In [1]:

```
#imports
import pandas as pd
import numpy as np
import datetime as dt
%matplotlib inline
import panel as pn
pn.extension('plotly')
import plotly.express as px
import hvplot.pandas
import matplotlib.pyplot as plt
# import os
from pathlib import Path
# from dotenv import load_dotenv
```

In [2]:

```
# Selected Stock Data
```

In [3]:

```
# Reading nasdaq returns
stock_price_csv = Path("Data/StockPriceData.csv")
nasdaq_stock_price = pd.read_csv(stock_price_csv, index_col=["Date"], parse_dates=Tr
# Clean CSV data
nasdaq_stock_price.drop("Unnamed: 0", axis=1, inplace=True)
#check Data
nasdaq_stock_price
```

Out[3]:

	Close	Ticker
Date		
2022-05-10 08:00:00+00:00	1.3000	AGRI
2022-05-10 12:00:00+00:00	1.4450	AGRI
2022-05-10 16:00:00+00:00	1.5200	AGRI
2022-05-10 20:00:00+00:00	1.5700	AGRI
2022-05-11 12:00:00+00:00	1.5150	AGRI
...	...	...
2022-05-13 16:00:00+00:00	25.1900	XOMAO
2022-05-16 12:00:00+00:00	25.3185	XOMAO
2022-05-16 16:00:00+00:00	25.0000	XOMAO
2022-05-17 12:00:00+00:00	25.7400	XOMAO
2022-05-17 16:00:00+00:00	25.0500	XOMAO

450 rows × 2 columns

In [4]:

```
# nasdaq plot to show returns
nasdaq_stock_price.hvplot.line(x='Date', y='Close', rot=90, width=800, groupby='Tick
```

Out[4]:

In [28]:

```
# Daily Standard Deviations

#Calculate Standard Deviation of Percentage Change per Ticker
df_daily_std_nasdaq_stock_price = nasdaq_stock_price
df_daily_std_nasdaq_stock_price['Close'].pct_change()
df_daily_std_nasdaq_stock_price = df_daily_std_nasdaq_stock_price.groupby(by = "Tick
df_daily_std_nasdaq_stock_price.columns = ["Standard_Deviation"]
#remove NaNs
df_daily_std_nasdaq_stock_price.dropna(inplace = True)
# Add Comparison to Market
df_daily_std_nasdaq_stock_price.loc['00_Market Mean'] = df_daily_std_nasdaq_stock_pr
# Check Output
df_daily_std_nasdaq_stock_price.sort_values(by = 'Ticker', inplace = True)
df_daily_std_nasdaq_stock_price.reset_index(inplace = True)

df_daily_std_nasdaq_stock_price
```

In [33]:

```
plot_df_daily_std_nasdaq_stock_price = df_daily_std_nasdaq_stock_price.hvplot.bar(
    x='Ticker',
    y='Standard_Deviation',
    rot=90,
    width=800,
    height = 300,
    title = "Standard Deviation",
    color = 'pink')

plot_df_daily_std_nasdaq_stock_price
```

Out[33]:

In [6]:

```
#stock Close Price change
```

In [7]:

```
stock_change_csv = Path("Data/StockPriceChange.csv")
nasdaq_stock_change = pd.read_csv(stock_change_csv, index_col=["Unnamed: 0"])
#check data output
nasdaq_stock_change.head()
```

Out[7]:

```
   Ticker  Start Price  End Price  Price Change  Price Change %
0    SRGA      0.17645       5.25       5.07355     2875.347124
1     PTE      0.16000       2.96       2.80000     1750.000000
2    TNXP      0.13620       2.30       2.16380     1588.693098
3    RMTI      0.27930       1.97       1.69070      605.334765
4     PXS      0.62610       2.76       2.13390      340.824149
```

In [8]:

```
# nasdaq plot to show returns

# set up ticker list to slice from
ticker_list = list(set(nasdaq_stock_price['Ticker']))
```

```
plot_nasdaq_stock_change = nasdaq_stock_change[nasdaq_stock_change['Ticker'].isin(ti
    x='Ticker',
    y='Price Change %',
    rot=90,
    width=800,
    height = 300,
    title = "Market Representation for Analysis",
    color = 'orange')

plot_nasdaq_stock_change
```

Out[8]:

In [9]:

```
#sentiment_analysis
```

In [10]:

```
# Reading nasdaq sentiment
stock_sentiment_csv = Path("Data\stock_tweet_sentiment.csv")
nasdaq_stock_sentiment = pd.read_csv(stock_sentiment_csv, index_col=["Date"], parse_
# drop invalid column
nasdaq_stock_sentiment.drop("Unnamed: 0", axis=1, inplace=True)
nasdaq_stock_sentiment.head()
```

Out[10]:

	Ticker	Sentiment_Score
Date		
2022-05-11 12:00:00+00:00	AGRI	0.000000
2022-05-11 16:00:00+00:00	AGRI	0.100719
2022-05-11 20:00:00+00:00	AGRI	0.103330
2022-05-12 08:00:00+00:00	AGRI	0.063434
2022-05-12 12:00:00+00:00	AGRI	0.008716

In [11]:

```
# Plot to show Sentiments
nasdaq_stock_sentiment.hvplot.line(x='Date', y='Sentiment_Score',rot=90, width=800,
```

Out[11]:

In [12]:

```
#Cross Analysis
```

In [13]:

```
#Combine
```

In [14]:

```
nasdaq_stock_price
```

Out[14]:

	Close	Ticker
Date		
2022-05-10 08:00:00+00:00	1.3000	AGRI
2022-05-10 12:00:00+00:00	1.4450	AGRI

	Close	Ticker
Date		
2022-05-10 16:00:00+00:00	1.5200	AGRI
2022-05-10 20:00:00+00:00	1.5700	AGRI
2022-05-11 12:00:00+00:00	1.5150	AGRI
...	...	...
2022-05-13 16:00:00+00:00	25.1900	XOMAO
2022-05-16 12:00:00+00:00	25.3185	XOMAO
2022-05-16 16:00:00+00:00	25.0000	XOMAO
2022-05-17 12:00:00+00:00	25.7400	XOMAO
2022-05-17 16:00:00+00:00	25.0500	XOMAO

450 rows × 2 columns

In [15]:

```
# Join the sentiment Score and Ticker Pricing for cross analysis

# Set up unique Index
# Reset index
nasdaq_stock_price.reset_index(inplace = True)
nasdaq_stock_sentiment.reset_index(inplace = True)

# Set up Referance to be able to match the data sets
nasdaq_stock_price['Ref'] = nasdaq_stock_price['Ticker'].astype(str) + nasdaq_stock_
nasdaq_stock_sentiment['Ref'] = nasdaq_stock_sentiment['Ticker'].astype(str) + nasda

# set Ref as new index
nasdaq_stock_price.set_index('Ref', inplace = True)
nasdaq_stock_sentiment.set_index('Ref', inplace = True)

# concatinate Stock price to Sentiment score
cross_analysis = pd.concat([nasdaq_stock_price, nasdaq_stock_sentiment],
                            join = 'outer',
                            axis = 'columns')

cross_analysis.reset_index(inplace = True)
#uniquily identify each coloum, and mark columns for deletion
cross_analysis.columns = ['Ref-Del','Date','Close','Ticker','Date-Del','Ticker-Del',
cross_analysis.drop(labels = ['Ref-Del','Date-Del','Ticker-Del'] ,axis = 'columns',
```

In [16]:

```
# nasdaq plot to show Sentiment vs Stock Price

cross_analysis['Sentiment_Score'] = cross_analysis['Sentiment_Score']*1000

cross_analysis.hvplot.scatter(x = 'Date',
                              y = 'Close',
                              c = 'Sentiment_Score',
                              size = 'Sentiment_Score',
                              rot = 90,
                              width = 800,
                              groupby = 'Ticker',
                              widget_location = 'left_top',
                              title = "Plot to Show Stock Price Against Sentiment Sc
```

Out[16]:

In [17]:

```
# Correlation
cross_analysis_corr = cross_analysis.groupby('Ticker')[['Close','Sentiment_Score']].

cross_analysis_corr.reset_index(inplace = True)

cross_analysis_corr = cross_analysis_corr[cross_analysis_corr['level_1'].isin(['Clos

cross_analysis_corr.drop(labels = ['level_1', 'Close'], inplace = True, axis = 'colu

cross_analysis_corr.columns = ['Ticker', 'Correlation']

cross_analysis_corr.dropna(inplace = True)

cross_analysis_corr.sort_values(by = 'Correlation',inplace = True, ascending = False
```

In [18]:

```
plot_cross_analysis_corr = cross_analysis_corr.hvplot.bar(x='Ticker', y='Correlation
```

In [19]:

```
plot_cross_analysis_corr
```

Out[19]:

In [20]:

```
cross_analysis_corr['Correlation'].mean()
```

Out[20]:

```
-0.008522832771524406
```

In [21]:

```
# Plot Market Representation next to

plot = (plot_nasdaq_stock_change + plot_cross_analysis_corr)
plot
```

Out[21]:

In [ ]: