Hedge Fund Financial Report

Analyzed leverage and profitability ratios across industries to provide data-driven insights for hedge fund investment strategies.

Instructions

Compute the two ratios:

- A debt-to-equity ratio or an equity multiplier ratio.
  Save this ratio in a column named `"leverage_ratio"` in a DataFrame called `df_ratios`.
- A gross margin ratio or an operating margin ratio.
  Save this ratio in a column named `"profitability_ratio"`, in a DataFrame called `df_ratios`.

The datasets have information on the type of industry a company belongs to in a column called comp_type. Your manager also needs you to answer these three questions:

1. Which company type (comp_type) has the lowest profitability ratio?
   Save this comp_type value as a string in a variable called lowest_profitability.

2. Which company type has the highest leverage ratio?
   Save this comp_type value as a string in a variable called highest_leverage.

3. What is the relationship between leverage and profitability in the real estate companies represented in this data?
   Is it "positive," "negative," or "no relationship?" Save one of these three strings in a variable called relationship.

Dataset

You have two datasets:

- `Balance_Sheet.xlsx`: Contains financial information related to assets, liabilities, and equity.
- `Income_Statement.xlsx`: Contains revenue, expenses, and profitability metrics.

Both these datasets have three columns in common:

- `"Company"`: The company's ticker name.

- `"comp_type"` The type of industry the company in question belongs to.

    - `"tech"` for companies in the technology industry
    - `"fmcg"` for companies in the fast-moving consumer goods industry
    - `"real_est"` for companies in the real estate industry.
- `"Year"`: The year the company's information is from.

The rest of the columns in the datasets contain information from the financial statement of the `"Company"` in question. Note that the columns in `Balance_Sheet.xlsx` only contain financial information from the balance sheet. Similarly, the columns in `Income_Statement.xlsx` only contain financial information from the income statement. The columns are named accordingly. For instance, the column `"Total Liab"` from `Balance_Sheet.xlsx` is the total liability.

In [49]:
```
import numpy as np
import pandas as pd
import seaborn as sns
# import openpyxl

# Read in the files
balance_sheet = pd.read_excel("../data_raw/Balance_Sheet.xlsx", index_col=0)
income_statement = pd.read_excel("../data_raw/Income_Statement.xlsx", index_
```

In [35]:
```
balance_sheet.head(10)
```

Out[35]:

	Year	comp_type	company	Accounts Payable	Cash	Inventory	Proj Equip
0	2019	tech	AAPL	46236000000	48844000000	4.106000e+09	3737800
1	2020	tech	AAPL	42296000000	38016000000	4.061000e+09	4533600
2	2021	tech	AAPL	54763000000	34940000000	6.580000e+09	4952700
3	2022	tech	AAPL	64115000000	23646000000	4.946000e+09	8423400
4	2019	tech	MSFT	9382000000	11356000000	2.063000e+09	4385600
5	2020	tech	MSFT	12530000000	13576000000	1.895000e+09	5290400
6	2021	tech	MSFT	15163000000	14224000000	2.636000e+09	7080300
7	2022	tech	MSFT	19000000000	13931000000	3.742000e+09	8754600
8	2018	tech	GOOG	4378000000	16701000000	1.107000e+09	5971900
9	2019	tech	GOOG	5561000000	18498000000	9.990000e+08	8458700

In [36]:
```
income_statement.head(10)
```

Out[36]:

	Year	comp_type	company	Cost Of Goods Sold	Gross Profit	Operating Income	C E
0	2019	tech	AAPL	161782000000	98392000000	63930000000	19624
1	2020	tech	AAPL	169559000000	104956000000	66288000000	20822
2	2021	tech	AAPL	212981000000	152836000000	108949000000	25686
3	2022	tech	AAPL	223546000000	170782000000	119437000000	27489
4	2019	tech	MSFT	42910000000	82933000000	42959000000	8288
5	2020	tech	MSFT	46078000000	96937000000	52959000000	9005
6	2021	tech	MSFT	52232000000	115856000000	69916000000	9817
7	2022	tech	MSFT	62650000000	135620000000	83383000000	11488
8	2018	tech	GOOG	59549000000	77270000000	32595000000	10422
9	2019	tech	GOOG	71896000000	89961000000	35928000000	12592

In [37]:
```
# Merge both the dataframes and call it df_ratios
df_ratios = pd.merge(income_statement, balance_sheet, on = ["Year", "company
df_ratios.head(10)
```

Out[37]:

	Year	comp_type	company	Cost Of Goods Sold	Gross Profit	Operating Income	C E
0	2019	tech	AAPL	161782000000	98392000000	63930000000	19624
1	2020	tech	AAPL	169559000000	104956000000	66288000000	20822
2	2021	tech	AAPL	212981000000	152836000000	108949000000	25686
3	2022	tech	AAPL	223546000000	170782000000	119437000000	27489
4	2019	tech	MSFT	42910000000	82933000000	42959000000	8288
5	2020	tech	MSFT	46078000000	96937000000	52959000000	9005
6	2021	tech	MSFT	52232000000	115856000000	69916000000	9817
7	2022	tech	MSFT	62650000000	135620000000	83383000000	11488
8	2018	tech	GOOG	59549000000	77270000000	32595000000	10422
9	2019	tech	GOOG	71896000000	89961000000	35928000000	12592

Key Computations

You only need to compute one profitability ratio, but since there is a choice, compute both the gross margin ratio and the operating margin ratio

- **Profitability Ratio**: Assesses a company's ability to generate profit from its revenue

- **Gross Margin Ratio** = (Total Revenue - Cost of Goods Sold) / Total Revenue
- **Operating Margin Ratio** = Operating Income / Total Revenue

In [38]:
```
# Compute gross margin ratio
df_ratios["profitability_ratio"] = (df_ratios["Total Revenue"] - df_ratios["
df_ratios.head(10)
```

Out[38]:

	Year	comp_type	company	Cost Of Goods Sold	Gross Profit	Operating Income	C E
0	2019	tech	AAPL	161782000000	98392000000	63930000000	19624
1	2020	tech	AAPL	169559000000	104956000000	66288000000	20822
2	2021	tech	AAPL	212981000000	152836000000	108949000000	25686
3	2022	tech	AAPL	223546000000	170782000000	119437000000	27489
4	2019	tech	MSFT	42910000000	82933000000	42959000000	8288
5	2020	tech	MSFT	46078000000	96937000000	52959000000	9005
6	2021	tech	MSFT	52232000000	115856000000	69916000000	9817
7	2022	tech	MSFT	62650000000	135620000000	83383000000	11488
8	2018	tech	GOOG	59549000000	77270000000	32595000000	10422
9	2019	tech	GOOG	71896000000	89961000000	35928000000	12592

In [39]:
```
# Compute operating margin ratio, but commenting it out
df_ratios["profitability_ratio_2"] = (df_ratios["Total Revenue"] - df_ratios
df_ratios.head(10)
```

Out[39]:

	Year	comp_type	company	Cost Of Goods Sold	Gross Profit	Operating Income	C E
0	2019	tech	AAPL	161782000000	98392000000	63930000000	19624
1	2020	tech	AAPL	169559000000	104956000000	66288000000	20822
2	2021	tech	AAPL	212981000000	152836000000	108949000000	25686
3	2022	tech	AAPL	223546000000	170782000000	119437000000	27489
4	2019	tech	MSFT	42910000000	82933000000	42959000000	8288
5	2020	tech	MSFT	46078000000	96937000000	52959000000	9005
6	2021	tech	MSFT	52232000000	115856000000	69916000000	9817
7	2022	tech	MSFT	62650000000	135620000000	83383000000	11488
8	2018	tech	GOOG	59549000000	77270000000	32595000000	10422
9	2019	tech	GOOG	71896000000	89961000000	35928000000	12592

You only need to compute one leverage ratio, but we are providing the code to compute both the debt-to-equity ratio and the equity multiplier ratio

- **Leverage Ratio**: Measures financial leverage using one of the following formulas
  - **Debt-to-Equity Ratio** = Total Liabilities / Total Stockholders' Equity
  - **Equity Multiplier Ratio** = Total Assets / Total Stockholders' Equity

In [40]:
```
# Compute debt-to-equity ratio
df_ratios["leverage_ratio"] = df_ratios["Total Liab"]/df_ratios["Total Stock
df_ratios.head(10)
```

Out[40]:

	Year	comp_type	company	Cost Of Goods Sold	Gross Profit	Operating Income	C E
0	2019	tech	AAPL	161782000000	98392000000	63930000000	19624
1	2020	tech	AAPL	169559000000	104956000000	66288000000	20822
2	2021	tech	AAPL	212981000000	152836000000	108949000000	25686
3	2022	tech	AAPL	223546000000	170782000000	119437000000	27489
4	2019	tech	MSFT	42910000000	82933000000	42959000000	8288
5	2020	tech	MSFT	46078000000	96937000000	52959000000	9005
6	2021	tech	MSFT	52232000000	115856000000	69916000000	9817
7	2022	tech	MSFT	62650000000	135620000000	83383000000	11488
8	2018	tech	GOOG	59549000000	77270000000	32595000000	10422
9	2019	tech	GOOG	71896000000	89961000000	35928000000	12592

10 rows × 21 columns

In [41]:
```
# Compute equity multiplier ratio, but commenting it out
df_ratios["leverage_ratio_2"] = df_ratios["Total Assets"]/df_ratios["Total S
df_ratios.head(10)
```

Out[41]:

	Year	comp_type	company	Cost Of Goods Sold	Gross Profit	Operating Income	C E
0	2019	tech	AAPL	161782000000	98392000000	63930000000	19624
1	2020	tech	AAPL	169559000000	104956000000	66288000000	20822
2	2021	tech	AAPL	212981000000	152836000000	108949000000	25686
3	2022	tech	AAPL	223546000000	170782000000	119437000000	27489
4	2019	tech	MSFT	42910000000	82933000000	42959000000	8288
5	2020	tech	MSFT	46078000000	96937000000	52959000000	9005
6	2021	tech	MSFT	52232000000	115856000000	69916000000	9817
7	2022	tech	MSFT	62650000000	135620000000	83383000000	11488
8	2018	tech	GOOG	59549000000	77270000000	32595000000	10422
9	2019	tech	GOOG	71896000000	89961000000	35928000000	12592

10 rows × 22 columns

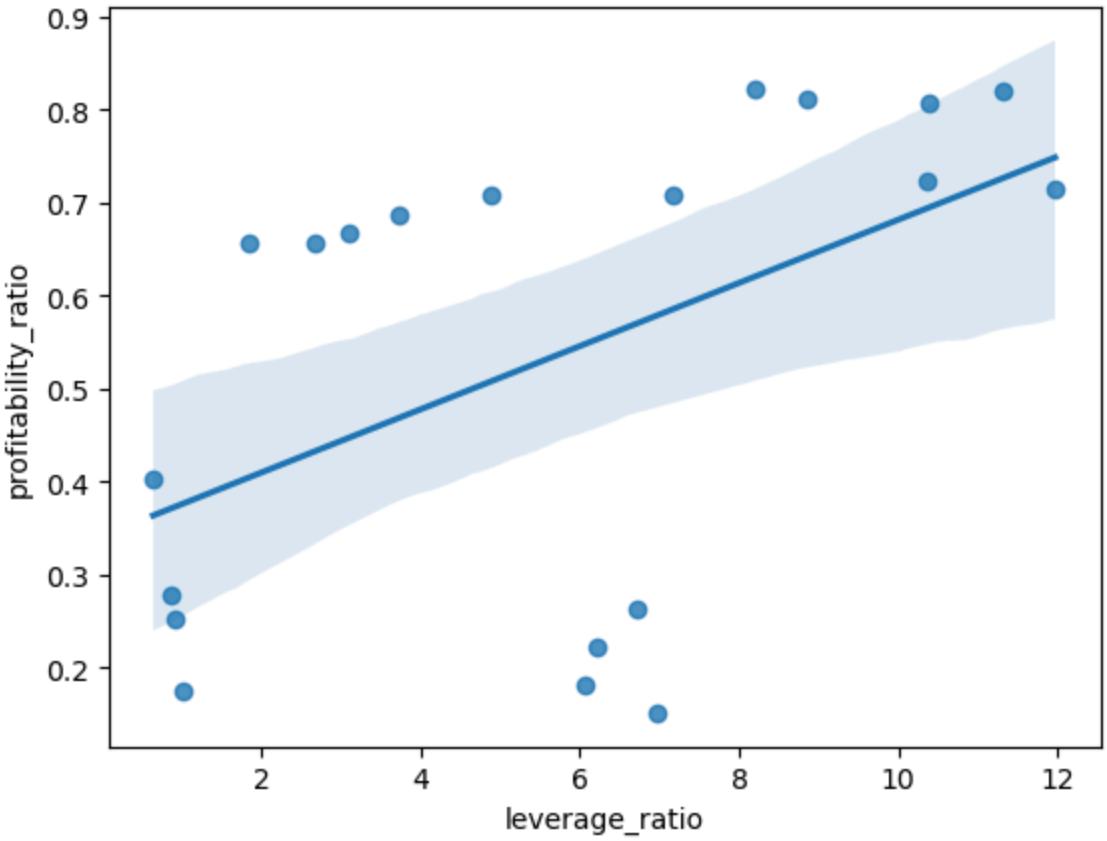
In [45]:
```
# Using pivot table to see the "comp_type" with the lowest average profitabi
print(df_ratios.pivot_table(index="comp_type", values="profitability_ratio")

# Using pivot table to see the "comp_type" with the highest average leverage
print(df_ratios.pivot_table(index="comp_type", values="leverage_ratio"))
```

```
           profitability_ratio
comp_type
fmcg                  0.514396
real_est              0.534848
tech                  0.572062
           leverage_ratio
comp_type
fmcg             2.997896
real_est         5.692041
tech             1.777448
```

In [46]:
```
lowest_profitability = "fmcg"
highest_leverage = "real_est"
```

In [47]:
```
# Plot the leverage ratio on x-axis and profitability on y axis to see if re
df_real_est = df_ratios.loc[df_ratios["comp_type"]=="real_est"]
plot = sns.regplot(data=df_real_est, x="leverage_ratio", y="profitability_ra
```



In [ ]:
```
relationship = "positive"
```

In [48]:
```
# jupyter nbconvert --to html "Banking Investment Optimization Framework/not
```