bcgx-project-mauricio-razon

November 17, 2024

BCG X Virtual Internship Task 1 by Mauricio Razon:

The first task will allow provide the foundation to develop a deep understanding of financial data analysis and its significance in AI applications.

The outline will consist of the following: Data extraction: Research and review 10-K documents. Focus on key financial figures and ratios.

Basic analysis: Identify significant financial trends and indicators. Assess the financial health and performance of the companies.

Data preparation: Format and clean the data for AI model integration.

Deliverable: A comprehensive data analysis report, which should include: your findings a summary providing insights into the financial health of the analyzed companies.

Importing Packages and Datasets:

[1]:
```
import pandas as pd
import numpy as np
```

[2]:
```
import os
print(os.getcwd())
os.chdir('/Users/razon/Downloads/')
```

```
c:\Users\razon\AppData\Local\Temp\1d6910e2-81ba-48a3-932a-f3b43c4fd5d9_BCGX_Proj
ect_Mauricio_Razon.zip.5d9
```

[3]:
```
df = pd.read_csv('financial_statements_BCGX_.csv')
```

[4]:
```
df
```

[4]:

	Company	Unnamed: 1	Unnamed: 2	\
0	Microsoft	NaN	NaN	
1	NaN	Total Revenue	NaN	
2	NaN	Net Income	NaN	
3	NaN	Total Assets	NaN	
4	NaN	Total Liabilities	NaN	
5	Cash Flow from Operating Activities	NaN	NaN	
6	NaN	NaN	NaN	
7	Tesla	NaN	NaN	

```
8                                    NaN      Total Revenue         NaN
9                                    NaN         Net Income         NaN
10                                   NaN       Total Assets         NaN
11                                   NaN  Total Liabilities         NaN
12  Cash Flow from Operating Activities                NaN         NaN
13                                   NaN                NaN         NaN
14                                 Apple                NaN         NaN
15                                   NaN      Total Revenue         NaN
16                                   NaN         Net Income         NaN
17                                   NaN       Total Assets         NaN
18                                   NaN  Total Liabilities         NaN
19  Cash Flow from Operating Activities                NaN         NaN

        2021      2022      2023  Unnamed: 6
0        NaN       NaN       NaN         NaN
1   168088.0  198270.0  211915.0         NaN
2    61271.0   72738.0   72361.0         NaN
3   333779.0  364840.0  411976.0         NaN
4   191791.0  198298.0  205753.0         NaN
5     2706.0    2368.0    2052.0         NaN
6        NaN       NaN       NaN         NaN
7        NaN       NaN       NaN         NaN
8    53823.0   81462.0   96773.0         NaN
9     5644.0   12587.0   14974.0         NaN
10   62131.0   82338.0  106618.0         NaN
11   30548.0   36440.0   43009.0         NaN
12   11497.0   14724.0   13256.0         NaN
13       NaN       NaN       NaN         NaN
14       NaN       NaN       NaN         NaN
15  365817.0  394328.0  383285.0         NaN
16   94680.0   99803.0   96995.0         NaN
17  351002.0  352755.0  352583.0         NaN
18  287912.0  302083.0  290437.0         NaN
19  104038.0  122151.0  110543.0         NaN
```

Tidying and Cleaning the data set:

[5]:
```
df = df.drop(columns=['Unnamed: 2', 'Unnamed: 6'])
df['Company'] = df['Company'].ffill()
df = df.dropna(subset=['Unnamed: 1'])
df = df.rename(columns={
    'Unnamed: 1': 'Metric',
    '2021': '2021',
    '2022': '2022',
    '2023': '2023'
})
```

```
df = df.pivot(index='Company', columns='Metric', values=['2021', '2022',␣
↪'2023'])
df.columns = [f'{year}_{metric}' for year, metric in df.columns]
df
```

[5]:

Company	2021_Net Income	2021_Total Assets	2021_Total Liabilities	2021_Total Revenue	2022_Net Income	2022_Total Assets	2022_Total Liabilities	2022_Total Revenue	2023_Net Income	2023_Total Assets	2023_Total Liabilities	2023_Total Revenue
Apple	94680.0	351002.0	287912.0	365817.0	99803.0	352755.0	302083.0	394328.0	96995.0	352583.0	290437.0	383285.0
Microsoft	61271.0	333779.0	191791.0	168088.0	72738.0	364840.0	198298.0	198270.0	72361.0	411976.0	205753.0	211915.0
Tesla	5644.0	62131.0	30548.0	53823.0	12587.0	82338.0	36440.0	81462.0	14974.0	106618.0	43009.0	96773.0

[6]:
```
df= df.astype('Int64')
df
```

[6]:

Company	2021_Net Income	2021_Total Assets	2021_Total Liabilities	2021_Total Revenue	2022_Net Income	2022_Total Assets	2022_Total Liabilities	2022_Total Revenue	2023_Net Income
Apple	94680	351002	287912	365817	99803	352755	302083	394328	96995
Microsoft	61271	333779	191791	168088	72738	364840			
Tesla	5644	62131	30548	53823	12587	82338			

```
Microsoft                  198298              198270            72361
Tesla                       36440               81462            14974

           2023_Total Assets  2023_Total Liabilities  2023_Total Revenue
Company
Apple                 352583                  290437              383285
Microsoft             411976                  205753              211915
Tesla                 106618                   43009               96773
```

[7]:
```
df = df.stack().reset_index()
df.columns = ['Company', 'Metric_Year', 'Value']
df[['Year', 'Metric']] = df['Metric_Year'].str.split('_', expand=True)
df = df.drop(columns=['Metric_Year'])
df = df.pivot(index=['Company', 'Year'], columns='Metric', values='Value').
 ↪reset_index()
```

[8]:
```
df.columns.name = None
df
```

[8]:

	Company	Year	Net Income	Total Assets	Total Liabilities	Total Revenue
0	Apple	2021	94680	351002	287912	365817
1	Apple	2022	99803	352755	302083	394328
2	Apple	2023	96995	352583	290437	383285
3	Microsoft	2021	61271	333779	191791	168088
4	Microsoft	2022	72738	364840	198298	198270
5	Microsoft	2023	72361	411976	205753	211915
6	Tesla	2021	5644	62131	30548	53823
7	Tesla	2022	12587	82338	36440	81462
8	Tesla	2023	14974	106618	43009	96773

Now, lets move on to analyzing trends with Pandas

I will begin by calculating the year-over-year changes for each financial metric.

[9]:
```
df['Revenue Growth (%)'] = df.groupby(['Company'])['Total Revenue'].
 ↪pct_change() * 100
df['Net Income Growth (%)'] = df.groupby(['Company'])['Net Income'].
 ↪pct_change() * 100
# Fill NA values that result from pct_change calculations with 0 or an␣
 ↪appropriate value
df.fillna(0, inplace=True)
```

[10]:
```
df
```

[10]:

	Company	Year	Net Income	Total Assets	Total Liabilities	\
0	Apple	2021	94680	351002	287912	
1	Apple	2022	99803	352755	302083	
2	Apple	2023	96995	352583	290437	

```
3  Microsoft  2021       61271        333779             191791
4  Microsoft  2022       72738        364840             198298
5  Microsoft  2023       72361        411976             205753
6      Tesla  2021        5644         62131              30548
7      Tesla  2022       12587         82338              36440
8      Tesla  2023       14974        106618              43009

   Total Revenue  Revenue Growth (%)  Net Income Growth (%)
0         365817                 0.0                    0.0
1         394328            7.793788               5.410858
2         383285           -2.800461              -2.813543
3         168088                 0.0                    0.0
4         198270           17.956071              18.715216
5         211915             6.88203              -0.518299
6          53823                 0.0                    0.0
7          81462           51.351653              123.015592
8          96773           18.795267               18.96401
```

Next, I will calculate the average financial performance by company. This will reveal which companies generally outperform in each metric.

[11]:
```
avg_metrics_by_company = df.groupby('Company').mean(numeric_only=True)
print("Average Financial Performance by Company:")
print(avg_metrics_by_company)
```

```
Average Financial Performance by Company:
             Net Income   Total Assets  Total Liabilities  Total Revenue  \
Company
Apple      97159.333333  352113.333333      293477.333333  381143.333333
Microsoft       68790.0  370198.333333           198614.0  192757.666667
Tesla      11068.333333   83695.666667       36665.666667   77352.666667

           Revenue Growth (%)  Net Income Growth (%)
Company
Apple                1.664442               0.865772
Microsoft            8.279367               6.065639
Tesla               23.382306              47.326534
```

As we can see from the figure above, Apple has the highest average net income, significantly outpacing Microsoft and Tesla. Tesla's net income is substantially lower than both companies, reflecting its smaller size or profitability. Microsoft holds the highest average total assets, followed by Apple. Tesla's average assets are considerably lower, indicating its smaller asset base compared to the other two tech giants. Apple has the highest average liabilities, which is expected given its large size. Microsoft follows with a smaller liabilities base, while Tesla has the lowest liabilities, which could indicate a more conservative debt structure. Apple leads in total revenue by a wide margin, followed by Microsoft. Tesla's revenue is the smallest, aligning with its smaller size relative to the other two companies. Tesla exhibits the highest average revenue growth rate, which suggests rapid expansion. Microsoft shows moderate growth, while Apple's revenue growth is the lowest,

possibly indicating a more mature stage in its business cycle. Tesla has the highest net income growth, indicating significant improvement in profitability. Microsoft follows with moderate growth, while Apple's growth in net income is slower, reflecting a more stable, less aggressive expansion.

Total Revenue and Net Income over Time (by Year) across all companies.

[12]:
```
total_revenue_net_income_by_year = df.groupby('Year')[['Total Revenue', 'Net␣
 ↪Income']].sum()
print("Total Revenue and Net Income by Year:")
print(total_revenue_net_income_by_year)
```

```
Total Revenue and Net Income by Year:
      Total Revenue  Net Income
Year
2021         587728      161595
2022         674060      185128
2023         691973      184330
```

Revenue across all companies increased from $587,728 million in 2021 to $691,973 million in 2023. The most significant growth occurred between 2021 and 2022, with smaller growth between 2022 and 2023.

Comparative Analysis of Asset and Liability Ratios for each company

[13]:
```
df['Assets to Liabilities Ratio'] = df['Total Assets'] / df['Total Liabilities']

assets_liabilities_ratio_by_company = df.groupby('Company')['Assets to␣
 ↪Liabilities Ratio'].mean()
print("Mean Assets-to-Liabilities Ratio by Company:")
print(assets_liabilities_ratio_by_company)
```

```
Mean Assets-to-Liabilities Ratio by Company:
Company
Apple        1.200282
Microsoft    1.860823
Tesla        2.257467
Name: Assets to Liabilities Ratio, dtype: Float64
```

Tesla has the highest mean assets-to-liabilities ratio of 2.26, reflecting its stronger asset base relative to liabilities. Microsoft follows with a ratio of 1.86, indicating a healthy balance sheet.

Performance Summary by Company and Year

[14]:
```
summary_pivot = df.pivot_table(index='Company', columns='Year', values=['Net␣
 ↪Income', 'Total Assets', 'Total Liabilities', 'Total Revenue'])
print("Performance Summary by Company and Year:")
print(summary_pivot)
```

```
Performance Summary by Company and Year:
          Net Income                    Total Assets                      \
Year            2021     2022     2023         2021      2022      2023
```

```
Company
Apple        94680.0  99803.0  96995.0     351002.0  352755.0  352583.0
Microsoft    61271.0  72738.0  72361.0     333779.0  364840.0  411976.0
Tesla         5644.0  12587.0  14974.0      62131.0   82338.0  106618.0

          Total Liabilities                    Total Revenue            \
Year                   2021      2022      2023          2021      2022
Company
Apple              287912.0  302083.0  290437.0      365817.0  394328.0
Microsoft          191791.0  198298.0  205753.0      168088.0  198270.0
Tesla               30548.0   36440.0   43009.0       53823.0   81462.0


Year           2023
Company
Apple      383285.0
Microsoft  211915.0
Tesla       96773.0
```

Apple experienced consistent net income, assets, and liabilities over the years, with revenue peaking in 2022. Microsoft showed steady growth in assets and liabilities, with slight fluctuations in net income. Tesla displayed the most profounding growth across all metrics, reflecting its expansion in revenue, assets, and net income.

1 Task 2: Developing an AI ChatBot

Building a fully functional AI chatbot for financial analysis is a complex process involving advanced programming and deep learning techniques. However, to fit our learning objectives and time constraints, we've tailored a simplified task. This streamlined version will introduce you to the basics of chatbot development, focusing on creating a prototype that responds to predefined financial queries. It's a first step into the world of AI chatbots, offering a glimpse into their potential without the need for extensive development time or advanced technical skills. Let's begin this journey, keeping an eye on the bigger picture while we tackle this accessible task.

I will use the following outline to integrate the chatbot: 1. Preparation 2. Chatbot design and data preparation 3. Basic chatbot development

Nevertheless, lets begin!

Step 1: Preparation

I will first ensure Python and essential libraries (like pandas for data handling and Flask for a simple web application, if applicable) are installed.

[15]:
```
#Both have been installed.
```

Step 2: Chatbot Design and Data Preparation Define predefined queries: 1. "What is the total revenue?" 2. "What is the net income?" 3. "How has net income changed over the last year?" 4. "What is the revenue growth percentage?" 5. "What is the net income growth percentage?"

Responses: 1. “The total revenue for [Company] is [amount].” 2. “The net income for [Company] is [amount].” 3. “The net income has changed by [percentage] over the last year.” 4. “The revenue growth percentage for [Company] is [percentage] per year.” 5. “The net income growth percentage for [Company] is [percentage] per year.”

Step 3: Basic Chatbot Development:

[16]:
```
def financial_chatbot(user_query, df):
    if user_query == "What is the total revenue?":
        response = "The total revenue for each company:\n"
        for company in df['Company'].unique():
            total_revenue = df[df['Company'] == company]['Total Revenue'].
 ↪iloc[-1]
            response += f"{company}: {total_revenue}\n"
        return response

    elif user_query == "What is the net income?":
        response = "The net income for each company:\n"
        for company in df['Company'].unique():
            net_income = df[df['Company'] == company]['Net Income'].iloc[-1]
            response += f"{company}: {net_income}\n"
        return response

    elif user_query == "What is the revenue growth percentage?":
        response = "The revenue growth percentage for each company:\n"
        for company in df['Company'].unique():
            revenue_growth = df[df['Company'] == company]['Revenue Growth (%)'].
 ↪iloc[-1]
            response += f"{company}: {revenue_growth}%\n"
        return response

    elif user_query == "What is the net income growth percentage?":
        response = "The net income growth percentage for each company:\n"
        for company in df['Company'].unique():
            net_income_growth = df[df['Company'] == company]['Net Income Growth␣
 ↪(%)'].iloc[-1]
            response += f"{company}: {net_income_growth}%\n"
        return response

    elif user_query == "How has net income changed over the last year?":
        response = "The net income change for each company:\n"
        for company in df['Company'].unique():
            last_year = df[df['Company'] == company]['Net Income'].iloc[-2]
            this_year = df[df['Company'] == company]['Net Income'].iloc[-1]
            change = ((this_year - last_year) / last_year) * 100
            response += f"{company}: {change:.2f}%\n"
        return response
```

```
    else:
        return "Sorry, I can only provide information on predefined queries."
while True:
    user_input = input("Please ask a financial query: ")
    if user_input.lower() == 'exit':
        print("Goodbye!")
        break
    print(financial_chatbot(user_input, df))
```

```
The total revenue for each company:
Apple: 383285
Microsoft: 211915
Tesla: 96773

The net income for each company:
Apple: 96995
Microsoft: 72361
Tesla: 14974

The revenue growth percentage for each company:
Apple: -2.800460530319937%
Microsoft: 6.8820295556564215%
Tesla: 18.79526650462793%

The net income growth percentage for each company:
Apple: -2.813542679077785%
Microsoft: -0.5182985509637361%
Tesla: 18.9640104870104%

The net income growth percentage for each company:
Apple: -2.813542679077785%
Microsoft: -0.5182985509637361%
Tesla: 18.9640104870104%

Sorry, I can only provide information on predefined queries.
The net income change for each company:
Apple: -2.81%
Microsoft: -0.52%
Tesla: 18.96%

Sorry, I can only provide information on predefined queries.
Goodbye!
```

Summary of the Chatbot:

The financial_chatbot is a Python-based chatbot designed to respond to predefined financial queries using data from a provided dataframe (df). It answers five key questions: total revenue, net income, year-over-year net income change, revenue growth percentage, and net income growth percentage,

providing formatted responses with company-specific data. The chatbot retrieves and processes financial metrics dynamically, ensuring accuracy based on the dataframe's completeness, which should include columns for company names, total revenue, net income, and growth percentages. Users can interact by asking predefined questions, and the chatbot outputs structured answers; however, it cannot handle queries outside its scope or in languages other than English. It operates in a simple query-response format and includes an "exit" option to terminate the session.