```
import seaborn as sns
import pandas as pd
import yfinance as yf
import numpy as np

data = pd.read_csv("../data/processed/weekly-delta-binary-
curated.csv")

df = pd.DataFrame()
df["index"] = data["index"]
df["rolling"] = data["stock_market"]

data
```

```
     Target  index  lag_1  debt  stocks  dow_jones  markets
unemployment  \
0         1      0      0  0.04    0.00       0.00    -0.04
0.00
1         1      1      1 -0.02   -0.06       0.00     0.00          -
0.01
2         0      2      1 -0.02   -0.06       0.00    -0.10
0.00
3         0      3      0 -0.01   -0.01      -0.01     0.05
0.00
4         1      4      0  0.00    0.00      -0.01     0.09
0.00
..      ...    ...    ...   ...     ...        ...      ...
...
850       1    850      0 -0.01   -0.19      -0.06    -0.02          -
0.19
851       1    851      1 -0.03   -0.10      -0.02    -0.04          -
0.16
852       1    852      1 -0.05   -0.07      -0.05    -0.06          -
0.08
853       0    853      1 -0.05    0.06       0.02    -0.01          -
0.12
854       1    854      0 -0.02    0.20       0.08     0.03          -
0.11

     money  stock_market  crisis  nasdaq  finance  invest
0     0.02          0.00    0.04   -0.01     0.01   -0.01
1    -0.05          0.00   -0.01   -0.07    -0.02   -0.02
2     0.02          0.00    0.07   -0.09     0.00    0.02
3     0.03         -0.01    0.07   -0.08     0.01   -0.04
4     0.06          0.00    0.06   -0.05    -0.02    0.05
..     ...           ...     ...     ...      ...     ...
850  -0.14         -0.05   -0.07   -0.01    -0.04   -0.33
851  -0.13         -0.04   -0.02    0.00    -0.05   -0.11
852  -0.11         -0.03    0.08   -0.05    -0.09   -0.14
853  -0.14          0.00    0.02    0.03     0.01   -0.01
```

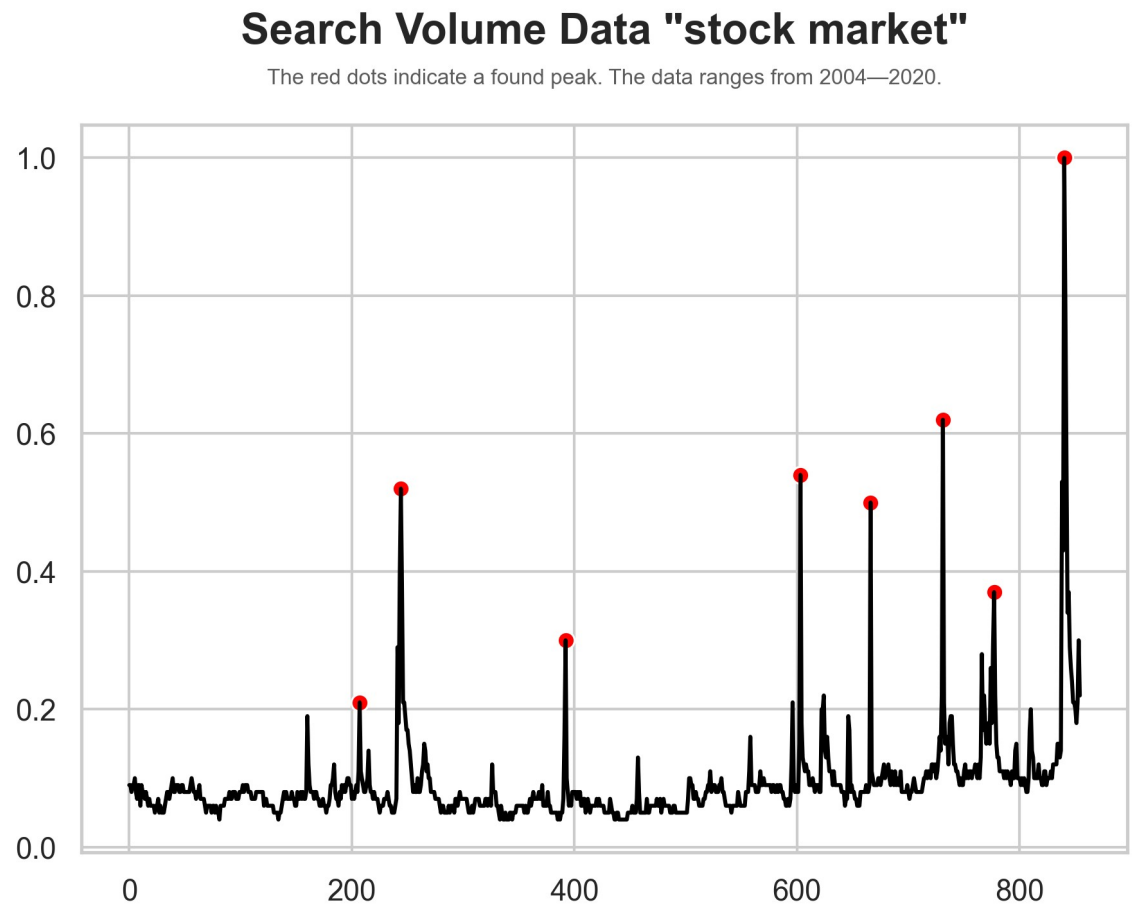
```
854  -0.07          0.10    0.00    0.22     0.09    0.12

[855 rows x 14 columns]
```

```
stock_market_df = pd.read_csv("../data/raw/weekly/stock_market.csv")
stock_market_df = stock_market_df[4:].reset_index()

df.insert(0, "actual", stock_market_df["Adjusted"])

ticker_df = yf.download("^DJI", period="max", interval="1wk")

ticker_df = ticker_df[-861:-5]
ticker_df = ticker_df.reset_index()

df["Close"] = ticker_df["Close"]
```

```
[*********************100%***********************]  1 of 1 completed
```

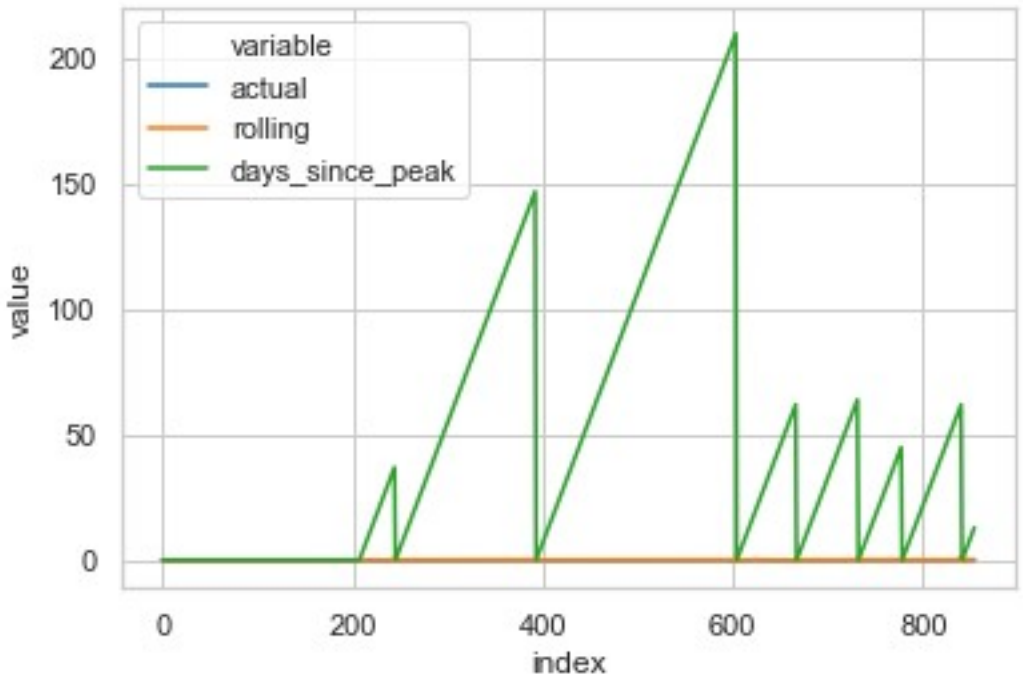
```
import matplotlib.pyplot as plt
from scipy.signal import find_peaks


sns.set_style("whitegrid", {"font.sans_serif": "Liberation Sans"})
sns.set_context("notebook")

fig, ax = plt.subplots(1, 1, figsize = (7, 5), dpi=300)

x = np.array(df["actual"])
peaks, _ = find_peaks(x, prominence=0.15, width=1)
sns.lineplot(y=x, x=df.index, color="black")
sns.scatterplot(peaks, x[peaks], color="r", s=40)

ax.set_ylabel('')
ax.set_xlabel('')
ax.text(x=0.5, y=1.1, s='Search Volume Data "stock market"',
fontsize=16, weight='bold', ha='center', va='bottom',
transform=ax.transAxes)
ax.text(x=0.5, y=1.05, s="The red dots indicate a found peak. The data
ranges from 2004–2020.", fontsize=8, alpha=0.75, ha='center',
va='bottom', transform=ax.transAxes)

# fig.savefig("graph.png")
```

```
Text(0.5, 1.05, 'The red dots indicate a found peak. The data ranges
from 2004–2020.')
```

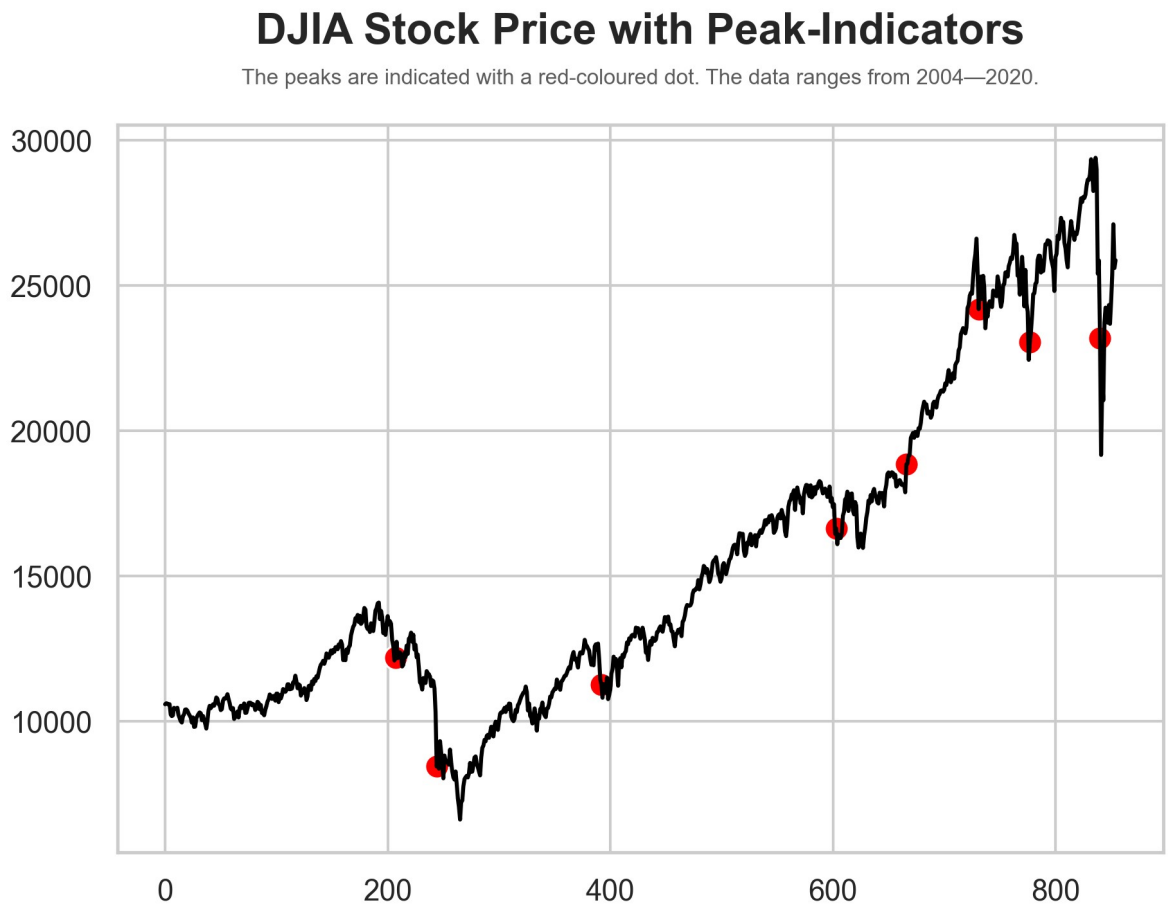



```
df["days_since_peak"] = 0

i = 0
j = 0
k = 0
while i < len(df):
    if i >= peaks[j]:
        df.loc[i, "days_since_peak"] = k
        k += 1

    try:
        if i == peaks[j+1]:
            k = 0
            j += 1
    except IndexError:
        pass

    i += 1

sns.lineplot(x="index", y="value", hue="variable", data=pd.melt(df,
["index", "Close"]))
```

```
<AxesSubplot:xlabel='index', ylabel='value'>
```

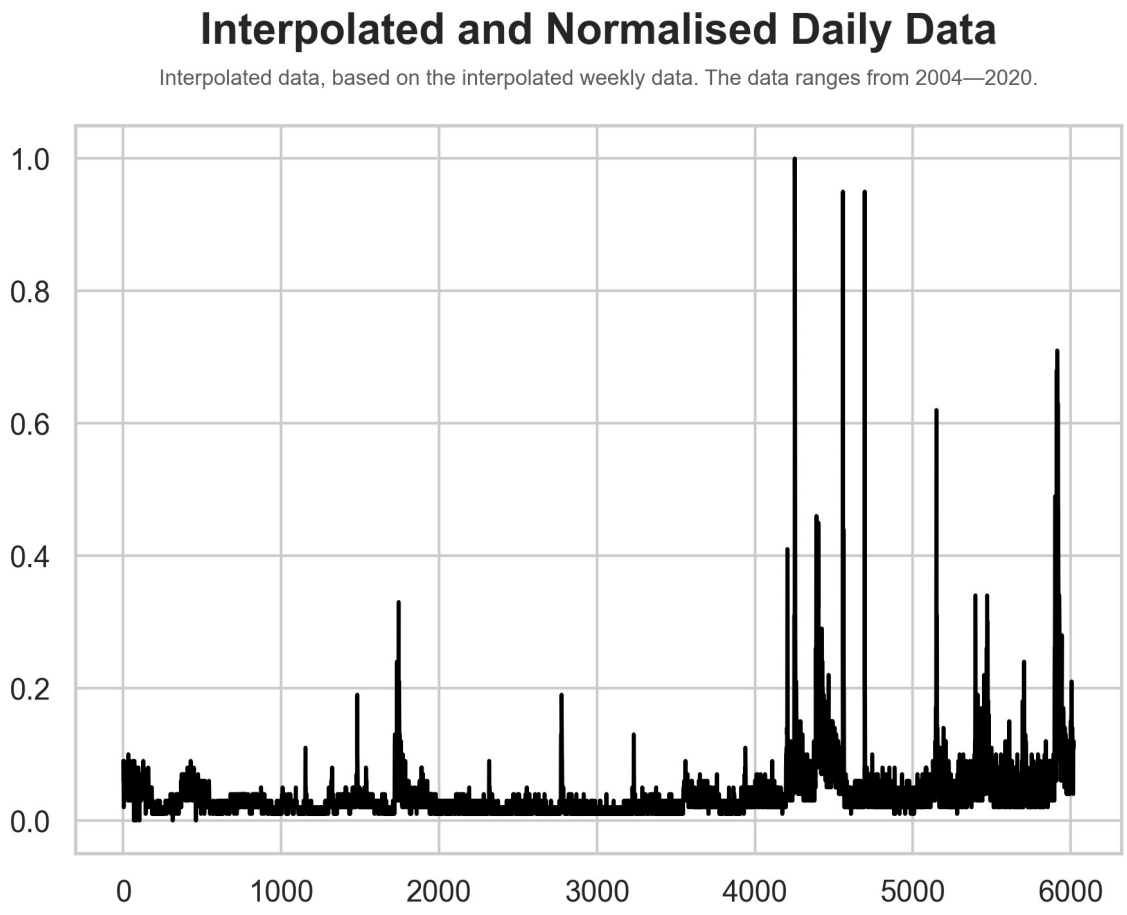



```
import matplotlib.pyplot as plt


sns.set_style("whitegrid", {"font.sans_serif": "Liberation Sans"})
sns.set_context("notebook")

fig, ax = plt.subplots(1, 1, figsize = (7, 5), dpi=300)

sns.lineplot(data=df["Close"], color="black")
sns.scatterplot(peaks, df["Close"][peaks], color="r", s=80)

ax.set_ylabel("")
ax.set_xlabel("")
ax.text(x=0.5, y=1.1, s="DJIA Stock Price with Peak-Indicators",
fontsize=16, weight='bold', ha='center', va='bottom',
transform=ax.transAxes)
ax.text(x=0.5, y=1.05, s="The peaks are indicated with a red-coloured
dot. The data ranges from 2004–2020.", fontsize=8, alpha=0.75,
ha='center', va='bottom', transform=ax.transAxes, )



# fig.savefig("graph1.png")

Text(0.5, 1.05, 'The peaks are indicated with a red-coloured dot. The
data ranges from 2004–2020.')
```



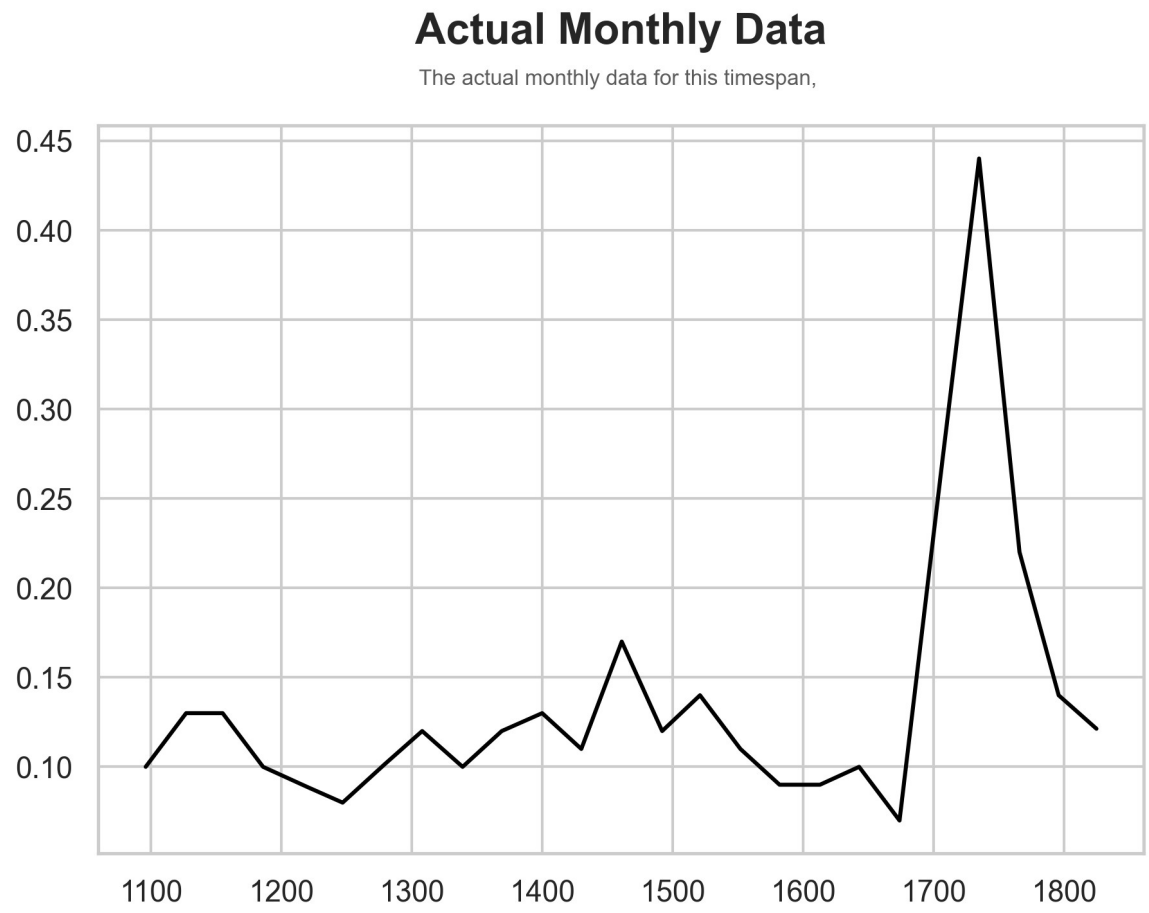

```
for i in range(10):
    df[f"peak_{i}"] = 0

i = 0
j = 0
k = 0
while i < len(df):
    if i >= peaks[j]:
        if k < 10:
            df.loc[i, f"peak_{k}"] = 1
        k += 1

    try:
        if i == peaks[j+1]:
            k = 0
            j += 1
    except IndexError:
        pass

    i += 1

daily_data = pd.read_csv("../data/raw/daily/stock_market.csv")
```

```
import matplotlib.pyplot as plt


sns.set_style("whitegrid", {"font.sans_serif": "Liberation Sans"})
sns.set_context("notebook")

fig, ax = plt.subplots(1, 1, figsize = (7, 5), dpi=300)

sns.lineplot(data=daily_data.Adjusted, color="black")

ax.set_ylabel("")
ax.set_xlabel("")
ax.text(x=0.5, y=1.1, s="Interpolated and Normalised Daily Data",
fontsize=16, weight='bold', ha='center', va='bottom',
transform=ax.transAxes)
ax.text(x=0.5, y=1.05, s="Interpolated data, based on the interpolated
weekly data. The data ranges from 2004–2020.", fontsize=8, alpha=0.75,
ha='center', va='bottom', transform=ax.transAxes, )



fig.savefig("graph1.png")
```

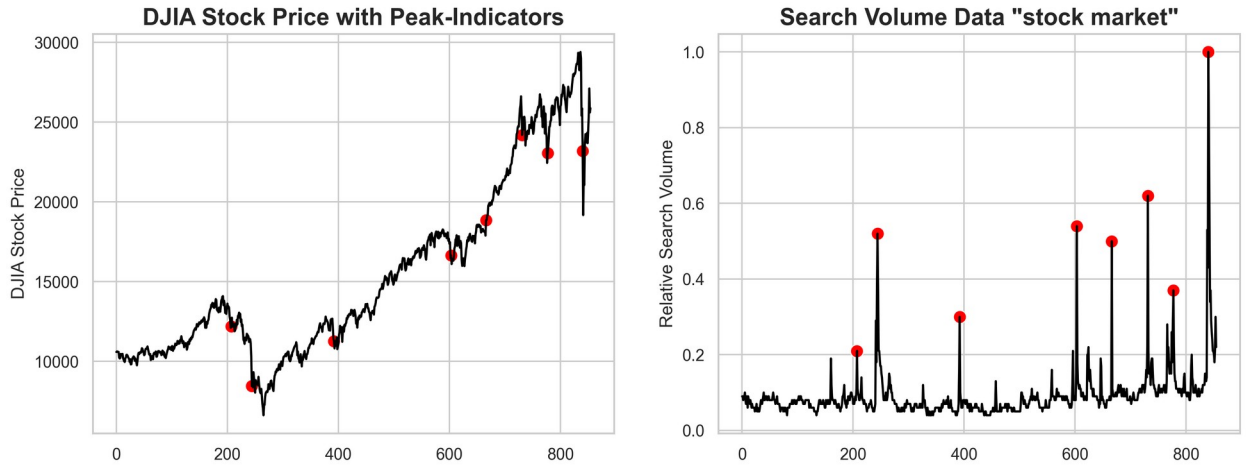


```
unadjusted_data = pd.read_csv("../deployment/data/data.csv")

import matplotlib.pyplot as plt


sns.set_style("whitegrid", {"font.sans_serif": "Liberation Sans"})
sns.set_context("notebook")

fig, ax = plt.subplots(1, 1, figsize = (7, 5), dpi=300)

sns.lineplot(data=unadjusted_data.monthly[1096:1826]/100,
color="black")

ax.set_ylabel("")
ax.set_xlabel("")
ax.text(x=0.5, y=1.1, s="Actual Monthly Data", fontsize=16,
weight='bold', ha='center', va='bottom', transform=ax.transAxes)
ax.text(x=0.5, y=1.05, s="The actual monthly data for this timespan,
", fontsize=8, alpha=0.75, ha='center', va='bottom',
transform=ax.transAxes, )



# fig.savefig("graph1.png")
```

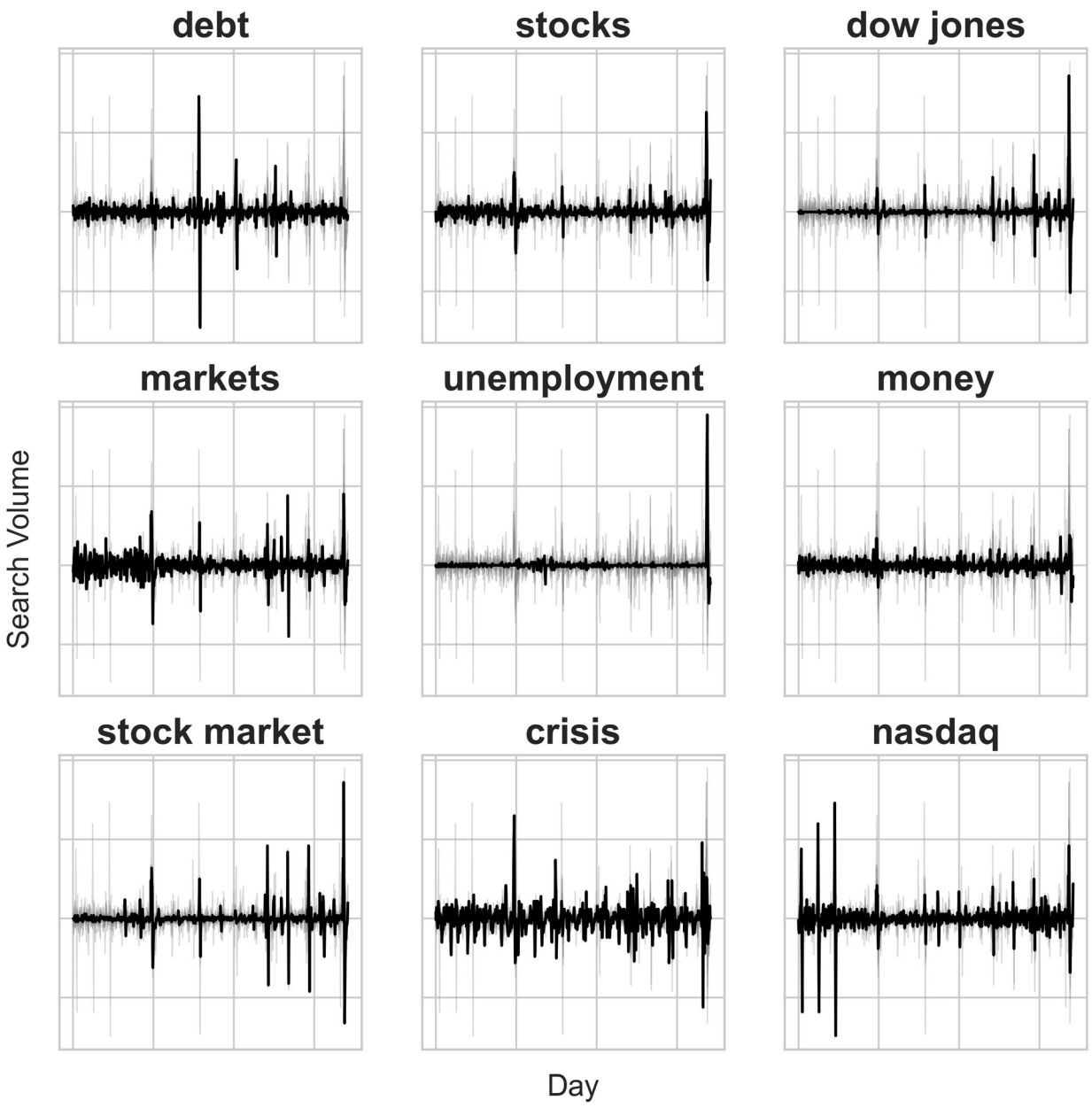
```
Text(0.5, 1.05, 'The actual monthly data for this timespan, ')
```




```
fig, axs = plt.subplots(ncols=2, figsize=(14, 5), dpi=300)

sns.lineplot(data=df["Close"], color="black", ax=axs[0])
sns.scatterplot(peaks, df["Close"][peaks], color="r", s=80, ax=axs[0])

sns.lineplot(y=x, x=df["index"], color="black", ax=axs[1])
sns.scatterplot(peaks, x[peaks], color="r", s=80, ax=axs[1])

axs[0].set_title("DJIA Stock Price with Peak-Indicators", fontsize=16,
weight="bold")
axs[0].set_xlabel("")
axs[0].set_ylabel("DJIA Stock Price")

axs[1].set_title('Search Volume Data "stock market"', fontsize=16,
weight="bold")
axs[1].set_xlabel("")
axs[1].set_ylabel("Relative Search Volume")

Text(0, 0.5, 'Relative Search Volume')
```




```
data = data.drop("lag_1", axis=1)

fig, axs = plt.subplots(ncols=3, nrows=3, figsize=(10, 10), dpi=300)

i = 2

for j in range(3):
    for k in range(3):
        l = 2
        while l <= 11:
            sns.lineplot(x="index", y=data.columns[l], marker='',
color='grey', linewidth=0.6, alpha=0.3, data=data, ax=axs[j][k])
            l += 1

        sns.lineplot(x="index", y=data.columns[i], data=data,
ax=axs[j][k], color="black")

        axs[j][k].set_xlabel("")
        axs[j][k].set_ylabel("")
        axs[j][k].set(xticklabels=[])
        axs[j][k].set(yticklabels=[])

        axs[j][k].set_title(data.columns[i].replace("_", " "),
fontweight="bold", fontsize=20)
        i += 1

axs[1][0].set_ylabel("Search Volume", fontsize=16)
axs[2][1].set_xlabel("Day", fontsize=16)

Text(0.5, 0, 'Day')
```




```
fig, axs = plt.subplots(ncols=3, nrows=3, figsize=(10, 10), dpi=300)

i = 2

for j in range(3):
    for k in range(3):
        sns.lineplot(x="Target", y=data.columns[i], data=data[1:],
ax=axs[j][k], color="black")

        axs[j][k].set_xlabel("")
        axs[j][k].set_ylabel("")
        axs[j][k].set(xticklabels=[])
```

```
        axs[j][k].set(yticklabels=[])

        axs[j][k].set_title(data.columns[i].replace("_", " "),
fontweight="bold", fontsize=20)
        i += 1

axs[1][0].set_ylabel("Search Volume", fontsize=16)
axs[2][1].set_xlabel("Target (left: stock price decrease, right: stock
price increase)", fontsize=16)
```

```
Text(0.5, 0, 'Target (left: stock price decrease, right: stock price
increase)')
```