 1.50, 1.75, 2.0, 2.25, 2.50, 2.
↪75, 3.0, 3.25, 3.50, 3.75])



for y, label, c in zip(Market_Value, labels, colors):
    plt.plot(Time, y, label=label, color=c)

plt.xlabel("Time")
plt.ylabel("Market Value")
plt.title("Limitted AI Integration Market Value Scatter Plot")
plt.legend(loc="center left", bbox_to_anchor=(2, 0.5))
plt.tight_layout()
plt.show()

for y, label, c in zip(Market_Value, labels, colors):
    plt.scatter(Time, y, label=label, color=c)

all_y = np.concatenate(Market_Value)
mask = ~np.isnan(all_y)
all_Time = np.tile(Time, len(Market_Value))


coef = np.polyfit(all_Time[mask], all_y[mask], 1)
reg_line = np.poly1d(coef)

plt.plot(Time, reg_line(Time), color='black', linestyle='--', label="Overall␣
↪Regression")

# Regression equation text
plt.text(1.0, max(all_y[mask]), f"y = {coef[0]:.4f}x + {coef[1]:.4f}",␣
↪fontsize=10, color="black")

plt.xlabel("Time")
plt.ylabel("Market Value")
plt.title("Limitted AI Integration Market Value Scatter Plot")
plt.legend(loc="center left", bbox_to_anchor=(1.25, 0.5))
plt.tight_layout()
plt.show()
```

```
/var/folders/9r/bk9mfv2548537j8q9xs0nbtc0000gn/T/ipykernel_605/349006615.py:17:
UserWarning: Tight layout not applied. The left and right margins cannot be made
large enough to accommodate all axes decorations.
  plt.tight_layout()
```





[26]:
```
from scipy.stats import ttest_ind

tech_campanies = [Apple_MV, Microsoft_MV, Alphabet_MV, Nvidia_MV, Meta_MV]
industrial_companies = [Berkshire_Hathaway_MV, UnitedHealth_MV,␣
↪Johnson_and_Johnson_MV, Visa_MV, ExxonMobil_MV, Eli_Lilly_MV]
```

```python
def compute_slope(series):
    Time = np.array([0.25, 0.50, 0.75, 1.0, 1.25, 1.50, 1.75, 2.0, 2.25, 2.50,␣
 ↪2.75, 3.0, 3.25, 3.50, 3.75])
    mask = ~np.isnan(series)
    slope, intercept = np.polyfit(Time[mask], series[mask], 1)
    return slope


slopes_tech = np.array([compute_slope(company) for company in tech_companies])
slopes_industrial = np.array([compute_slope(company) for company in␣
 ↪industrial_companies])
```

```
/var/folders/9r/bk9mfv2548537j8q9xs0nbtc0000gn/T/ipykernel_605/4159545331.py:14:
RankWarning: Polyfit may be poorly conditioned
  slopes_industrial = np.array([compute_slope(company) for company in
industrial_companies])
```

[27]:
```python
tstat, pvalue = ttest_ind(slopes_tech, slopes_industrial, alternative =␣
 ↪'greater')

print("t-statistic:", tstat)
print()

print("p-value (one-sided):", pvalue)
if pvalue < 0.05:
    print("Conclusion: Tech companies grow significantly faster than industrial␣
 ↪companies")

else:
    print("Conclusion: There is no big difference")
```

```
t-statistic: 2.1737353249547033

p-value (one-sided): 0.023686192576885902
Conclusion: Tech companies grow significantly faster than industrial companies
```

[29]:
```python
mean_tech = np.mean(slopes_tech)
mean_industrial = np.mean(slopes_industrial)

std_tech = np.std(slopes_tech, ddof = 1)
std_industrial = np.std(slopes_industrial, ddof = 1)

spooled = np.sqrt(((len(slopes_tech)-1)*std_tech**2 +␣
 ↪(len(slopes_industrial)-1)*std_industrial**2)/(len(slopes_tech)+␣
 ↪len(slopes_industrial)-2))

Cohens_d = (mean_tech-mean_industrial)/spooled
```

```python
print("Cohen's d:", Cohens_d)
print()
if Cohens_d > 1.0:
    print("The effect size is very large(it is a land slide difference)")

elif Cohens_d >0.8:
    print("The effect size is large")

elif Cohens_d > 0.5:
    print("The effect size is medium")

else:
    print("The effect size is small")
```

```
Cohen's d: 1.1225120950149778

The effect size is very large(it is a land slide difference)
```

[ ]:

```python
```

[ ]:

```python
```