```
#pandas and NumPy imports
import pandas as pd
from pandas import Series,DataFrame
import numpy as np

# For Visualization
import matplotlib.pyplot as plt
import seaborn as sns
sns.set_style('whitegrid')
%matplotlib inline



BATA = pd.read_csv('/content/Bata.csv')
print(BATA.head())
```

```
                       Data   Adj Close       Close        High         Low  \
0  2014-01-01 00:00:00+00:00  493.201904  533.825012  542.000000  525.625000
1  2014-01-02 00:00:00+00:00  475.948090  515.150024  539.700012  510.549988
2  2014-01-03 00:00:00+00:00  480.867767  520.474976  522.250000  510.000000
3  2014-01-06 00:00:00+00:00  474.100281  513.150024  522.474976  511.450012
4  2014-01-07 00:00:00+00:00  473.291779  512.275024  518.900024  506.174988

          Open  Volume
0  528.000000  424144
1  537.474976  774618
2  510.000000  180640
3  520.500000  145714
4  513.150024  148746
```

```
INFY= pd.read_csv('/content/INFY.csv')
print(INFY.head())
```

```
         Date  Adj Close    Close     High      Low     Open   Volume
0  2014-01-02   5.219006  6.94125  7.03125  6.92625  7.03125  3642400
1  2014-01-03   5.371264  7.14375  7.19750  7.11125  7.14375  8421600
2  2014-01-06   5.292315  7.03875  7.11375  7.02125  7.11125  4820000
3  2014-01-07   5.271638  7.01125  7.04875  6.95625  6.97625  6201600
4  2014-01-08   5.240623  6.97000  6.97000  6.90500  6.93750  9640000
```

```
TAMO= pd.read_csv('/content/TATAMOTORS.NS.csv')
print(TAMO.head())
```

```
         Date   Adj Close       Close        High         Low        Open  \
0  2014-01-01  366.963165  370.970978  374.780121  370.080505  373.543365
1  2014-01-02  364.418579  368.398560  372.207703  365.281982  370.970978
2  2014-01-03  354.974121  358.850952  364.094727  356.179626  364.094727
3  2014-01-06  359.133575  363.055847  366.568176  360.335053  363.600037
4  2014-01-07  357.322998  361.225494  366.073486  359.889832  363.105347

    Volume
0  1364747
1  3015089
2  5004049
3  5490077
4  4020899
```

```
TTML= pd.read_csv('/content/TTML.csv')
print(TTML.head())
```

```
                       Date  Adj Close     Close      High       Low  \
0  2014-01-01 00:00:00+00:00   7.511293  7.511293  7.559135  7.080709
1  2014-01-02 00:00:00+00:00   7.319922  7.319922  7.941876  7.224237
2  2014-01-03 00:00:00+00:00   7.989719  7.989719  8.420303  7.606978
3  2014-01-06 00:00:00+00:00   7.654821  7.654821  8.085404  7.463450
4  2014-01-07 00:00:00+00:00   7.559135  7.559135  7.798349  7.511293

       Open    Volume
0  7.224237   4210915
1  7.559135   4508978
2  7.846191  14955567
3  8.037562   3995497
4  7.750506   2825451
```

```
# Summary Stats for BATA stocks
BATA.describe()
```

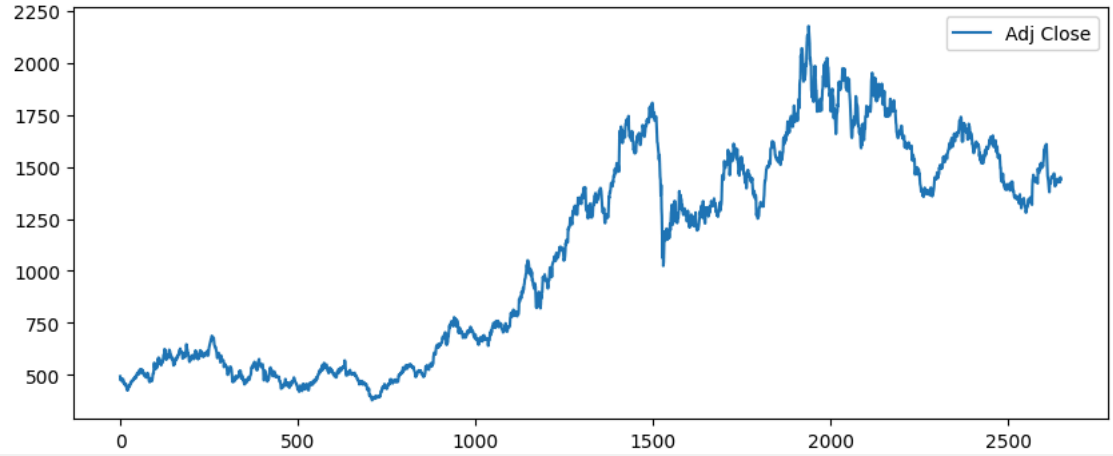
	Adj Close	Close	High	Low	Open	Volume
count	2649.000000	2649.000000	2649.000000	2649.000000	2649.000000	2.649000e+03
mean	1093.973713	1133.256851	1148.773347	1118.628540	1134.327624	5.646764e+05
std	506.415936	508.793239	514.994875	502.843340	509.289649	4.928113e+05
min	378.681732	402.549988	411.000000	399.200012	403.799988	0.000000e+00
25%	548.515747	585.000000	595.500000	576.849976	587.099976	2.749050e+05
50%	1239.916748	1286.550049	1310.000000	1265.599976	1284.699951	4.356020e+05
75%	1548.845703	1584.150024	1602.400024	1569.000000	1586.900024	6.925690e+05
max	2175.689941	2229.600098	2262.000000	2183.850098	2238.500000	8.481663e+06

```
# General Info about BATA Stock
BATA.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2649 entries, 0 to 2648
Data columns (total 7 columns):
 #   Column     Non-Null Count  Dtype  
---  ------     --------------  -----  
 0   Data       2649 non-null   object 
 1   Adj Close  2649 non-null   float64
 2   Close      2649 non-null   float64
 3   High       2649 non-null   float64
 4   Low        2649 non-null   float64
 5   Open       2649 non-null   float64
 6   Volume     2649 non-null   int64  
dtypes: float64(5), int64(1), object(1)
memory usage: 145.0+ KB
```

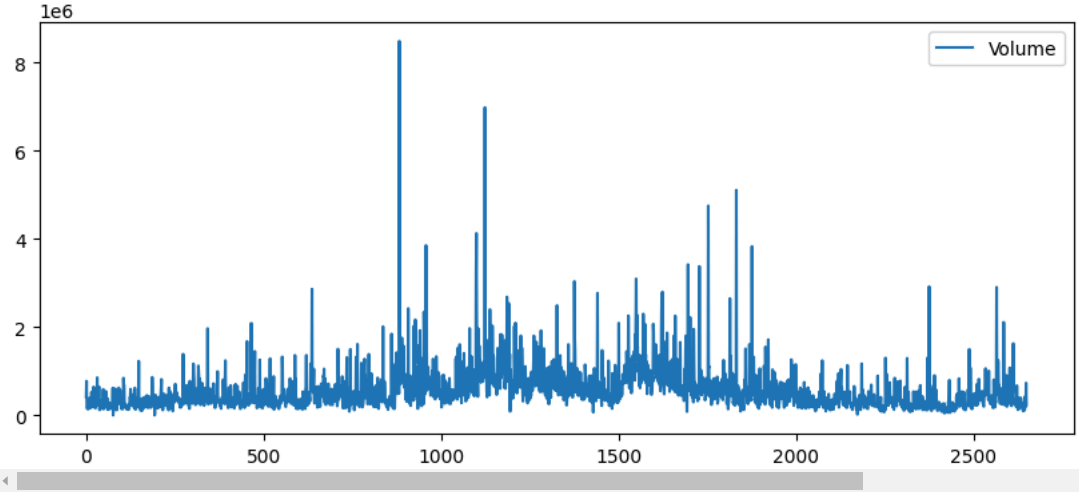
```
# Historical view of the closing price of BATA stock
BATA['Adj Close'].plot(legend=True,figsize=(10,4))
```

```
<Axes: >
```



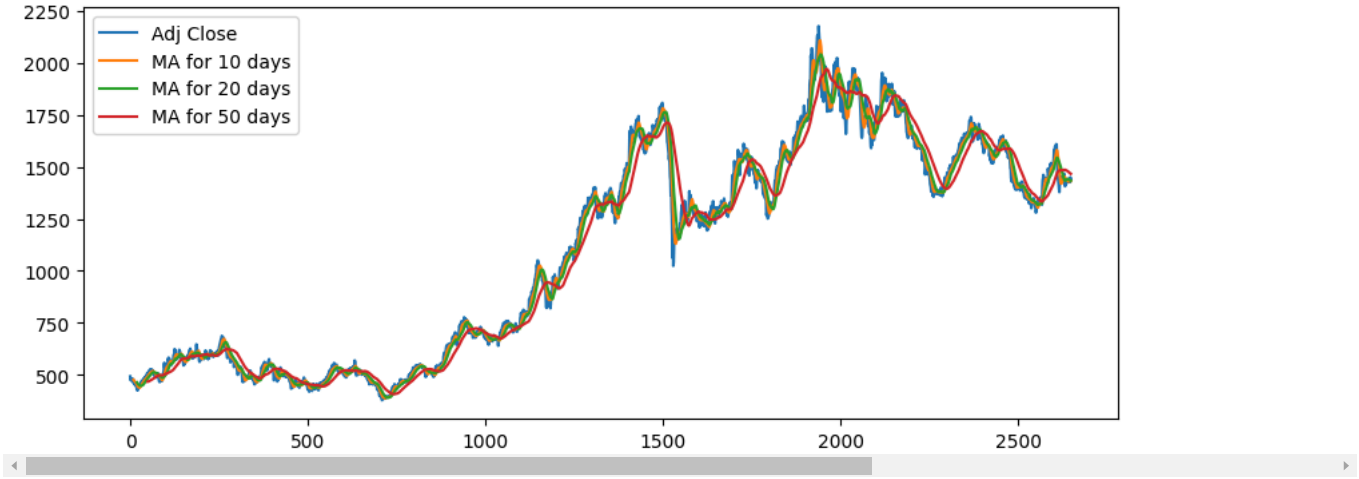

```
# Historical view of the total volume of BATA stock traded each day
BATA['Volume'].plot(legend=True,figsize=(10,4))
```

```
<Axes: >
```



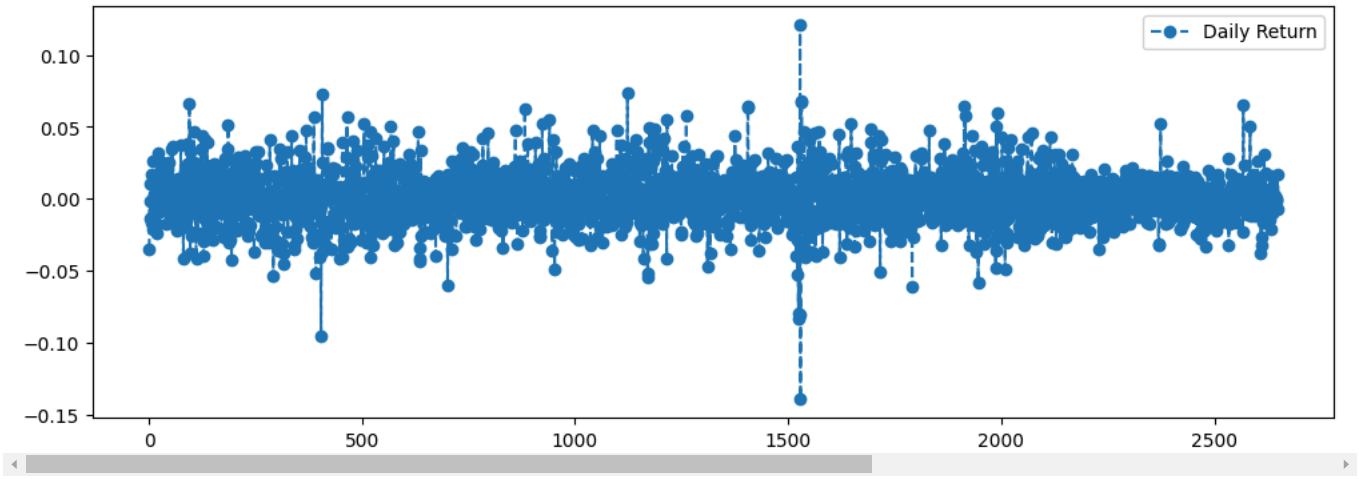
```
import pandas as pd

ma_day = [10,20,50]

for ma in ma_day:
    column_name = "MA for %s days" %(str(ma))
    # Use the rolling method instead of the deprecated rolling_mean function.
    # We are calculating the mean of the 'Adj Close' column over a window of 'ma' days.
    BATA[column_name] = BATA['Adj Close'].rolling(window=ma,center=False).mean()


# Hostorical view of the moving averages of Closing Price of BATA Stock
BATA[['Adj Close','MA for 10 days','MA for 20 days','MA for 50 days']].plot(subplots=False,figsize=(10,4))
```

```
<Axes: >
```



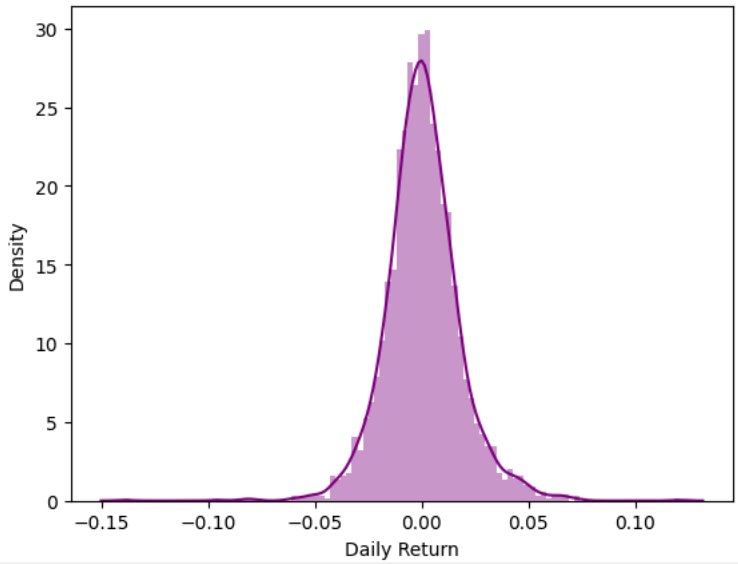

```
#Calculation to find he percent change for each day of BATA stock
BATA['Daily Return'] = BATA['Adj Close'].pct_change()


# Visualization of the percent change for each day of BATA stock
BATA['Daily Return'].plot(figsize=(12,4),legend=True,linestyle='--',marker='o')
```

```
<Axes: >
```




```
# Histogram to visualiza the average daily return of BATA stock
import seaborn as sns
sns.distplot(BATA['Daily Return'].dropna(),bins=100,color='purple')
```

```
<ipython-input-26-3923a4eac864>:3: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(BATA['Daily Return'].dropna(),bins=100,color='purple')
<Axes: xlabel='Daily Return', ylabel='Density'>
```




```
DATA = pd.read_csv('/content/Portfolio_dataset1.csv')

# Print the head of the data


# Limit the columns to 'DATE', 'Symbol', and 'Close'
DATA = DATA[['DATE', 'Symbol', 'Close']]

# Pivot the data to reorganize it
DATA = DATA.pivot(index='DATE', columns='Symbol', values='Close')
print(DATA.head())
```

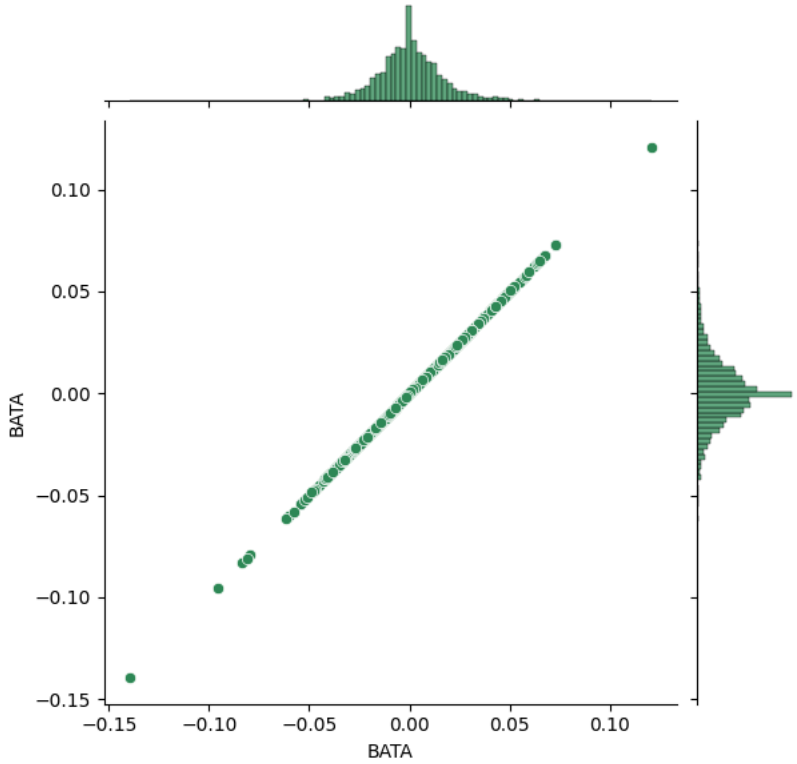
```
Symbol                           BATA     INFY        TAMO      TTML
DATE
2014-01-01 00:00:00+00:00  533.825012  6.45375  370.970978  7.511293
2014-01-02 00:00:00+00:00  515.150024  6.94125  368.398560  7.319922
2014-01-03 00:00:00+00:00  520.474976  7.14375  358.850952  7.989719
2014-01-06 00:00:00+00:00  513.150024  7.03875  363.055847  7.654821
2014-01-07 00:00:00+00:00  512.275024  7.01125  361.225494  7.559135
```

```
# Calculate the daily return percent of all stocks and store them in a new tech returns DataFrame
tech_rets = DATA.pct_change()
```

```
<ipython-input-32-f93928a83b4f>:2: FutureWarning: The default fill_method='pad' in DataFrame.pct_change is deprecated and will be re
  tech_rets = DATA.pct_change()
```

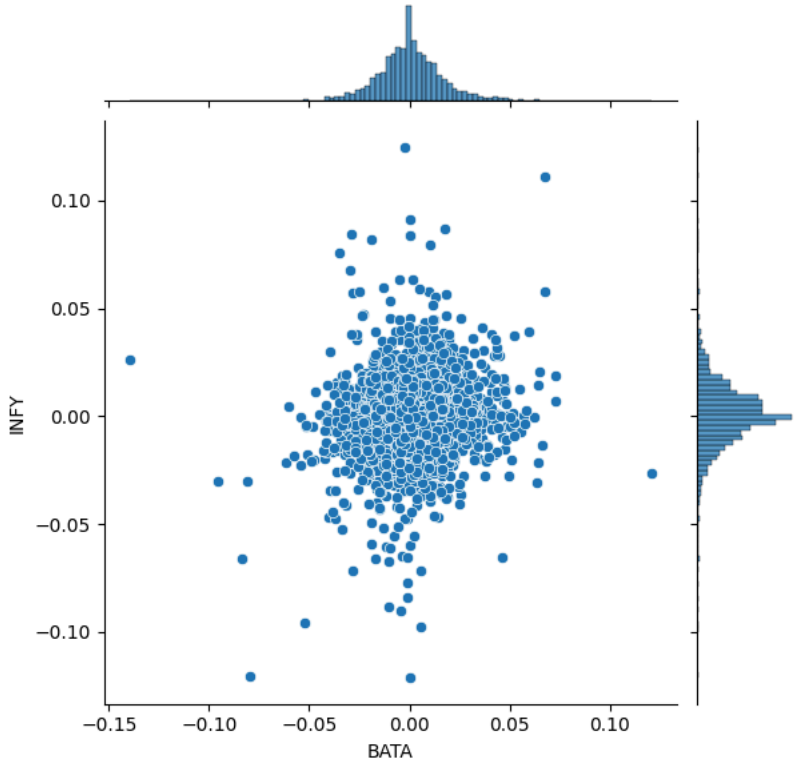
```
# Correct usage of sns.jointplot()
sns.jointplot(x='BATA', y='BATA', data=tech_rets, kind='scatter', color='seagreen')
```

```
<seaborn.axisgrid.JointGrid at 0x78857a3c2800>
```



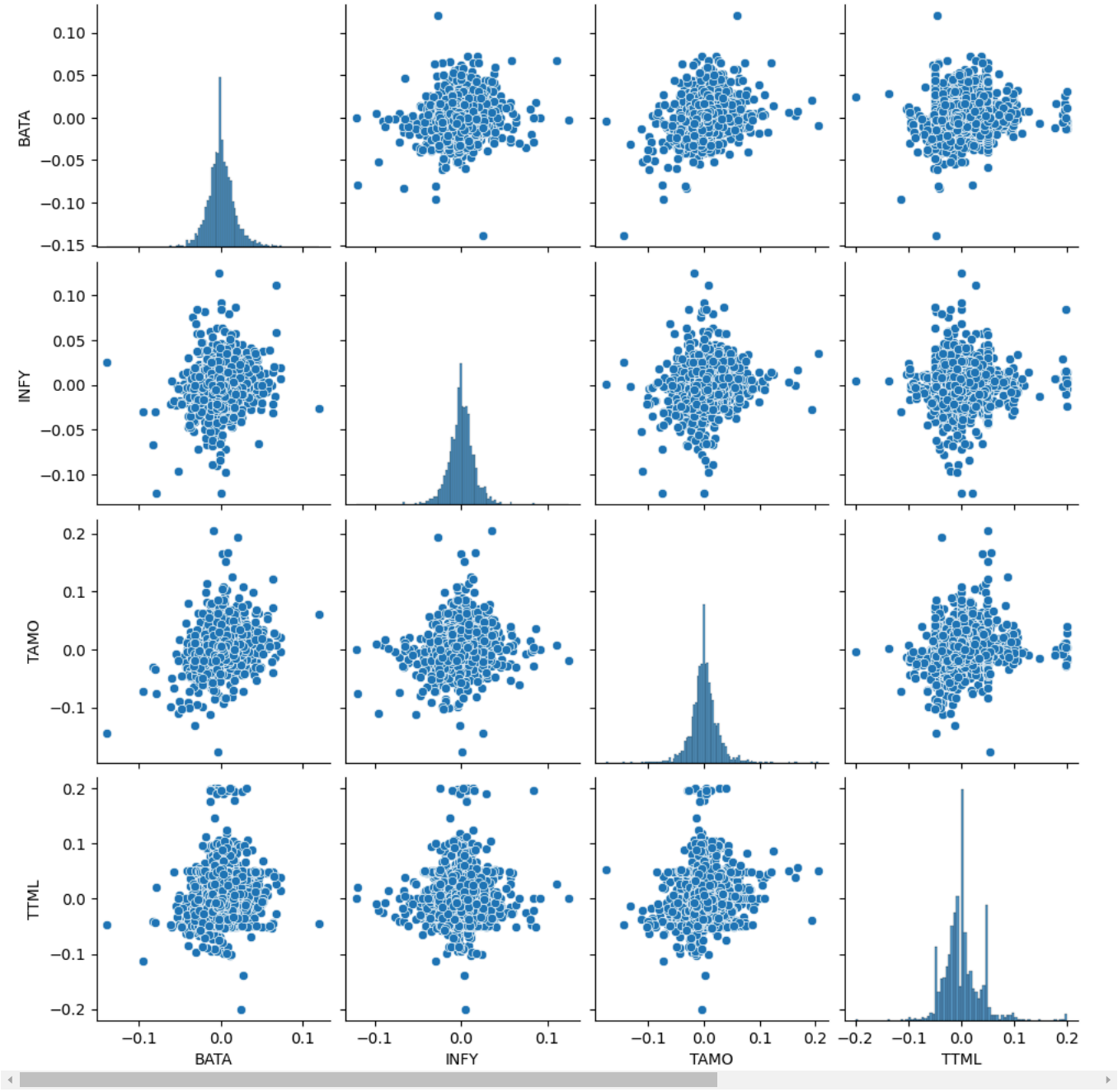

```
#Joinplot to compare the daily returns of Google and Microsoft
sns.jointplot(x='BATA', y='INFY', data=tech_rets, kind='scatter') # Pass 'BATA', 'INFY' as keyword arguments for x and y, and tech_rets
```

```
<seaborn.axisgrid.JointGrid at 0x78857a9c8bb0>
```



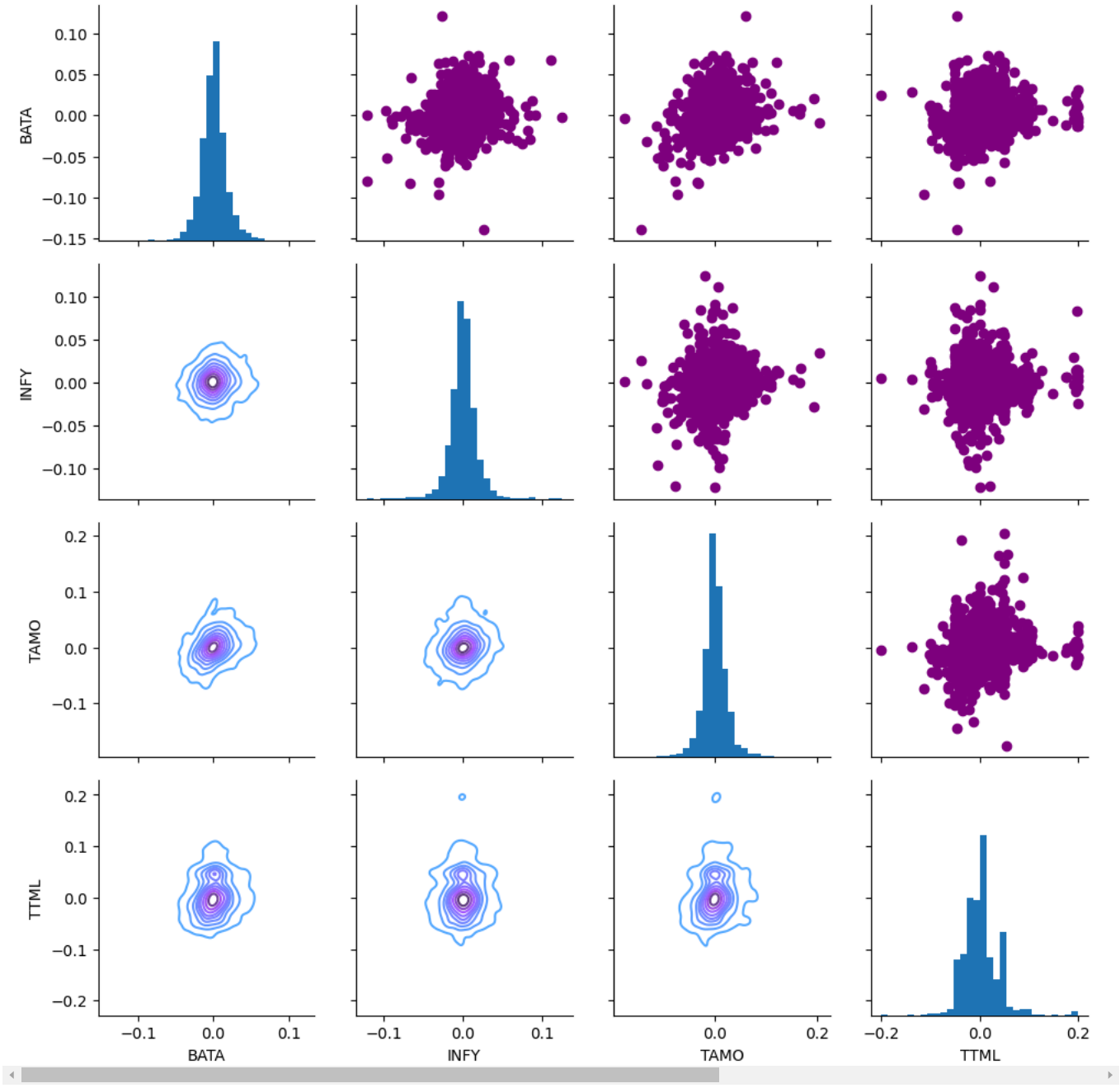

```
#Correlation analysis for every possible combination of stocks in our technology stock ticker list.
import seaborn as sns
sns.pairplot(tech_rets.dropna())
```

```
<seaborn.axisgrid.PairGrid at 0x7885bdb09510>
```



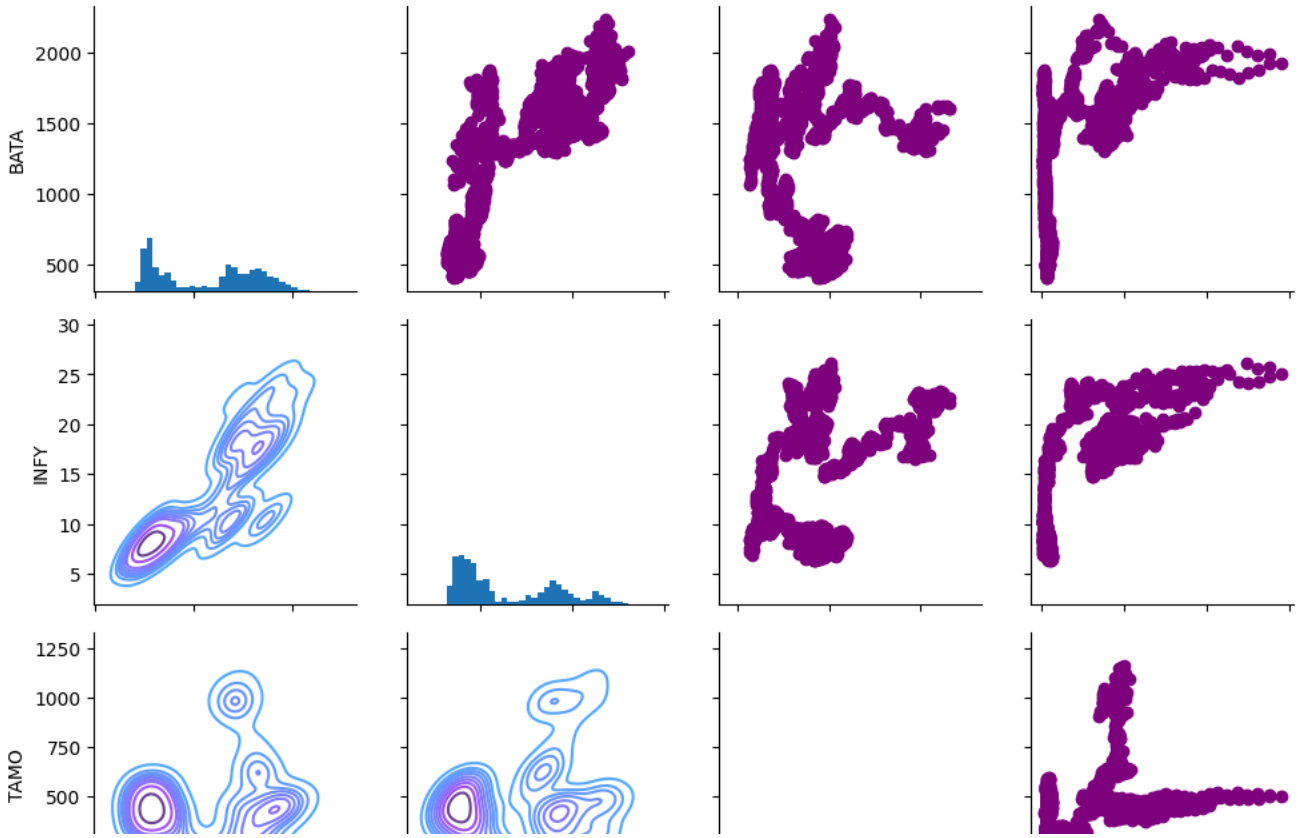

```
# Mixed plot to visualize the correlation between all technology stocks
import seaborn as sns
import matplotlib.pyplot as plt # Import the matplotlib library and assign it to the alias 'plt'

returns_fig = sns.PairGrid(tech_rets.dropna())

returns_fig.map_upper(plt.scatter,color='purple') # Now 'plt' is recognized and the scatter plot will be generated

returns_fig.map_lower(sns.kdeplot,cmap='cool_d')

returns_fig.map_diag(plt.hist,bins=30) # Now 'plt' is recognized and the histogram will be generated
```

```
<seaborn.axisgrid.PairGrid at 0x7885741efb50>
```




```
# Correlation analysis by using mixed types of plots for the closing price of all technology stocks
returns_fig = sns.PairGrid(DATA)

returns_fig.map_upper(plt.scatter,color='purple')

returns_fig.map_lower(sns.kdeplot,cmap='cool_d')

returns_fig.map_diag(plt.hist,bins=30)
```

```
<seaborn.axisgrid.PairGrid at 0x788573493af0>
```




```
import seaborn as sns
import matplotlib.pyplot as plt
import numpy as np

# Calculate the correlation matrix
corr_matrix = tech_rets.dropna().corr()

# Generate a mask for the upper triangle
mask = np.triu(np.ones_like(corr_matrix, dtype=bool))

# Set up the matplotlib figure
f, ax = plt.subplots(figsize=(11, 9))

# Generate a custom diverging colormap
cmap = sns.diverging_palette(230, 20, as_cmap=True)

# Draw the heatmap with the mask and correct aspect ratio
sns.heatmap(corr_matrix, mask=mask, cmap=cmap, vmax=.3, center=0,
            square=True, linewidths=.5, cbar_kws={"shrink": .5}, annot=True)

plt.show() # Display the plot
```