Comprehensive Financial Analysis and Valuation of Publicly Traded Companies Using Python

```
ticker = 'AAPL'
stock = yf.Ticker(ticker)
income_statement = stock.financials.T
balance_sheet = stock.balance_sheet.T
cash_flow = stock.cashflow.T
```

Downloaded financial data for Apple (AAPL)

```
import yfinance as yf
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
from mpl_toolkits.mplot3d import Axes3D
import seaborn as sns
import scipy.stats as stats
```

Retrieved financial statements and transposed them for easier handling and print first few rows of each financial statement to understand the structure

```
print("Income Statement:")
print(income_statement.head())
```

```
Income Statement:
            Tax Effect Of Unusual Items Tax Rate For Calcs Normalized
EBITDA  \
2023-09-30                          0.0              0.147
129188000000.0
2022-09-30                          0.0              0.162
133138000000.0
2021-09-30                          0.0              0.133
123136000000.0
2020-09-30                          0.0              0.144
81020000000.0

            Net Income From Continuing Operation Net Minority Interest
\
2023-09-30                                        96995000000.0

2022-09-30                                        99803000000.0
```

```
2021-09-30                                          94680000000.0

2020-09-30                                          57411000000.0


            Reconciled Depreciation Reconciled Cost Of Revenue
EBITDA  \
2023-09-30           11519000000.0             214137000000.0
129188000000.0
2022-09-30           11104000000.0             223546000000.0
133138000000.0
2021-09-30           11284000000.0             212981000000.0
123136000000.0
2020-09-30           11056000000.0             169559000000.0
81020000000.0

                       EBIT Net Interest Income Interest
Expense  ...  \
2023-09-30  117669000000.0        -183000000.0     3933000000.0  ...

2022-09-30  122034000000.0        -106000000.0     2931000000.0  ...

2021-09-30  111852000000.0         198000000.0     2645000000.0  ...

2020-09-30   69964000000.0         890000000.0     2873000000.0  ...


            Interest Expense Non Operating Interest Income Non
Operating  \
2023-09-30                    3933000000.0
3750000000.0
2022-09-30                    2931000000.0
2825000000.0
2021-09-30                    2645000000.0
2843000000.0
2020-09-30                    2873000000.0
3763000000.0

            Operating Income Operating Expense Research And Development
\
2023-09-30   114301000000.0     54847000000.0             29915000000.0

2022-09-30   119437000000.0     51345000000.0             26251000000.0

2021-09-30   108949000000.0     43887000000.0             21914000000.0

2020-09-30    66288000000.0     38668000000.0             18752000000.0


            Selling General And Administration    Gross Profit Cost Of
Revenue  \
```

```
2023-09-30                         24932000000.0  169148000000.0
214137000000.0
2022-09-30                         25094000000.0  170782000000.0
223546000000.0
2021-09-30                         21973000000.0  152836000000.0
212981000000.0
2020-09-30                         19916000000.0  104956000000.0
169559000000.0

             Total Revenue Operating Revenue
2023-09-30  383285000000.0    383285000000.0
2022-09-30  394328000000.0    394328000000.0
2021-09-30  365817000000.0    365817000000.0
2020-09-30  274515000000.0    274515000000.0

[4 rows x 39 columns]
```

```
print("\nBalance Sheet:")
print(balance_sheet.head())
```

```
Balance Sheet:
           Treasury Shares Number Ordinary Shares Number   Share
Issued  \
2023-09-30                    0.0          15550061000.0
15550061000.0
2022-09-30                    NaN          15943425000.0
15943425000.0
2021-09-30                    NaN          16426786000.0
16426786000.0
2020-09-30                    NaN          16976763000.0
16976763000.0
2019-09-30                    NaN                    NaN
NaN

                 Net Debt      Total Debt Tangible Book Value  \
2023-09-30  81123000000.0  123930000000.0       62146000000.0
2022-09-30  96423000000.0  132480000000.0       50672000000.0
2021-09-30  89779000000.0  136522000000.0       63090000000.0
2020-09-30  74420000000.0  122278000000.0       65339000000.0
2019-09-30            NaN             NaN                 NaN

           Invested Capital Working Capital Net Tangible Assets  \
2023-09-30   173234000000.0   -1742000000.0       62146000000.0
2022-09-30   170741000000.0  -18577000000.0       50672000000.0
2021-09-30   187809000000.0    9355000000.0       63090000000.0
2020-09-30   177775000000.0   38321000000.0       65339000000.0
2019-09-30              NaN             NaN                 NaN

           Capital Lease Obligations  ... Other Current Assets
```

```
Inventory  \
2023-09-30             12842000000.0  ...        14695000000.0
6331000000.0
2022-09-30             12411000000.0  ...        21223000000.0
4946000000.0
2021-09-30             11803000000.0  ...        14111000000.0
6580000000.0
2020-09-30              9842000000.0  ...        11264000000.0
4061000000.0
2019-09-30                       NaN  ...                  NaN
NaN

              Receivables Other Receivables Accounts Receivable  \
2023-09-30  60985000000.0     31477000000.0       29508000000.0
2022-09-30  60932000000.0     32748000000.0       28184000000.0
2021-09-30  51506000000.0     25228000000.0       26278000000.0
2020-09-30  37445000000.0     21325000000.0       16120000000.0
2019-09-30            NaN               NaN                 NaN

           Cash Cash Equivalents And Short Term Investments  \
2023-09-30                                     61555000000.0
2022-09-30                                     48304000000.0
2021-09-30                                     62639000000.0
2020-09-30                                     90943000000.0
2019-09-30                                               NaN

           Other Short Term Investments Cash And Cash Equivalents  \
2023-09-30                31590000000.0             29965000000.0
2022-09-30                24658000000.0             23646000000.0
2021-09-30                27699000000.0             34940000000.0
2020-09-30                52927000000.0             38016000000.0
2019-09-30                          NaN                       NaN

           Cash Equivalents Cash Financial
2023-09-30     1606000000.0  28359000000.0
2022-09-30     5100000000.0  18546000000.0
2021-09-30    17635000000.0  17305000000.0
2020-09-30    20243000000.0  17773000000.0
2019-09-30              NaN            NaN

[5 rows x 68 columns]
```

```
print("\nCash Flow Statement:")
print(cash_flow.head())
```

```
Cash Flow Statement:
             Free Cash Flow Repurchase Of Capital Stock Repayment Of
Debt  \
2023-09-30   99584000000.0              -77550000000.0    -
```

```
11151000000.0
2022-09-30  111443000000.0               -89402000000.0     -
9543000000.0
2021-09-30   92953000000.0               -85971000000.0     -
8750000000.0
2020-09-30   73365000000.0               -72358000000.0    -
12629000000.0
2019-09-30             NaN                           NaN
NaN

            Issuance Of Debt Issuance Of Capital Stock Capital
Expenditure  \
2023-09-30     5228000000.0                         NaN      -
10959000000.0
2022-09-30     5465000000.0                         NaN      -
10708000000.0
2021-09-30    20393000000.0               1105000000.0      -
11085000000.0
2020-09-30    16091000000.0                880000000.0       -
7309000000.0
2019-09-30              NaN                781000000.0
NaN

            Interest Paid Supplemental Data Income Tax Paid
Supplemental Data  \
2023-09-30                    3803000000.0
18679000000.0
2022-09-30                    2865000000.0
19573000000.0
2021-09-30                    2687000000.0
25385000000.0
2020-09-30                    3002000000.0
9501000000.0
2019-09-30                             NaN
NaN

            End Cash Position Beginning Cash Position  ... Change In
Inventory  \
2023-09-30     30737000000.0           24977000000.0  ...       -
1618000000.0
2022-09-30     24977000000.0           35929000000.0  ...
1484000000.0
2021-09-30     35929000000.0           39789000000.0  ...       -
2642000000.0
2020-09-30     39789000000.0           50224000000.0  ...        -
127000000.0
2019-09-30               NaN                     NaN  ...
NaN

            Change In Receivables Changes In Account Receivables  \
```

```
2023-09-30          -417000000.0                   -1688000000.0
2022-09-30         -9343000000.0                   -1823000000.0
2021-09-30        -14028000000.0                  -10125000000.0
2020-09-30          8470000000.0                    6917000000.0
2019-09-30                   NaN                             NaN

            Other Non Cash Items Stock Based Compensation  Deferred Tax
\
2023-09-30        -2227000000.0             10833000000.0           NaN

2022-09-30         1006000000.0              9038000000.0   895000000.0

2021-09-30        -4921000000.0              7906000000.0 -4774000000.0

2020-09-30          -97000000.0              6829000000.0  -215000000.0

2019-09-30                  NaN                       NaN  -340000000.0


            Deferred Income Tax Depreciation Amortization Depletion  \
2023-09-30                  NaN                        11519000000.0
2022-09-30         895000000.0                        11104000000.0
2021-09-30       -4774000000.0                        11284000000.0
2020-09-30        -215000000.0                        11056000000.0
2019-09-30        -340000000.0                                  NaN

            Depreciation And Amortization Net Income From Continuing
Operations
2023-09-30                  11519000000.0
96995000000.0
2022-09-30                  11104000000.0
99803000000.0
2021-09-30                  11284000000.0
94680000000.0
2020-09-30                  11056000000.0
57411000000.0
2019-09-30                           NaN
NaN

[5 rows x 53 columns]
```

Displayed the columns of each financial statement to identify available data points

```
print("Income Statement Columns:")
print(income_statement.columns)

print("\nBalance Sheet Columns:")
print(balance_sheet.columns)

print("\nCash Flow Statement Columns:")
print(cash_flow.columns)
```

```
Income Statement Columns:
Index(['Tax Effect Of Unusual Items', 'Tax Rate For Calcs',
       'Normalized EBITDA',
       'Net Income From Continuing Operation Net Minority Interest',
       'Reconciled Depreciation', 'Reconciled Cost Of Revenue',
'EBITDA',
       'EBIT', 'Net Interest Income', 'Interest Expense', 'Interest
Income',
       'Normalized Income',
       'Net Income From Continuing And Discontinued Operation',
       'Total Expenses', 'Total Operating Income As Reported',
       'Diluted Average Shares', 'Basic Average Shares', 'Diluted
EPS',
       'Basic EPS', 'Diluted NI Availto Com Stockholders',
       'Net Income Common Stockholders', 'Net Income',
       'Net Income Including Noncontrolling Interests',
       'Net Income Continuous Operations', 'Tax Provision', 'Pretax
Income',
       'Other Income Expense', 'Other Non Operating Income Expenses',
       'Net Non Operating Interest Income Expense',
       'Interest Expense Non Operating', 'Interest Income Non
Operating',
       'Operating Income', 'Operating Expense', 'Research And
Development',
       'Selling General And Administration', 'Gross Profit', 'Cost Of
Revenue',
       'Total Revenue', 'Operating Revenue'],
      dtype='object')

Balance Sheet Columns:
Index(['Treasury Shares Number', 'Ordinary Shares Number', 'Share
Issued',
       'Net Debt', 'Total Debt', 'Tangible Book Value', 'Invested
Capital',
       'Working Capital', 'Net Tangible Assets', 'Capital Lease
Obligations',
       'Common Stock Equity', 'Total Capitalization',
       'Total Equity Gross Minority Interest', 'Stockholders Equity',
       'Gains Losses Not Affecting Retained Earnings',
       'Other Equity Adjustments', 'Retained Earnings', 'Capital
Stock',
       'Common Stock', 'Total Liabilities Net Minority Interest',
       'Total Non Current Liabilities Net Minority Interest',
       'Other Non Current Liabilities', 'Tradeand Other Payables Non
Current',
       'Long Term Debt And Capital Lease Obligation',
       'Long Term Capital Lease Obligation', 'Long Term Debt',
       'Current Liabilities', 'Other Current Liabilities',
       'Current Deferred Liabilities', 'Current Deferred Revenue',
       'Current Debt And Capital Lease Obligation',
```

```
       'Current Capital Lease Obligation', 'Current Debt',
       'Other Current Borrowings', 'Commercial Paper',
       'Payables And Accrued Expenses', 'Payables', 'Total Tax
Payable',
       'Income Tax Payable', 'Accounts Payable', 'Total Assets',
       'Total Non Current Assets', 'Other Non Current Assets',
       'Non Current Deferred Assets', 'Non Current Deferred Taxes
Assets',
       'Investments And Advances', 'Other Investments',
       'Investmentin Financial Assets', 'Available For Sale
Securities',
       'Net PPE', 'Accumulated Depreciation', 'Gross PPE', 'Leases',
       'Other Properties', 'Machinery Furniture Equipment',
       'Land And Improvements', 'Properties', 'Current Assets',
       'Other Current Assets', 'Inventory', 'Receivables', 'Other
Receivables',
       'Accounts Receivable',
       'Cash Cash Equivalents And Short Term Investments',
       'Other Short Term Investments', 'Cash And Cash Equivalents',
       'Cash Equivalents', 'Cash Financial'],
      dtype='object')

Cash Flow Statement Columns:
Index(['Free Cash Flow', 'Repurchase Of Capital Stock', 'Repayment Of
Debt',
       'Issuance Of Debt', 'Issuance Of Capital Stock', 'Capital
Expenditure',
       'Interest Paid Supplemental Data', 'Income Tax Paid
Supplemental Data',
       'End Cash Position', 'Beginning Cash Position', 'Changes In
Cash',
       'Financing Cash Flow', 'Cash Flow From Continuing Financing
Activities',
       'Net Other Financing Charges', 'Cash Dividends Paid',
       'Common Stock Dividend Paid', 'Net Common Stock Issuance',
       'Common Stock Payments', 'Common Stock Issuance',
       'Net Issuance Payments Of Debt', 'Net Short Term Debt
Issuance',
       'Net Long Term Debt Issuance', 'Long Term Debt Payments',
       'Long Term Debt Issuance', 'Investing Cash Flow',
       'Cash Flow From Continuing Investing Activities',
       'Net Other Investing Changes', 'Net Investment Purchase And
Sale',
       'Sale Of Investment', 'Purchase Of Investment',
       'Net Business Purchase And Sale', 'Purchase Of Business',
       'Net PPE Purchase And Sale', 'Purchase Of PPE', 'Operating Cash
Flow',
       'Cash Flow From Continuing Operating Activities',
       'Change In Working Capital', 'Change In Other Working Capital',
       'Change In Other Current Liabilities', 'Change In Other Current
```

```
Assets',
       'Change In Payables And Accrued Expense', 'Change In Payable',
       'Change In Account Payable', 'Change In Inventory',
       'Change In Receivables', 'Changes In Account Receivables',
       'Other Non Cash Items', 'Stock Based Compensation', 'Deferred
Tax',
       'Deferred Income Tax', 'Depreciation Amortization Depletion',
       'Depreciation And Amortization',
       'Net Income From Continuing Operations'],
      dtype='object')
```

Extracted specific financial metrics from the balance sheet and cash flow statement

```
total_assets = balance_sheet['Total Assets']
total_liabilities = balance_sheet['Total Liabilities Net Minority
Interest']
total_equity = balance_sheet['Stockholders Equity']
operating_cash_flow = cash_flow['Operating Cash Flow']
capital_expenditure = cash_flow['Capital Expenditure']
free_cash_flow = cash_flow['Free Cash Flow']
```

Extracted specific financial metrics from the income statement

```
gross_profit = income_statement['Gross Profit']
operating_income = income_statement['Operating Income']
net_income = income_statement['Net Income Continuous Operations']
```

Calculated financial ratios to assess profitability and leverage

```
profit_margin = net_income / income_statement['Total Revenue']
return_on_assets = net_income / total_assets
return_on_equity = net_income / total_equity
debt_to_equity = total_liabilities / total_equity
```

Combined the extracted metrics and ratios into a single DataFrame for analysis

```
metrics = pd.DataFrame({
    'Gross Profit': gross_profit,
    'Operating Income': operating_income,
    'Net Income': net_income,
    'Operating Cash Flow': operating_cash_flow,
    'Capital Expenditure': capital_expenditure,
    'Free Cash Flow': free_cash_flow,
    'Profit Margin': profit_margin,
    'Return on Assets': return_on_assets,
    'Return on Equity': return_on_equity,
    'Debt to Equity': debt_to_equity
})
```

Displayed the compiled metrics DataFrame

```
print(metrics)
```

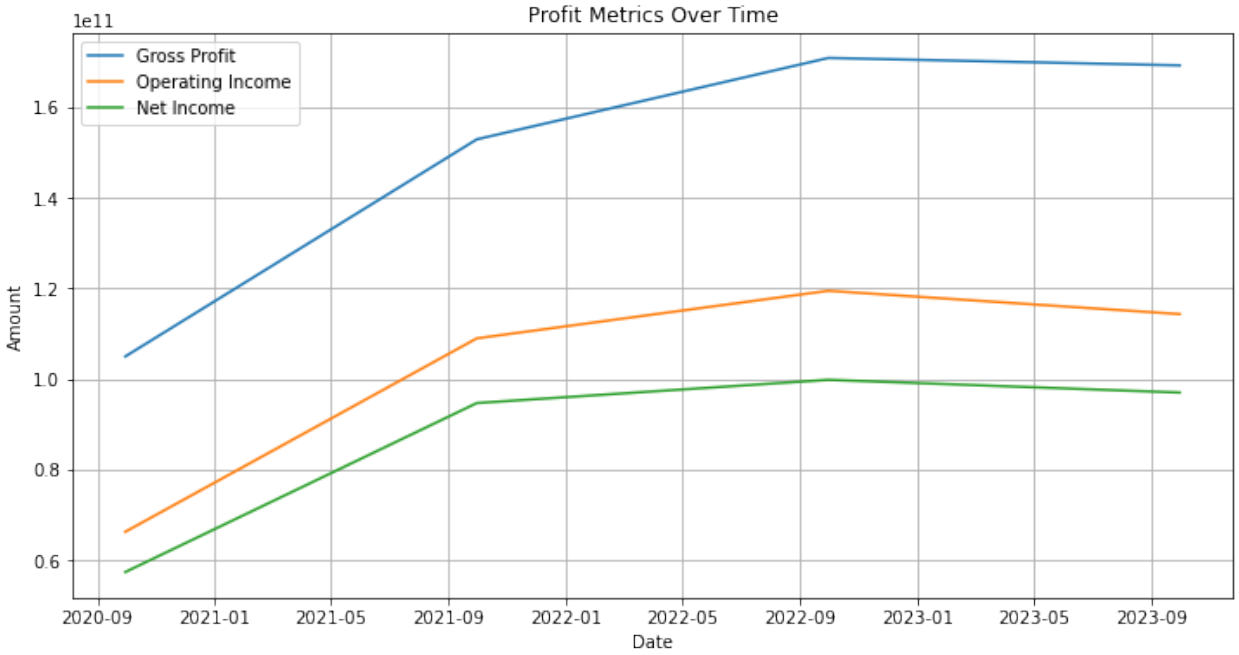
```
              Gross Profit Operating Income     Net Income  \
2019-09-30             NaN              NaN            NaN
2020-09-30  104956000000.0    66288000000.0  57411000000.0
2021-09-30  152836000000.0   108949000000.0  94680000000.0
2022-09-30  170782000000.0   119437000000.0  99803000000.0
2023-09-30  169148000000.0   114301000000.0  96995000000.0

            Operating Cash Flow Capital Expenditure  Free Cash Flow  \
2019-09-30                  NaN                 NaN             NaN
2020-09-30       80674000000.0       -7309000000.0   73365000000.0
2021-09-30      104038000000.0      -11085000000.0   92953000000.0
2022-09-30      122151000000.0      -10708000000.0  111443000000.0
2023-09-30      110543000000.0      -10959000000.0   99584000000.0

            Profit Margin Return on Assets Return on Equity Debt to
Equity
2019-09-30           NaN               NaN              NaN
NaN
2020-09-30      0.209136          0.177256         0.878664
3.957039
2021-09-30      0.258818          0.269742         1.500713
4.563512
2022-09-30      0.253096          0.282924         1.969589
5.961537
2023-09-30      0.253062          0.275098          1.56076
4.673462
```

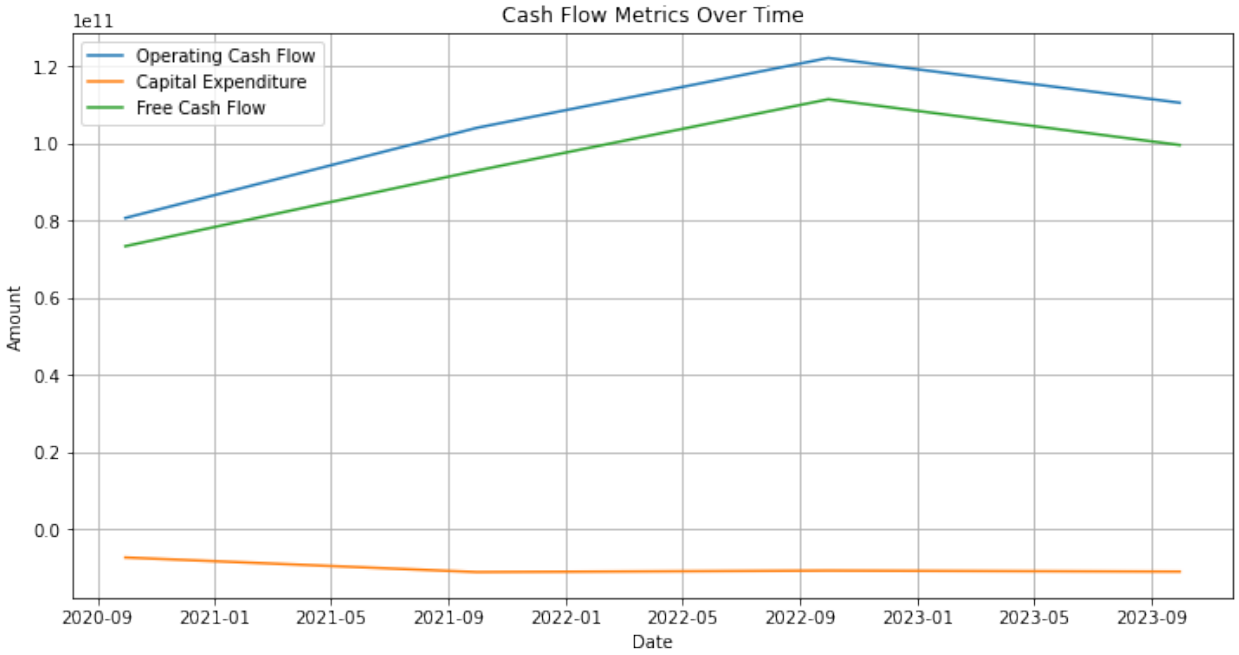
Plotted Gross Profit, Operating Income, and Net Income to visualize profit metrics over time

```
plt.figure(figsize=(12, 6))
plt.plot(gross_profit, label='Gross Profit')
plt.plot(operating_income, label='Operating Income')
plt.plot(net_income, label='Net Income')
plt.xlabel('Date')
plt.ylabel('Amount')
plt.title('Profit Metrics Over Time')
plt.legend()
plt.grid(True)
plt.show()
```



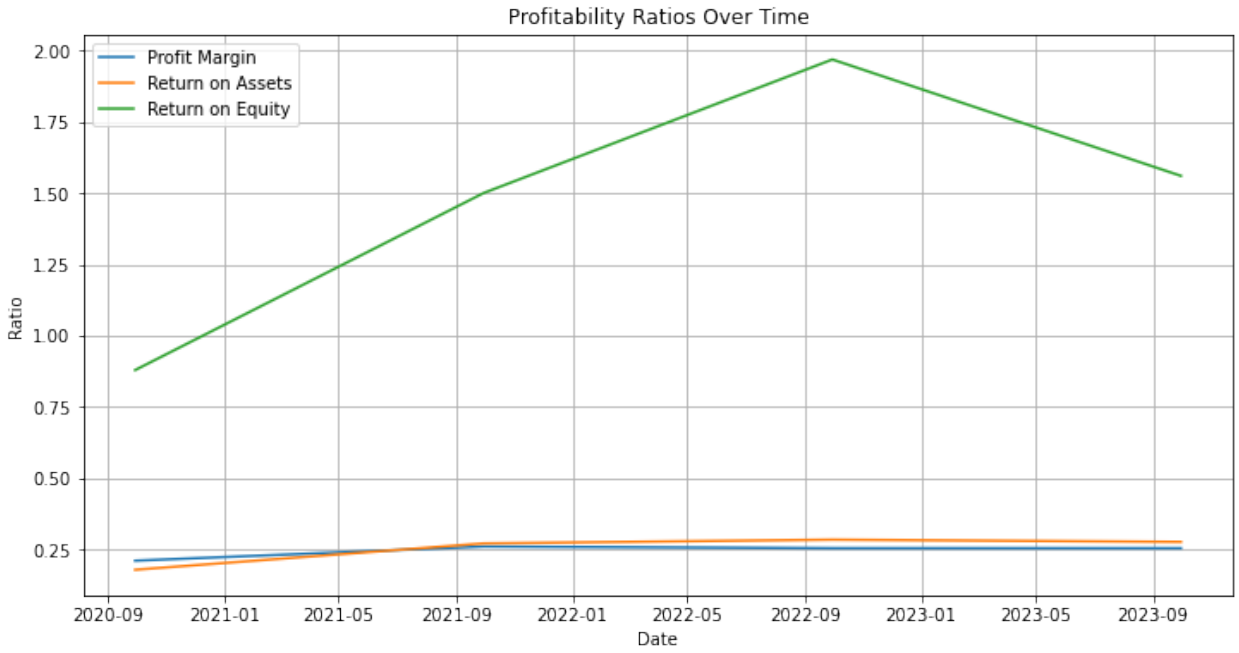
Plotted Operating Cash Flow, Capital Expenditure, and Free Cash Flow to visualize cash flow metrics over time

```
plt.figure(figsize=(12, 6))
plt.plot(operating_cash_flow, label='Operating Cash Flow')
plt.plot(capital_expenditure, label='Capital Expenditure')
plt.plot(free_cash_flow, label='Free Cash Flow')
plt.xlabel('Date')
plt.ylabel('Amount')
plt.title('Cash Flow Metrics Over Time')
plt.legend()
plt.grid(True)
plt.show()
```



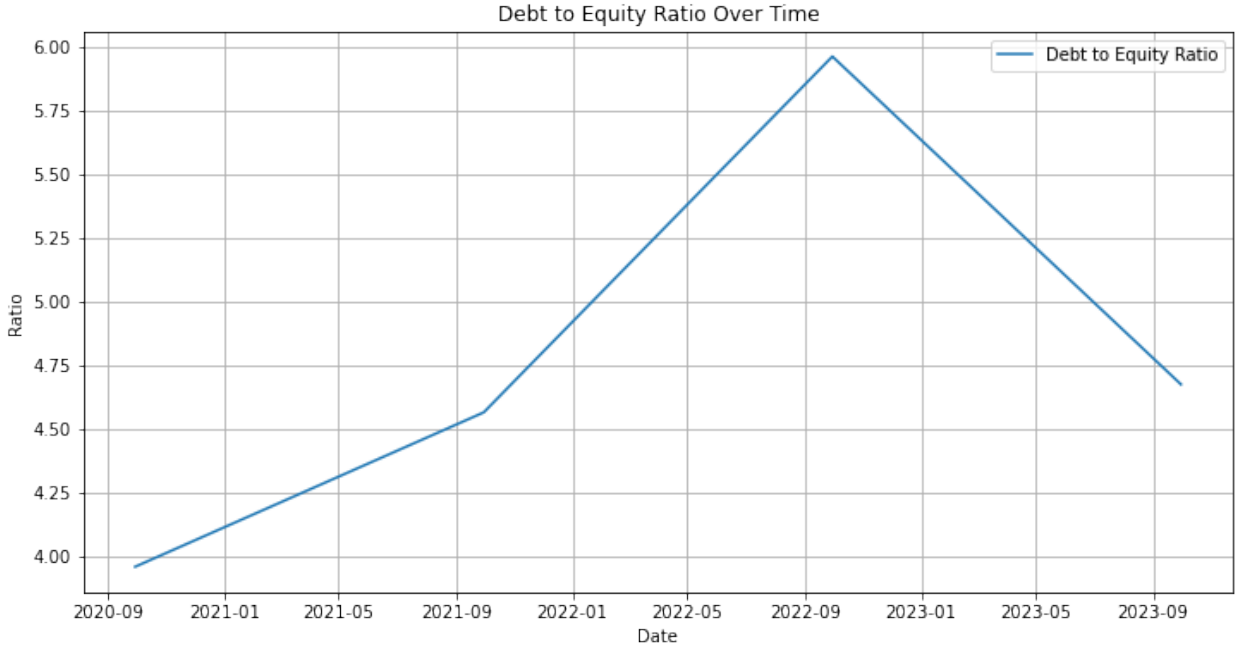
Plotted Profit Margin, Return on Assets, and Return on Equity to visualize profitability ratios over time

```
plt.figure(figsize=(12, 6))
plt.plot(profit_margin, label='Profit Margin')
plt.plot(return_on_assets, label='Return on Assets')
plt.plot(return_on_equity, label='Return on Equity')
plt.xlabel('Date')
plt.ylabel('Ratio')
plt.title('Profitability Ratios Over Time')
plt.legend()
plt.grid(True)
plt.show()
```



Plotted Debt to Equity Ratio to visualize the leverage over time

```
plt.figure(figsize=(12, 6))
plt.plot(debt_to_equity, label='Debt to Equity Ratio')
plt.xlabel('Date')
plt.ylabel('Ratio')
plt.title('Debt to Equity Ratio Over Time')
plt.legend()
plt.grid(True)
plt.show()
```



Discounted Cash Flow (DCF) and Sensitivity analysis

Defined base parameters for Discounted Cash Flow (DCF) analysis

```
base_discount_rate = 0.08
base_growth_rate = 0.05
base_terminal_growth_rate = 0.03
```

Defined ranges for sensitivity analysis of the DCF model

```
discount_rate_range = np.arange(0.07, 0.09, 0.01)
growth_rate_range = np.arange(0.04, 0.06, 0.01)
terminal_growth_rate_range = np.arange(0.02, 0.04, 0.01)
```

Function to calculate DCF value given free cash flows, discount rate, growth rate, and terminal growth rate

```
def calculate_dcf(free_cash_flows, discount_rate, growth_rate,
terminal_growth_rate):
    present_value = sum([cf / (1 + discount_rate) ** (i + 1) for i, cf
in enumerate(free_cash_flows)])
    terminal_value = free_cash_flows[-1] * (1 + terminal_growth_rate)
/ (discount_rate - terminal_growth_rate)
    present_value += terminal_value / (1 + discount_rate) **
len(free_cash_flows)
    return present_value
```

Example free cash flows for the DCF calculation

```
free_cash_flows = [100, 110, 121, 133, 146]
```

Conducted sensitivity analysis by varying discount rate, growth rate, and terminal growth rate

```
sensitivity_results = pd.DataFrame(columns=['Discount Rate', 'Growth
Rate', 'Terminal Growth Rate', 'DCF Value'])
results = []
for dr in discount_rate_range:
    for gr in growth_rate_range:
        for tgr in terminal_growth_rate_range:
            dcf_value = calculate_dcf(free_cash_flows, dr, gr, tgr)
            results.append({
                'Discount Rate': dr,
                'Growth Rate': gr,
                'Terminal Growth Rate': tgr,
                'DCF Value': dcf_value
            })
sensitivity_results = pd.DataFrame(results)
```

Displayed the first few rows of the sensitivity analysis results

```
print(sensitivity_results.head())
```

```
   Discount Rate  Growth Rate  Terminal Growth Rate    DCF Value
0           0.07         0.04                  0.02  2617.427330
1           0.07         0.04                  0.03  3174.340834
2           0.07         0.05                  0.02  2617.427330
3           0.07         0.05                  0.03  3174.340834
4           0.08         0.04                  0.02  2169.285176
```

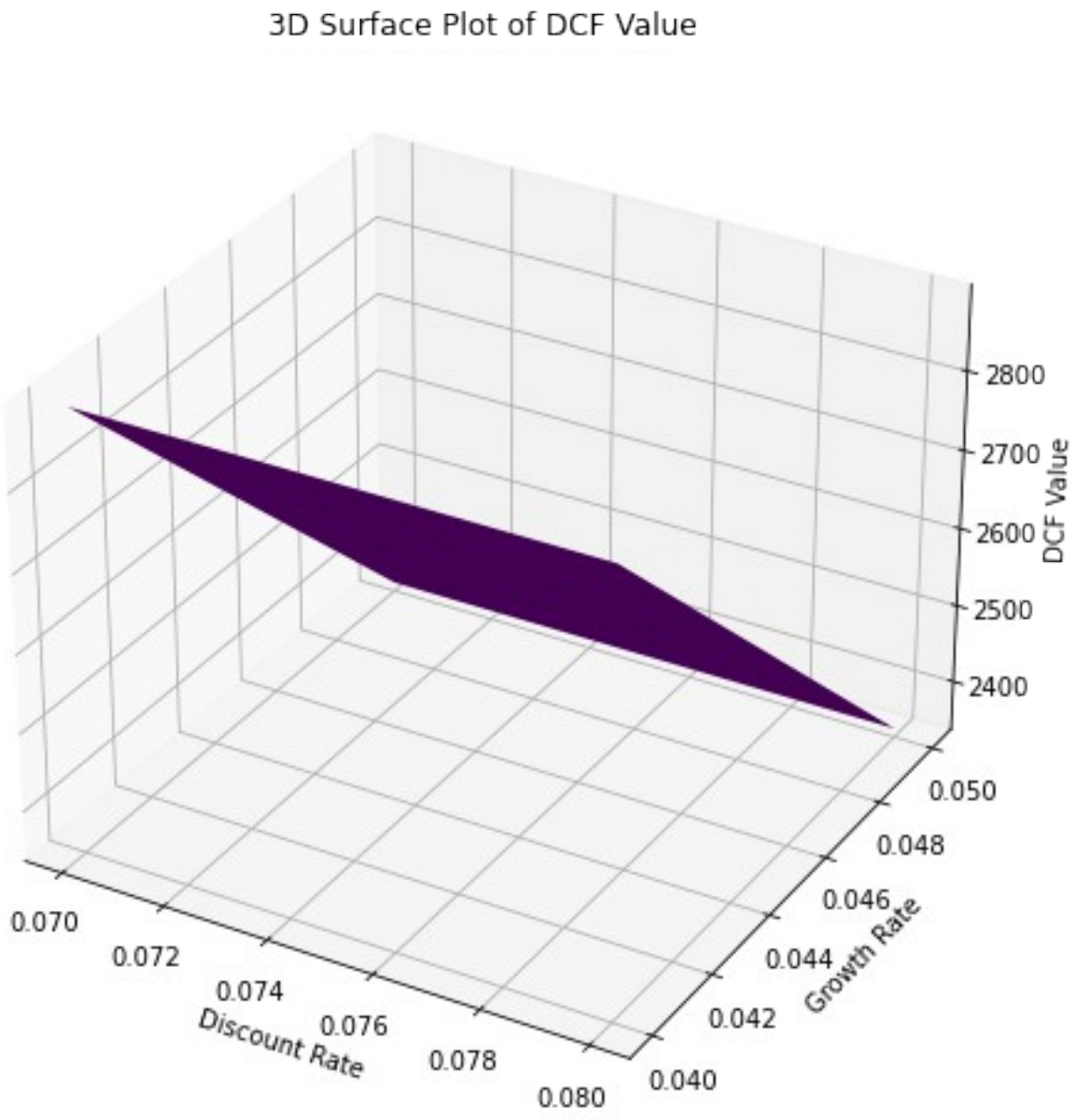
```
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
pivot_df = sensitivity_results.pivot_table(index='Discount Rate',
columns='Growth Rate', values='DCF Value')
x = pivot_df.index
y = pivot_df.columns
x, y = np.meshgrid(x, y)
z = pivot_df.values
```

Plotted 3D surface plot of DCF value against discount rate and growth rate to visualize the sensitivity analysis

```
fig = plt.figure(figsize=(12, 8))
ax = fig.add_subplot(111, projection='3d')
ax.plot_surface(x, y, z, cmap='viridis')

ax.set_xlabel('Discount Rate')
ax.set_ylabel('Growth Rate')
ax.set_zlabel('DCF Value')
ax.set_title('3D Surface Plot of DCF Value')

plt.show()
```



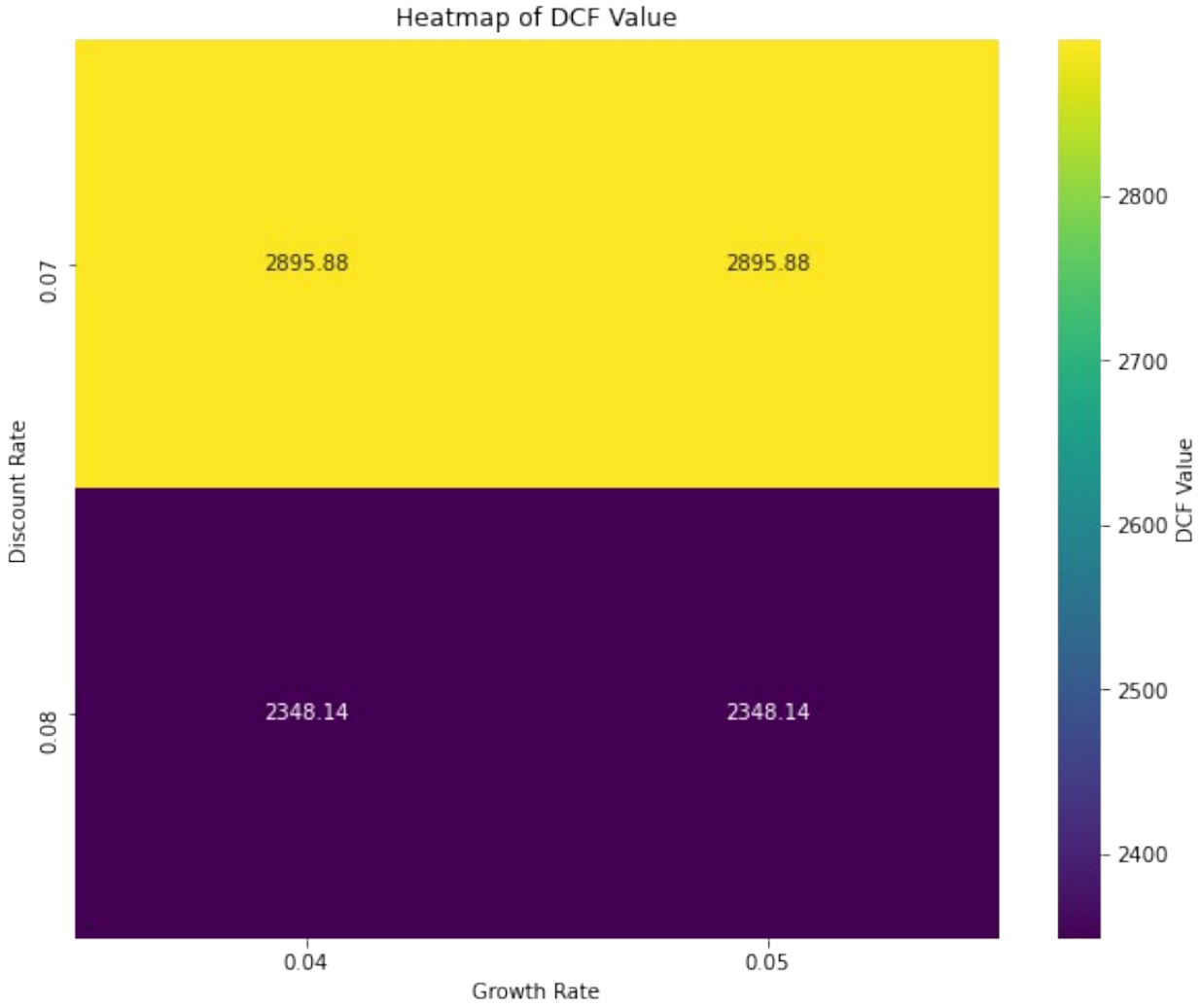
Plotted heatmap of DCF value to visualize the sensitivity analysis in 2D

```
import seaborn as sns
heatmap_df = sensitivity_results.pivot_table(index='Discount Rate',
columns='Growth Rate', values='DCF Value')

plt.figure(figsize=(10, 8))
sns.heatmap(heatmap_df, cmap='viridis', annot=True, fmt='.2f',
cbar_kws={'label': 'DCF Value'})

plt.title('Heatmap of DCF Value')
plt.xlabel('Growth Rate')
```

```
plt.ylabel('Discount Rate')
plt.show()
```

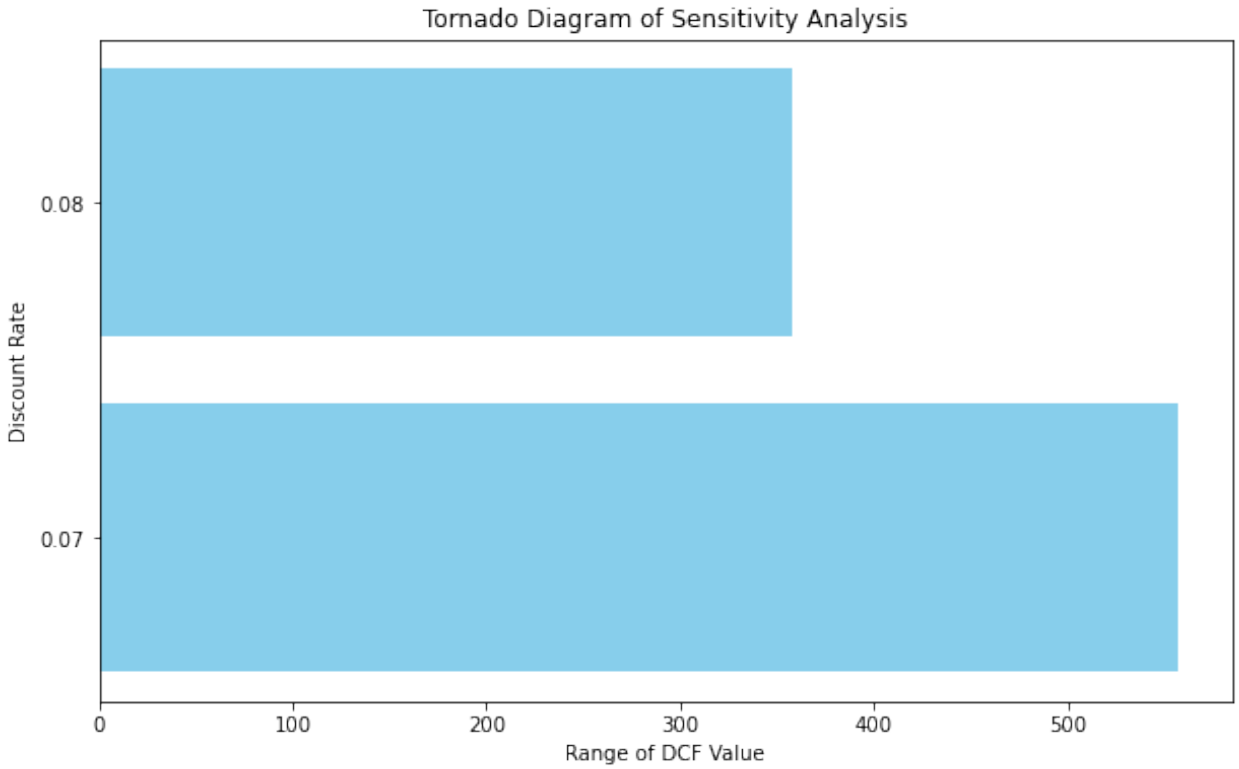


Summarized the sensitivity results to understand the range of DCF values for each discount rate and plotted Tornado Diagram to visualize the range of DCF values for each discount rate

```
sensitivity_summary = sensitivity_results.groupby('Discount
Rate').agg({'DCF Value': ['min', 'max']})
sensitivity_summary.columns = sensitivity_summary.columns.droplevel()
sensitivity_summary['Range'] = sensitivity_summary['max'] -
sensitivity_summary['min']
sensitivity_summary = sensitivity_summary.sort_values('Range',
ascending=False)

plt.figure(figsize=(10, 6))
plt.barh(sensitivity_summary.index.astype(str),
sensitivity_summary['Range'], color='skyblue')
```

```
plt.xlabel('Range of DCF Value')
plt.ylabel('Discount Rate')
plt.title('Tornado Diagram of Sensitivity Analysis')
plt.show()
```



Calculated 95% confidence interval for the DCF value using statistical analysis

```
confidence_level = 0.95
df = len(sensitivity_results) - 1
mean_dcf = sensitivity_results['DCF Value'].mean()
std_dev = sensitivity_results['DCF Value'].std()
confidence_interval = stats.t.interval(confidence_level, df, mean_dcf,
std_dev / np.sqrt(len(sensitivity_results)))

print(f"95% Confidence Interval for DCF Value: {confidence_interval}")
```

```
95% Confidence Interval for DCF Value: (2300.0553897586537,
2943.9711325551966)
```

Comparison between different Companies

Downloaded financial data for multiple companies and calculate valuation multiples

```
tickers = ['AAPL', 'MSFT', 'GOOGL']
```

```
data = {}
for ticker in tickers:
    stock = yf.Ticker(ticker)
    info = stock.info
    data[ticker] = {
        'P/E Ratio': info.get('trailingPE'),
        'EV/EBITDA': info.get('enterpriseToEbitda'),
        'P/B Ratio': info.get('priceToBook'),
        'P/S Ratio': info.get('priceToSalesTrailing12Months')
    }
```

Displayed the calculated valuation multiples for each company

```
df = pd.DataFrame(data).T
print(df)
```

```
       P/E Ratio  EV/EBITDA  P/B Ratio  P/S Ratio
AAPL   33.464230     25.666  50.173440   8.668951
MSFT   34.588480     23.631  11.310812  12.386984
GOOGL  23.911049     17.127   6.828089   6.279017
```

Calculated average valuation multiples to use as a benchmark

```
average_multiples = df.mean()
print("Average Multiples:")
print(average_multiples)
```

```
Average Multiples:
P/E Ratio    30.654586
EV/EBITDA    22.141333
P/B Ratio    22.770780
P/S Ratio     9.111651
dtype: float64
```

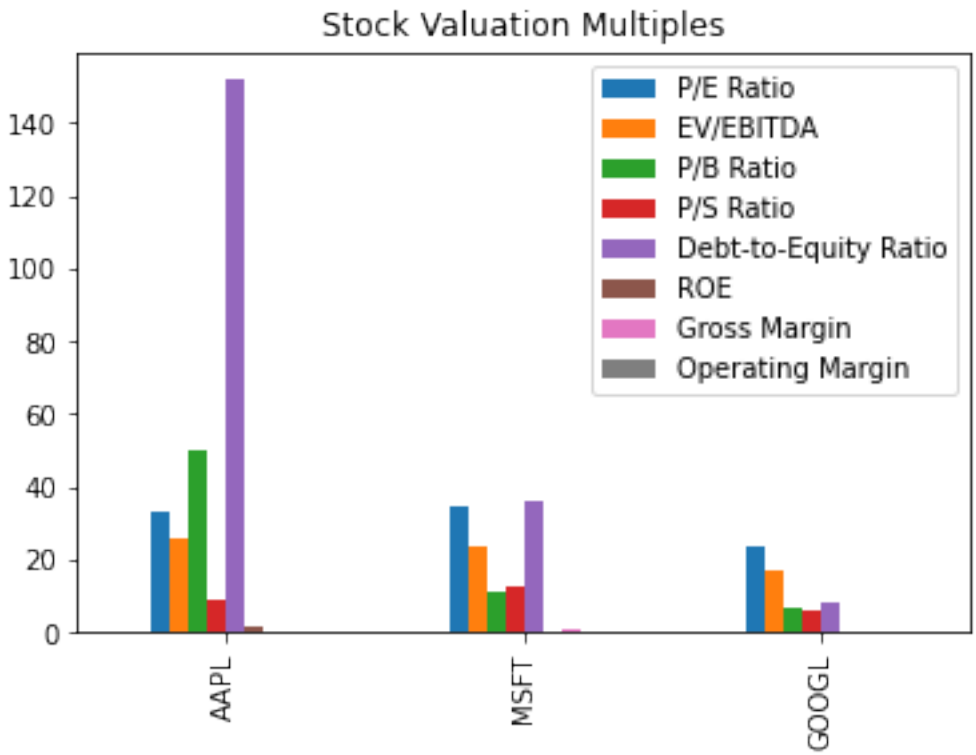
Estimated the value of Apple using the average P/E Ratio multiple

```
target_ticker = 'AAPL'
target_stock = yf.Ticker(target_ticker)
target_info = target_stock.info
target_eps = target_info.get('trailingEps')
estimated_value = average_multiples['P/E Ratio'] * target_eps
print(f"Estimated Value for {target_ticker} using P/E Ratio: $
{estimated_value:.2f}")
```

```
Estimated Value for AAPL using P/E Ratio: $201.40
```

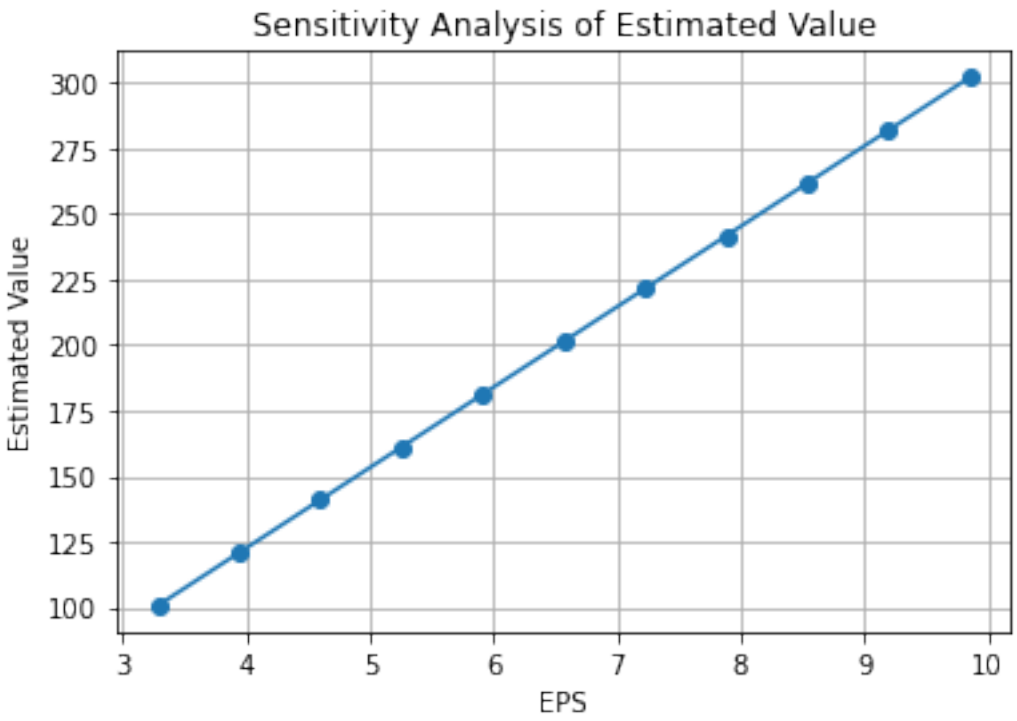
Plotted valuation multiples for visual comparison across companies

```
df.plot(kind='bar')
plt.title('Stock Valuation Multiples')
plt.show()
```



Conducted sensitivity analysis on the estimated value of Apple by varying EPS

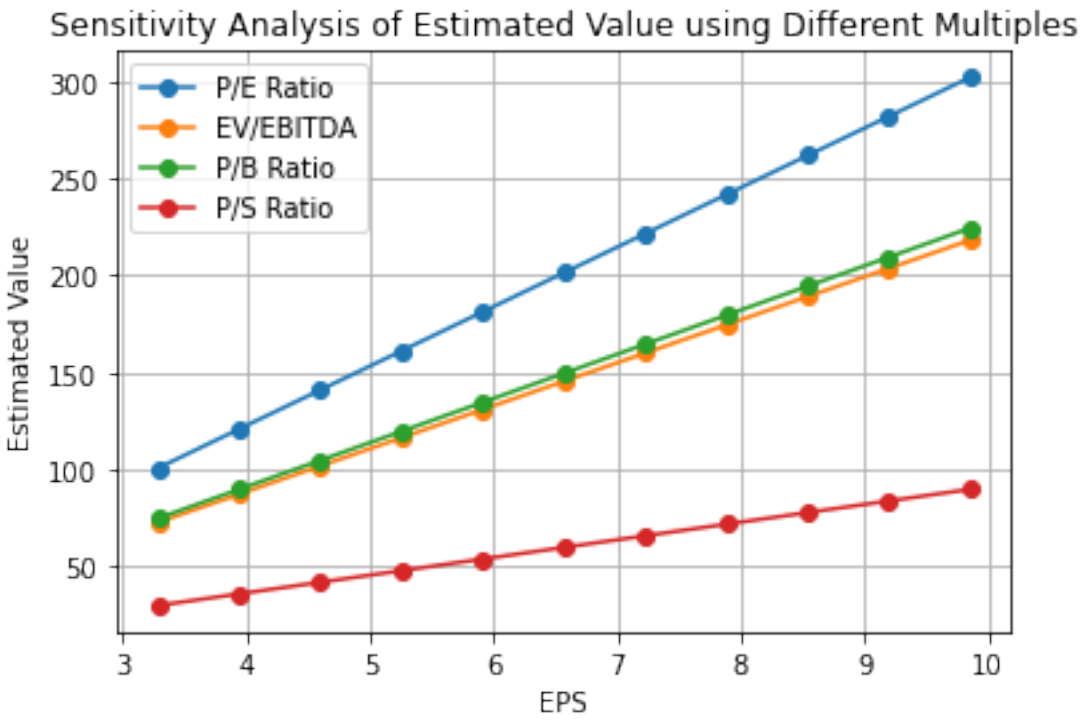
```
eps_range = [target_eps * (1 + i/10) for i in range(-5, 6)]
values = [average_multiples['P/E Ratio'] * eps for eps in eps_range]
plt.plot(eps_range, values, marker='o')
plt.xlabel('EPS')
plt.ylabel('Estimated Value')
plt.title('Sensitivity Analysis of Estimated Value')
plt.grid(True)
plt.show()
```



Conducted sensitivity analysis on the estimated value of Apple by varying EPS

```
multiples = ['P/E Ratio', 'EV/EBITDA', 'P/B Ratio', 'P/S Ratio']
for multiple in multiples:
    values = [average_multiples[multiple] * eps for eps in eps_range]
    plt.plot(eps_range, values, marker='o', label=multiple)

plt.xlabel('EPS')
plt.ylabel('Estimated Value')
plt.title('Sensitivity Analysis of Estimated Value using Different
Multiples')
plt.legend()
plt.grid(True)
plt.show()
```

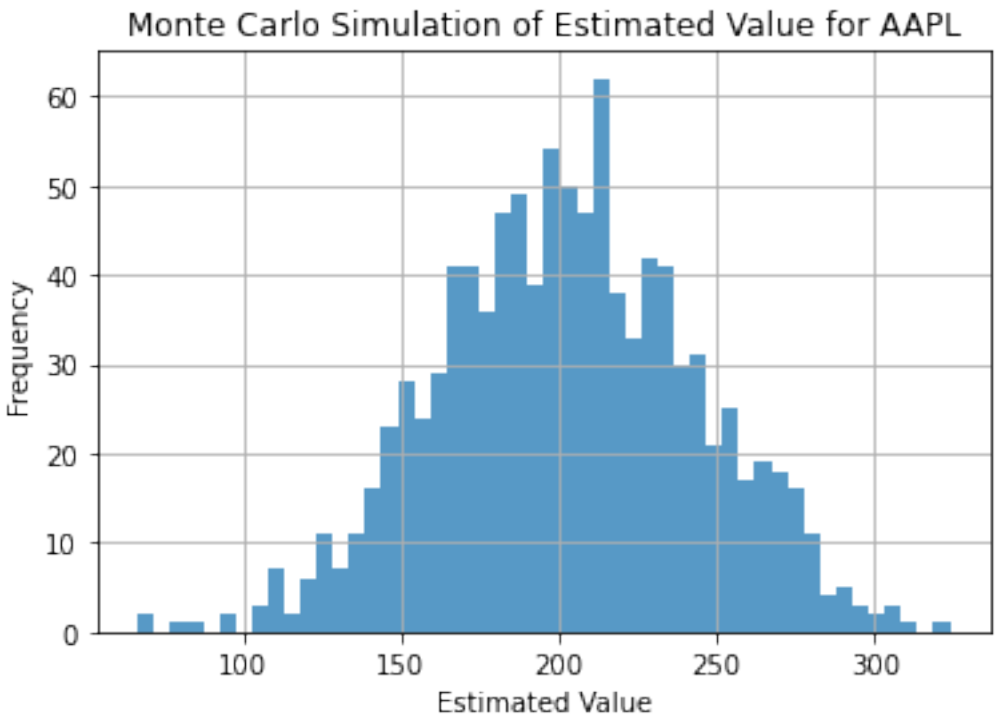


Conducted Monte Carlo simulation to estimate the value of Apple based on EPS distribution

```
num_simulations = 1000
eps_mean = target_eps
eps_std = target_eps * 0.2
simulated_eps = np.random.normal(eps_mean, eps_std, num_simulations)
simulated_values = average_multiples['P/E Ratio'] * simulated_eps

plt.hist(simulated_values, bins=50, alpha=0.75)
plt.xlabel('Estimated Value')
plt.ylabel('Frequency')
plt.title('Monte Carlo Simulation of Estimated Value for AAPL')
plt.grid(True)
plt.show()

# Print summary statistics
print(f"Mean Estimated Value: ${np.mean(simulated_values):.2f}")
print(f"Median Estimated Value: ${np.median(simulated_values):.2f}")
print(f"10th Percentile: ${np.percentile(simulated_values, 10):.2f}")
print(f"90th Percentile: ${np.percentile(simulated_values, 90):.2f}")
```




```
Mean Estimated Value: $202.65
Median Estimated Value: $202.82
10th Percentile: $151.00
90th Percentile: $256.90
```