COGS 108 - Trump's Social Media Posts and Market Volatility

Permissions

Place an X in the appropriate bracket below to specify if you would like your group's project to be made available to the public, including the names of the authors.

- [X] YES - make available
- ☐ NO - keep private

Link to video

https://drive.google.com/file/d/1XR1wVeRRXsolYOkwBCAnoPaQPwYYTo3g/view?usp=drive_link

Abstract

In the modern world, prominent figures, including presidents, can deliver their messages to millions of people instantly through social media platforms. Such posts can spark reactions across political, social, and economic spaces. Since financial markets are highly sensitive to information covered by mainstream media, it is reasonable to wonder whether social media posts by presidents have the same strength to create meaningful fluctuations in financial markets. This study explores this question by analyzing Donald Trump's economic and financial posts across his two presidential terms along with trends seen in market volatility in U.S. stock indices and cryptocurrency markets.

We obtained over 60 thousand posts from X (Twitter) and Truth Social, published between January 2016 and November 2025, which were then sorted to identify posts containing economic keywords such as "tariff", "rate", "trade", "jobs", "inflation", and "Fed". Our data for financial markets include ETFs for S&P 500, Nasdaq Composite, Dow Jones Industrial Average, Bitcoin prices and cryptocurrency market caps. Donald Trump's posting activity was then paired with financial market data to perform event-window analyses around the time each keyword post was published. As a control, randomly selected periods were used, allowing for a baseline comparison of asset prices before and after each post.

Using Parkinson's range-based estimator, we converted equity volatility to abnormal volatility z-scores. The plots showed small day-to-day movement around Trump's economic posts, but these shifts are not statistically meaningful. Bitcoin shows even less of a pattern, where both the event-window results and the slope comparisons look mostly like regular crypto noise, making it hard to associate any changes in value directly to the posts. The rest of the cryptocurrency market behaves the same way, with most keyword categories showing no noticeable or statistically significant changes. Overall, while there are a few isolated spikes here and there, Trump's economic posts don't show a strong or reliable connection to market volatility.

Authors

- Kento Bushey: Conceptualization, Background Research, Crypto Data wrangling, Analysis, Visualization
- Ryan Luutuyen: Data curation, Analysis, S&P500 Data Wrangling, Analysis, Visualization
- Kevin Mata: Data curation, Experimental investigation, Writing - original draft
- Mahdi Najjar: Conceptualization, Writing - original draft, Writing - review & editing
- Alessio Yu: Data curation, Visualization, Analysis, Social Media Web Scraping and Data Wrangling

Research Question

Does the incident of Trump’s social media posts containing economic or financial keywords (e.g., “tariff,” “rate,” “trade,” “jobs,” “inflation,” “Fed”) correlate with increased volatility in the U.S. stock and global cryptocurrency markets during the days following the posts, and does this relationship differ between his first and second term?

Background and Prior Work

The president of the United States is one of the most powerful roles in the world. Donald Trump was first elected in 2016 and served for four years, and has been elected again for his second term starting in 2025. Due to the president's great power, their public communications are under scrutiny both nationally and internationally.

One method of public communication the president has is social media. Twitter has become a popular platform for presidents to communicate with the public. Donald Trump also has his own platform called Truth Social. Social media allows for immediate dissemination of statements, potentially affecting public opinion and financial markets simultaneously. Researchers have investigated whether tweets by prominent figures, such as Trump, can create measurable effects on market behavior.[1]

The global market cap is estimated to be 135 trillion dollars, with the United States alone holding 70 trillion dollars.[2] Approximately 40% of the U.S. stock market is held in Americans' 401(k) accounts, so even small market fluctuations can result in substantial gains or losses for individual investors.[3] Understanding the drivers of market volatility, including social media communications, is therefore of both economic and societal interest.

Previous research by Wolff analyzed the potential effects of Donald Trump's tweets on stock prices during his first term. Wolff found some short-term anomalies in social media activity and financial markets but concluded that the overall market response was not statistically significant across all sectors.[1]

In addition to traditional markets, the cryptocurrency market, with a total market capitalization of approximately 3.74 trillion dollars,[4] may be particularly sensitive to news and communications such as presidential tweets. Unlike equities held largely in retirement accounts, crypto investments are typically more liquid and speculative, which can lead to rapid price reactions. Recent research shows that Bitcoin and Ethereum volatilities respond significantly to macroeconomic data releases, particularly U.S. monetary policy announcements, with pronounced effects in the pre-announcement period and heightened sensitivity during the pandemic.[5] This suggests that the inherent volatility of crypto assets makes them highly responsive to news and social signals, supporting the rationale for analyzing the effects of political communications on these markets.

1. ^ Wolff, L. (October 2019) *Financial Anomalies in Social Media – Analyzing Potential Effects of Donald Trump's Tweets on the Stock Market.* Lund University, Department of Economics (NEKH02). https://lup.lub.lu.se/luur/download?func=downloadFile&recordOId=9012527&fileOId=9012533

2. ^ Rosenthal, S. M. & Austin, L. S. (May 16 2016) *The Dwindling Taxable Share of U.S. Corporate Stock.* Tax Notes, Vol. 151, No. 6, pp. 923–934. https://www.taxpolicycenter.org/sites/default/files/alfresco/publication-pdfs/2000790-The-Dwindling-Taxable-Share-of-U.S.-Corporate-Stock.pdf

3. ^ "Companies ranked by Market Cap – CompaniesMarketCap.com." https://companiesmarketcap.com/ (companiesmarketcap.com)
4. ^ "Cryptocurrency Market Capitalization – CoinMarketCap Charts." https://coinmarketcap.com/charts/

5. ^ Chundakkadan, R. et al. (2025) *Cryptocurrency price volatility responses to macroeconomic news: Evidence from Bitcoin and Ethereum.* Finance Research Letters, Vol. 54, 103757. https://www.sciencedirect.com/science/article/pii/S1059056025006720

Hypothesis

We hypothesize that Trump's social media posts referencing economic or financial topics, such as tariffs, trade, or interest rates, are associated with changes in market volatility in the U.S. stock market and cryptocurrency market. Specifically, we expect that U.S. stock volatility can be measured using the daily high and low difference of the market-cap weighted S&P 500 index fund, with daily closing prices capturing end-of-day market reactions. Cryptocurrency volatility is expected to be higher and more immediate, with Bitcoin hourly price data providing granular insight into short-term responses. Additionally, daily total cryptocurrency market capitalization will be used to assess broader market movements.

We predict that volatility will be most pronounced on the day immediately following a relevant post, particularly for cryptocurrency assets, and that the effect will gradually taper over subsequent days. Posts on topics directly related to trade or monetary policy are expected to produce larger market reactions. We also hypothesize that during Trump's second term, market sensitivity to these posts will be stronger, reflecting increased attention or media amplification compared with the first term. This framework allows us to compare the magnitude and timing of market responses across asset classes and presidential terms.

Data

Data overview

For each dataset include the following information

- Dataset #1 Crypto Market Cap Daily data

  - Link to the dataset: https://www.coingecko.com/en/charts
  - Number of observations: 18,514
  - Number of variables: 4
  - Description of the variables most relevant to this project: Market cap in USD,total value of cryptocurrency in circulation, and trading volume, total value of coins traded in a 24-hour period.
  - Descriptions of any shortcomings this dataset has with repsect to the project: Volume data is missing for all records prior to December 26, 2013
- Dataset #2 Bitcoin Volume and Trading Data
  - Link to the dataset: www.cryptodatadownload.com/data/gemini/#google_vignette
  - Number of observations: Hourly and Daily Crypto Data from October 8, 2015, through November 11, 2025
  - Number of variables: 4
  - Description of the variables most relevant to this project: open, high, low, close (hourly prices in USD per Bitcoin) and Volume BTC, Volume USD (trading volumes exchanged during that hour)
  - Descriptions of any shortcomings this dataset has with repsect to the project: Some records show zero volume, which indicates missing or unreported data
- Dataset #3 Trump Social Media (X and Truth) Posts
  - Link to the dataset: Roll Call Factbase Twitter
  - Number of observations: 61995
  - Number of variables: 5 (platform, timestamp_et, timestamp_epoch, link, description)
  - Description of the variables most relevant to this project
    - description: Trump's X/Truth post content, this can allow us to find keywords
    - timestamp_et/timestamp_epoch: The timestamp in which Trump posted
    - platform: whether the post was published on Truth Social or X (formerly Twitter)
  - Descriptions of any shortcomings this dataset has with repsect to the project
    - The data accuracy is dependent on Roll Call's record keeping. Since we scraped the data from Roll Call's public facing site, if any record is not accurate or complete, then our analysis may be affected.
- ETFs Daily Trading Values
  - Dataset Name: SPX, COMP, DJIA Daily Trading Values
  - Link to the dataset: Historical Quotes SPX, Historical Quotes DJIA, Historical Quotes COMP
  - Number of observations: 2478
  - Number of variables: 5
  - High: Highest market value of share in day, Low: Lowest market value of share in day, Close: Closing value of share at end of day Purely numerical data doesn’t explain the contextual data behind sudden drastic changes in the market value of shares

```
# Run this code every time when you're actively developing modules
in .py files.  It's not needed if you aren't making modules
#
## this code is necessary for making sure that any modules we load are
updated here
## when their source code .py files are modified

%load_ext autoreload
%autoreload 2

# Setup code -- Run only once after cloning!!!
#
# this code downloads the data from its source to the `data/00-raw/`
directory
# if the data hasn't updated you don't need to do this again!

# if you don't already have these packages (you should!) uncomment
this line
# %pip install requests tqdm

import sys
sys.path.append('./modules') # this tells python where to look for
modules to import

import get_data # this is where we get the function we need to
download data
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
from datetime import date, timedelta, datetime
import seaborn as sns
from modules.data_cleanup import *
import re
from nltk.sentiment import SentimentIntensityAnalyzer
import nltk
from modules.hourly_data_functions import *
from modules.volatility_analysis import run_btc_volatility_analysis
from modules.crypto_volatility_module import *
from modules.ETF_volatility_functions import *

# replace the urls and filenames in this list with your actual
datafiles
# yes you can use Google drive share links or whatever
# format is a list of dictionaries;
# each dict has keys of
#   'url' where the resource is located
#   'filename' for the local filename where it will be stored
# datafiles = [
#     { 'url':
'https://raw.githubusercontent.com/fivethirtyeight/data/refs/heads/
master/airline-safety/airline-safety.csv', 'filename':'airline-
safety.csv'},
#     { 'url':
'https://raw.githubusercontent.com/fivethirtyeight/data/refs/heads/
master/bad-drivers/bad-drivers.csv', 'filename':'bad-drivers.csv'}
# ]

# get_data.get_raw(datafiles,destination_directory='data/00-raw/')

import warnings
warnings.filterwarnings('ignore')
```

Dataset #1 | Global Cryptocurrency Market Capitalization and Volume (Daily, 2013–2025)

This dataset contains daily global cryptocurrency market data from April 29, 2013, through November 11, 2025, compiled by CoinGecko. It includes 4,580 daily observations with three columns: a Unix timestamp (snapped_at), total market capitalization, and total 24-hour trading volume, all expressed in U.S. dollars (USD). Each row represents one full trading day across the global cryptocurrency market. Cryptocurrencies trade continuously, these daily values reflect full-day aggregates rather than exchange-specific trading hours.

Market capitalization measures the total value of all circulating cryptocurrencies, calculated as price × supply. Typical values range from about 1 billion dollars in 2013 to more than 2 trillion dollars in 2025, indicating massive market expansion. Total trading volume measures the dollar value of all trades executed within a 24-hour period, serving as a liquidity indicator. Both metrics are in USD, with volume and market cap typically correlated during high-volatility periods.

Missing data occur for total volume before December 26, 2013, affecting 241 early records (5.3 % of total rows). Because this period lies outside our analytical window, the data are excluded from analyses. Outliers are defined as single-day market cap or volume changes exceeding three standard deviations from a 7-day rolling mean. These reflect genuine market shocks (e.g., 2017 boom, 2021 crash) rather than data errors.

A potential limitation is that CoinGecko aggregates data across exchanges and coins, with early-period coverage skewed toward large-cap cryptocurrencies such as Bitcoin. Smaller or regional markets may be underrepresented. Despite this, the dataset provides a comprehensive, globally aggregated record of cryptocurrency market activity suitable for time-series analysis of volatility and long-term trends.

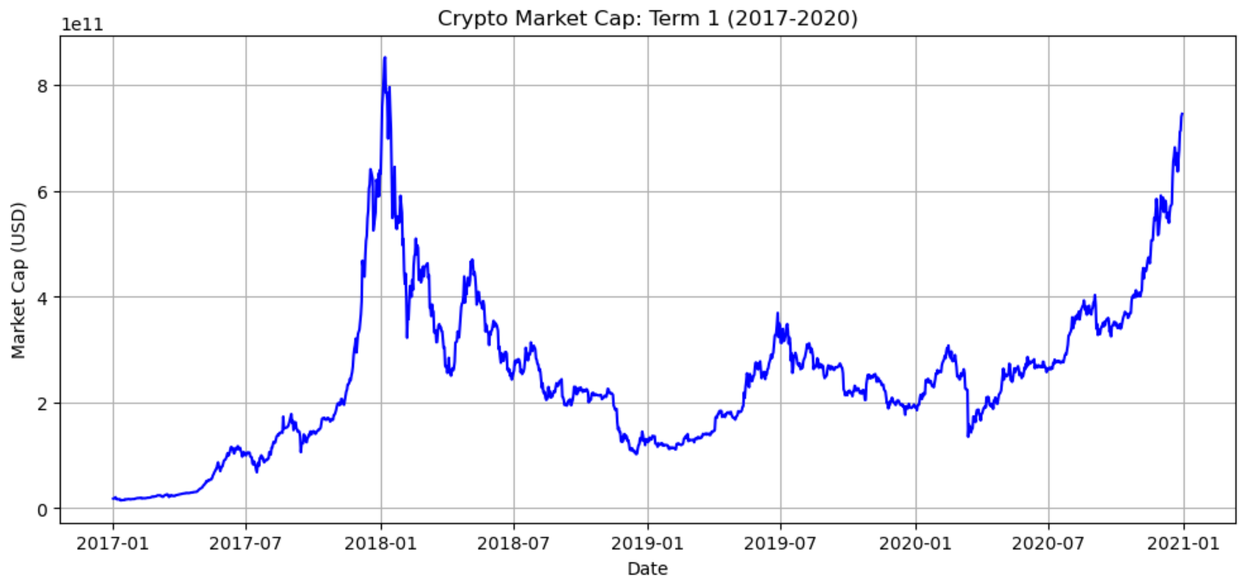
```
df1 = pd.read_csv("data/00-raw/crypto_marketcap_daily.csv")
df1["datetime"] = pd.to_datetime(df1["snapped_at"], unit="ms")
print(df1.head())
```

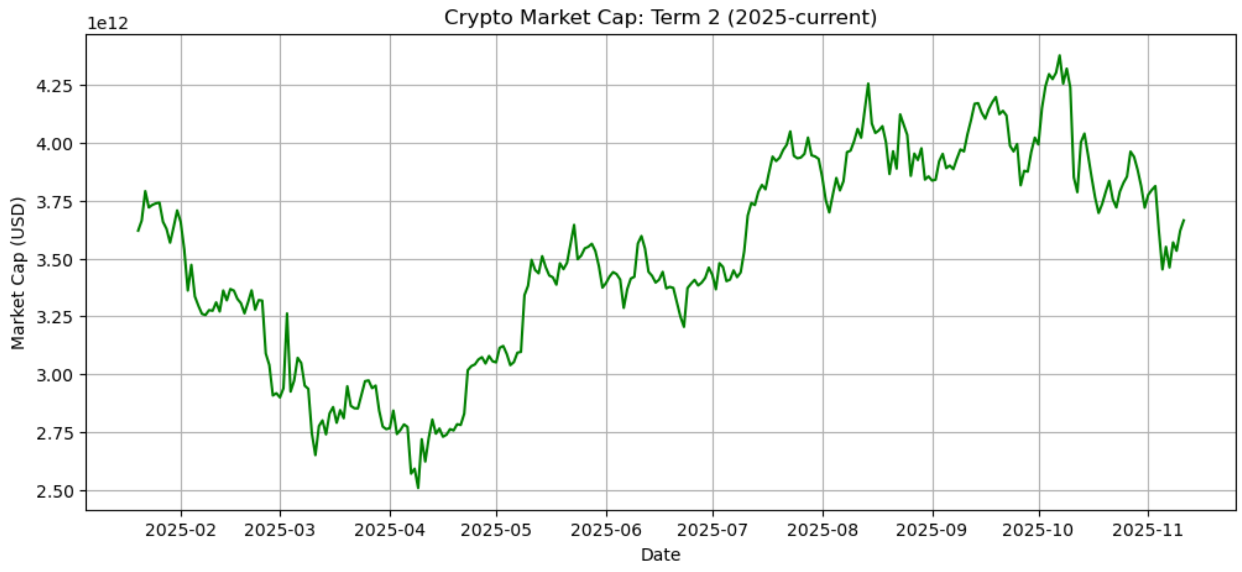
```
      snapped_at    market_cap  total_volume   datetime
0  1367193600000  1.661442e+09           0.0 2013-04-29
1  1367280000000  1.592765e+09           0.0 2013-04-30
2  1367366400000  1.378705e+09           0.0 2013-05-01
```

```
3  1367452800000  1.220763e+09           0.0 2013-05-02
4  1367539200000  1.075224e+09           0.0 2013-05-03

# Term 1: 1/1/2017-12/31/2020
term1 = df1[(df1["datetime"] >= "2017-01-01") & (df1["datetime"] <
"2020-12-31")]

# Term 2: 1/1/2025-current
term2 = df1[df1["datetime"] >= "2025-01-20"]

term1.head(5)
```

```
        snapped_at    market_cap  total_volume   datetime
1336  1483228800000  1.841179e+10  3.924458e+09 2017-01-01
1337  1483315200000  1.883194e+10  5.077314e+09 2017-01-02
1338  1483401600000  1.923852e+10  4.989160e+09 2017-01-03
1339  1483488000000  2.104879e+10  9.438407e+09 2017-01-04
1340  1483574400000  1.857421e+10  1.286286e+10 2017-01-05
```

```
# Plot Term 1: Market Cap
plt.figure(figsize=(12, 5))
plt.plot(term1["datetime"], term1["market_cap"], color="blue")
plt.title("Crypto Market Cap: Term 1 (2017-2020)")
plt.xlabel("Date")
plt.ylabel("Market Cap (USD)")
plt.grid(True)
plt.show()

# Plot Term 2: Market Cap
plt.figure(figsize=(12, 5))
plt.plot(term2["datetime"], term2["market_cap"], color="green")
plt.title("Crypto Market Cap: Term 2 (2025-current)")
plt.xlabel("Date")
plt.ylabel("Market Cap (USD)")
plt.grid(True)
plt.show()
```





Dataset #2 - Bitcoin Price and Volume (Hourly, 2015–2025)

This dataset contains hourly Bitcoin (BTC) trading data denominated in U.S. dollars (USD) from October 8, 2015, at 13:00 UTC through November 11, 2025. Each record represents one hour of trading activity and includes the opening, high, low, and closing prices of BTC in USD, along with trading volumes measured both in BTC and USD. Prices are expressed as USD per Bitcoin, while volumes represent the total amount of BTC or equivalent USD traded within each hourly interval.

Because cryptocurrency exchanges operate continuously, timestamps are in Coordinated Universal Time (UTC), ensuring uniform comparison across all trading hours without regard to regional market closures. This time resolution allows for finer-grained analysis of intraday volatility, liquidity changes, and short-term market reactions to events.

Data completeness is high overall. For Term 1 (January 1, 2017 – December 31, 2020), 756 out of 43,822 records (1.73%) show missing BTC or USD volume data, primarily early in the dataset. For Term 2 (January 1, 2025 – present), only 3 out of 7,536 records (0.04%) have missing volume entries. Missingness appears random and not associated with specific time periods or market conditions.

Outliers, particularly extreme price or volume spikes—are expected to correspond to real market events such as sudden demand surges or flash crashes. The data is tidy: each column represents a distinct variable, each row corresponds to a one-hour observation, and all entries are atomic with consistent types. This structure makes the dataset suitable for statistical and time-series analysis of Bitcoin's short-term market behavior, price volatility, and trading volume.

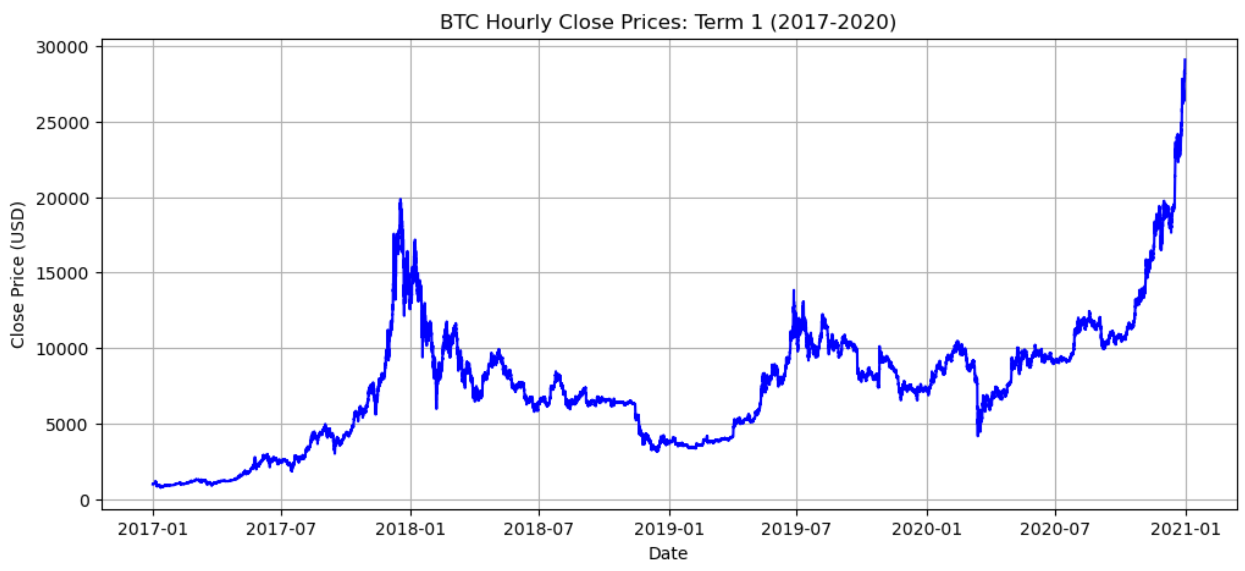
```
df2 = pd.read_csv("data/00-raw/BTC_hourly.csv")
df2.head(4)
```

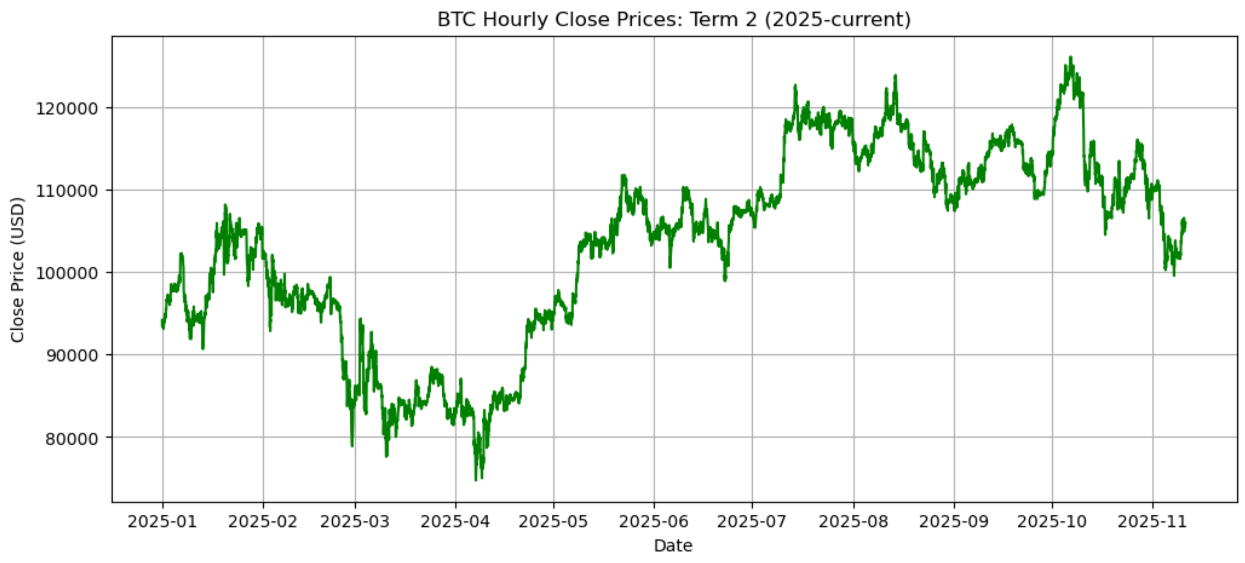
```
            unix                 date   symbol       open
high  \
0  1762815600000  2025-11-10 23:00:00  BTC/USD  105992.49  106261.48

1  1762812000000  2025-11-10 22:00:00  BTC/USD  105570.78  106106.60

2  1762808400000  2025-11-10 21:00:00  BTC/USD  105995.16  105995.16

3  1762804800000  2025-11-10 20:00:00  BTC/USD  105769.15  106260.00


         low      close  Volume BTC    Volume USD
0  105867.10  105957.91    6.502861  6.890296e+05
1  105348.03  105992.49    5.499902  5.829483e+05
2  105251.73  105570.78   19.287521  2.036199e+06
3  105769.15  105995.16   27.902652  2.957546e+06
```

```
# Convert the 'date' column to datetime
df2["datetime"] = pd.to_datetime(df2["date"])
df2 = df2.drop(columns="date")

# Term 1: 1/1/2017 - 12/31/2020
crypto_hourly_term1 = df2[(df2["datetime"] >= "2017-01-01") &
(df2["datetime"] <= "2020-12-31")]

# Term 2: 1/1/2025 - current
crypto_hourly_term2 = df2[df2["datetime"] >= "2025-01-01"]

crypto_hourly_term1.head(5)
```

```
                unix   symbol      open      high       low     close
\
42623  1609372800000  BTC/USD  28898.55  29316.49  28889.66  29096.60

42624  1609369200000  BTC/USD  28713.18  28930.98  28661.93  28898.55
```

```
42625  1609365600000  BTC/USD  28916.95  28943.74  28606.89  28713.18

42626  1609362000000  BTC/USD  28783.62  28998.00  28559.79  28916.95

42627  1609358400000  BTC/USD  28788.87  28998.00  28637.55  28783.62


       Volume BTC    Volume USD            datetime
42623  149.898636  4.361541e+06 2020-12-31 00:00:00
42624   49.153198  1.420456e+06 2020-12-30 23:00:00
42625   94.474535  2.712664e+06 2020-12-30 22:00:00
42626  273.429226  7.906739e+06 2020-12-30 21:00:00
42627  197.972330  5.698360e+06 2020-12-30 20:00:00

# Plot Term 1
plt.figure(figsize=(12, 5))
plt.plot(crypto_hourly_term1["datetime"],
crypto_hourly_term1["close"], color="blue")
plt.title("BTC Hourly Close Prices: Term 1 (2017-2020)")
plt.xlabel("Date")
plt.ylabel("Close Price (USD)")
plt.grid(True)
plt.show()

# Plot Term 2
plt.figure(figsize=(12, 5))
plt.plot(crypto_hourly_term2["datetime"],
crypto_hourly_term2["close"], color="green")
plt.title("BTC Hourly Close Prices: Term 2 (2025-current)")
plt.xlabel("Date")
plt.ylabel("Close Price (USD)")
plt.grid(True)
plt.show()
```





```
print("Term1:")
missing_volume(crypto_hourly_term1)
print("Term2:")
missing_volume(crypto_hourly_term2)
```

```
Term1:
Missing BTC volume: 204 out of 35038 (0.58%)
Missing USD volume: 204 out of 35038 (0.58%)
Term2:
Missing BTC volume: 3 out of 7536 (0.04%)
Missing USD volume: 3 out of 7536 (0.04%)
```

Dataset #3 | Trump Social Media (X and Truth) Posts

The dataset contains public posts made by Donald Trump on X (formerly Twitter) and Truth Social. Each entry includes a timestamp (both in eastern time and in Unix epoch), the platform, the full post text, and link. From these, we can derive further metric during EDA. The timestamp is the most important field because the Unix epoch format allows us to cleanly join this dataset with our financial datasets, which also rely on time based records.

This dataset is relevant because it allows us to compare the content of Trump's posts, specifically, the presence of certain keywords, with movements in financial markets. By aligning each post's epoch timestamp with hourly or daily market data, we can test whether posts containing particular terms coincide with shifts in market prices.

There are a few limitations of this dataset. The long gap in X posts during Trump's suspension (January 8, 2021 to November 19, 2022) creates a missing period unrelated to real posting behavior. Since we base our observations of post data by the text description, posts that do not contain any text, such as image or video only posts cannot be quantified easily.

Overall, the dataset is structured, consistent, and straightforward to merge with our financial data, making it well-suited for evaluating whether specific types of posts or keywords align with market reactions.

```
original_posts =
pd.read_csv("data/00-raw/trump_social_posts_2016_to_now.csv")
original_posts.head()
```

```
  platform               timestamp_et  timestamp_epoch  \
0  twitter  2025-11-09T17:51:18-05:00       1762728678
1  twitter  2025-11-09T16:53:22-05:00       1762725202
2  twitter  2025-11-09T16:38:31-05:00       1762724311
3  twitter  2025-11-09T16:27:19-05:00       1762723639
4  twitter  2025-11-09T16:21:39-05:00       1762723299

                                                link link_type  \
0  https://truthsocial.com/@realDonaldTrump/posts...      post
1  https://truthsocial.com/@realDonaldTrump/posts...      post
2  https://truthsocial.com/@realDonaldTrump/posts...      post
3  https://truthsocial.com/@realDonaldTrump/posts...      post
4  https://truthsocial.com/@realDonaldTrump/posts...      post

                                         description
0  RT @ NewtGingrich The New York Post report on ...
1                                                NaN
2  https:// nypost.com/2025/10/24/us-news/ kash-p...
3  https:// thefederalist.com/2025/10/23/l indsey...
4  DHS sees biggest jump in public approval among...
```

```
original_posts['platform'].value_counts()
```

```
platform
twitter    61995
Name: count, dtype: int64
```

```python
posts_raw = pd.read_csv("./data/01-interim/posts.csv")
posts_raw.head()
```

```
  platform               timestamp_et  timestamp_epoch  \
0    truth  2025-11-09T17:51:18-05:00       1762728678
1    truth  2025-11-09T16:53:22-05:00       1762725202
2    truth  2025-11-09T16:38:31-05:00       1762724311
3    truth  2025-11-09T16:27:19-05:00       1762723639
4    truth  2025-11-09T16:21:39-05:00       1762723299

                                                link link_type  \
0  https://truthsocial.com/@realDonaldTrump/posts...      post
1  https://truthsocial.com/@realDonaldTrump/posts...      post
2  https://truthsocial.com/@realDonaldTrump/posts...      post
3  https://truthsocial.com/@realDonaldTrump/posts...      post
4  https://truthsocial.com/@realDonaldTrump/posts...      post

                                         description platform_ignore
0  RT @ NewtGingrich The New York Post report on ...         twitter
1                                                NaN         twitter
2  https:// nypost.com/2025/10/24/us-news/ kash-p...         twitter
3  https:// thefederalist.com/2025/10/23/l indsey...         twitter
4  DHS sees biggest jump in public approval among...         twitter
```

```python
platformtypes = posts_raw['platform_ignore'].value_counts()
linktypes = posts_raw['link_type'].value_counts()
print(platformtypes, linktypes)
```

```
platform_ignore
twitter    61995
Name: count, dtype: int64 link_type
post    61995
Name: count, dtype: int64
```

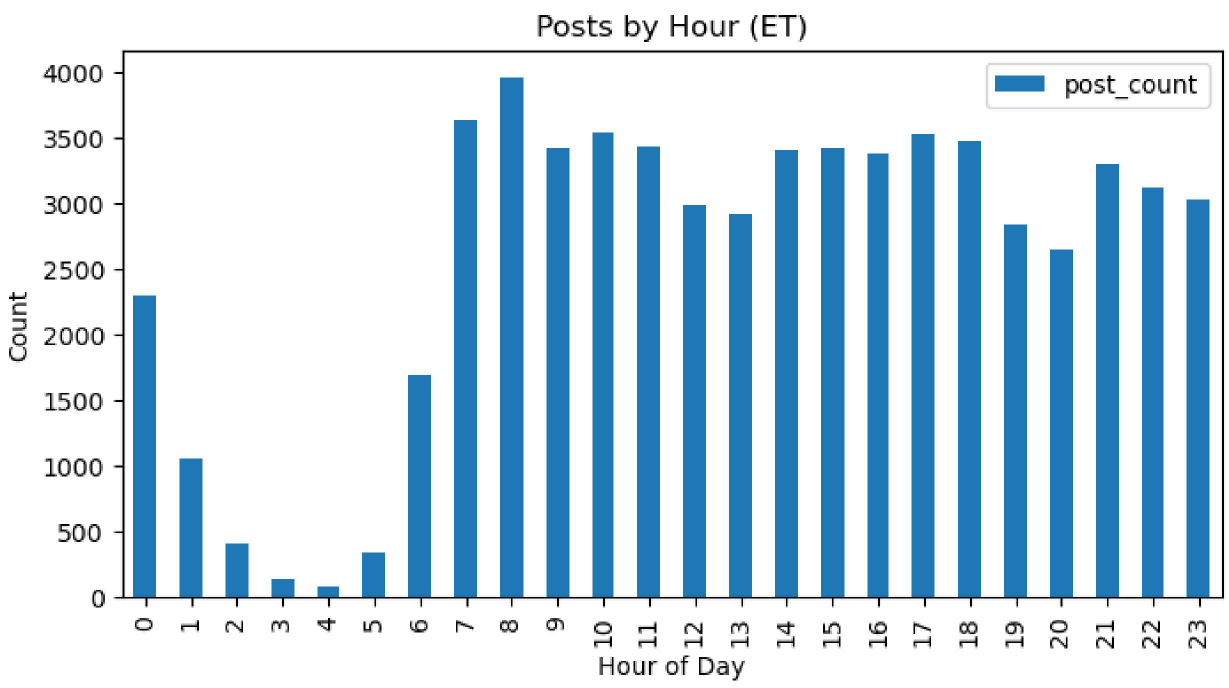
```python
posts_clean = pd.read_csv("./data/02-processed/posts.csv")
posts_clean = posts_clean.drop(columns = "link_type")
posts_clean = posts_clean.drop(columns = "platform_ignore")
posts_clean.head()
```

```
  platform               timestamp_et  timestamp_epoch  \
0    truth  2025-11-09T17:51:18-05:00       1762728678
1    truth  2025-11-09T16:53:22-05:00       1762725202
2    truth  2025-11-09T16:38:31-05:00       1762724311
3    truth  2025-11-09T16:27:19-05:00       1762723639
4    truth  2025-11-09T16:21:39-05:00       1762723299

                                                link  \
0  https://truthsocial.com/@realDonaldTrump/posts...
1  https://truthsocial.com/@realDonaldTrump/posts...
2  https://truthsocial.com/@realDonaldTrump/posts...
3  https://truthsocial.com/@realDonaldTrump/posts...
4  https://truthsocial.com/@realDonaldTrump/posts...

                                         description
0  RT @ NewtGingrich The New York Post report on ...
1                                                NaN
2  https:// nypost.com/2025/10/24/us-news/ kash-p...
3  https:// thefederalist.com/2025/10/23/l indsey...
4  DHS sees biggest jump in public approval among...
```

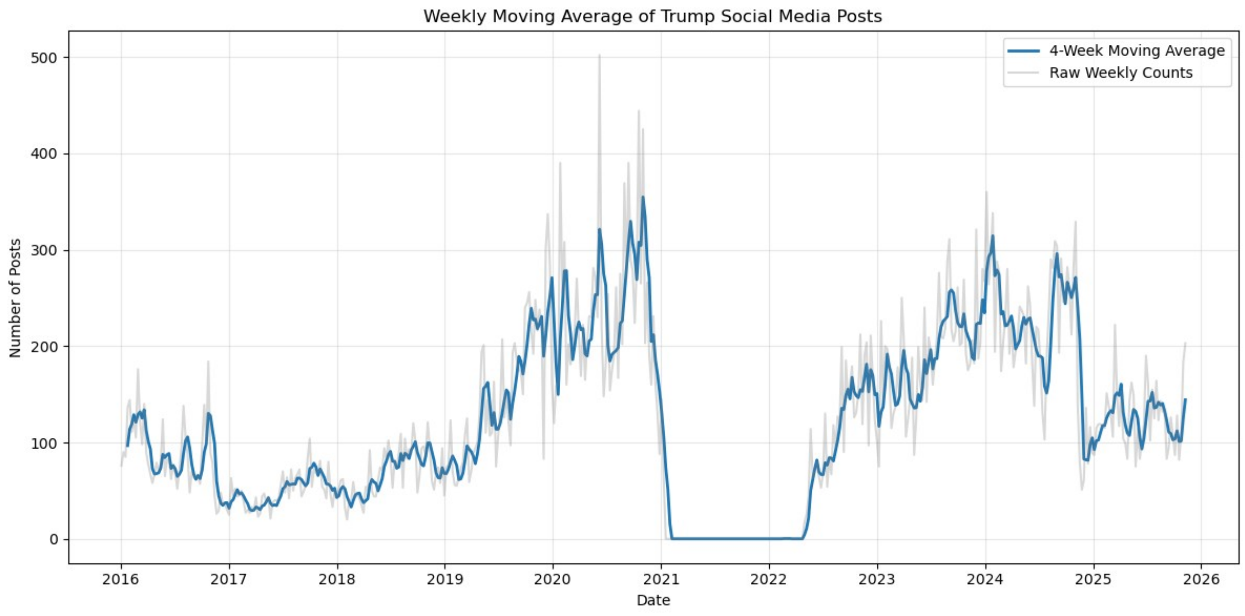
```
nan_counts = posts_clean.isna().sum().to_frame(name='NaN_count')
nan_counts
```

```
                 NaN_count
platform                 0
timestamp_et             0
timestamp_epoch          0
link                     0
description           5195
```

```
nan_description =
posts_clean[posts_clean['description'].isna()].copy()
posts_clean['description'] =
posts_clean['description'].fillna("").astype(str)

ts_utc = pd.to_datetime(posts_clean['timestamp_et'], utc=True)
posts_clean['timestamp_et'] = ts_utc.dt.tz_convert('America/New_York')
posts_clean['date'] = posts_clean['timestamp_et'].dt.date
posts_clean['hour'] = posts_clean['timestamp_et'].dt.hour

posts_by_date =
posts_clean.groupby('date').size().to_frame('post_count')
posts_by_hour =
posts_clean.groupby('hour').size().to_frame('post_count')

#graph posts per hour of the day
posts_by_hour.plot(kind='bar', figsize=(8,4), title='Posts by Hour
(ET)')
plt.xlabel('Hour of Day')
plt.ylabel('Count')
plt.show()
```



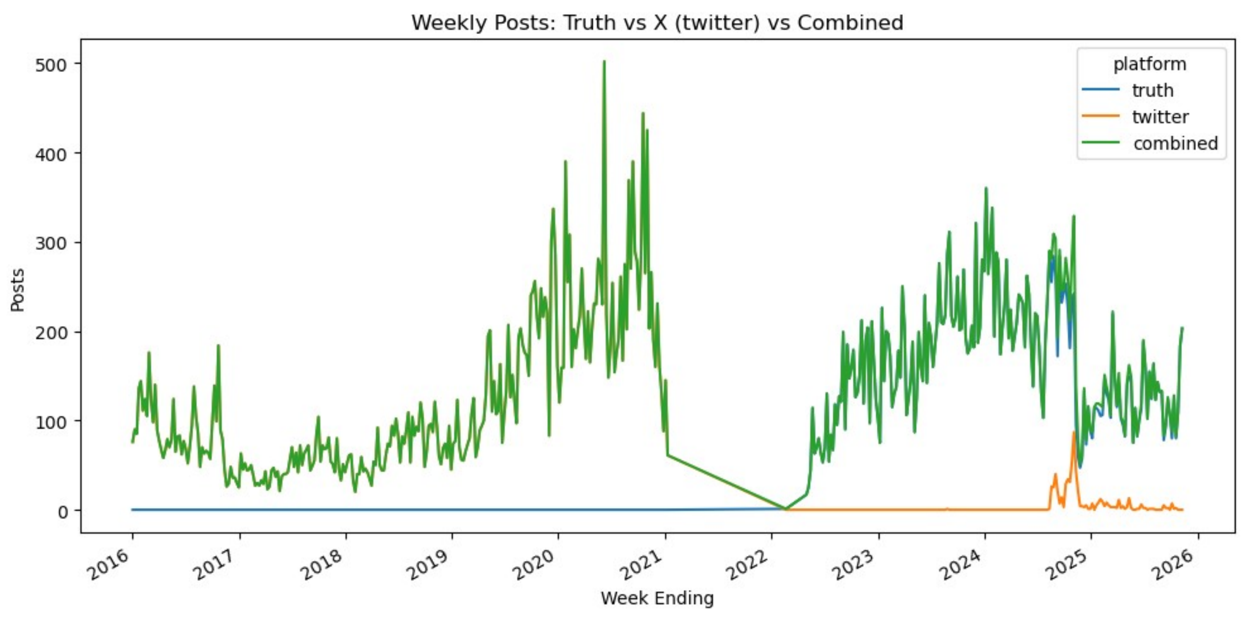
```
if not
pd.api.types.is_datetime64_any_dtype(posts_clean['timestamp_et']):
    posts_clean['timestamp_et'] =
pd.to_datetime(posts_clean['timestamp_et'], utc=True)

weekly_counts =
posts_clean.set_index('timestamp_et').resample('W').size()

rolling_mean = weekly_counts.rolling(window=4).mean()

plt.figure(figsize=(12, 6))

plt.plot(rolling_mean.index, rolling_mean, label='4-Week Moving
Average', color='tab:blue', linewidth=2)

plt.plot(weekly_counts.index, weekly_counts, label='Raw Weekly
Counts', color='gray', alpha=0.3)

plt.title('Weekly Moving Average of Trump Social Media Posts')
plt.xlabel('Date')
plt.ylabel('Number of Posts')
plt.legend()
plt.grid(True, alpha=0.3)

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



```python
df = pd.read_csv("./data/01-interim/posts.csv")

# Parse as timezone-aware UTC datetimes
df["timestamp_et"] = pd.to_datetime(df["timestamp_et"], utc=True,
errors="coerce")

# Define date range as UTC-aware timestamps
start = pd.Timestamp("2021-06-01", tz="UTC")
end   = pd.Timestamp("2022-02-01", tz="UTC")   # end is exclusive

# Filter rows in the range [start, end)
mask = (df["timestamp_et"] >= start) & (df["timestamp_et"] < end)
subset = df[mask]

print("Number of posts between June 2021 and Jan 2022:", len(subset))
```

```
Number of posts between June 2021 and Jan 2022: 0
```

```python
weekly_platform = (posts_clean.groupby([pd.Grouper(key='timestamp_et',
freq='W-SUN'),
'platform']).size().unstack(fill_value=0).reindex(columns=['truth',
'twitter'], fill_value=0))
weekly_platform['combined'] = weekly_platform.sum(axis=1)

ax = weekly_platform[['truth', 'twitter',
'combined']].plot(kind='line', figsize=(10, 5), title='Weekly Posts:
Truth vs X (twitter) vs Combined')
ax.set_xlabel('Week Ending')
ax.set_ylabel('Posts')
plt.tight_layout()
plt.show()
```



```
posts = posts_clean.copy()
posts.head()
```

```
  platform              timestamp_et  timestamp_epoch  \
0    truth 2025-11-09 17:51:18-05:00       1762728678
1    truth 2025-11-09 16:53:22-05:00       1762725202
2    truth 2025-11-09 16:38:31-05:00       1762724311
3    truth 2025-11-09 16:27:19-05:00       1762723639
4    truth 2025-11-09 16:21:39-05:00       1762723299

                                                link  \
0  https://truthsocial.com/@realDonaldTrump/posts...
1  https://truthsocial.com/@realDonaldTrump/posts...
2  https://truthsocial.com/@realDonaldTrump/posts...
3  https://truthsocial.com/@realDonaldTrump/posts...
4  https://truthsocial.com/@realDonaldTrump/posts...

                                         description        date  hour

0  RT @ NewtGingrich The New York Post report on ...  2025-11-09    17

1                                                     2025-11-09    16

2  https:// nypost.com/2025/10/24/us-news/ kash-p...  2025-11-09    16

3  https:// thefederalist.com/2025/10/23/l indsey...  2025-11-09    16

4  DHS sees biggest jump in public approval among...  2025-11-09    16
```

Some interesting observations are that there was a period where there were no posts on Twitter or Truth Social. After a quick google search, it seems like it was a result of his Twitter account suspension which began on January 8, 2021 to November 19, 2022. Trump also created his own social media called Truth Social which contains the majority of his posts since 2022.

Dataset #4 - S&P500, Dow Jones, NASDAQ Daily Values

This dataset contains daily historical market data for three major U.S. stock market indices represented by exchange-traded funds (ETFs): the S&P 500 (SPX), the Nasdaq Composite (COMP), and the Dow Jones Industrial Average (DJIA). Each record includes the financial metrics: open, high, low, and close prices (in U.S. dollars per share). The high and low indicate the day's trading range, with their difference offering insight into the volatility of the ETF, showcasing investor uncertainty which could be related outside influences such as social media posts. The close price is the end of day market price of the ETF, a metric which can be used to analyze general trends or sudden bursts within the market in short or long periods.

There are several limitations and considerations about this dataset. The lack of after-market activity-trading outside of the 'open' trading period (9:30am-4pm EST) could miss events occuring outside of those hours, i.e. social media posts that don't occur within that timeframe. Additionally, global, political, or macroeconomic events could cause abrupt changes in values that cannot be explained through the numerical data without additional context. Finally, ETFs are traded by public investors, being influenced greatly by sentiment, speculation, and other, personal concerns that also cannot be explained by purely numerical data.

```
SPX = pd.read_csv('data/00-raw/S&P500 Data - S&P.csv')
COMP = pd.read_csv('data/00-raw/NASDAQ Data - NASDAQ.csv')
DJIA = pd.read_csv('data/00-raw/DOW - DOW.csv')
#updating column names to specify ETF
SPX.columns = [col if col == 'Date' else 'SPX_' + col for col in
SPX.columns]
COMP.columns = [col if col == 'Date' else 'COMP_' + col for col in
COMP.columns]
DJIA.columns = [col if col == 'Date' else 'DJIA_' + col for col in
DJIA.columns]
#merging seperate datasets into one
df4 = SPX.merge(COMP, on='Date', how='outer').merge(DJIA, on='Date',
how='outer')
df4['Date'] = pd.to_datetime(df4['Date'], format = '%m/%d/%Y')
#sorting by date
df4 = df4.sort_values(by='Date')
for col in df4.columns:
    if col != 'Date':
        df4[col] = df4[col].astype(str).str.replace(',', '',
regex=False).astype(float)
df4.head()
```

```
         Date  SPX_Open  SPX_High  SPX_Low  SPX_Close  COMP_Open
COMP_High  \
13 2016-01-04   2038.20   2038.20  1989.68    2012.66    4897.65
4903.09
21 2016-01-05   2013.78   2021.94  2004.17    2016.71    4917.84
4926.73
```

```
28 2016-01-06   2011.71   2011.71  1979.05    1990.26    4813.76
4866.04
35 2016-01-07   1985.32   1985.32  1938.83    1943.09    4736.40
4788.02
41 2016-01-08   1945.97   1960.40  1918.46    1922.03    4722.02
4742.57

    COMP_Low  COMP_Close  DJIA_Open  DJIA_High  DJIA_Low  DJIA_Close
13   4846.98     4903.09   17405.48   17405.48  16957.63    17148.94
21   4872.74     4891.43   17147.50   17195.84  17038.61    17158.66
28   4804.69     4835.76   17154.83   17154.83  16817.62    16906.51
35   4688.17     4689.43   16888.36   16888.36  16463.63    16514.10
41   4637.85     4643.63   16519.17   16651.89  16314.57    16346.45

# Term 1: 1/1/2017-12/31/2020
term1_index = df4[(df4["Date"] >= datetime.strptime("2017-01-01", '%Y-
%m-%d')) & (df4["Date"] < datetime.strptime("2020-12-31", '%Y-%m-
%d'))]

# Term 2: 1/1/2025-current
term2_index = df4[df4["Date"] >= datetime.strptime("2025-01-20", '%Y-
%m-%d')]

term1_index.head(5)

         Date  SPX_Open  SPX_High  SPX_Low  SPX_Close  COMP_Open
COMP_High  \
5  2017-01-03   2251.57   2263.88  2245.13    2257.83    5425.62
5452.57
14 2017-01-04   2261.60   2272.82  2261.60    2270.75    5440.91
5482.35
22 2017-01-05   2268.18   2271.50  2260.45    2269.00    5474.39
5495.86
29 2017-01-06   2271.14   2282.10  2264.06    2276.98    5499.08
5536.52
48 2017-01-09   2273.59   2275.49  2268.90    2268.90    5527.58
5541.08

    COMP_Low  COMP_Close  DJIA_Open  DJIA_High  DJIA_Low  DJIA_Close
5    5397.99     5429.08   19872.86   19938.53  19775.93    19881.76
14   5440.24     5477.00   19890.94   19956.14  19878.83    19942.16
22   5464.36     5487.94   19924.56   19948.60  19811.12    19899.29
29   5482.81     5521.06   19906.96   19999.63  19834.08    19963.80
48   5517.14     5531.82   19931.41   19943.78  19887.38    19887.38
```

Results

Exploratory Data Analysis

Tweet/Truth Keyword Filtering and Sentiment Analysis

First, we will make the description more easy to analyze by setting all characters to lower case and seperate URLs, which will make keyword matching and sentiment analysis work better.

```
posts["posts_lc"] = posts["description"].str.lower()
posts["posts_lc"] = (posts["posts_lc"]
    .str.replace(r"http\S+", " ", regex=True)
    .str.replace(r"\s+", " ", regex=True)
    .str.strip()
)
posts[["description", "posts_lc"]].head()
```

```
                                         description  \
0  RT @ NewtGingrich The New York Post report on ...
1
2  https:// nypost.com/2025/10/24/us-news/ kash-p...
3  https:// thefederalist.com/2025/10/23/l indsey...
4  DHS sees biggest jump in public approval among...

                                            posts_lc
0  rt @ newtgingrich the new york post report on ...
1
2  nypost.com/2025/10/24/us-news/ kash-patel-skew...
3  thefederalist.com/2025/10/23/l indsey-halligan...
4  dhs sees biggest jump in public approval among...
```

Now, in order to examine our hypothesis, we must identify which posts use vocabulary that are relevant to the economy and that we believe would potentially lead to market acting more volatile. Some keywords we want examine are placed into categories and then we do a keyword count.

```
econ_keywords = {
    "tariff": ["tariff", "tariffs"],
    "rate": ["rate", "rates", "interest rate", "hike", "cut"],
    "trade": ["trade", "china", "deal", "exports", "imports"],
    "jobs": ["jobs", "employment", "unemployment"],
    "inflation": ["inflation", "cpi", "prices", "cost of living"],
    "fed": ["fed", "federal reserve", "powell"],
    "market": ["market", "stock", "stocks", "dow", "nasdaq", "sp500",
"s&p"],
}

for cat, words in econ_keywords.items():
    pattern = r"\b(?:" + "|".join(re.escape(w) for w in words) + r")\
b"  # non-capturing group
    posts[f"kw_{cat}"] = posts["posts_lc"].str.contains(pattern,
regex=True)


kw_cols = [c for c in posts.columns if c.startswith("kw_")]
posts["contains_econ_keyword"] =
posts[kw_cols].any(axis=1).astype(int)

posts.head(5)
print(posts.head(5))

  platform              timestamp_et  timestamp_epoch  \
0    truth 2025-11-09 17:51:18-05:00       1762728678
1    truth 2025-11-09 16:53:22-05:00       1762725202
2    truth 2025-11-09 16:38:31-05:00       1762724311
3    truth 2025-11-09 16:27:19-05:00       1762723639
4    truth 2025-11-09 16:21:39-05:00       1762723299

                                                link  \
0  https://truthsocial.com/@realDonaldTrump/posts...
1  https://truthsocial.com/@realDonaldTrump/posts...
2  https://truthsocial.com/@realDonaldTrump/posts...
3  https://truthsocial.com/@realDonaldTrump/posts...
4  https://truthsocial.com/@realDonaldTrump/posts...

                                         description        date  hour
\
0  RT @ NewtGingrich The New York Post report on ...  2025-11-09    17

1                                                     2025-11-09    16

2  https:// nypost.com/2025/10/24/us-news/ kash-p...  2025-11-09    16

3  https:// thefederalist.com/2025/10/23/l indsey...  2025-11-09    16

4  DHS sees biggest jump in public approval among...  2025-11-09    16


                                            posts_lc  kw_tariff
kw_rate  \
0  rt @ newtgingrich the new york post report on ...      False
False
1                                                         False
False
2  nypost.com/2025/10/24/us-news/ kash-patel-skew...      False
False
3  thefederalist.com/2025/10/23/l indsey-halligan...      False
False
4  dhs sees biggest jump in public approval among...      False
False
```

```
   kw_trade  kw_jobs  kw_inflation  kw_fed  kw_market
contains_econ_keyword
0     False    False         False   False      False
0
1     False    False         False   False      False
0
2     False    False         False   False      False
0
3     False    False         False   False      False
0
4     False    False         False   False      False
0
```

It would also be helpful determining the tone of the post. Since this is a more qualitative value, we will use the Vader Sentiment library to gain a sentiment score. Using this score we can say if it is positive, neutral, or negative.

```
nltk.download("vader_lexicon")

sia = SentimentIntensityAnalyzer()

posts["sentiment_compound"] = posts["posts_lc"].apply(
    lambda x: sia.polarity_scores(x)["compound"]
)

posts["sentiment_label"] =
posts["sentiment_compound"].apply(classify_sentiment)

posts[["posts_lc", "sentiment_compound", "sentiment_label"]].head(5)
```

```
[nltk_data] Downloading package vader_lexicon to
[nltk_data]     /Users/kento/nltk_data...
[nltk_data]   Package vader_lexicon is already up-to-date!

                                            posts_lc
sentiment_compound  \
0  rt @ newtgingrich the new york post report on ...
0.9953
1
0.0000
2  nypost.com/2025/10/24/us-news/ kash-patel-skew...
0.0000
3  thefederalist.com/2025/10/23/l indsey-halligan...
0.0000
4  dhs sees biggest jump in public approval among...
0.4767

  sentiment_label
0        positive
```

```
1         neutral
2         neutral
3         neutral
4        positive
```

Bitcoin Hourly Data Analysis

The primary goal of this analysis is to visually detect deviations from the expected market behavior—the "abnormal returns" and "abnormal volatility" following social media post events.

In an ideal scenario, the control group's average price change and volatility lines would hover close to zero throughout the 24-hour window, and the 95% confidence band would consistently encompass the zero line.

First lets do one last round of cleanup on the post data.

```
btc_term1 = pd.read_csv("data/02-processed/btc_hourly_term1.csv")
posts = pd.read_csv("data/02-processed/posts_analyzed.csv")

posts_term1 = posts[(posts['date'] >= '2017-01-01') & (posts['date'] <
'2021-01-01')].copy()

posts_term1.tail(2)

tariff_tweets = posts_term1[posts_term1['kw_tariff'] == True].copy()
tariff_tweets['date'] = pd.to_datetime(tariff_tweets['date'])
btc_term1['datetime'] =
pd.to_datetime(btc_term1['datetime']).dt.tz_localize('UTC')

tariff_tweets.head(2)

      platform               timestamp_et  timestamp_epoch  \
32124  twitter  2020-11-21 23:54:18-05:00       1606020858
32134  twitter  2020-11-21 23:04:31-05:00       1606017871

                                                     link  \
32124  https://x.com/realdonaldtrump/status/133037402...
32134  https://x.com/realdonaldtrump/status/133036149...

                                             description       date
hour  \
32124  Thanks Mark. It's all a continuation of the ne... 2020-11-21
23
32134  Thanks Mark. It's all a continuation of the ne... 2020-11-21
23

                                                posts_lc  kw_tariff
kw_rate  \
32124  thanks mark. it's all a continuation of the ne...       True
False
```

```
32134  thanks mark. it's all a continuation of the ne...       True
False

       kw_trade  kw_jobs  kw_inflation  kw_fed  kw_market  \
32124     False    False         False   False      False
32134     False    False         False   False      False

       contains_econ_keyword  sentiment_compound sentiment_label
32124                      1             -0.3711        negative
32134                      1             -0.3711        negative
```

Let's just double check that we can access data.

```
btc_term1_indexed = btc_term1.set_index('datetime').sort_index()
first_timestamp = pd.to_datetime(tariff_tweets['timestamp_et'],
utc=True).to_list()[0]

first_tweet_hour = first_timestamp.floor('h')
btc_price = btc_term1_indexed.loc[first_tweet_hour, 'close']

print(f"First tariff tweet timestamp (Full): {first_timestamp}")
print(f"Hour start for lookup: {first_tweet_hour}")
print(f"BTC Close Price at that hour: {btc_price}")

First tariff tweet timestamp (Full): 2020-11-22 04:54:18+00:00
Hour start for lookup: 2020-11-22 04:00:00+00:00
BTC Close Price at that hour: 18605.81
```

Great, now lets create a dataframe which includes what we plan to use.

```
tariff_tweets['tweet_datetime_full'] =
pd.to_datetime(tariff_tweets['timestamp_et'], utc=True)

# 2. Round down all tweet times to the nearest hour (t=0)
tweet_hours = tariff_tweets['tweet_datetime_full'].dt.floor('h')

# 3. Extract the BTC closing price for each rounded hour
# We use the full Series of UTC hours to lookup prices in the
correctly timezone-aware btc_term1_indexed.
btc_prices_at_tweet_hour = btc_term1_indexed.loc[tweet_hours,
'close'].reset_index(drop=True)

# 4. Create the final DataFrame
hourly_prices_df = pd.DataFrame({
    'tweet_datetime_utc':
tariff_tweets['tweet_datetime_full'].reset_index(drop=True),
    'tweet_hour_start_utc': tweet_hours.reset_index(drop=True),
    'btc_close_price_t0': btc_prices_at_tweet_hour
})
```

```
hourly_prices_df.head()

          tweet_datetime_utc       tweet_hour_start_utc
btc_close_price_t0
0 2020-11-22 04:54:18+00:00 2020-11-22 04:00:00+00:00
18605.81
1 2020-11-22 04:04:31+00:00 2020-11-22 04:00:00+00:00
18605.81
2 2020-11-02 19:29:20+00:00 2020-11-02 19:00:00+00:00
13576.79
3 2020-10-01 02:47:30+00:00 2020-10-01 02:00:00+00:00
10821.83
4 2020-09-11 02:15:09+00:00 2020-09-11 02:00:00+00:00
10268.87
```

Now that we have the close price at the time of the tweet, lets also get the hours around when the tweet was sent so that we can create some price over time graphs.

```
price_columns = [str(i) for i in range(-24, 25)]
HOURS_TO_EXTRACT = 49
tweet_timeline_data = []
reference_column = '0'

if 'tweet_datetime_full' not in tariff_tweets.columns:
    tariff_tweets['tweet_datetime_full'] =
pd.to_datetime(tariff_tweets['timestamp_et'], utc=True)

for _, tweet in tariff_tweets.iterrows():
    tweet_hour_start = tweet['tweet_datetime_full'].floor('h')
    series_start_time = tweet_hour_start - pd.Timedelta(hours=24)

    start_pos = btc_term1_indexed.index.get_loc(series_start_time)

    # Exclude posts whose window would lie outside of the term.
    if start_pos + HOURS_TO_EXTRACT <= len(btc_term1_indexed):
        price_series = btc_term1_indexed['close'].iloc[start_pos :
start_pos + HOURS_TO_EXTRACT]

        row_data = {
            'link': tweet['link'],
            'tweet_datetime_utc': tweet['tweet_datetime_full']
        }
        row_data.update(dict(zip(price_columns, price_series.values)))
        tweet_timeline_data.append(row_data)

# 3. CREATE DataFrame: tweet_timeline_term1
final_columns = ['link', 'tweet_datetime_utc'] + price_columns
tweet_timeline_term1 = pd.DataFrame(tweet_timeline_data,
columns=final_columns)
```

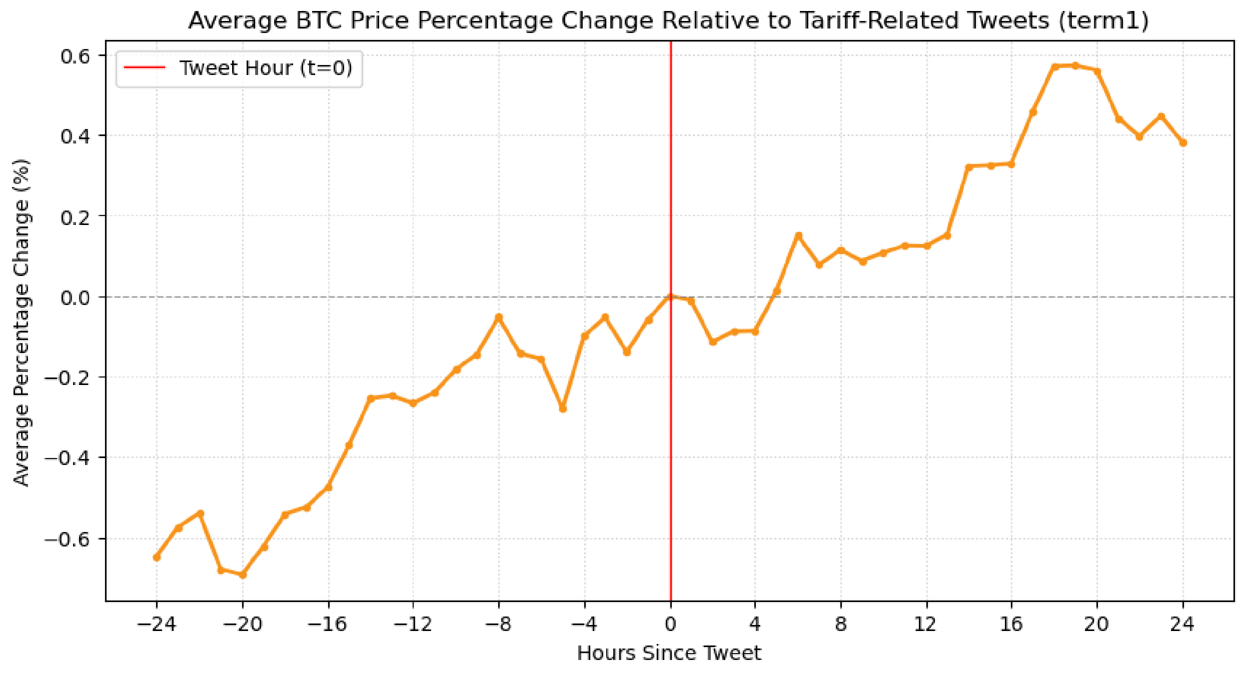
```
tweet_timeline_term1[price_columns] =
tweet_timeline_term1[price_columns].apply(pd.to_numeric,
errors='coerce')

price_t0 = tweet_timeline_term1[[reference_column]].values

price_data = tweet_timeline_term1[price_columns].values
percentage_change_data = ((price_data / price_t0) - 1) * 100

tweet_timeline_term1[price_columns] = percentage_change_data
tweet_timeline_term1[reference_column] = 0.0

new_column_names = {col: f'pct_change_{col}' for col in price_columns}
tweet_timeline_term1.rename(columns=new_column_names, inplace=True)

tweet_timeline_term1.head()

                                                link  \
0  https://x.com/realdonaldtrump/status/133037402...
1  https://x.com/realdonaldtrump/status/133036149...
2  https://x.com/realdonaldtrump/status/132334647...
3  https://x.com/realdonaldtrump/status/131149794...
4  https://x.com/realdonaldtrump/status/130424204...

           tweet_datetime_utc  pct_change_-24  pct_change_-23
pct_change_-22  \
0 2020-11-22 04:54:18+00:00       -0.018650        0.391759
0.240946
1 2020-11-22 04:04:31+00:00       -0.018650        0.391759
0.240946
2 2020-11-02 19:29:20+00:00        1.653631        1.904132
1.783706
3 2020-10-01 02:47:30+00:00       -0.402982       -0.663659        -
0.707182
4 2020-09-11 02:15:09+00:00        1.028059        1.107230
1.221751

   pct_change_-21  pct_change_-20  pct_change_-19  pct_change_-18  \
0        0.445130        1.029195        0.125821        0.056058
1        0.445130        1.029195        0.125821        0.056058
2        0.959947        1.472292        1.816851        1.354370
3       -1.061928       -0.957601       -1.070614       -1.012213
4        1.127485        0.756753        0.147923        0.115397

   pct_change_-17  ...  pct_change_15  pct_change_16  pct_change_17  \
0       -0.293295  ...      -0.054123      -0.157639      -0.037623
1       -0.293295  ...      -0.054123      -0.157639      -0.037623
2        1.203156  ...      -0.117775      -0.275102      -0.335131
3       -1.170504  ...      -3.158616      -2.158415      -1.872234
```

```
4        0.145293  ...       0.314640       0.518947       0.633857

   pct_change_18  pct_change_19  pct_change_20  pct_change_21  
pct_change_22  \
0       0.247826      -0.843285      -2.554525      -1.985724      -
2.057476
1       0.247826      -0.843285      -2.554525      -1.985724      -
2.057476
2       0.949488       1.639047       1.126997       1.200873       
1.272981
3      -1.938951      -2.144000      -1.948284      -1.785558      -
2.025628
4       0.606006       0.869326       1.089117       1.304330       
0.823070

   pct_change_23  pct_change_24
0      -1.144212      -0.669522
1      -1.144212      -0.669522
2       1.404308       1.202051
3      -2.058802      -2.019899
4       1.033512       0.820149

[5 rows x 51 columns]
```

Lets take the average of each column and graph our results.

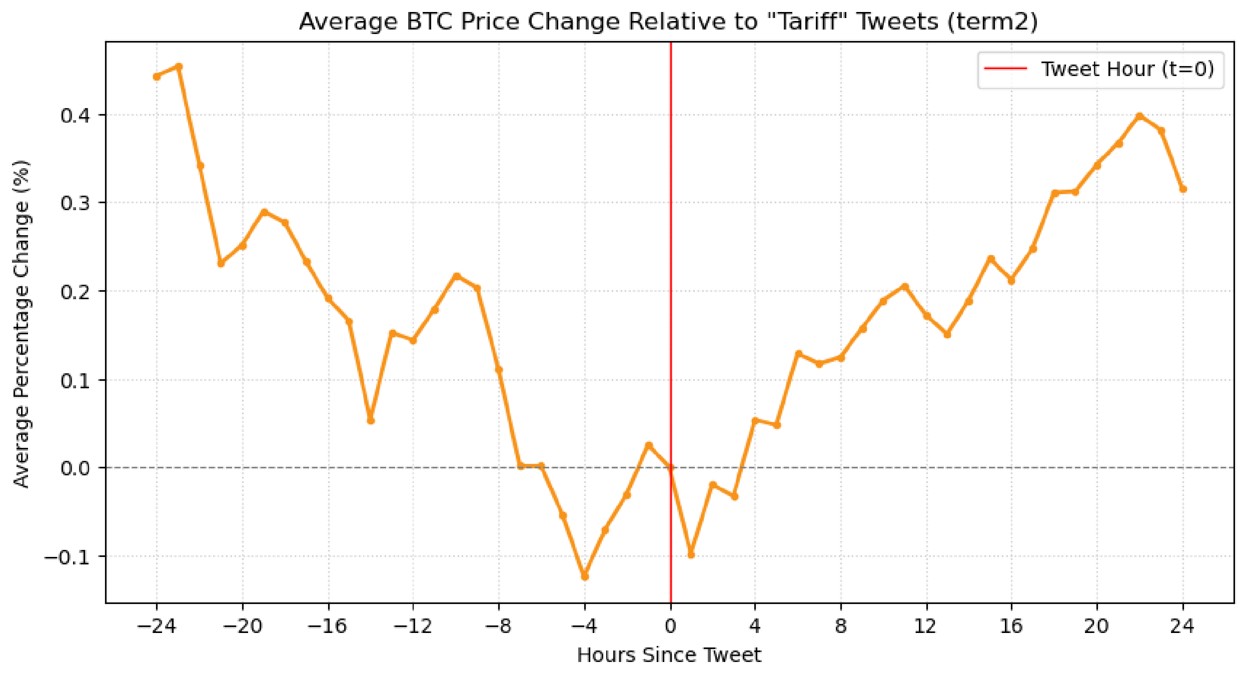
```python
pct_change_columns = [f'pct_change_{i}' for i in range(-24, 25)]

# 2. Calculate the average percentage change across all tweets for
each hour
avg_change = tweet_timeline_term1[pct_change_columns].mean()

# 3. Wrangle the data into a clean DataFrame for plotting
plot_data = avg_change.reset_index()
plot_data.columns = ['Hour_String', 'Avg_Pct_Change']
plot_data['Hour'] = plot_data['Hour_String'].str.split('_').str[-
1].astype(int)
plot_data = plot_data.sort_values(by='Hour')

# 4. Generate the plot (using a concise format as requested)
plt.figure(figsize=(10, 5))
plt.plot(plot_data['Hour'], plot_data['Avg_Pct_Change'], marker='.',
linewidth=2, color='#F7931A') # BTC Orange

# Add formatting for better readability
plt.axhline(0, color='gray', linestyle='--', linewidth=0.8)
plt.axvline(0, color='red', linestyle='-', linewidth=1, label='Tweet
Hour (t=0)')
plt.title('Average BTC Price Percentage Change Relative to Tariff-
Related Tweets (term1)')
```

```
plt.xlabel('Hours Since Tweet')
plt.ylabel('Average Percentage Change (%)')
plt.grid(True, linestyle=':', alpha=0.6)
plt.xticks(range(-24, 25, 4))
plt.legend()
plt.show()
```



The above is the average price change before, and after a tariff related tweet. Now I have generalized the above steps to a function in `modules/hourly_data_functions.py`

```
plt_object, timeline_df = analyze_btc_impact_auto(
    term="term2",
    keyword_category="tariff",
    time_range_hours=24
)
```



Analysis Summary: Time-Series Event Study

The next cells of code perform a **Time-Series Event Study** to measure the average short-term **impact of specific keyword mentions on the price of Bitcoin (BTC)** across two distinct time periods.

---

1. Data Calculation Methodology

The code iterates through all of the **Keyword Categorys** (e.g., 'tariff', 'fed') and a **Term** ('term1': 2017-2020, 'term2': 2025-Present) to aggregate the price data.

1. **Event Data Alignment:** The external function extracts the BTC hourly percentage price change for a window of ± 72 **hours** around every keyword mention (event time $t=0$).
2. **Averaging (The Mean Line):** The **Average Percentage Change (**Avg_Pct_Change**)** is calculated by taking the **mean** of the percentage price changes across all events in the sample for each specific hour in the ± 72 window.

$$\overline{\Delta P_t} = \frac{1}{N} \sum_{i=1}^{N} \Delta P_{i,t}$$

   where N is the sample size, and $\Delta P_{i,t}$ is the price change for event i at hour t.
3. **Volatility (The Shaded Area):** The **Standard Deviation (**Std_Pct_Change**)** is calculated to measure the variability of price changes around the average effect at each hour.

---

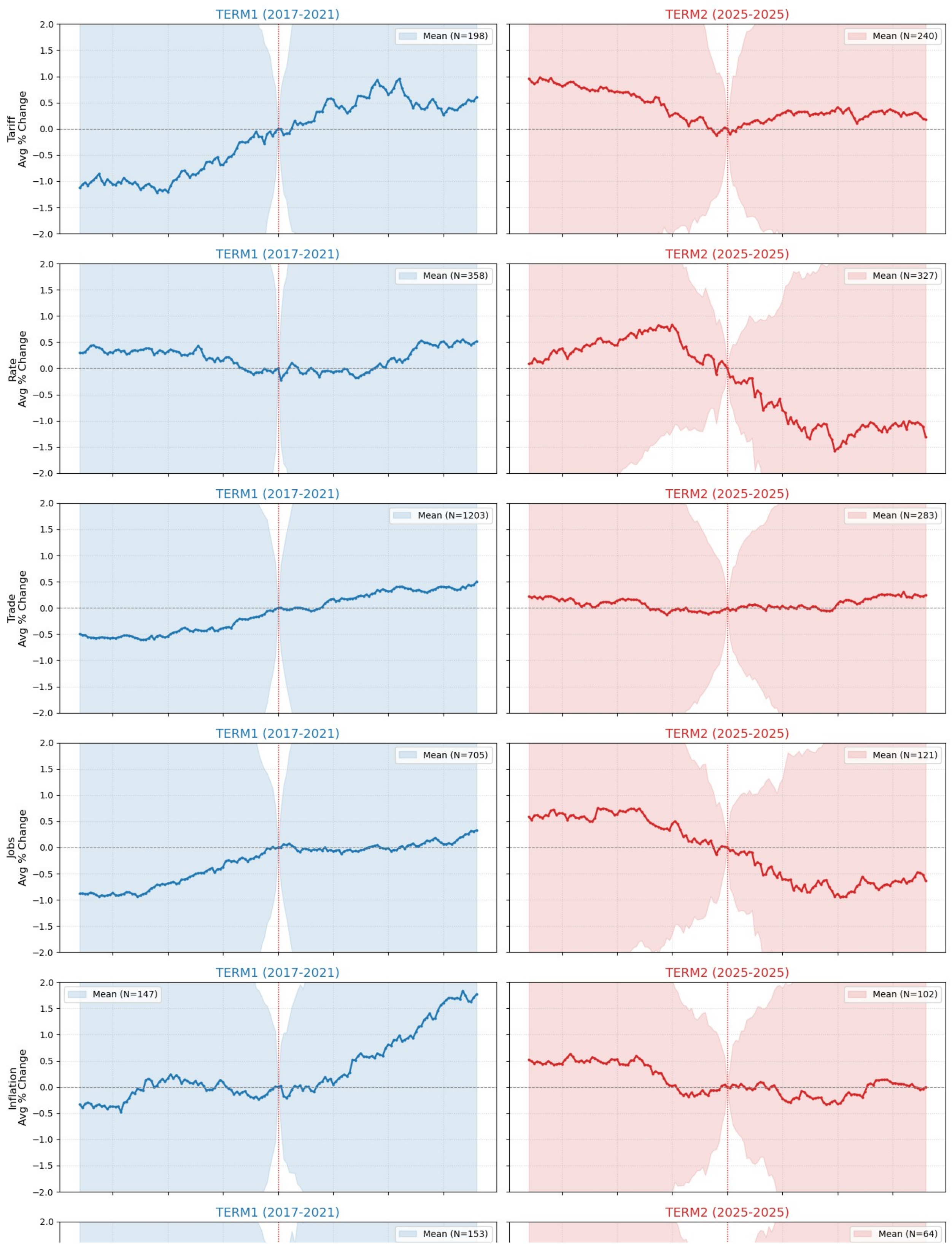
2. Graph Visualization

A multi-panel grid is generated, where each subplot visualizes the aggregated effect for one Keyword Category during one Term.

Element	Data Source	Purpose
Main Line Plot	`Avg_Pct_Change`	Shows the **average price trajectory** of BTC relative to the event time ($t = 0$).
Shaded Area	`Avg_Pct_Change` $\pm$ `Std_Pct_Change`	Represents the ± 1 **Standard Deviation (± 1 SD)**, indicating the **volatility** (or uncertainty) of the price movement.
Vertical Red Line	$t = 0$	Marks the **exact time** the keyword event occurred.
Horizontal Gray Line	0%	Marks the **zero percent price change** baseline.

```
run_full_btc_impact_grid(
    analyze_func=analyze_btc_impact_auto,
    terms=["term1", "term2"],
    categories=["tariff", "rate", "trade", "jobs", "inflation", "fed",
"market", "econ"],
    time_range=72,
    y_min=-2,
    y_max=2
)
```

Average BTC Price Change Relative to Keyword Mentions (±72h)



Improvements: Statistical Trend Analysis and Control Group

The following updated code significantly enhances the analysis by introducing a **statistical framework** and generating a **control group** for robust comparison.

---

1. Statistical Trend Analysis

The new function `calculate_p_trend_shift` formalizes the assessment of price momentum around the event time ($t=0$).

- **Linear Regression:** It fits a **linear trend** (a straight line) to the average price change data both in the **pre-event period** (Hour -72 to -1) and the **post-event period** (Hour $+1$ to $+72$).
- **Significance Test:** It performs a two-sample **t-test** comparing the **slope** of the pre-event trend ($\text{slope}_{\text{pre}}$) against the slope of the post-event trend ($\text{slope}_{\text{post}}$). This tests for a statistically significant change in momentum or price direction following the event.
- **P-Value Output:** The result is a **p-value** ($\text{p}_{\text{keyword}}$ or $\text{p}_{\text{control}}$) which represents the probability of observing such a difference in slopes under the null hypothesis that the pre-event and post-event trends are the same (i.e., no real change in price momentum occurred).

$$\text{T-Statistic}=\frac{\text{slope}_{\text{post}}-\text{slope}_{\text{pre}}}{\sqrt{\text{SE}_{\text{post}}^2+\text{SE}_{\text{pre}}^2}}$$

---

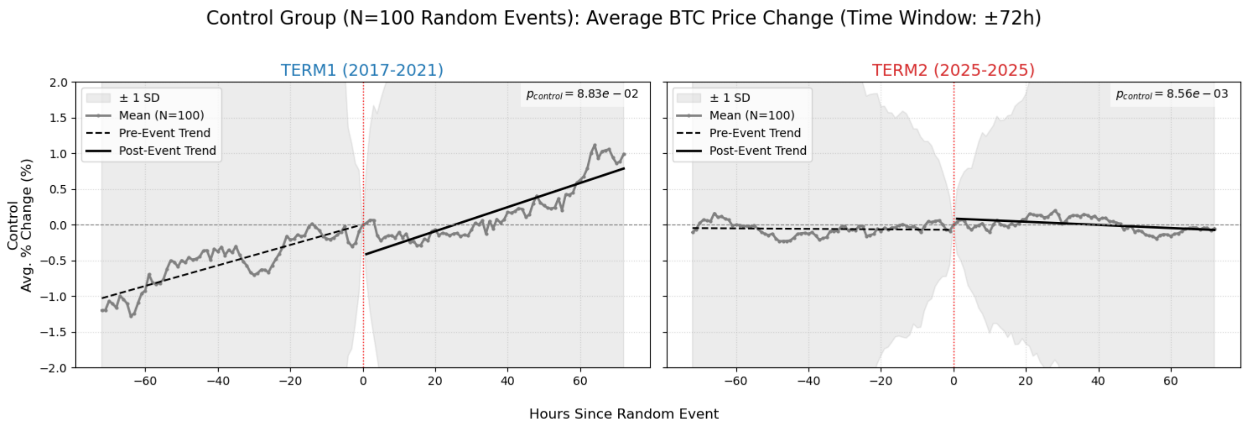
2. Control Group Generation and Plotting

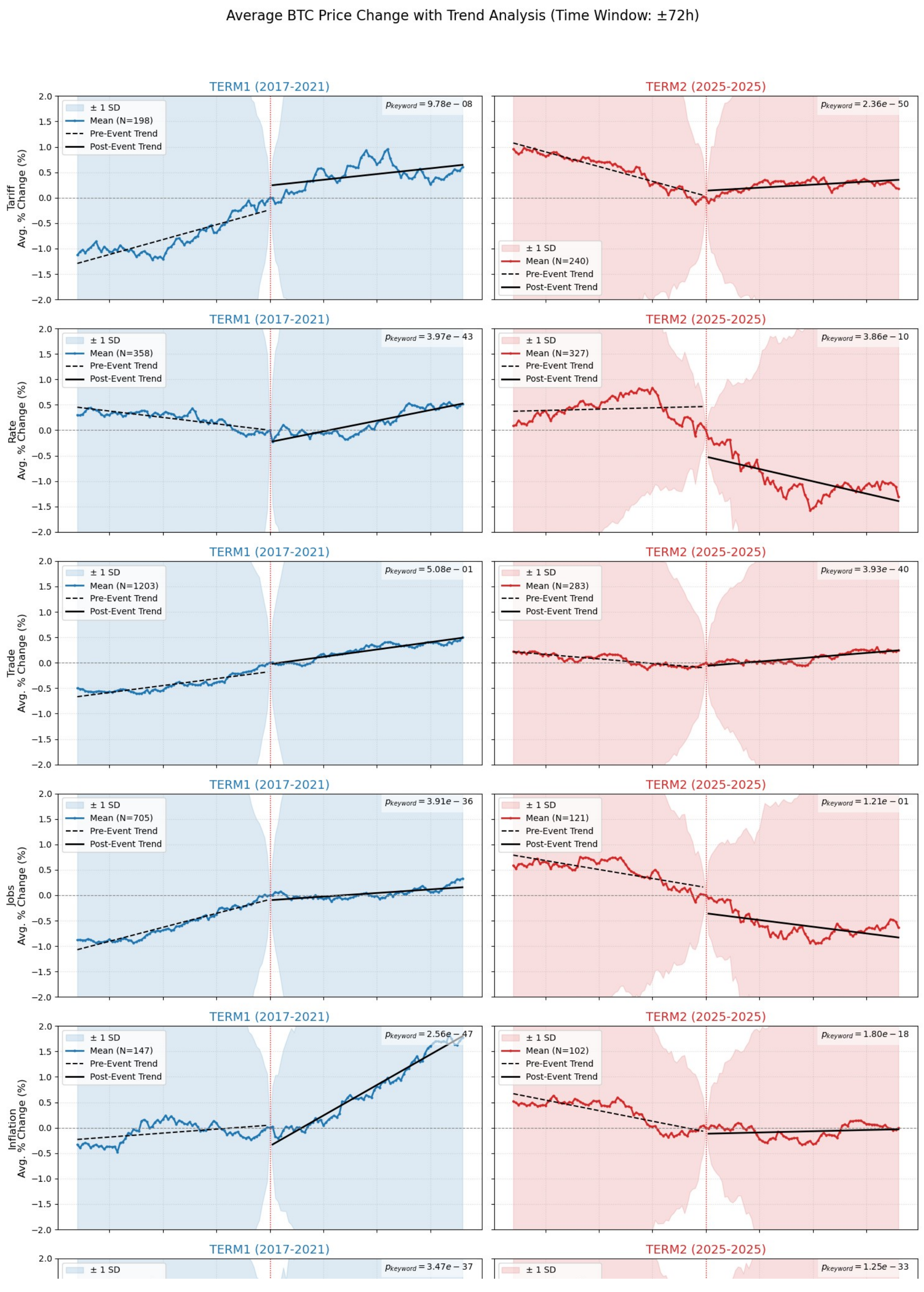
To validate the findings, the code now compares the keyword effects against a random baseline.

New Feature	Function / Logic	Purpose
Control Group Generation	`generate_control_data`	Randomly selects $N=100$ time points from the historical BTC data (`btc_hourly_{term}.csv`) to serve as "random events." This creates a baseline to establish typical market noise.
Control Plot (`fig_control`)	New dedicated plot	Visualizes the average BTC price change, ± 1 SD, and trend lines for the **random events**.
Trend Line Visualization	Linear trend fit on Avg_Pct_Change	**Dashed black lines** show the fitted trend *before* the event ($t=-72$ to -1), and **solid**

New Feature	Function / Logic	Purpose
		black lines show the trend *after* the event ($t = +1$ to $+72$).
Statistical Annotation	p_{keyword} and p_{control}	The p-value from the trend shift test is displayed on the top right of each chart, quantitatively measuring if the average price experienced a **statistically significant change in momentum** around the event time.

```
run_btc_analysis()
```




Average BTC Price Change with Trend Analysis (Time Window: ±72h)
TERM1 (2017-2021)
TERM2 (2025-2025)
Tariff
Avg. % Change (%)
$p_{keyword} = 9.78e-08$
Mean (N=198)
$p_{keyword} = 2.36e-50$
Mean (N=240)
Rate
$p_{keyword} = 3.97e-43$
Mean (N=358)
$p_{keyword} = 3.86e-10$
Mean (N=327)
Trade
$p_{keyword} = 5.08e-01$
Mean (N=1203)
$p_{keyword} = 3.93e-40$
Mean (N=283)
Jobs
$p_{keyword} = 3.91e-36$
Mean (N=705)
$p_{keyword} = 1.21e-01$
Mean (N=121)
Inflation
$p_{keyword} = 2.56e-47$
Mean (N=147)
$p_{keyword} = 1.80e-18$
Mean (N=102)
$p_{keyword} = 3.47e-37$
$p_{keyword} = 1.25e-33$
± 1 SD
Pre-Event Trend
Post-Event Trend
2.0
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−2.0

Interpretation of P-Values and Analysis Expansion

P-Value Interpretation

The statistical test currently performed is a two-sample t-test comparing the slope of the average price change before the event ($\text{slope}_{\text{pre}}$) versus after the event ($\text{slope}_{\text{post}}$).

The **Null Hypothesis (H_0)** for this test is that there is **no difference** between the two slopes: $\text{slope}_{\text{pre}} = \text{slope}_{\text{post}}$.

- **Volatility and Noise:** In volatile markets like Bitcoin, the price trajectory is inherently noisy and rarely linear, even when averaged. This high underlying **volatility** violates the test's underlying assumption of a stable, linear market.
- **False Significance:** Since the market fundamentally deviates from the test's linear assumption, the statistical test frequently finds even small, non-meaningful differences in slopes to be **highly significant** ($\text{p-value} \ll 0.05$). This is why the **control group p-values (p_{control})** are also small, indicating a statistical artifact of market noise rather than true explanatory power.

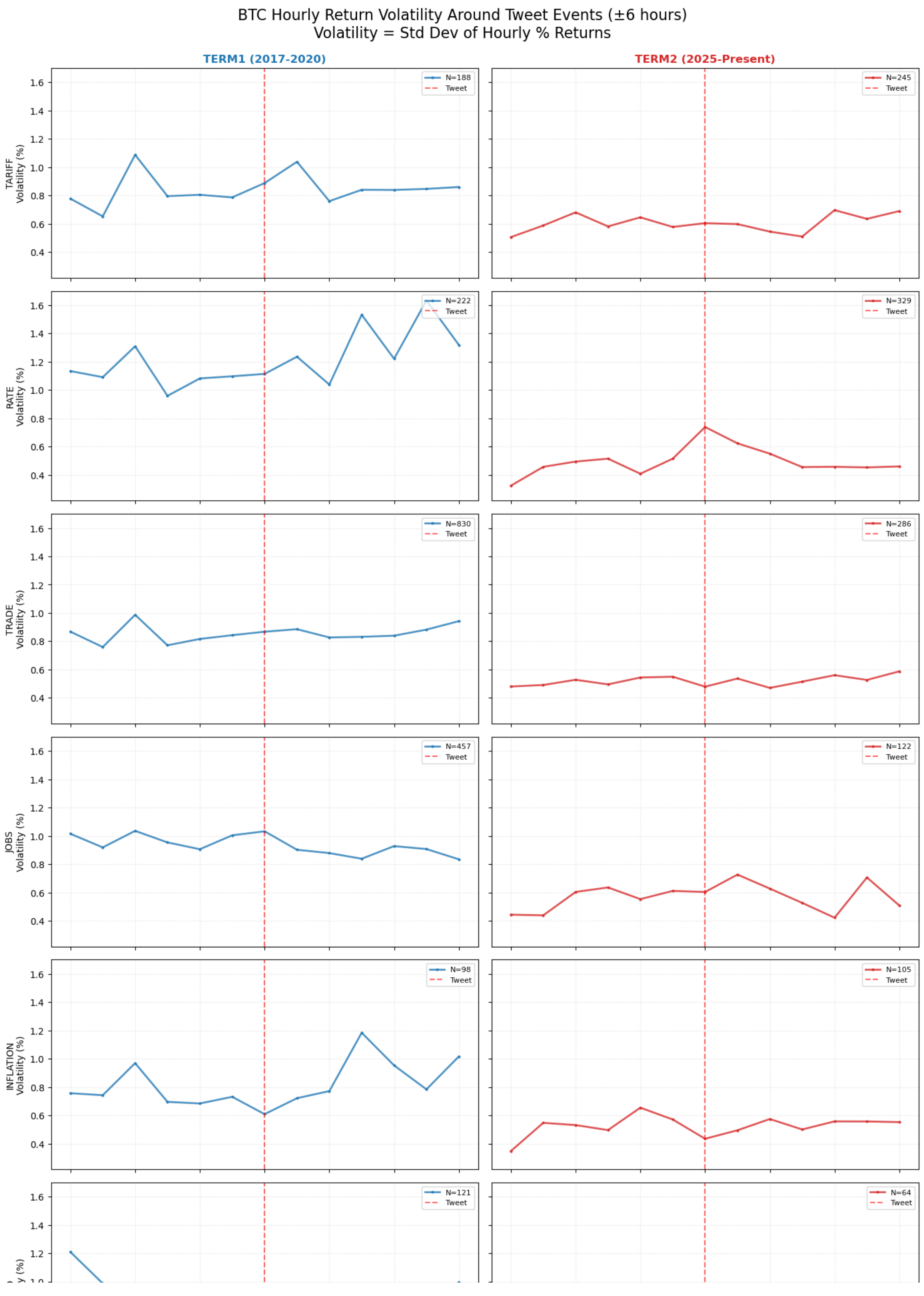
Improved Null Hypothesis

To establish a more meaningful statistical baseline, this research could be expanded by the following:

1. **Expanded Null Hypothesis:** Instead of comparing p_{keyword} to a static $\alpha = 0.05$ threshold, we would compare it to the **distribution of p-values** generated from hundreds or thousands of randomly generated control event sets.
2. **True Baseline:** This expansion would establish a **true empirical baseline** for "random market momentum shifts" (p_{random}), allowing us to determine if the keyword events are statistically significant **relative to typical market noise.**

However, generating, processing, and calculating the trend shift p-value for thousands of control datasets would be **too computationally expensive and time-consuming** to complete in a reasonable timeframe within this analysis environment, making the current single control group a compromising simplification.

```
run_btc_volatility_analysis()
```


BTC Hourly Return Volatility Around Tweet Events (±6 hours)
Volatility = Std Dev of Hourly % Returns
TERM1 (2017-2020)
TERM2 (2025-Present)
TARIFF Volatility (%)
RATE Volatility (%)
TRADE Volatility (%)
JOBS Volatility (%)
INFLATION Volatility (%)
N=188
N=245
N=222
N=329
N=830
N=286
N=457
N=122
N=98
N=105
N=121
N=64
Tweet
0.4
0.6
0.8
1.0
1.2
1.4
1.6

Bitcoin Volatility Analysis: Methodology

For each tweet event:

1. Round the tweet timestamp to the nearest hour (since BTC data is hourly)
2. Extract BTC prices from 12 hours before to 12 hours after the tweet
3. This creates a 25-point price series: hours [-12, -11, ..., -1, 0, 1, ..., 11, 12]

The result is a DataFrame where:

- Each row = one tweet event
- Each column = the BTC price at a specific hour relative to the tweet
- Example: Column `-5` contains the BTC price 5 hours before each tweet

Volatility Calculation

We calculate volatility as the **standard deviation of hourly percentage returns**:

1. **Hourly Returns**: For each event and each hour `t`, calculate:

```
  Return(t) = [(Price(t) - Price(t-1)) / Price(t-1)] × 100
```

This gives the percentage change from the previous hour.

1. **Cross-Event Volatility**: For each hour offset (e.g., hour 0, hour +3, etc.), calculate the standard deviation of returns across all events:

```
  Volatility(hour) = std_dev(Returns at that hour across all events)
```

Control Group

1. Select 100 random timestamps from the BTC dataset (avoiding edges)
2. Apply the same window extraction and volatility calculation
3. This shows typical volatility patterns during non-tweet periods

Analysis Results: No Significant Correlation Between Trump Tweets and Bitcoin Volatility

1. **No systematic volatility spikes at tweet time**: Visual inspection of keyword-term combinations shows relatively flat volatility patterns around hour 0 (the tweet moment). There are no unusual pattern immediately following tweets.

Crypto Market Cap Daily Data Analysis

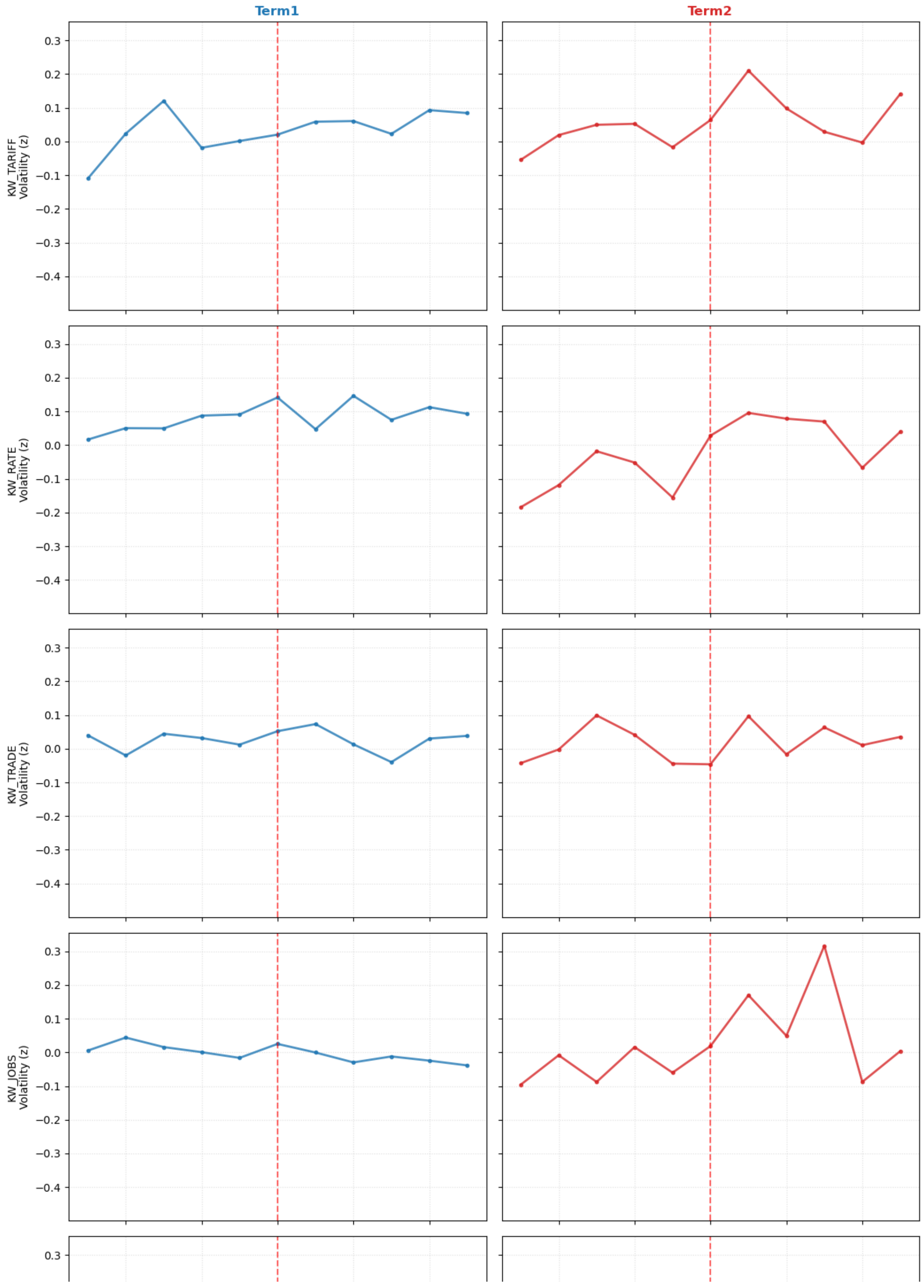
```
crypto_term1 = pd.read_csv('data/02-processed/crypto_term1.csv')
crypto_term2 = pd.read_csv('data/02-processed/crypto_term2.csv')
crypto_term1.head(3)
```

```
      snapped_at    market_cap  total_volume    datetime
0  1451606400000  7.124298e+09  1.505954e+09  2016-01-01
1  1451692800000  7.131191e+09  8.657891e+08  2016-01-02
2  1451779200000  7.080195e+09  1.076885e+09  2016-01-03
```

```
crypto_term2.head(3)
```

```
       snapped_at    market_cap  total_volume    datetime
0  1737331200000  3.621159e+12  3.592565e+11  2025-01-20
1  1737417600000  3.664661e+12  4.010621e+11  2025-01-21
2  1737504000000  3.792118e+12  2.651147e+11  2025-01-22
```

```
results = run_keyword_vol_analysis(
    "data/02-processed/crypto_term1.csv",
    "data/02-processed/crypto_term2.csv",
    posts_path="data/02-processed/posts_analyzed.csv",
    window=(-5,5)
)
```



Term1
Term2
KW_TARIFF Volatility (z)
KW_RATE Volatility (z)
KW_TRADE Volatility (z)
KW_JOBS Volatility (z)
0.3
0.2
0.1
0.0
−0.1
−0.2
−0.3
−0.4

Event-Window Analysis of Daily Crypto Volatility

We analyzed day-to-day crypto volatility for two datasets, **term1** and **term2**, using standardized daily absolute returns (`Close_Vol`) as a proxy for volatility. Each day's volatility was converted to a **z-score (`Close_Vol_z`)**, so the plotted values represent the number of standard deviations away from the mean volatility. This allows comparison across time and between datasets.

What Close_Vol means:
Close_Vol measures the magnitude of day-to-day changes in cryptocurrency value, computed as the absolute log-return of either price or market capitalization:

$$\text{Close\_Vol}_t = \left|\log\left(\text{Value}_t / \text{Value}_{t-1}\right)\right|$$

It captures how much the crypto value **swung in a single day**, regardless of direction. It does **not** measure trading volume or the amount sold, but instead reflects **price or market-cap volatility**.

```
results = run_keyword_vol_analysis(
    term1_path='data/02-processed/crypto_term1.csv',
    term2_path='data/02-processed/crypto_term2.csv',
    window=(-2,2),
    plot=False
)

significance = test_post_event_volatility(results, pre_window=(-2,-1),
post_window=(0,2))
for kw, stats in significance.items():
    print(f"Keyword: {kw}")
    for term, vals in stats.items():
        print(f"  {term.capitalize()}: t={vals['t_stat']:.2f},
p={vals['p_value']:.3f}")
```

```
Keyword: kw_tariff
  Term1: t=3.34, p=0.045
  Term2: t=1.89, p=0.156
Keyword: kw_rate
  Term1: t=0.70, p=0.558
  Term2: t=3.07, p=0.150
Keyword: kw_trade
  Term1: t=1.20, p=0.320
  Term2: t=0.21, p=0.849
Keyword: kw_jobs
  Term1: t=0.34, p=0.755
  Term2: t=1.70, p=0.190
Keyword: kw_inflation
  Term1: t=-0.41, p=0.727
  Term2: t=5.30, p=0.105
Keyword: kw_fed
  Term1: t=2.48, p=0.122
  Term2: t=5.10, p=0.015
Keyword: kw_market
```

```
Term1: t=0.12, p=0.915
Term2: t=2.40, p=0.105
```

Assessing Statistical Significance of Post-Event Volatility

We compared average z-scored volatility before and after keyword events, including the event day (`t=0`). The z-scores are standardized relative to the entire dataset, so they reflect how extreme a day's volatility is compared to normal fluctuations:

- Pre-event: t = -2 to -1
- Post-event: t = 0 to 2

Two-sample t-tests were used to assess whether post-event volatility was higher.

- **Null hypothesis (H_0):** There is no difference in average volatility between the pre-event and post-event periods (i.e., the posts have no effect on market volatility).
- **t-value:** measures the difference in means relative to variability
- **p-value:** probability of observing a difference as extreme as (or more extreme than) the one observed, assuming the null hypothesis is true

The t-statistic is calculated as:

$$t = \frac{\acute{X}_{\text{post}} - \acute{X}_{\text{pre}}}{\sqrt{\frac{s_{\text{post}}^2}{n_{\text{post}}} + \frac{s_{\text{pre}}^2}{n_{\text{pre}}}}}$$

Results:

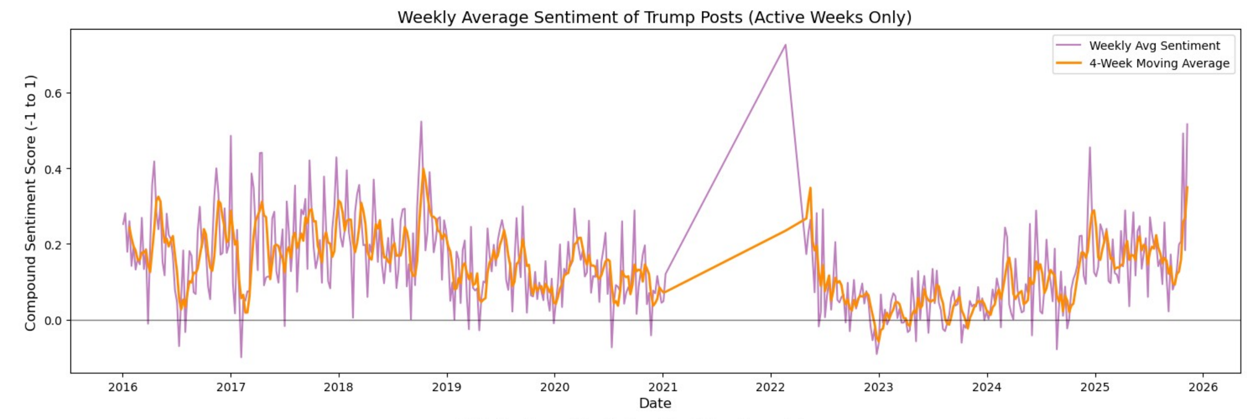
- Most keywords: p > 0.1, no significant rise
- kw_tariff (term1): t = 3.34, p = 0.045, marginally significant
- kw_fed (term2): t = 5.10, p = 0.015, significant
- Others: not significant

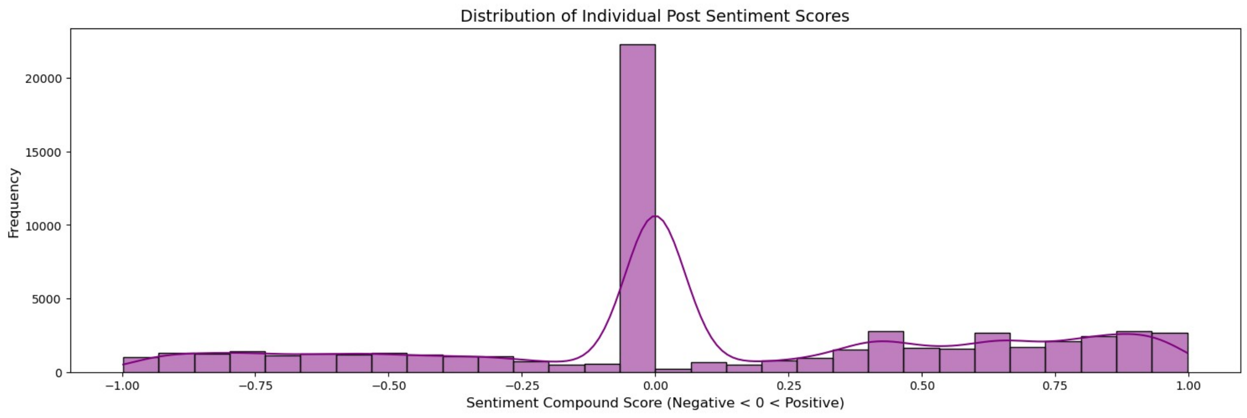
Including t=0 captures volatility the day of the post.

Analysis of Social media posts data

```
df = pd.read_csv('./data/02-processed/posts_analyzed.csv')

df['timestamp_et'] = pd.to_datetime(df['timestamp_et'], utc=True,
errors='coerce')

weekly_stats = df.set_index('timestamp_et').resample('W')
['sentiment_compound'].agg(['mean', 'count'])
```

```python
weekly_stats = weekly_stats[weekly_stats['count'] > 0]

fig, (ax1, ax2) = plt.subplots(2, 1, figsize=(15, 10))
ax1.plot(weekly_stats.index, weekly_stats['mean'], label='Weekly Avg
Sentiment', color='purple', alpha=0.5)

rolling_sentiment = weekly_stats['mean'].rolling(window=4).mean()
ax1.plot(rolling_sentiment.index, rolling_sentiment, label='4-Week
Moving Average', color='darkorange', linewidth=2)

ax1.set_title('Weekly Average Sentiment of Trump Posts (Active Weeks
Only)', fontsize=14)
ax1.set_xlabel('Date', fontsize=12)
ax1.set_ylabel('Compound Sentiment Score (-1 to 1)', fontsize=12)
ax1.axhline(0, color='black', linewidth=1, linestyle='-', alpha=0.5)
ax1.legend()

sns.histplot(df['sentiment_compound'], bins=30, ax=ax2, kde=True,
color='purple')
ax2.set_title('Distribution of Individual Post Sentiment Scores',
fontsize=14)
ax2.set_xlabel('Sentiment Compound Score (Negative < 0 < Positive)',
fontsize=12)
ax2.set_ylabel('Frequency', fontsize=12)

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





These graphs analyze the emotional tone of the social media posts using the VADER compound sentiment score.

Weekly Average Sentiment: This line chart tracks the average sentiment of posts by week. Weeks with zero activity have been removed to maintain a focus on active periods. A moving average is included to highlight becoming more negative or positive over time.

The weekly average sentiment fluctuates significantly, often hovering above the neutral line. This means theres a generally positive tone in his posts on average. There are some dips below zero, indicating periods of negative sentiment.

Distribution of Sentiment Scores: This histogram displays the overall spread of sentiment for individual posts. It reveals whether the posts are generally neutral, highly positive, highly negative, or skewed in one direction.

This histogram has a peak aronud neutral (0,0). This indicates that a large portion of the posts are classified as neutral likely short updates, links, or retweets without strong emotional keywords. The opinionated content tends to be strongly positive rather than strongly negative.

```
print("-" * 80)
print("EXAMPLES OF TWEETS BY SENTIMENT SCORE")
print("-" * 80)

idx_pos = df['sentiment_compound'].idxmax()
idx_neg = df['sentiment_compound'].idxmin()
```

```
# 1. Highly Positive Example
print(f"\n[+] HIGHLY POSITIVE (Score: {df.loc[idx_pos,
'sentiment_compound']}):")
print(f"\"{df.loc[idx_pos, 'description']}\"")

# 3. Highly Negative Example
print(f"\n[-] HIGHLY NEGATIVE (Score: {df.loc[idx_neg,
'sentiment_compound']}):")
print(f"\"{df.loc[idx_neg, 'description']}\"")
print("-" * 80)
```

```
----------------------------------------------------------------------
----------
EXAMPLES OF TWEETS BY SENTIMENT SCORE
----------------------------------------------------------------------
----------

[+] HIGHLY POSITIVE (Score: 0.9977):
"I had a truly great meeting with President Xi of China. There is
enormous respect between our two Countries, and that will only be
enhanced with what just took place. We agreed on many things, with
others, even of high importance, being very close to resolved. I was
extremely honored by the fact that President Xi authorized China to
begin the purchase of massive amounts of Soybeans, Sorghum, and other
Farm products. Our Farmers will be very happy! In fact, as I said once
before during my first Administration, Farmers should immediately go
out and buy more land and larger tractors. I would like to thank
President Xi for this! Additionally, China has agreed to continue the
flow of Rare Earth, Critical Minerals, Magnets, etc., openly and
freely. Very significantly, China has strongly stated that they will
work diligently with us to stop the flow of Fentanyl into our Country.
They will help us end the Fentanyl Crisis. China also agreed that they
will begin the process of purchasing American Energy. In fact, a very
large scale transaction may take place concerning the purchase of Oil
and Gas from the Great State of Alaska. Chris Wright, Doug Burgum, and
our respective Energy teams will be meeting to see if such an Energy
Deal can be worked out. The agreements reached today will deliver
Prosperity and Security to millions of Americans. After this Historic
trip to Asia, I am now heading back to Washington, D.C. I want to
thank the Great Countries of Malaysia, Japan, and South Korea for
being so generous, gracious, and hospitable – Also, Australia, Canada,
New Zealand, Singapore, Thailand, and Vietnam, who were at the Dinner
last night hosted by His Excellency Lee Jae Myung. Hundreds of
Billions of Dollars are being brought into our Country because of
them. Our Nation is Strong, Respected, and Admired Again and, THE BEST
IS YET TO COME!"

[-] HIGHLY NEGATIVE (Score: -0.9976):
"I have been briefed on the deadly shooting at the ICE Field Office in
Dallas, Texas. It has now been revealed the deranged shooter wrote
```

```
“Anti-ICE” on his shell casings. This is despicable! The Brave Men and
Women of ICE are just trying to do their jobs, and remove the “WORST
of the WORST” Criminals out of our Country, but they are facing an
unprecedented increase in threats, violence, and attacks by Deranged
Radical Leftists. This violence is the result of the Radical Left
Democrats constantly demonizing Law Enforcement, calling for ICE to be
demolished, and comparing ICE Officers to “Nazis.” The continuing
violence from Radical Left Terrorists, in the aftermath of Charlie
Kirk’s assassination, must be stopped. ICE Officers, and other Brave
Members of Law Enforcement, are under grave threat. We have already
declared ANTIFA a Terrorist Organization, and I will be signing an
Executive Order this week to dismantle these Domestic Terrorism
Networks. I AM CALLING ON ALL DEMOCRATS TO STOP THIS RHETORIC AGAINST
ICE AND AMERICA’S LAW ENFORCEMENT, RIGHT NOW! The Trump Administration
is fully committed to backing Law Enforcement, Strong Borders,
securing our Homeland, deporting Violent Illegal Criminals, and fully
rooting out the Left Wing Domestic Terrorism that is terrorizing our
Country. Thank you for your attention to this matter!"
----------------------------------------------------------------------
----------
```

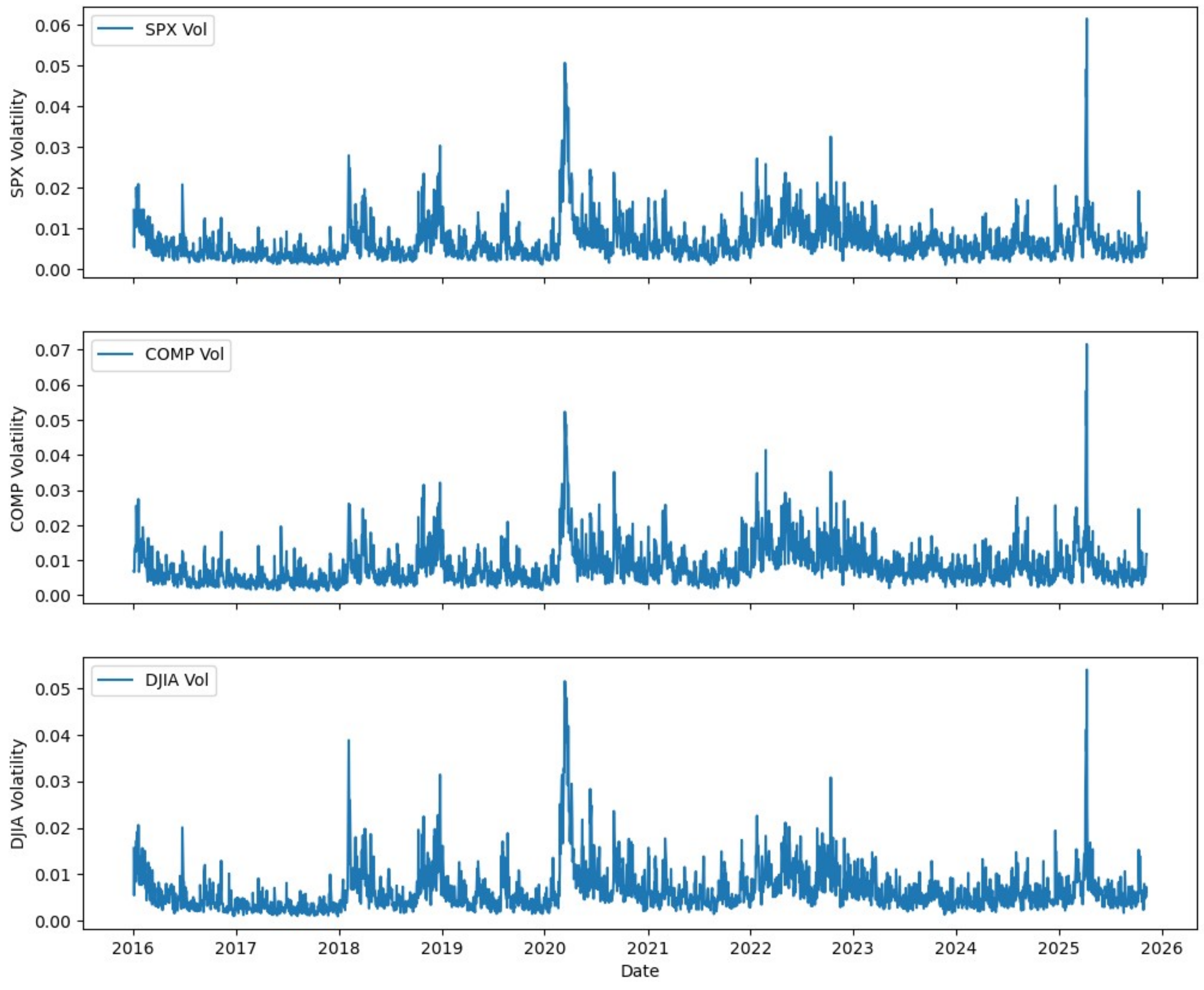
ETF Data Analysis

The quantitative data from the stock market itself needs to be altered to analyze effectively, including computing daily volatility of an ETF, standardizing it, and comparing volatility at certain dates where Trump's social media posts include keywords with dates where they don't or he hasn't posted at all as a base.

Volatility in the market is the value we are interested in analyzing, computed using the highs and lows of each day through Parkinson's Volatility. It measures 0.005 as low volatility, 0.01 as moderate volatility, 0.02 as high volatility, and anything above 0.03 as extreme, showing uncertainty in the market that we can then try to attribute to Trump's social media posts. We also standardized the volatility to show how each day's volatility compares to the average volatility in a market day.

```
df4['SPX_Vol'] = [Parkinson_Volatility(df4['SPX_High'][x],
df4['SPX_Low'][x]) for x in df4.index]
df4['COMP_Vol'] = [Parkinson_Volatility(df4['COMP_High'][x],
df4['COMP_Low'][x]) for x in df4.index]
df4['DJIA_Vol'] = [Parkinson_Volatility(df4['DJIA_High'][x],
df4['DJIA_Low'][x]) for x in df4.index]
standardize_P(df4, 'SPX_Vol')
standardize_P(df4, 'COMP_Vol')
standardize_P(df4, 'DJIA_Vol')

fig, axs = plt.subplots(3, 1, figsize=(12, 10), sharex=True)

axs[0].plot(df4['Date'], df4['SPX_Vol'], label='SPX Vol')
axs[0].set_ylabel('SPX Volatility')
```

```
axs[0].legend(loc='upper left')

axs[1].plot(df4['Date'], df4['COMP_Vol'], label='COMP Vol')
axs[1].set_ylabel('COMP Volatility')
axs[1].legend(loc='upper left')

axs[2].plot(df4['Date'], df4['DJIA_Vol'], label='DJIA Vol')
axs[2].set_xlabel('Date')
axs[2].set_ylabel('DJIA Volatility')
axs[2].legend(loc='upper left')

plt.legend()
plt.show()
```



To fully analyze the effect, if any, Trump's social media posts have on the market, we need to establish a baseline to compare to. Here we separate dates where Trump posts with our keywords of interest and either those without keywords or where there are no posts at all.

```
posts_with_keyword = posts[posts['contains_econ_keyword'] >
0].rename(columns={'date':'Date'})
dates_with_keyword = posts_with_keyword['Date'].unique()
num_post_key = posts_with_keyword['Date'].value_counts()

posts_without_keyword = posts[posts['contains_econ_keyword'] ==
0].rename(columns={'date':'Date'})
dates_without_keyword = posts_without_keyword['Date'].unique()
num_post_without = posts_without_keyword['Date'].value_counts()

#creates date range from all posts in dataset
start_date = date.fromisoformat('2016-01-04')
end_date = date.fromisoformat('2025-11-09')
date_range = [(end_date - timedelta(days=i)).strftime('%Y-%m-%d') for
i in range((end_date - start_date).days)]
dates_no_post = [date for date in date_range if date not in
dates_with_keyword and date not in dates_without_keyword]

no_posts = pd.DataFrame()
df4_strdate = df4.copy()
df4_strdate['Date'] = [d.strftime('%Y-%m-%d') for d in df4['Date']]
df4_dind = df4_strdate.set_index('Date')
no_posts['Date'] = [date for date in dates_no_post if date in
df4_strdate['Date'].to_list()]
no_posts['SPX_Vol_z'] = [df4_dind['SPX_Vol_z'][date] for date in
no_posts['Date']]
no_posts['COMP_Vol_z'] = [df4_dind['COMP_Vol_z'][date] for date in
no_posts['Date']]
no_posts['DJIA_Vol_z'] = [df4_dind['DJIA_Vol_z'][date] for date in
no_posts['Date']]
no_posts
```

```
           Date  SPX_Vol_z  COMP_Vol_z  DJIA_Vol_z
0    2024-11-06   0.113602    0.038729    1.238807
1    2022-07-28   1.186099    1.062864    0.997576
2    2022-06-03  -0.048789    0.276387   -0.246623
3    2022-04-27   0.862391    0.797176    0.782769
4    2022-04-26   1.545202    2.021179    1.093185
..          ...        ...         ...         ...
331  2017-10-06  -1.033221   -1.067977   -1.094201
332  2017-06-08  -0.748646   -0.762334   -0.601041
333  2016-11-25  -0.917355   -1.058072   -0.954241
334  2016-11-14  -0.469010   -0.348045   -0.570904
335  2016-10-07  -0.156875   -0.484575   -0.203838

[336 rows x 4 columns]
```

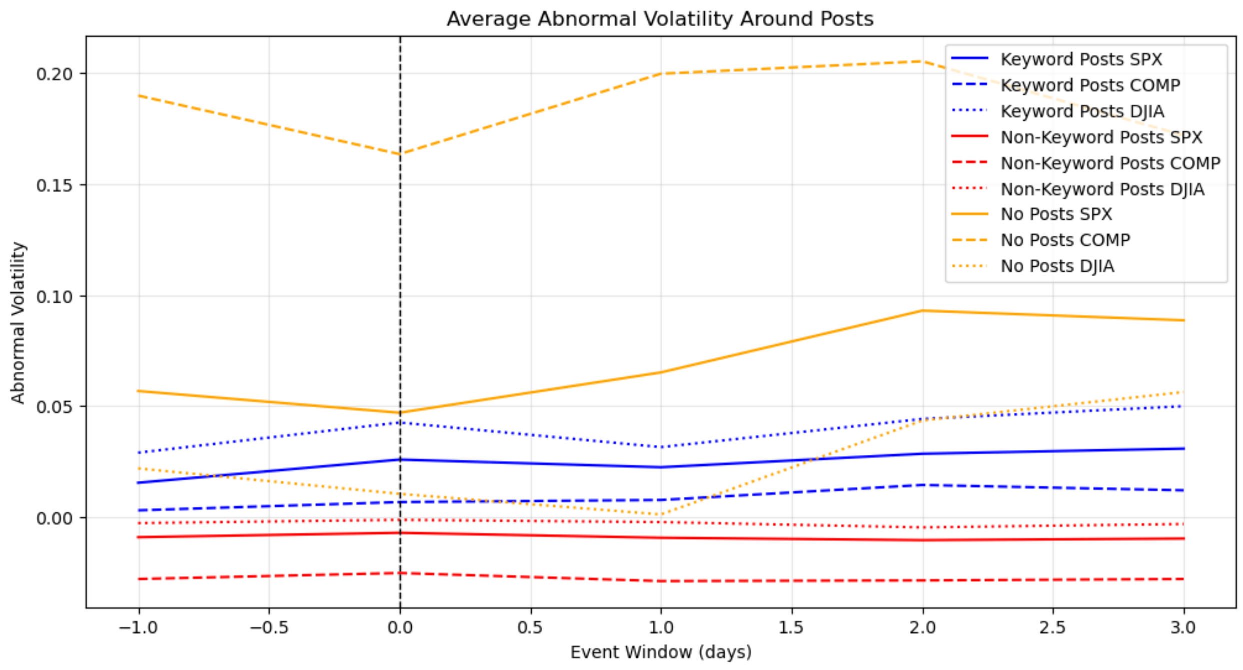
The volatility day by day doesn't matter much by itself, which is why we'll show it over a 4 day window to visualize volatility changes.

```python
event_window_key = event_windows(df4,
posts_with_keyword.drop_duplicates(subset=['Date']), 'SPX_Vol_z',
'COMP_Vol_z', 'DJIA_Vol_z')
average_key = average_event_windows(event_window_key, ['SPX_Vol_z',
'COMP_Vol_z', 'DJIA_Vol_z'])

event_window_nok = event_windows(df4,
posts_without_keyword.drop_duplicates(subset=['Date']), 'SPX_Vol_z',
'COMP_Vol_z', 'DJIA_Vol_z')
average_nok = average_event_windows(event_window_nok, ['SPX_Vol_z',
'COMP_Vol_z', 'DJIA_Vol_z'])

event_window_nop = event_windows(df4, no_posts, 'SPX_Vol_z',
'COMP_Vol_z', 'DJIA_Vol_z')
average_nop = average_event_windows(event_window_nop, ['SPX_Vol_z',
'COMP_Vol_z', 'DJIA_Vol_z'])

average_key
```

```
   t  SPX_Vol_z  COMP_Vol_z  DJIA_Vol_z
0 -1   0.015386    0.002955    0.028936
1  0   0.025823    0.006683    0.042522
2  1   0.022372    0.007624    0.031434
3  2   0.028448    0.014381    0.044129
4  3   0.030760    0.011975    0.049873
```

```python
plt.figure(figsize=(12, 6))
days = np.arange(-1,4)
plt.plot(days, average_key['SPX_Vol_z'], label="Keyword Posts SPX",
color='b')
plt.plot(days, average_key['COMP_Vol_z'], label="Keyword Posts COMP",
color='b', linestyle='--')
plt.plot(days, average_key['DJIA_Vol_z'], label="Keyword Posts DJIA",
color='b', linestyle=':')
plt.plot(days, average_nok['SPX_Vol_z'], label="Non-Keyword Posts
SPX", color='r')
plt.plot(days, average_nok['COMP_Vol_z'], label="Non-Keyword Posts
COMP", color='r', linestyle='--')
plt.plot(days, average_nok['DJIA_Vol_z'], label="Non-Keyword Posts
DJIA", color='r', linestyle=':')
plt.plot(days, average_nop['SPX_Vol_z'], label="No Posts SPX",
color='orange')
plt.plot(days, average_nop['COMP_Vol_z'], label="No Posts COMP",
color='orange', linestyle='--')
plt.plot(days, average_nop['DJIA_Vol_z'], label="No Posts DJIA",
color='orange', linestyle=':')

plt.axvline(0, color="black", linestyle="--", linewidth=1)  # event
day marker
```

```
plt.title("Average Abnormal Volatility Around Posts")
plt.xlabel("Event Window (days)")
plt.ylabel("Abnormal Volatility")
plt.legend()
plt.grid(True, alpha=0.3)
plt.show()
```



Generally, the average abnormal volatility seems to spike on the event days (day of a post) when the post is a keyword post, with there being almost no change or even dips in volatility otherwise. This is indicative that the keyword posts do have some effect on market volatility on event days, even increasing volatility a couple of days after in comparison to the day prior to the event day. However, when there are no posts or a post without keywords, the volatility is difficult to predict, with no posts having random dips and spikes while posts without keywords seem to have very little change in volatility over the window period.

Despite the appearance of an effect of Trump's social media posts upon the volatility of these three stock indices, the average change in volatility is not significant enough to conclude there is a causal relationship between the two. This can be seen when comparing post days to non-post days, where the average volatility change is greater than days when Trump post, indicating that the volatility changes on post days are from normal market changes. As such, we it is inconclusive as to whether or not Trump's posts affect the market to a significant degree under the constraints of our data and lack of context surrounding such a large period of time.

Interpretation of P-Values and Analysis Expansion

P-Value Interpretation

The statistical test currently performed is a two-sample t-test comparing the slope of the average price change before the event ($\text{slope}_{\text{pre}}$) versus after the event ($\text{slope}_{\text{post}}$).

The **Null Hypothesis (H_0)** for this test is that there is **no difference** between the two slopes: $\text{slope}_{\text{pre}} = \text{slope}_{\text{post}}$.

- **Volatility and Noise:** In volatile markets like Bitcoin, the price trajectory is inherently noisy and rarely linear, even when averaged. This high underlying **volatility** violates the test's underlying assumption of a stable, linear market.
- **False Significance:** Since the market fundamentally deviates from the test's linear assumption, the statistical test frequently finds even small, non-meaningful differences in slopes to be **highly significant** ($\text{p-value} \ll 0.05$). This is why the **control group p-values (p_{control})** are also small, indicating a statistical artifact of market noise rather than true explanatory power.

Improved Null Hypothesis

To establish a more meaningful statistical baseline, this research could be expanded by the following:

1. **Expanded Null Hypothesis:** Instead of comparing p_{keyword} to a static $\alpha = 0.05$ threshold, we would compare it to the **distribution of p-values** generated from hundreds or thousands of randomly generated control event sets.
2. **True Baseline:** This expansion would establish a **true empirical baseline** for "random market momentum shifts" (p_{random}), allowing us to determine if the keyword events are statistically significant **relative to typical market noise.**

However, generating, processing, and calculating the trend shift p-value for thousands of control datasets would be **too computationally expensive and time-consuming** to complete in a reasonable timeframe within this analysis environment, making the current single control group a compromising simplification.

Ethics

A. Data Collection

- ☒ **A.1 Informed consent**: If there are human subjects, have they given informed consent, where subjects affirmatively opt-in and have a clear understanding of the data uses to which they consent?

  Example of how to use the checkbox, and also of how you can put in a short paragraph that discusses the way this checklist item affects your project. Remove this paragraph and the X in the checkbox before you fill this out for your project

A. Data Collection

- ☒ **A.1 Informed consent**: If there are human subjects, have they given informed consent, where subjects affirmatively opt-in and have a clear understanding of the data uses to which they consent?

  Example of how to use the checkbox, and also of how you can put in a short paragraph that discusses the way this checklist item affects your project. Remove this paragraph and the X in the checkbox before you fill this out for your project

- ☒ **A.3 Limit PII exposure**: Have we considered ways to minimize exposure of personally identifiable information (PII) for example through anonymization or not collecting information that isn't relevant for analysis?
- ☒ **A.4 Downstream bias mitigation**: Have we considered ways to enable testing downstream results for biased outcomes (e.g., collecting data on protected group status like race or gender)?

B. Data Storage

- ☒ **B.1 Data security**: Do we have a plan to protect and secure data (e.g., encryption at rest and in transit, access controls on internal users and third parties, access logs, and up-to-date software)?

  Yes, most of our data will be publicly available and accessible, so there is no need to worry about that.

- ☒ **B.2 Right to be forgotten**: Do we have a mechanism through which an individual can request their personal information be removed?

  Yes, any individual can request their personal information to be removed by contancting our support email: kemata@ucsd.edu

- ☒ **B.3 Data retention plan**: Is there a schedule or plan to delete the data after it is no longer needed?

  Not Applicable as it is all publicly available data, and we are not collecting our own.

C. Analysis

- ☒ **C.1 Missing perspectives**: Have we sought to address blindspots in the analysis through engagement with relevant stakeholders (e.g., checking assumptions and discussing implications with affected communities and subject matter experts)?
- ☒ **C.2 Dataset bias**: Have we examined the data for possible sources of bias and taken steps to mitigate or address these biases (e.g., stereotype perpetuation, confirmation bias, imbalanced classes, or omitted confounding variables)?

  Quantitative data is used, though it is difficult to not have biased analysis of tweets since sentiment analysis of text is inherently biased. We can try to find an algorithm that uses dictionary definitions, but cultural significance to certain phrases could be missed in doing so.

- ☒ **C.3 Honest representation**: Are our visualizations, summary statistics, and reports designed to honestly represent the underlying data?
- ☒ **C.4 Privacy in analysis**: Have we ensured that data with PII are not used or displayed unless necessary for the analysis?

  The only real data with PII are essentially the name of Trump and perhaps other things included in his tweets, which are publically available. No PII aside from his name will be shared though, as sentiment analysis will be performed on the tweets, so no other identifiable information should be included in the project.

D. Modeling

- ☒ **D.1 Proxy discrimination**: Have we ensured that the model does not rely on variables or proxies for variables that are unfairly discriminatory?

  Data from markets is entirely quantitative

- ☒ **D.2 Fairness across groups**: Have we tested model results for fairness with respect to different affected groups (e.g., tested for disparate error rates)?
- ☒ **D.3 Metric selection**: Have we considered the effects of optimizing for our defined metrics and considered additional metrics?
- ☒ **D.4 Explainability**: Can we explain in understandable terms a decision the model made in cases where a justification is needed?

  Clear visualizations and analysis should be easily understood by the common person

- ☒ **D.5 Communicate limitations**: Have we communicated the shortcomings, limitations, and biases of the model to relevant stakeholders in ways that can be generally understood?

  This will be discussed in the ethics portion of our final project.

E. Deployment

- ☒ **E.1 Monitoring and evaluation**: Do we have a clear plan to monitor the model and its impacts after it is deployed (e.g., performance monitoring, regular audit of sample predictions, human review of high-stakes decisions, reviewing downstream impacts of errors or low-confidence decisions, testing for concept drift)?
- ☒ **E.2 Redress**: Have we discussed with our organization a plan for response if users are harmed by the results (e.g., how does the data science team evaluate these cases and update analysis and models to prevent future harm)?
- ☒ **E.3 Roll back**: Is there a way to turn off or roll back the model in production if necessary?
- ☒ **E.4 Unintended use**: Have we taken steps to identify and prevent unintended uses and abuse of the model and do we have a plan to monitor these once the model is deployed?

  We will include a warning that there are a variety of factors that also affect the market, and that the correlation we are exploring is not at all representative as to why the market moves and shouldn't be used to formulate decisions around the purchase of stocks.

Discussion and Conclusion

Our key results show that the relationship between Trump's social media posts and financial market behavior is *inconclusive*. Across our event-window style comparisons (aligning market data around post timestamps) and our sentiment + keyword filtering pipeline, we do not see a stable, repeatable pattern where volatility reliably increases (or decreases) after posts. There are occasional apparent spikes in certain windows or subsets, but those effects are inconsistent across time periods and do not persist when we compare against reasonable baselines (such as other days or other post categories), which suggests the signal, if present at all, is weak relative to normal market noise. Essentially, there is no definitive correlation or patterns between the social media posts and the financial market.

Referring back to our Background and prior work, our project extends earlier “Twitter/Trump effect” framing by focusing specifically on economy-related content. In theory, if political communication were strongly shaping market uncertainty, we would expect clearer post-event structure (for example, consistent volatility changes sometime after a post) and a more clear separation between economy-related posts and other content. In contrast, our results indicate that any influence is difficult to isolate in the presence of economic news, scheduled announcements, broader media cycles, and platform/timing effects (for example, posting time versus market trading hours).

Putting everything together, we found that it’s hard to clearly detect a “ tweet causes market change ” effect with our current approach. Markets are noisy, and volatility moves for many reasons. Therefore, keyword and sentiment methods can’t effectively isolate the impact of Trump’s posts very well. Overall, we can’t confidently claim a strong correlation or causal link between economy-related posts and the financial data we analyzed. A stronger test would need better controls (like placebo events and major-news filtering) and more detailed data with statistical testing. Unfortunately, we don't exactly have the technical capacity to have tested ***very*** large quantities of data and dataframes, but the probability is that our results would likely remain inconclusive.

Our results were somewhat confusing since they contradict our initial hypothesis that social media post containing specific keywords would correlate with spikes in market volatility, specifically in crypto. Our data shows that during the time frame of the tweet being published, the market seems to have typical behavior, contradicting our hypothesis. We expected fear and greed to drive the market after these tweets but our sentiment portion of the flagged post were mostly neutral. Meaning with our keywords, the post were not intented to influence the market in any way.

Our findings align with prior research examining Trump's social media influence on financial. Wolff's 2019 analysis of Trump's tweets during his first presidential term similarly concluded that while some anomalies appeared in the data, the overall market response was not statistically significant across all sectors.[1]

Our project had many limitations that constrain the generalizability of our findings. We had a gap in our tweet data due to the suspension of Trumps twitter account being banned between January 2021 and November 2022. This gap disrupted the time-series continuity, making it hard to preform a comparison between the first and second term and seeing how his influence has changed over time.

Another limitation is that we relied on HOURLY bitcoin data. Since the crypto market is open 24/7 there can be variabled that influence the market within minuetes or even seconds. Since we used by hour, we may have smoothed out certain important volatility spikes.

If this investigation were to continue, the next step would be to get higher frequency price data, like minute by minute instead of hourly. Since crypto markets react to news pretty much instantly, using higher frequency price data will potentially show volatility spikes we are currently missing.

As mentioned in the BTC hourly data analysis, statistical significance could be determined by comparing our observed data against empirically sampled baseline data from time ranges when no economy-related posts occurred. However, implementing such statistical testing would require processing substantially larger datasets than our current computational resources can support.