```
!pip install yfinance

Requirement already satisfied: yfinance in
/usr/local/lib/python3.10/dist-packages (0.2.49)
Requirement already satisfied: pandas>=1.3.0 in
/usr/local/lib/python3.10/dist-packages (from yfinance) (2.2.2)
Requirement already satisfied: numpy>=1.16.5 in
/usr/local/lib/python3.10/dist-packages (from yfinance) (1.26.4)
Requirement already satisfied: requests>=2.31 in
/usr/local/lib/python3.10/dist-packages (from yfinance) (2.32.3)
Requirement already satisfied: multitasking>=0.0.7 in
/usr/local/lib/python3.10/dist-packages (from yfinance) (0.0.11)
Requirement already satisfied: lxml>=4.9.1 in
/usr/local/lib/python3.10/dist-packages (from yfinance) (5.3.0)
Requirement already satisfied: platformdirs>=2.0.0 in
/usr/local/lib/python3.10/dist-packages (from yfinance) (4.3.6)
Requirement already satisfied: pytz>=2022.5 in
/usr/local/lib/python3.10/dist-packages (from yfinance) (2024.2)
Requirement already satisfied: frozendict>=2.3.4 in
/usr/local/lib/python3.10/dist-packages (from yfinance) (2.4.6)
Requirement already satisfied: peewee>=3.16.2 in
/usr/local/lib/python3.10/dist-packages (from yfinance) (3.17.8)
Requirement already satisfied: beautifulsoup4>=4.11.1 in
/usr/local/lib/python3.10/dist-packages (from yfinance) (4.12.3)
Requirement already satisfied: html5lib>=1.1 in
/usr/local/lib/python3.10/dist-packages (from yfinance) (1.1)
Requirement already satisfied: soupsieve>1.2 in
/usr/local/lib/python3.10/dist-packages (from beautifulsoup4>=4.11.1-
>yfinance) (2.6)
Requirement already satisfied: six>=1.9 in
/usr/local/lib/python3.10/dist-packages (from html5lib>=1.1->yfinance)
(1.16.0)
Requirement already satisfied: webencodings in
/usr/local/lib/python3.10/dist-packages (from html5lib>=1.1->yfinance)
(0.5.1)
Requirement already satisfied: python-dateutil>=2.8.2 in
/usr/local/lib/python3.10/dist-packages (from pandas>=1.3.0->yfinance)
(2.8.2)
Requirement already satisfied: tzdata>=2022.7 in
/usr/local/lib/python3.10/dist-packages (from pandas>=1.3.0->yfinance)
(2024.2)
Requirement already satisfied: charset-normalizer<4,>=2 in
/usr/local/lib/python3.10/dist-packages (from requests>=2.31-
>yfinance) (3.4.0)
Requirement already satisfied: idna<4,>=2.5 in
/usr/local/lib/python3.10/dist-packages (from requests>=2.31-
>yfinance) (3.10)
Requirement already satisfied: urllib3<3,>=1.21.1 in
/usr/local/lib/python3.10/dist-packages (from requests>=2.31-
>yfinance) (2.2.3)
```

```
Requirement already satisfied: certifi>=2017.4.17 in
/usr/local/lib/python3.10/dist-packages (from requests>=2.31-
>yfinance) (2024.8.30)

pip install python-louvain

Requirement already satisfied: python-louvain in
/usr/local/lib/python3.10/dist-packages (0.16)
Requirement already satisfied: networkx in
/usr/local/lib/python3.10/dist-packages (from python-louvain) (3.4.2)
Requirement already satisfied: numpy in
/usr/local/lib/python3.10/dist-packages (from python-louvain) (1.26.4)

import yfinance as yf
import pandas as pd
import numpy as np
import networkx as nx
import matplotlib.pyplot as plt
import seaborn as sns
from datetime import datetime

tickers = [
    'AAPL', 'MSFT', 'GOOGL', 'AMZN', 'META', 'TSLA', 'NVDA', 'BRK-B',
    'JNJ', 'V', 'WMT', 'JPM', 'UNH', 'MA', 'PG', 'HD', 'DIS', 'ADBE',
    'NFLX', 'PYPL'
]

def fetch_and_process_data():
    data = yf.download(tickers, start='2020-01-01', end='2020-12-31')
['Close']

    print("\nMissing values in dataset:")
    print(data.isnull().sum())

    data = data.fillna(method='ffill')
    return data

def create_correlation_network(data, threshold=0.6):
    corr_matrix = data.corr()

    G = nx.Graph()

    for ticker in tickers:
        G.add_node(ticker)

    for i in range(len(tickers)):
        for j in range(i+1, len(tickers)):
            correlation = abs(corr_matrix.loc[tickers[i], tickers[j]])
            if correlation > threshold:
                G.add_edge(tickers[i], tickers[j], weight=correlation)
```

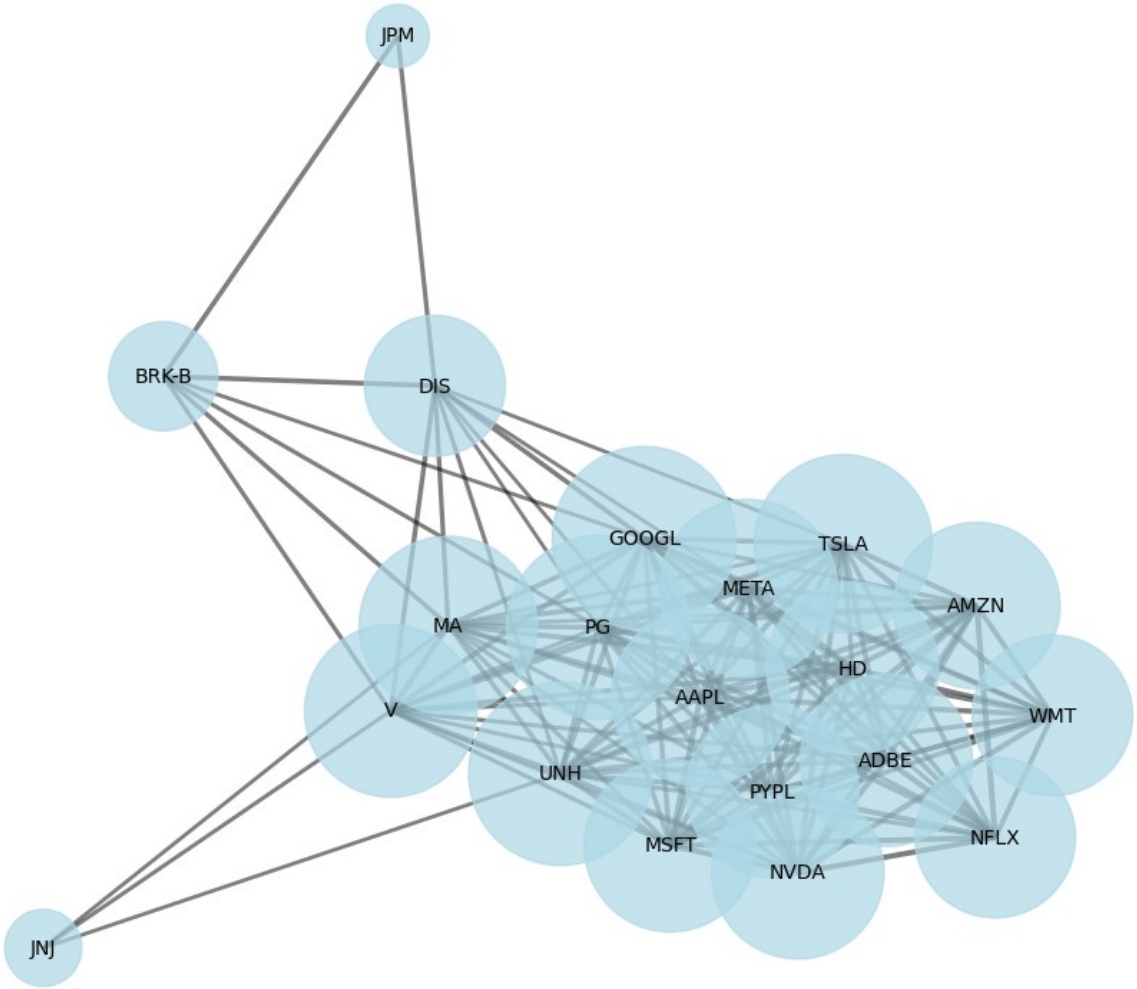
```
    return G, corr_matrix

def visualize_network(G, corr_matrix):
    plt.figure(figsize=(11, 10))

    degrees = dict(nx.degree(G))
    node_sizes = [v * 500 for v in degrees.values()]

    pos = nx.spring_layout(G, k=1, iterations=50)

    nx.draw_networkx_nodes(G, pos, node_size=node_sizes,
                           node_color='lightblue', alpha=0.7)

    edge_weights = nx.get_edge_attributes(G, 'weight')
    nx.draw_networkx_edges(G, pos, edgelist=edge_weights.keys(),
                           width=[w * 3 for w in
edge_weights.values()],
                           alpha=0.5)

    nx.draw_networkx_labels(G, pos, font_size=10)

    plt.title("Stock Correlation Network (2020)\nEdge thickness
represents correlation strength",
              fontsize=12, pad=20)
    plt.axis('off')

    plt.figure(figsize=(12, 10))
    sns.heatmap(corr_matrix, annot=True, cmap='coolwarm',
                fmt='.2f', square=True)
    plt.title("Stock Correlation Heatmap")

def analyze_network(G, corr_matrix):
    density = nx.density(G)
    avg_clustering = nx.average_clustering(G)
    communities = nx.community.louvain_communities(G)
    degree_cent = nx.degree_centrality(G)
    betweenness_cent = nx.betweenness_centrality(G)

    print("\nNetwork Analysis Results:")
    print(f"Number of nodes: {G.number_of_nodes()}")
    print(f"Number of edges: {G.number_of_edges()}")
    print(f"Network density: {density:.3f}")
    print(f"Average clustering coefficient: {avg_clustering:.3f}")

    print("\nTop 5 Central Stocks (by degree centrality):")
    sorted_degree = sorted(degree_cent.items(), key=lambda x: x[1],
reverse=True)[:5]
    for stock, cent in sorted_degree:
        print(f"{stock}: {cent:.3f}")
```

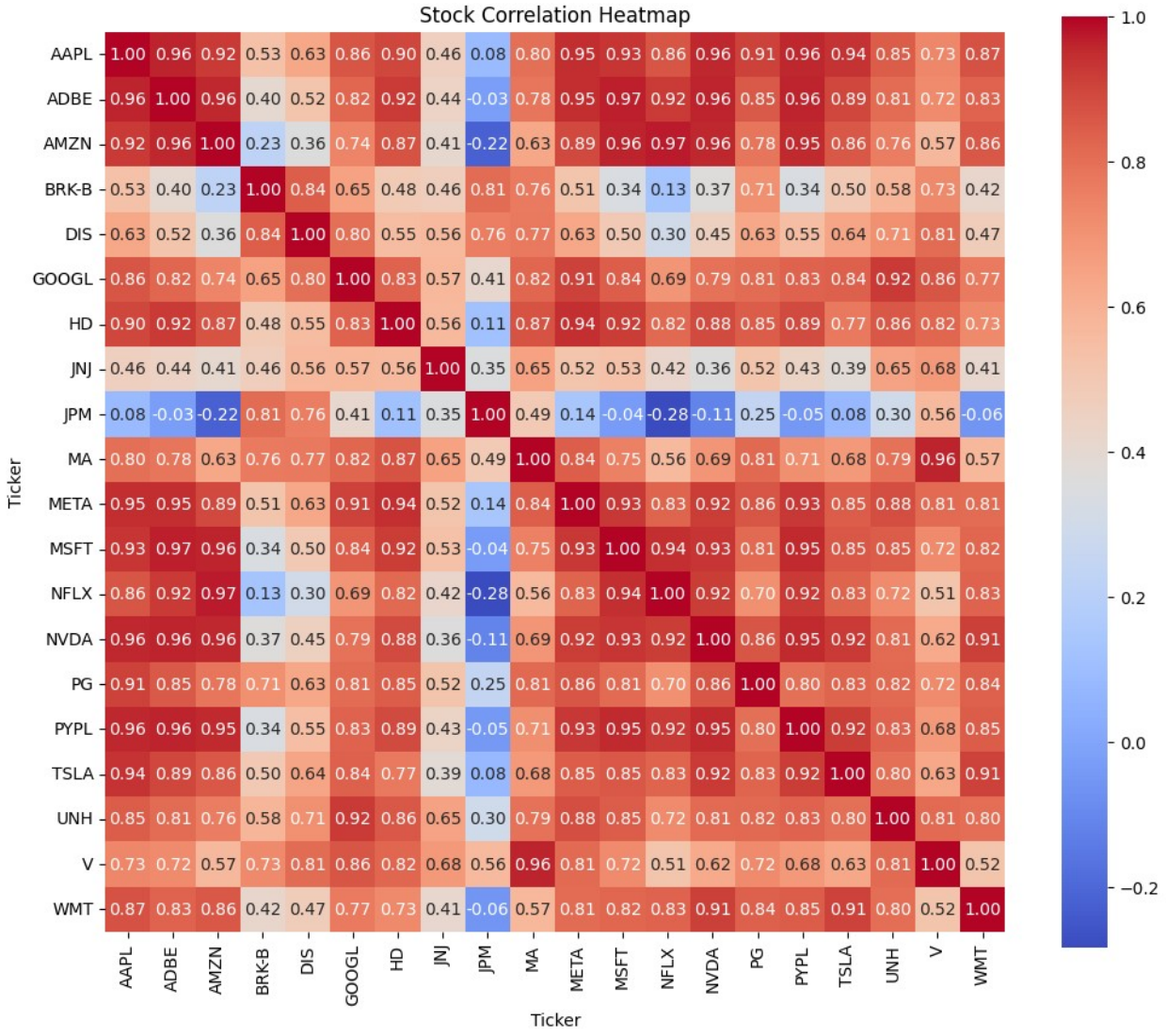
```
    print("\nIdentified Communities:")
    for i, community in enumerate(communities, 1):
        print(f"Community {i}: {', '.join(community)}")

def main():
    data = fetch_and_process_data()

    G, corr_matrix = create_correlation_network(data, threshold=0.6)

    visualize_network(G, corr_matrix)

    analyze_network(G, corr_matrix)

    plt.show()

if __name__ == "__main__":
    main()
```

```
[**********************100%**********************]  20 of 20 completed
<ipython-input-7-a8eacd14d153>:21: FutureWarning: DataFrame.fillna
with 'method' is deprecated and will raise in a future version. Use
obj.ffill() or obj.bfill() instead.
  data = data.fillna(method='ffill')


Missing values in dataset:
Ticker
AAPL     0
ADBE     0
AMZN     0
BRK-B    0
DIS      0
GOOGL    0
HD       0
JNJ      0
JPM      0
MA       0
META     0
MSFT     0
NFLX     0
NVDA     0
PG       0
PYPL     0
TSLA     0
UNH      0
V        0
WMT      0
dtype: int64

Network Analysis Results:
```

```
Number of nodes: 20
Number of edges: 133
Network density: 0.700
Average clustering coefficient: 0.891

Top 5 Central Stocks (by degree centrality):
GOOGL: 0.895
UNH: 0.895
PG: 0.895
AAPL: 0.842
META: 0.842

Identified Communities:
Community 1: TSLA, MSFT, WMT, AAPL, PG, HD, AMZN, ADBE, NFLX, NVDA,
META, PYPL
Community 2: DIS, GOOGL, UNH, BRK-B, V, JNJ, JPM, MA
```

Stock Correlation Network (2020)
Edge thickness represents correlation strength





##Implication:

Tech Exposure: Due to the small size and close proximity of the tech industry, investors should be wary of expecting too much diversity from holding a number of tech equities.

Sector Analysis: The community structure predominantly corresponds with conventional sector classifications, thereby supporting sector-based analytical methods.

Market Indicators: The prominent stocks (MSFT, AAPL, V, JPM, MA) may serve as indicators for the overall market trends.

Diversification: The evident community structure indicates that genuine diversification may include investment across these distinct communities rather than only across various stocks.

Unique Opportunities: Stocks such as Tesla that exhibit distinctive behavior may provide diversification advantages; however, they may also be associated with distinctive risks.

##Interpretation: High Clustering: Two stocks are likely to be linked with each other if they are connected with a third stock, according to the high average clustering coefficient. This points to the market's high level of interdependence.

Tech Dominance: Tech stocks make up most of the biggest group, which shows that there are strong connections within the tech sector. This means that these stocks moved together a lot in 2020. This could be because the tech industry was affected by similar market forces.

Diverse Stable Stocks: The fourth community comprises a combination of consumer goods, healthcare, and services stocks. These stocks are frequently regarded as more stable and defensive, which may account for their classification.

Financial Sector: The limited financial community (BRK-B and JPM) indicates that these stocks exhibited comparable price movements, potentially mirroring overarching trends in the financial sector.

Tesla's Uniqueness: Tesla's establishment of its own community underscores its distinctive status in 2020. This may result from its accelerated growth and market enthusiasm, distinguishing it from other stocks.

Central Stocks: The significant centrality of MSFT, AAPL, V, JPM, and MA demonstrates that these stocks exerted considerable influence on overall market movements. Investors and analysts may closely observe these stocks as reflections of overarching market trends.