INFO 2950: Phase II

Group Members: Anusha Bishayee, Katheryn Ding

Research Question:

How do ESG score and stock performance (price) align across different industries? What associations can we find between company industry, stock performance, and ESG ratings?

note: ESG score refers to a quantiative metric measuring a company's environmental, social, and governance performance; 'environmental' pertains to aspects like waste management and energy emissions, 'social' pertains to aspects like customer satisfaction and DEI in the workplace, and 'governance' pertains to aspects like operating efficiencies and risk management. ESG scores are typically examined by independent investors, business analysts, and even competitior companies to assess risk or opportunities associated with a specific company's practices.

Data Collection and Cleaning:

In [27]:
```
import contextlib
import os
import sys
import warnings

import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import seaborn as sns
import statsmodels.api as sm
import yfinance as yf
from scipy import stats
from sklearn.linear_model import LinearRegression, LogisticRegression
from sklearn.metrics import accuracy_score, root_mean_squared_error, r2_scor
from sklearn.model_selection import train_test_split
```

our original dataset with ESG information for different large/mid-cap companies came in a csv format, which we downloaded from Kaggle. this had about 722 rows, each corresponding to a unique publicly traded company. further description of the columns here can be found in the 'Dataset Description' portion of this notebook.

we first dropped all rows that had null values, which eliminated 27 companies. we then filtered this dataset for just USD currency, excluding companies that are traded in CNY or any other currency. this allows us to have greater familiarity with the industries and companies we analyze - this process eliminated 15 more of our rows, and left us with 680 companies.

In [2]:
```
esg = pd.read_csv("esg_data.csv")
print(f"original data shape: {esg.shape}")

esg = esg.dropna()
print(f"non-null data shape: {esg.shape}")

esg = esg[esg["currency"] == "USD"]
print(f"refined data shape: {esg.shape}")
```

```
original data shape: (722, 21)
non-null data shape: (695, 21)
refined data shape: (680, 21)
```

then, we converted the 'last_processing_date' column in our dataset to DateTime format - a lot of rows had a differing date formats as well, so we had to convert them all to m/d/y. after that, we sorted the dataset by ascending and descending 'last_processing_date' to see the range of processing dates in the data.

In [3]:
```
esg["last_processing_date"] = pd.to_datetime(esg["last_processing_date"], fo
esg["last_processing_date"] = esg["last_processing_date"].dt.strftime('%m-%d

esg = esg.sort_values(by = "last_processing_date", ascending = False)
print(f"latest dates:\n{esg["last_processing_date"].head(2)}")

esg = esg.sort_values(by = "last_processing_date", ascending = True)
print(f"\nearliest dates:\n{esg["last_processing_date"].head(2)}")
```

```
latest dates:
720    11-15-2022
716    11-15-2022
Name: last_processing_date, dtype: object

earliest dates:
658    02-08-2022
36     04-16-2022
Name: last_processing_date, dtype: object
```

after this, we realized we have two 'Energy' values for the 'industry' column - one is 'Energy ' and one is 'Energy'. we renamed all the 'Energy ' values, and also re-formatted some other industry column values.

In [4]:
```
esg['industry'] = esg['industry'].replace('Energy ', 'Energy')
esg['industry'] = esg['industry'].replace('Hotels, Restaurants & Leisure', '
esg['industry'] = esg['industry'].replace('Hotels Restaurants and Leisure',
esg['industry'] = esg['industry'].replace('Consumer products', 'Consumer Pro
esg['industry'] = esg['industry'].replace('Logistics and Transportation', 'L
```

```
esg['industry'] = esg['industry'].replace('Life Sciences Tools and Services'
esg['industry'] = esg['industry'].replace('Commercial Services and Supplies'
esg['industry'] = esg['industry'].replace('Road and Rail', 'Road & Rail')
esg['industry'] = esg['industry'].replace('Metals and Mining', 'Metals & Mir
esg['industry'] = esg['industry'].replace('Aerospace and Defense', 'Aerospac
esg['industry'] = esg['industry'].replace('Textiles Apparel and Luxury Goods
esg['industry'] = esg['industry'].replace('Trading Companies and Distributor

print(esg['industry'].unique())
```

```
['Leisure Products' 'Semiconductors' 'Health Care' 'Chemicals'
 'Telecommunication' 'Consumer Products' 'Airlines' 'Insurance'
 'Communications' 'Building' 'Technology' 'Logistics & Transportation'
 'Biotechnology' 'Banking' 'Pharmaceuticals' 'Financial Services'
 'Life Sciences Tools & Services' 'Electrical Equipment' 'Real Estate'
 'Machinery' 'Retail' 'Food Products' 'Industrial Conglomerates'
 'Hotels, Restaurants, & Leisure' 'Utilities' 'Beverages' 'Tobacco'
 'Media' 'Auto Components' 'Energy' 'Commercial Services & Supplies'
 'Packaging' 'Road & Rail' 'Metals & Mining'
 'Textiles, Apparel, & Luxury Goods' 'Trading Companies & Distributors'
 'Aerospace & Defense' 'Automobiles' 'Distributors'
 'Professional Services' 'Construction' 'Marine'
 'Diversified Consumer Services']
```

now, we wanted to add our finance data from the yfinance library onto to our esg dataset. we used the ticker column to match up companies from the yfinance library and our esg dataset, and we set our dates of the finance data to range from 2/1/21 to 12/31/22, as all of the 'last processing date' values for the esg data range from 2/8/22 to 11/15/22. in specific, we calculated a stock percentage change over this period for each company, a volatility index, a 50-day moving average, and a cumulative return metric.

In [5]:

```
# prevents some annoying yfinance outputs from printing, gpt was used to ass
@contextlib.contextmanager
def suppress_output():
    with open(os.devnull, 'w') as devnull:
        old_stdout = sys.stdout
        old_stderr = sys.stderr
        sys.stdout = devnull
        sys.stderr = devnull
        try:
            yield
        finally:
            sys.stdout = old_stdout
            sys.stderr = old_stderr

tickers = esg["ticker"].tolist()
stock_data = []

for ticker in tickers:
    try:
        # suppresses all of the outputs when grabbing data from yfinance
        with suppress_output():
```

```
            stock = yf.download(ticker, start = "2021-02-01", end = "2022-12

        if not stock.empty:
            # get closing price for 01/01/2021 and 12/31/2022
            close_2021_02_01 = stock.loc["2021-02-01"]["Close"] if "2021-02-
            close_2022_12_31 = stock.loc["2022-12-30"]["Close"] if "2022-12-

            # calculating percentage change
            percentage_change = ((stock["Close"].iloc[-1] - stock["Close"].i

            # calculating volatility (sd of daily returns)
            daily_returns = stock["Close"].pct_change()
            volatility = daily_returns.std()

            # calculating 50-day moving average
            stock["50_day_SMA"] = stock["Close"].rolling(window=50).mean()
            sma_50_day = stock["50_day_SMA"].iloc[-1]

            # calculating cumulative return
            cumulative_return = (stock["Close"].iloc[-1] / stock["Close"].il

            stock_data.append({
                'ticker': ticker,
                'start_close': close_2021_02_01,
                'end_close': close_2022_12_31,
                'percentage_change': percentage_change,
                'volatility': volatility,
                '50_day_SMA': sma_50_day,
                'cumulative_return': cumulative_return
            })

    # also helps to suppress annoying outputs
    except (yf.YFTzMissingError, yf.YFPricesMissingError):
        pass
```

now, we needed to convert the stock data we extracted from yfinance to a dataframe, so that we can merge it with our original esg dataframe.

In [6]:
```
stock_df = pd.DataFrame(stock_data)
merged_df = esg.merge(stock_df, on = 'ticker', how = 'left')
print(f"current data shape: {merged_df.shape}")
print(f"\n{merged_df.head()}")
```

```
current data shape: (680, 27)

  ticker                           name currency  \
0   poww                       Ammo Inc      USD
1   acls        Axcelis Technologies Inc      USD
2   achc  Acadia Healthcare Company Inc      USD
3     cf     CF Industries Holdings Inc      USD
4      t                       AT&T Inc      USD

                         exchange           industry  \
0     NASDAQ NMS - GLOBAL MARKET   Leisure Products
1     NASDAQ NMS - GLOBAL MARKET     Semiconductors
2     NASDAQ NMS - GLOBAL MARKET        Health Care
3  NEW YORK STOCK EXCHANGE, INC.          Chemicals
4  NEW YORK STOCK EXCHANGE, INC.  Telecommunication

                                                logo  \
0  https://static.finnhub.io/logo/8decc6ca0564a89...
1  https://static.finnhub.io/logo/88b5f730-80df-1...
2  https://static.finnhub.io/logo/4b6b2e5a4cfce5b...
3  https://static.finnhub.io/logo/9b57a636-80eb-1...
4  https://static.finnhub.io/logo/7d20269e-80ec-1...

                            weburl environment_grade environment_level  \
0              https://ammoinc.com/                 B            Medium
1          https://www.axcelis.com/                 A              High
2  https://www.acadiahealthcare.com/               BB            Medium
3      https://www.cfindustries.com/                 A              High
4              https://www.att.com/                 B            Medium

  social_grade  ... last_processing_date total_grade total_level      cik  \
0            B  ...           02-08-2022           B      Medium  1015383
1           BB  ...           04-16-2022         BBB        High  1113232
2            B  ...           04-16-2022          BB      Medium  1520697
3           BB  ...           04-16-2022         BBB        High  1324404
4            B  ...           04-16-2022           B      Medium   732717

   start_close  end_close  percentage_change volatility 50_day_SMA  \
0     5.400000   1.730000         -67.962963   0.040577     2.3782
1    36.150002  79.360001         119.529730   0.038259    73.6454
2    52.990002  82.320000          55.350061   0.021303    82.6706
3    42.970001  85.199997          98.277856   0.027433   101.2440
4    21.638973  18.410000         -14.922027   0.015132    18.5720

  cumulative_return
0         -0.679630
1          1.195297
2          0.553501
3          0.982779
4         -0.149220

[5 rows x 27 columns]
```

after that, we needed to drop any rows where the finance data join has left null values.

this eliminates 80 more of our rows, leaving us with 600 companies.

In [7]:
```python
merged_df = merged_df.dropna()
print(f"non-null finance data shape: {merged_df.shape}\n")

merged_df = merged_df.sort_values(by = "name", ascending = True)
print(merged_df.head)
```

```
non-null finance data shape: (599, 27)

<bound method NDFrame.head of     ticker                                    name cu
rrency  \
71     mmm                               3M Co      USD
142    aos                       A O Smith Corp      USD
661   abvc                   ABVC Biopharma Inc      USD
17    acad          ACADIA Pharmaceuticals Inc      USD
41    aciw                   ACI Worldwide Inc      USD
..     ...                                 ...      ...
500    zts                          Zoetis Inc      USD
571    zuo                           Zuora Inc      USD
643    zws   Zurn Elkay Water Solutions Corp      USD
647   zyme                       Zymeworks Inc      USD
189   ebay                            eBay Inc      USD

                          exchange                     industry  \
71   NEW YORK STOCK EXCHANGE, INC.  Industrial Conglomerates
142  NEW YORK STOCK EXCHANGE, INC.                     Building
661     NASDAQ NMS - GLOBAL MARKET                Biotechnology
17      NASDAQ NMS - GLOBAL MARKET                Biotechnology
41      NASDAQ NMS - GLOBAL MARKET                   Technology
..                             ...                          ...
500  NEW YORK STOCK EXCHANGE, INC.              Pharmaceuticals
571  NEW YORK STOCK EXCHANGE, INC.                   Technology
643  NEW YORK STOCK EXCHANGE, INC.                     Building
647  NEW YORK STOCK EXCHANGE, INC.                Biotechnology
189     NASDAQ NMS - GLOBAL MARKET                       Retail

                                                  logo  \
71   https://static.finnhub.io/logo/2a1802fa-80ec-1...
142  https://static.finnhub.io/logo/73381be8-80eb-1...
661  https://static.finnhub.io/logo/2b1c9fbcfafa70d...
17   https://static.finnhub.io/logo/2d87da57d47f7a5...
41   https://static.finnhub.io/logo/875c5a76-80df-1...
..                                                 ...
500  https://static.finnhub.io/logo/aae505a6-80cd-1...
571  https://static.finnhub.io/logo/181c3c26-80db-1...
643  https://static.finnhub.io/logo/166822dc-80c9-1...
647  https://static.finnhub.io/logo/d5054e357d522c9...
189  https://static.finnhub.io/logo/919a6270-826a-1...

                          weburl environment_grade environment_level  \
71             https://www.3m.com/                 A              High
142        https://www.aosmith.com/                 A              High
```

```
661          https://abvcpharma.com/                 B           Medium
17              https://acadia.com/                  B           Medium
41     https://www.aciworldwide.com/                 A             High
..                               ...               ...              ...
500          https://www.zoetis.com/                 A             High
571           https://www.zuora.com/                 B           Medium
643  https://zurnwatersolutions.com/                 A             High
647       https://www.zymeworks.com/                 B           Medium
189         https://www.ebayinc.com/                 A             High

    social_grade  ... last_processing_date total_grade total_level      cik  \
71            BB  ...           04-16-2022         BBB        High    66740
142           BB  ...           04-16-2022         BBB        High    91142
661            B  ...           11-06-2022           B      Medium  1173313
17             B  ...           04-16-2022          BB      Medium  1070494
41            BB  ...           04-16-2022         BBB        High   935036
..           ...  ...                  ...         ...         ...      ...
500           BB  ...           04-20-2022         BBB        High  1555280
571            B  ...           09-24-2022           B      Medium  1423774
643           BB  ...           11-06-2022         BBB        High  1439288
647           BB  ...           11-06-2022          BB      Medium  1403752
189           BB  ...           04-17-2022         BBB        High  1065088

     start_close   end_close  percentage_change volatility  50_day_SMA  \
71    146.070236  100.267555         -31.356615   0.014472  104.066388
142    56.029999   57.240002           2.159563   0.018196   57.343600
661    50.000000    6.250000         -87.500000   0.089921    7.173200
17     46.580002   15.920000         -65.822242   0.045374   15.391400
41     39.590000   23.000000         -41.904522   0.021739   21.438400
..           ...         ...                ...        ...         ...
500   155.580002  146.550003          -5.804087   0.016947  147.633001
571    14.240000    6.360000         -55.337077   0.034177    7.057200
643    38.889999   21.150000         -45.615840   0.032776   23.315800
647    35.639999    7.860000         -77.946127   0.049518    7.356000
189    58.470001   41.470001         -29.074739   0.022532   42.343200

    cumulative_return
71          -0.313566
142          0.021596
661         -0.875000
17          -0.658222
41          -0.419045
..                ...
500         -0.058041
571         -0.553371
643         -0.456158
647         -0.779461
189         -0.290747

[599 rows x 27 columns]>
```

this is still a lot of data, but will be helpful for getting industry-level and other general overviews of the data. merged_df will be our main dataframe.

we also want to create a sample of these 600 companies so that we are able to look at trends and associations at the individual company-level as well.

since we want to take random sample stratum on industries and total ESG score:

1. We first list all industries with descending total_score, then separate the industries based on total_score into three groups: high/medium/low ESG based on their rank.
2. Then, we use .sample() to randomly choose two industries from each group, which generates 6 industries.
3. For each of the randomly chosen industries, we randomly pick 5 companies from each industry, which generates 30 companies in the sample_companies.

By applying this sampling process, we ensure our sample by industries is representative of all levels of ESG scores. gpt was used here for assistance with the sampling. sample_companies will be our 2nd dataframe.

In [32]:
```
warnings.filterwarnings("ignore", category = DeprecationWarning)
np.random.seed(123)

avg_esg_by_industry = merged_df.groupby('industry')['total_score'].mean().re
avg_esg_by_industry.columns = ['Industry', 'Average Total ESG Score']
avg_esg_by_industry = avg_esg_by_industry.sort_values(by = 'Average Total ES

third = len(avg_esg_by_industry) // 3
high_group = avg_esg_by_industry.iloc[:third]
medium_group = avg_esg_by_industry.iloc[third: 2 * third]
low_group = avg_esg_by_industry.iloc[2 * third:]

random_high_industries = high_group.sample(2)['Industry'].tolist()
random_medium_industries = medium_group.sample(2)['Industry'].tolist()
random_low_industries = low_group.sample(2)['Industry'].tolist()

print(f"randomly selected high ESG industries: {random_high_industries}")
print(f"randomly selected medium ESG industries: {random_medium_industries}"
print(f"randomly selected low ESG industries: {random_low_industries}")

high_industry_companies = merged_df[merged_df['industry'].isin(random_high_i
medium_industry_companies = merged_df[merged_df['industry'].isin(random_medi
low_industry_companies = merged_df[merged_df['industry'].isin(random_low_inc

high_industry_companies_sampled = high_industry_companies.groupby('industry'
high_industry_companies_sampled['ESG score level'] = 'High'
medium_industry_companies_sampled = medium_industry_companies.groupby('indus
medium_industry_companies_sampled['ESG score level'] = 'Medium'
low_industry_companies_sampled = low_industry_companies.groupby('industry',
low_industry_companies_sampled['ESG score level'] = 'Low'

sample_companies = pd.concat([high_industry_companies_sampled, medium_indust
```

```
print(sample_companies)
```

```
randomly selected high ESG industries: ['Building', 'Road & Rail']
randomly selected medium ESG industries: ['Professional Services', 'Real Est
ate']
randomly selected low ESG industries: ['Biotechnology', 'Banking']
  ticker                               name currency  \
0     tt           Trane Technologies PLC      USD
1    mas                         Masco Corp      USD
2    zws   Zurn Elkay Water Solutions Corp      USD
3    aos                      A O Smith Corp      USD
4   aaon                           Aaon Inc      USD
5    xpo                  XPO Logistics Inc      USD
6    unp                  Union Pacific Corp      USD
7    csx                           CSX Corp      USD
8    nsc               Norfolk Southern Corp      USD
9   odfl      Old Dominion Freight Line Inc      USD
0   ldos                Leidos Holdings Inc      USD
1    efx                        Equifax Inc      USD
2   upwk                         Upwork Inc      USD
3   vrsk              Verisk Analytics Inc      USD
4    rhi      Robert Half International Inc      USD
5   well                  Welltower OP LLC      USD
6    amt                American Tower Corp      USD
7    vtr                         Ventas Inc      USD
8    dlr             Digital Realty Trust Inc      USD
9   eqix                        Equinix Inc      USD
0    cfg       Citizens Financial Group Inc      USD
1    mtb                      M&T Bank Corp      USD
2    tfc               Truist Financial Corp      USD
3    pnc  PNC Financial Services Group Inc      USD
4    hbt                 HBT Financial Inc      USD
5   mrna                        Moderna Inc      USD
6   adma                ADMA Biologics Inc      USD
7   urgn                  Urogen Pharma Ltd      USD
8   acet                    Adicet Bio Inc      USD
9   abus             Arbutus Biopharma Corp      USD

                        exchange               industry  \
0  NEW YORK STOCK EXCHANGE, INC.               Building
1  NEW YORK STOCK EXCHANGE, INC.               Building
2  NEW YORK STOCK EXCHANGE, INC.               Building
3  NEW YORK STOCK EXCHANGE, INC.               Building
4     NASDAQ NMS - GLOBAL MARKET               Building
5  NEW YORK STOCK EXCHANGE, INC.            Road & Rail
6  NEW YORK STOCK EXCHANGE, INC.            Road & Rail
7     NASDAQ NMS - GLOBAL MARKET            Road & Rail
8  NEW YORK STOCK EXCHANGE, INC.            Road & Rail
9     NASDAQ NMS - GLOBAL MARKET            Road & Rail
0  NEW YORK STOCK EXCHANGE, INC.  Professional Services
1  NEW YORK STOCK EXCHANGE, INC.  Professional Services
2     NASDAQ NMS - GLOBAL MARKET  Professional Services
3     NASDAQ NMS - GLOBAL MARKET  Professional Services
4  NEW YORK STOCK EXCHANGE, INC.  Professional Services
```

```
5  NEW YORK STOCK EXCHANGE, INC.           Real Estate
6  NEW YORK STOCK EXCHANGE, INC.           Real Estate
7  NEW YORK STOCK EXCHANGE, INC.           Real Estate
8  NEW YORK STOCK EXCHANGE, INC.           Real Estate
9     NASDAQ NMS - GLOBAL MARKET           Real Estate
0  NEW YORK STOCK EXCHANGE, INC.               Banking
1  NEW YORK STOCK EXCHANGE, INC.               Banking
2  NEW YORK STOCK EXCHANGE, INC.               Banking
3  NEW YORK STOCK EXCHANGE, INC.               Banking
4     NASDAQ NMS - GLOBAL MARKET               Banking
5     NASDAQ NMS - GLOBAL MARKET         Biotechnology
6     NASDAQ NMS - GLOBAL MARKET         Biotechnology
7     NASDAQ NMS - GLOBAL MARKET         Biotechnology
8     NASDAQ NMS - GLOBAL MARKET         Biotechnology
9     NASDAQ NMS - GLOBAL MARKET         Biotechnology

                                                  logo  \
0  https://static.finnhub.io/logo/a16640a1e06f411...
1  https://static.finnhub.io/logo/f733abfaa82424a...
2  https://static.finnhub.io/logo/166822dc-80c9-1...
3  https://static.finnhub.io/logo/73381be8-80eb-1...
4  https://static.finnhub.io/logo/8819421c-80df-1...
5  https://static.finnhub.io/logo/67e2a608-826a-1...
6  https://static.finnhub.io/logo/92d19634-80ec-1...
7  https://static.finnhub.io/logo/b034979a-80eb-1...
8  https://static.finnhub.io/logo/40989d8c-80ec-1...
9  https://static.finnhub.io/logo/8b7ba59a5161457...
0  https://static.finnhub.io/logo/6a70aab0-80ec-1...
1  https://static.finnhub.io/logo/ca2b7d6c-80eb-1...
2  https://static.finnhub.io/logo/04e63e39dfa2f09...
3  https://static.finnhub.io/logo/fe905336-80fe-1...
4  https://static.finnhub.io/logo/611b69c8-80ec-1...
5  https://static.finnhub.io/logo/f36a2b02-80eb-1...
6  https://static.finnhub.io/logo/72058208-80eb-1...
7  https://static.finnhub.io/logo/e63599b4-8279-1...
8  https://static.finnhub.io/logo/c066b1b0-80eb-1...
9  https://static.finnhub.io/logo/2905ab26-80e0-1...
0  https://static.finnhub.io/logo/7aa22fa8-80d0-1...
1  https://static.finnhub.io/logo/30d4609c-80ec-1...
2  https://static.finnhub.io/logo/50ba697e87554ae...
3  https://static.finnhub.io/logo/595585fe-80ec-1...
4  https://static.finnhub.io/logo/ce6355c1d526b56...
5  https://static.finnhub.io/logo/a024deee-80da-1...
6  https://static.finnhub.io/logo/cba1b436-80d0-1...
7  https://static.finnhub.io/logo/34ed2f05d06c4b4...
8  https://static.finnhub.io/logo/ea701be36180b02...
9  https://static.finnhub.io/logo/0068df02-80ca-1...

                                       weburl environment_grade  \
0              https://www.tranetechnologies.com/                AA
1                           https://masco.com/                 A
2             https://zurnwatersolutions.com/                 A
3                       https://www.aosmith.com/                 A
```

```
4                                https://www.aaon.com/                 A
5                                 https://www.xpo.com/                 A
6                       https://www.up.com/index.htm                   B
7                                 https://www.csx.com/                 A
8                               http://www.nscorp.com/                BB
9                                https://www.odfl.com/               BBB
0                              https://www.leidos.com/                 A
1                             https://www.equifax.com/                BB
2                              https://www.upwork.com/                 B
3                               https://www.verisk.com                AA
4                           https://www.roberthalf.com/                B
5                               https://welltower.com/               BBB
6                       https://www.americantower.com/                 A
7                           https://www.ventasreit.com/                A
8                        https://www.digitalrealty.com                 A
9                             https://www.equinix.com/                 A
0                        https://www.citizensbank.com/                 B
1                                 https://www.mtb.com/                 A
2                              https://www.truist.com/               BBB
3                                 https://www.pnc.com/               BBB
4  https://ir.hbtfinancial.com/investor-relations                      B
5                            https://www.modernatx.com/                A
6                        https://www.admabiologics.com/                B
7                              https://www.urogen.com/                 B
8                            https://www.adicetbio.com/                B
9                           https://www.arbutusbio.com/                B

  environment_level social_grade  ... total_level      cik start_close  \
0         Excellent           BB  ...        High  1466258  142.750000
1              High           BB  ...        High    62996   55.080002
2              High           BB  ...        High  1439288   38.889999
3              High           BB  ...        High    91142   56.029999
4              High           BB  ...        High   824142   49.973331
5              High           BB  ...        High  1166003   39.738617
6            Medium            B  ...      Medium   100885  198.600006
7              High           BB  ...        High   277948   29.016666
8            Medium           BB  ...        High   702165  239.910004
9              High          BBB  ...        High   878927   99.550003
0              High           BB  ...        High  1336920  105.059998
1            Medium           BB  ...        High    33185  180.309998
2            Medium            B  ...      Medium  1627475   45.110001
3         Excellent           BB  ...        High  1442145  185.940002
4            Medium           BB  ...      Medium   315213   68.699997
5              High           BB  ...        High   766704   62.509998
6              High           BB  ...        High  1053507  235.570007
7              High           BB  ...        High   740260   47.560001
8              High           BB  ...        High  1297996  148.210007
9              High           BB  ...        High  1101239  756.719971
0            Medium            B  ...      Medium   759944   37.209999
1              High           BB  ...        High    36270  132.949997
2              High           BB  ...        High    92230   48.380001
3              High           BB  ...        High   713676  145.860001
4            Medium            B  ...      Medium   775215   15.060000
```

```
5            High          BB  ...       High  1682852  157.479996
6          Medium           B  ...     Medium  1368514    2.450000
7          Medium           B  ...     Medium  1668243   21.870001
8          Medium           B  ...     Medium  1720580   12.120000
9          Medium           B  ...     Medium  1447028    3.880000

    end_close  percentage_change  volatility  50_day_SMA cumulative_return
\
0  168.089996          17.751311    0.017016  169.433000          0.177513
1   46.669998         -15.268706    0.018719   48.071000         -0.152687
2   21.150000         -45.615840    0.032776   23.315800         -0.456158
3   57.240002           2.159563    0.018196   57.343600          0.021596
4   50.213333           0.480260    0.020646   49.237200          0.004803
5   33.290001         -16.227580    0.028413   34.973681         -0.162276
6  207.070007           4.264854    0.014568  207.042801          0.042649
7   30.980000           6.766226    0.015786   30.759800          0.067662
8  246.419998           2.713515    0.016159  241.700600          0.027135
9  141.889999          42.531386    0.021535  144.182400          0.425314
0  105.190002           0.123743    0.015973  105.201200          0.001237
1  194.360001           7.792138    0.021956  186.466199          0.077921
2   10.440000         -76.856574    0.044098   12.001400         -0.768566
3  176.419998          -5.119933    0.016003  176.626599         -0.051199
4   73.830002           7.467256    0.019270   75.301000          0.074673
5   65.550003           4.863230    0.016966   65.386000          0.048632
6  211.860001         -10.064951    0.016860  210.900200         -0.100650
7   45.049999          -5.277548    0.017761   43.266000         -0.052775
8  100.269997         -32.346001    0.018200  104.035600         -0.323460
9  655.030029         -13.438253    0.019535  637.711002         -0.134383
0   39.369999           5.804891    0.021152   39.932600          0.058049
1  145.059998           9.108688    0.020756  159.723800          0.091087
2   43.029999         -11.058293    0.019116   43.858000         -0.110583
3  157.940002           8.281915    0.017795  158.468999          0.082819
4   19.570000          29.946874    0.017339   20.027400          0.299469
5  179.619995          14.058928    0.048083  172.877399          0.140589
6    3.880000          58.367349    0.041680    3.206800          0.583673
7    8.870000         -59.442160    0.043236    9.235600         -0.594422
8    8.940000         -26.237627    0.054050   15.558900         -0.262376
9    2.330000         -39.948457    0.046707    2.484200         -0.399485

  stock_increase ESG score level
0              1            High
1              0            High
2              0            High
3              1            High
4              1            High
5              0            High
6              1            High
7              1            High
8              1            High
9              1            High
0              1          Medium
1              1          Medium
2              0          Medium
```

```
3               0          Medium
4               1          Medium
5               1          Medium
6               0          Medium
7               0          Medium
8               0          Medium
9               0          Medium
0               1             Low
1               1             Low
2               0             Low
3               1             Low
4               1             Low
5               1             Low
6               1             Low
7               0             Low
8               0             Low
9               0             Low

[30 rows x 29 columns]
```

finally, we also extracted some general S&P 500 data from yfinance, ranging from the dates of 2/1/21 and 12/31/22 for the same reason. we are pulling this data so that we can compare stock performance of the individual companies to the overall S&P 500 in the same time range. sp500 will be our 3rd dataset.

In [9]:

```python
sp500data = yf.download('^GSPC', start = '2021-02-01', end = '2022-12-31', p
sp500 = pd.DataFrame({
    'Date': sp500data.index,
    'Start Price': sp500data['Open'],
    'End Price': sp500data['Close'],
    'Rate of Change': ((sp500data['Close'] - sp500data['Open']) / sp500data[

sp500.set_index('Date', inplace = True)
print(sp500)
```

```
            Start Price    End Price  Rate of Change
Date
2021-02-01  3731.169922  3773.860107        1.144150
2021-02-02  3791.840088  3826.310059        0.909057
2021-02-03  3840.270020  3830.169922       -0.263005
2021-02-04  3836.659912  3871.739990        0.914339
2021-02-05  3878.300049  3886.830078        0.219942
...                 ...          ...             ...
2022-12-23  3815.110107  3844.820068        0.778745
2022-12-27  3843.340088  3829.250000       -0.366610
2022-12-28  3829.560059  3783.219971       -1.210063
2022-12-29  3805.449951  3849.280029        1.151771
2022-12-30  3829.060059  3839.500000        0.272650

[484 rows x 3 columns]
```

Data Description

1. Where can your raw source data be found, if applicable? Provide a link to the raw data (hosted on Github, in a Cornell Google Drive or Cornell Box).

- We have 3 main datasets: 1 main dataset (merged_df), 1 "sample" dataset that selects 30 rows from this main dataset (sample_companies), and 1 additional dataset (sp500). Our raw data for the first 2 datasets can be found on Kaggle, here: https://github.com/anushabishayee/info2950_finalproject/blob/main/raw%20data/esg%2 and the actual csv is here: https://github.com/anushabishayee/info2950_finalproject/blob/main/esg_data.csv.
- Even more specifically, the Kaggle author states that they pulled the data for their csv from multiple APIs, like ESG Enterprise, a publicly-available API. They grabbed financial and company data from Finnhub. 3 of these links can be found here: https://github.com/anushabishayee/info2950_finalproject/blob/main/raw%20data/esg%2
- The finance data that the 3rd dataset is comprised of, and the finance data that is joined to the 1st and 2nd datasets is found in the yfinance library in Python (Yahoo Finance data, https://github.com/anushabishayee/info2950_finalproject/blob/main/raw%20data/yfinan

---

2. If people are involved, were they aware of the data collection and if so, what purpose did they expect the data to be used for?

- Individuals are not involved in the data directly, as each observation corresponds to an entire company.

---

3. What preprocessing was done, and how did the data come to be in the form that you are using?

- Our preprocessing of these datasets is detailed above. Generally, we imported the yfinance library, downloaded the Kaggle csv with the company ESG data, cleaned the dataset for NaNs and unneeded values, and reformatted some date values for ease of manipulation. Then, we joined the ESG data to the yfinance stock data, matching on company ticker (we created 4 new stock metric columns), and dropped NaNs for the creation of merged_df. For the 2nd dataset (sample_companies), we stratified and randomly selected 30 specific companies from this main dataset (specific methodology is outlined above in the 'Data Collection and Cleaning' section. For the 3rd dataset, we extracted the data straight from the yfinance library, and calculated a rate of change variable for the stock change as well for sp500.
- For the Kaggle csv, the author notes that they used company stock ticker as a unique identifier, then pulled and collated data from various APIs. in specific, they utilized ESG Enterprise (https://www.esgenterprise.com/), a publicly-available API,

and pulled their ratings methodology from https://app.esgenterprise.com/uploads/ESG-Enterprise-Risk-Ratings-MethodologyV3.pdf. They grabbed financial and company data from Finnhub (https://finnhub.io/).

---

4. What processes might have influenced what data was observed and recorded and what was not?

- For the ESG data, the Kaggle author of the csv specifically mentioned that only mid/large-cap companies are included, so this influences the specific companies that are recorded in the initial data - smaller companies (that also might have an ESG score) will not be 'observed' here. The author pulled data from ESG Enterprise and Finnhub, so any companies that do not have data available there will not be observed in the dataset. We also dropped any company that had a NaN or blank column value for the ESG columns, and dropped any company that didn't have stock data available in Yahoo Finance (or had NaNs for any specific finance column).

---

5. Who funded the creation of the dataset?

- We created these 3 analysis-ready datasets from two data sources: a 'Public Company ESG Ratings Dataset' Kaggle dataset from user Alistair King (https://www.kaggle.com/alistairking), a New York-based Kaggle Datasets Grandmaster, as well as the yfinance Python library, created by Ran Aroussi (https://aroussi.com/) as a way around the 2017 Yahoo Finance API deprecation. It is unclear if these datasets were 'funded', but their organization and accumulation were spearheaded by the two aforementioned people, respectively.

---

6. Why was this dataset created?

- We formulated our main analysis-ready dataset (merged_df) to examine associations between some of the largest USD-utilizing companies' ESG scores and their stock performances (as well as industry-specific analyses). Then, we formulated our sample dataset (sample_companies) so that we could take a look at some company-level analyses of the general data and research question (620 companies are kinda hard to visualize simultaneously). Finally, we formulated the sp500 dataset so that we could contrast company stock performance from the specified range of 2/1/21 - 12/31/22 to the overall performance of the S&P 500. (the rationale for the range of 2/1/21 - 12/31/22 is mentioned above, it's due to the fact that most companies have a 'last processing date' of February 2022 - Novermber 2022 for their ESG score.)
- The original ESG csv was created and uploaded by Kaggle user Alistair King,

perhaps for personal enrichment or curiosity (they do have a Kaggle Datasets Grandmaster rank, so perhaps they just enjoy creating and uploading datasets). The original yfinance Python library was created by Ran Aroussi to have a simple way to download historical market data from Yahoo Finance, due to the Yahoo Finance API deprecation.

---

7. What are the observations (rows) and attributes (columns)?

- For the S&P 500 dataset (sp500), the rows each correspond to a specific date where the S&P 500 was measured, within the range from 2/1/21 - 12/31/22. The columns for this dataset are Start Price, End Price, and Rate of Change (aka the starting price of the S&P 500 when the US market opened on a specific day, the ending price of the S&P 500 when the US market closed on the same specific day, and the percentage change that this stock exhibited between the start and close times of that specific day (100 * (end price - start price) / start price)).

- For both the merged_df and sample_companies dataset, each row corresponds to an unique, mid- to large-cap company that is publicly-traded and utilizes USD. merged_df, our main dataset, has 620 companies, while sample_companies has 30 companies for now. merged_df and sample_companies have the same columns, they are:

- ticker - a unique combo of letters and numbers that represent a particular stock
- name - the official name of the company
- currency - the currency the company is traded in (this was filtered to only be USD)
- exchange - what market the company is exchanged on
- industry - the type of output the company produces
- logo - a link to the company logo, potentially for joining with other datasets (MIGHT BE DROPPED LATER)
- weburl - a link to the company website, potentially for joining with other datasets or scraping for text sentiment analysis (MIGHT BE DROPPED LATER)
- environment_grade - a letter score given to the company that measures how well it complies to environmental standards, ranging from AAA being the best to CCC being the worst
- environment_level - a categorical classification of a company's overall environmental performance (low, medium, high)
- social_grade - a letter score given to the company that measures how well it complies to social standards, ranging from AAA being the best to CCC being the worst
- social_level - a categorical classification of a company's overall social performance (low, medium, high)
- governance_grade - a letter score given to the company that measures how well it

complies to governance standards, ranging from AAA being the best to CCC being the worst

- governance_level - a categorical classification of a company's overall governance performance (low, medium, high)
- environment_score - a numerical measure of how well a company performs on environment-related factors, ranging from 0-1000
- social_score - a numerical measure of how well a company performs on social-related factors, ranging from 0-1000
- governance_score - a numerical measure of how well a company performs on governance-related factors, ranging from 0-1000
- total_score - a numerical measure of how well a company performs on environment, social, and governance-related factors, ranging from 0-1500]
- cik - central index key, a unique identifier assigned by the SEC to any company that files documents with the SEC (MIGHT BE DROPPED LATER) the following columns are ones that we created, using the yfinance data:
- percent_change - the percent change in the company stock price from close time on 2/1/21 to close time on 12/31/22 (100 * (end price - start price) / start price))
- start_close - the closing price of the company stock on 2/1/21
- end_close - the closing price of the company stock on 12/31/22
- volatility - standard deviation of daily returns of the company stock, aka the percentage change in the stock price from day to day (indicator of how much stock price fluctuates in a given period, higher volatility is riskier, lower volatility has more stability). specifically, daily return is calculated by closing price on day x+1 - closing price on day x divided by closing price on day x, so all daily returns in the time period 2/1/21-12/31/22 are calculated for the specific company stock, and then the standard deviation is taken to get the volatility
- 50_day_SMA - 50 day simple moving average, or the sum of closing price of a company stock for the last 50 days before 12/31/22, divided by 50 (if current stock price is above the 50-day SMA, the company is in uptrend, and vice versa)
- cumulative_return - cumulative return of the company stock over the entire period ((close price on 12/31/22 / close price on 2/1/21) - 1), positive values represent returns, and negative values represent losses

---

Data Limitations

1. ESG is typically evaluated annually, which might mean the scores in our dataset don't reflect the most accurate performance of the company, which directly impacts the analyses and conclusions we might draw from our EDA. in other words, when considering the short-term impact of the company's ESG and other policies, it's likely that policy change affect stocks immediately, but these changes might not

also be reflected in the company's ESG rating. Bascially, since ESG scores lag behind the stock fluctuations due to immediate events (mergers, acqusitions, freak events like the CrowdStrike failure), any significant events that occur during 2/1/21-12/31/22 may result in stock price changes that do not perfectly correlate to ESG metrics. This could skew our correlation or regression model analysis, so we want to be careful to not falsely attributing any stock changes to ESG scores (in case of possible confounding variables). As a caveat, we should also be careful not to infer any causal relations when correlation exists.

2. ESG is a constant value that is gathered from different days for each company in 2022, though stock prices for these companies change over time every day. We cannot perform any time-series analyses with ESG due to this fact, which limits what we can do for our final phases and EDA.

3. Due to the nature of the Kaggle csv and yfinance data, our data is restricted to the variable types of stock data, industry type, company name, and ESG score - which actually does help us narrow down the scope of our research question, but limits the breadth of the analyses we can perform as well.

4. Some specific data from the yfinance library is missing - we had to drop all companies that didn't have the specified data we wanted in our specified time range. We also had to drop all companies from the original ESG csv that had missing or blank data. Overall, this means that our analyses will not be perfectly representative of all companies that use USD and have an ESG rating (can't perfectly generalize to the population). Additionally, we filtered our original ESG dataframe to be just companies traded in USD, so we can't do any inter-country comparison (although this also helps us narrow the scope of our project). Since we are also only using companies that are existent and large/mid-cap within 2/1/21 - 12/31/22, any company that stopped their operations in this time frame will be excluded. In other words, our findings might disproportionally overestimate the relationship between ESG scores and stock performance (companies that went bankrupt or have poor ESG / financial outcomes are not represented, which might skew interpretation of ESG positively).

5. For our sample dataset (companies_sample), the current 30 companies were chosen with a stratified sample. As a reminder, essentially, we looked at all of the different industries, and ranked them by total ESG score. then, we divided up the industries into 3 groups: high ESG score, medium ESG score, and low ESG score. Then, from each group, we chose 2 industries from a random sample. After this, we then randomly selected 5 companies from each industry, giving us a dataset of 30 companies. This sample is not fully representative of all USD-using companies with ESG ratings (this actually ties into one of our questions for reviewers). After getting feedback, we may also consider complete random sampling (no stratification) to

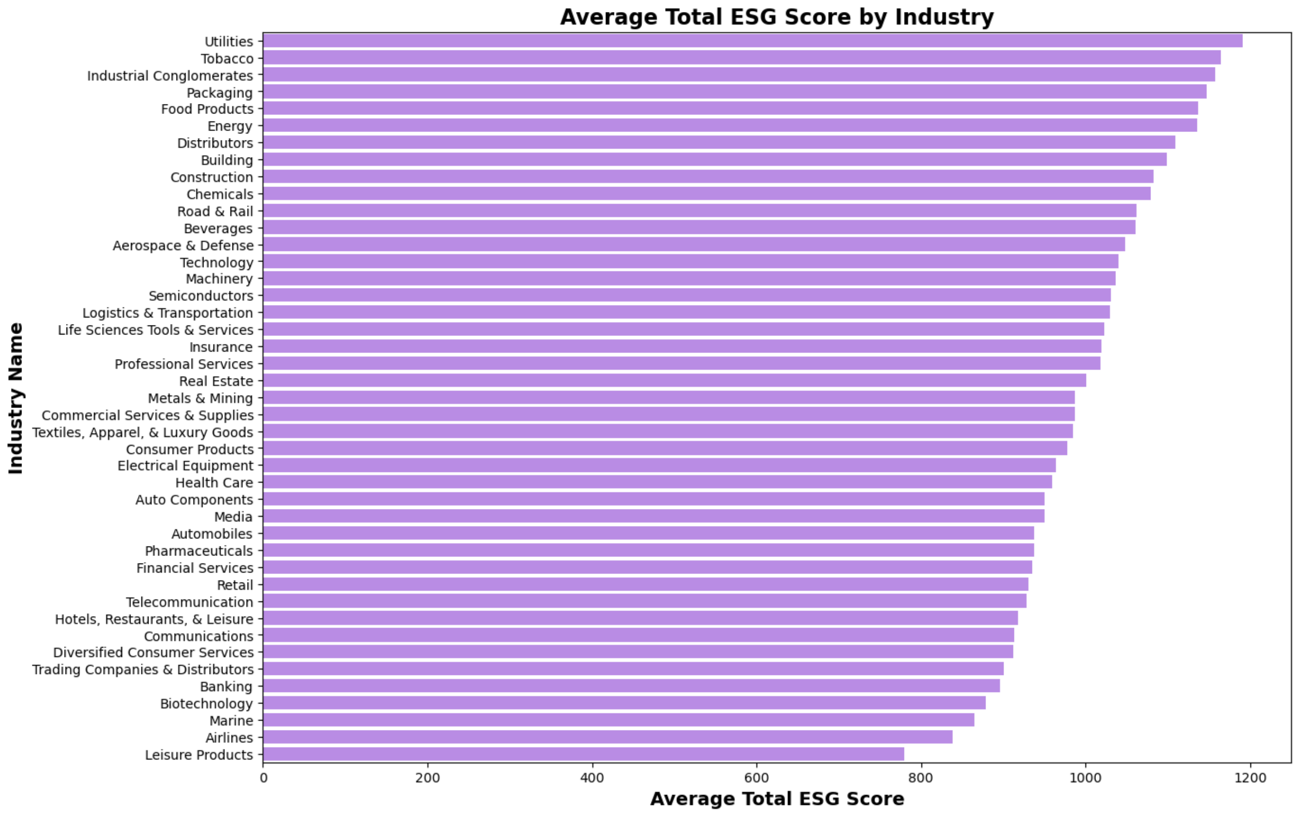
expand the representativeness of our sample dataset, but since we are exploring potential connections between ESG ratings and company stock performance, we may need to sample not only by industry but also by ESG rating levels to ensure a more balanced and comprehensive analysis of the different ESG performance tiers (for our sample dataset). Furthermore, though our sample size is sufficient to assume normality in distribution, it is still rather small compared to merged_df's number of companies included, so our findings might reflect trends only in specific industries or companies rather than the average and general market trends.

6. We currently plan on comparing the rate of change of the sample stocks (in companies_sample) to the S&P 500's rate of change. We also plan on taking a look at volatility, cumulative returns, and the 50 day simple moving average, but other measures of stock performance might provide more valuable insights (but we will proceed with these 4 for now). Additionally, due to the last processing date of the ESG scores for the companies, we also restricted our stock data to be from 2/1/21 - 12/31/22, which poses a limitation on the amount of analyses we can garner as we cannot extrapolate our conclusions to beyond this time frame.

Exploratory Data Analysis:

part one - exploring different average environmental, social, governance, and total ESG scores by industry

In [10]:
```
plt.figure(figsize = (14, 10))
sns.barplot(x = 'Average Total ESG Score', y = 'Industry', data = avg_esg_by
plt.title('Average Total ESG Score by Industry', horizontalalignment = 'cent
plt.xlabel('Average Total ESG Score', fontsize = 14, fontweight = 'bold')
plt.ylabel('Industry Name', fontsize = 14, fontweight = 'bold')
plt.show()
```



interestingly (and somewhat predictably) - the industries with the lowest ESG scores are Leisure Products, Airlines, Marine, Biotechnology, and Banking. the industries with the highest ESG scores are Utilities, Tobacco, Industrial Conglomerates, Packaging, and Industrial Conglomerates.

future steps: sort different industries by just Environmental score, just Social score, and just Governance score to see if these differ significantly.

In [11]:
```
avg_enviro_by_industry = merged_df.groupby('industry')['environment_score'].
avg_enviro_by_industry.columns = ['Industry', 'Average Environmental Score']

avg_enviro_by_industry = avg_enviro_by_industry.sort_values(by = 'Average En
print("best average Environment scores")
print(avg_enviro_by_industry.head(6))

avg_enviro_by_industry = avg_enviro_by_industry.sort_values(by = 'Average En
print("\nworst average Environment scores")
print(avg_enviro_by_industry.head(6))
```

```
best average Environment scores
                    Industry  Average Environmental Score
42                 Utilities                   550.966667
21  Industrial Conglomerates                   534.666667
30                 Packaging                   530.000000
13              Distributors                   525.000000
16                    Energy                   522.555556
18             Food Products                   517.076923

worst average Environment scores
                            Industry  Average Environmental Score
23                  Leisure Products                   252.333333
27                            Marine                   305.333333
6                      Biotechnology                   315.210526
41  Trading Companies & Distributors                   319.000000
1                           Airlines                   326.714286
4                            Banking                   341.000000
```

In [12]:

```python
avg_social_by_industry = merged_df.groupby('industry')['social_score'].mean(
avg_social_by_industry.columns = ['Industry', 'Average Social Score']

avg_social_by_industry = avg_social_by_industry.sort_values(by = 'Average Sc
print("best average social scores")
print(avg_social_by_industry.head(6))

avg_social_by_industry = avg_social_by_industry.sort_values(by = 'Average Sc
print("\nworst average social scores")
print(avg_social_by_industry.head(6))
```

```
best average social scores
                            Industry  Average Social Score
42                         Utilities            357.400000
41  Trading Companies & Distributors            335.000000
35                      Road & Rail             329.400000
40                           Tobacco            327.000000
30                         Packaging            322.666667
21          Industrial Conglomerates            318.000000

worst average social scores
            Industry  Average Social Score
3        Automobiles            236.333333
23  Leisure Products            259.333333
1           Airlines            265.714286
6      Biotechnology            270.526316
10    Communications            271.500000
2    Auto Components            272.500000
```

In [13]:

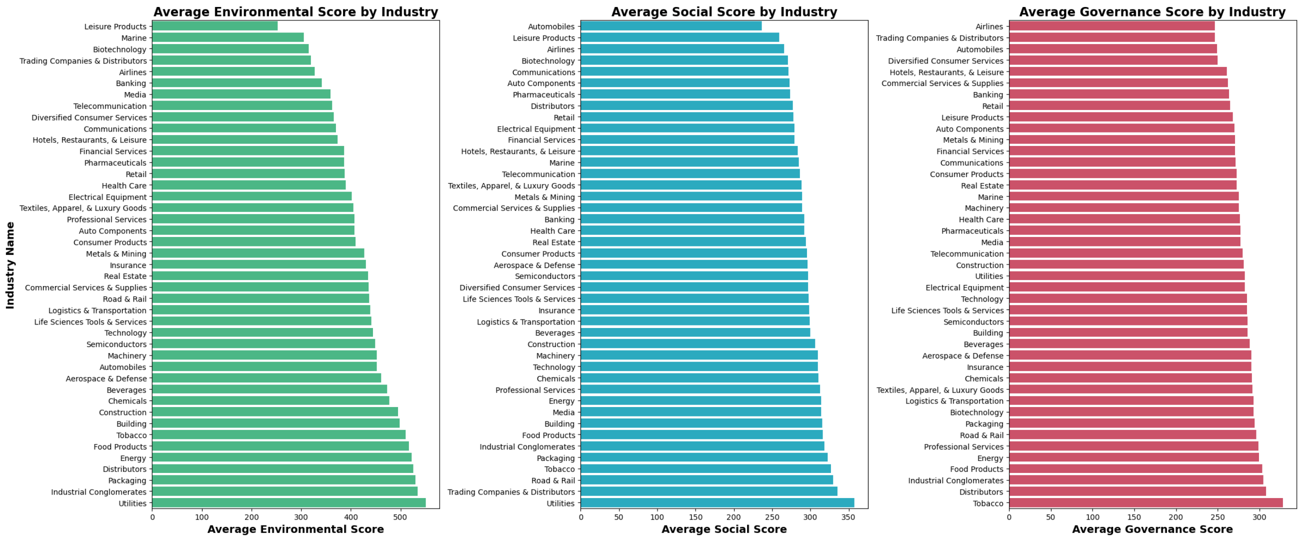
```python
avg_gov_by_industry = merged_df.groupby('industry')['governance_score'].mear
avg_gov_by_industry.columns = ['Industry', 'Average Governance Score']

avg_gov_by_industry = avg_gov_by_industry.sort_values(by = 'Average Governar
print("best average governance scores")
print(avg_gov_by_industry.head(6))
```

```
avg_gov_by_industry = avg_gov_by_industry.sort_values(by = 'Average Governar
print("\nworst average governance scores")
print(avg_gov_by_industry.head(6))
```

```
best average governance scores
                    Industry  Average Governance Score
40                   Tobacco                328.000000
13              Distributors                307.666667
21  Industrial Conglomerates                304.333333
18             Food Products                303.076923
16                    Energy                299.611111
32     Professional Services                298.600000

worst average governance scores
                            Industry  Average Governance Score
1                           Airlines                246.285714
41  Trading Companies & Distributors                246.666667
3                        Automobiles                249.333333
14     Diversified Consumer Services                250.000000
20    Hotels, Restaurants, & Leisure                260.913043
9     Commercial Services & Supplies                262.200000
```

In [14]:

```
fig, axes = plt.subplots(1, 3, figsize=(24, 10))  # Adjust the figsize to ma

# Plot 1: Average Environmental Score by Industry
sns.barplot(x='Average Environmental Score', y='Industry', data=avg_enviro_b
axes[0].set_title('Average Environmental Score by Industry', horizontalalign
axes[0].set_xlabel('Average Environmental Score', fontsize=14, fontweight='b
axes[0].set_ylabel('Industry Name', fontsize=14, fontweight='bold')

# Plot 2: Average Social Score by Industry
sns.barplot(x='Average Social Score', y='Industry', data=avg_social_by_indus
axes[1].set_title('Average Social Score by Industry', horizontalalignment='c
axes[1].set_xlabel('Average Social Score', fontsize=14, fontweight='bold')
axes[1].set_ylabel('')  # Remove y-axis label to avoid redundancy

# Plot 3: Average Governance Score by Industry
sns.barplot(x='Average Governance Score', y='Industry', data=avg_gov_by_indu
axes[2].set_title('Average Governance Score by Industry', horizontalalignmer
axes[2].set_xlabel('Average Governance Score', fontsize=14, fontweight='bolc
axes[2].set_ylabel('')

plt.tight_layout()
plt.show()
```



the industries with the lowest Environmental scores are also Leisure Products, Marine, Biotechnology, Trading Companies & Distributors, and Airlines. the industries with the highest Environmental scores are also Utilities, Industrial Conglomerates, Packaging, Distributors, and Energy.

the industries with the lowest Social scores are Automobiles, Leisure Products, Airlines, Biotechnology, and Communications. the industries with the highest Social scores are also Utilities, Trading Companies & Distributors, Road & Rail, Tobacco, and Packaging.

Regression

In [15]:

```
#Regression
X = merged_df[['environment_score', 'social_score','governance_score']]   #
y = merged_df['percentage_change']   # Dependent variable

model = LinearRegression().fit(X, y)
print(f"Environmental Score Coefficient: {model.coef_[0]}")
print(f"Social Score Coefficient: {model.coef_[1]}")
print(f"Governance Score Coefficient: {model.coef_[2]}")
print(f"Intercept: {model.intercept_}")
```

```
Environmental Score Coefficient: 0.12695165695146626
Social Score Coefficient: -0.0010317972363383901
Governance Score Coefficient: -0.16957002113899453
Intercept: -4.314223587566138
```

Regression Coefficients Interpretation

For every 1-unit increase in the Environmental score (assuming all other factors remain constant), the stock return is expected to increase by 0.1079 units. The positive coefficient suggests that higher Environmental scores are associated with better stock performance (or higher returns).

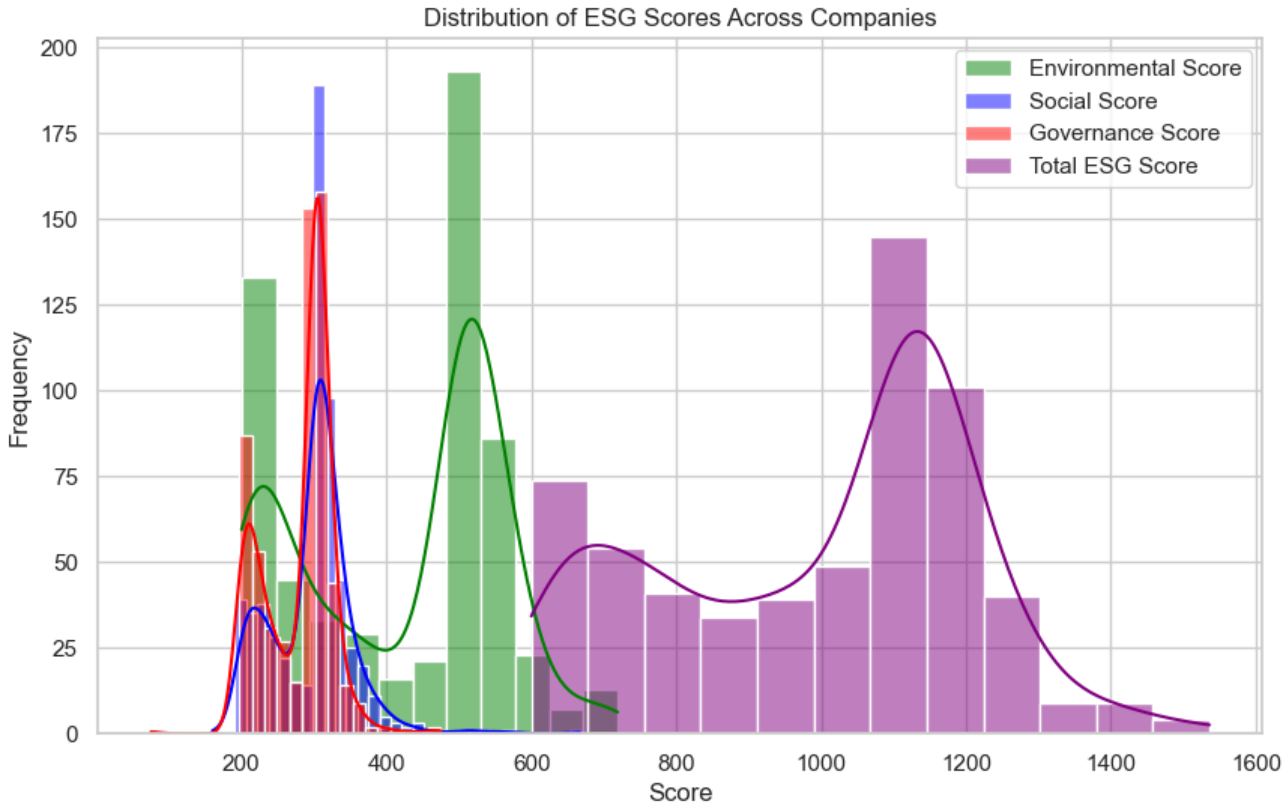
For every 1-unit increase in the Social score (with other variables constant), the stock return is expected to decrease by 0.0237 units. The negative coefficient indicates that better Social scores might be associated with lower stock performance, but generally, since the coefficient is so close to 0, Social scores seem to have little impact on stock performance.

For every 1-unit increase in the Governance score (with other variables constant), stock return is expected to decrease by 0.0750 units. This negative coefficient suggests that improvements in governance (e.g., stricter regulation or more ethical practices) are associated with slightly lower stock returns. This could imply that governance improvements come at a financial cost.

the industries with the lowest Governance scores are Airlines, Trading Companies & Distributors, Automobiles, Diversified Consumer Services, and Hotels, Restaurants, & Leisure. the industries with the highest Governance scores are also Tobacco, Distributors, Industrial Conglomerates, Food Products, and Tobacco.

Distribution of ESG Scores

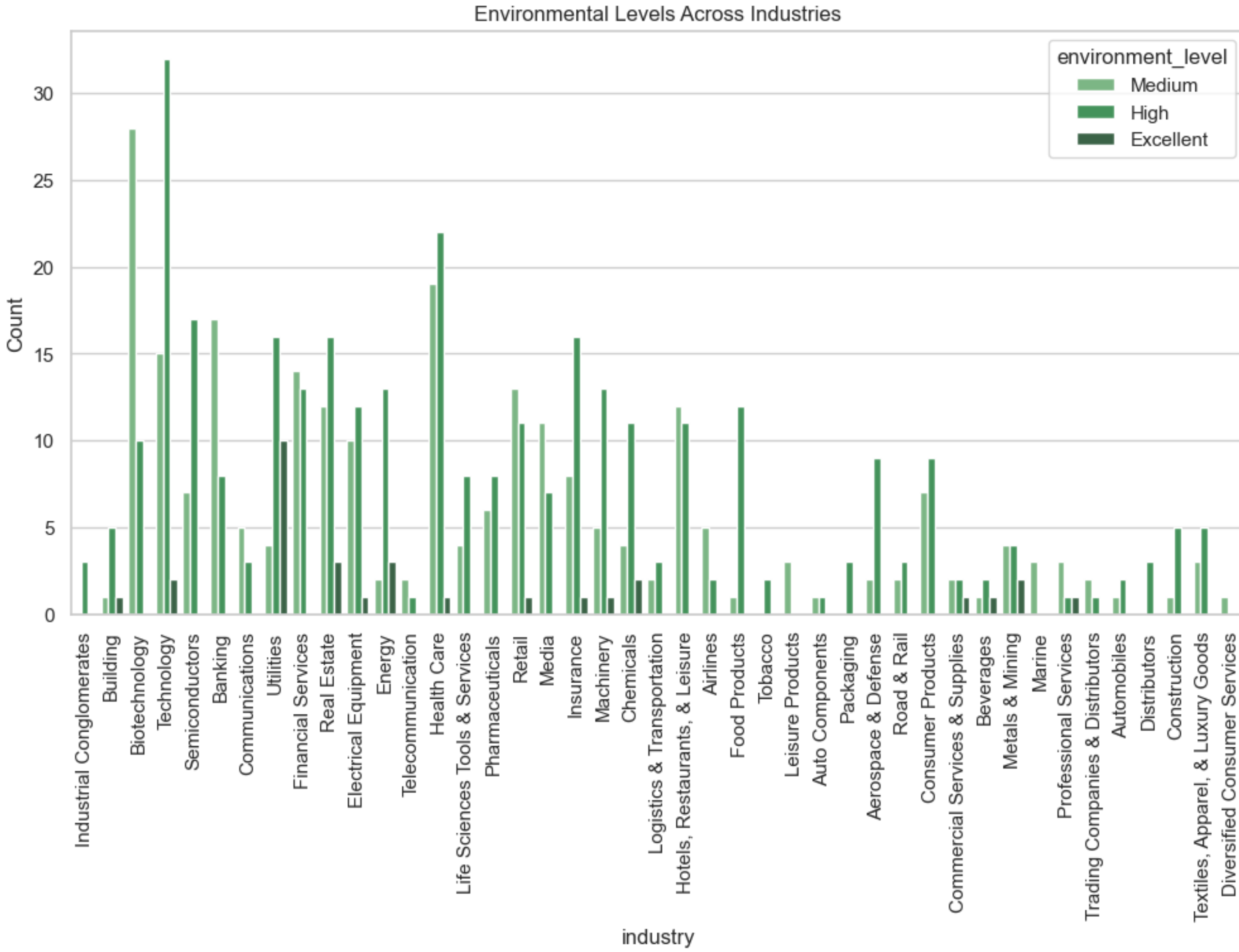
In [16]:
```
# Setting up Seaborn style
sns.set(style="whitegrid")

# 1. Distribution of ESG Scores (Environmental, Social, Governance, and Tota

plt.figure(figsize=(10, 6))
sns.histplot(merged_df['environment_score'], kde=True, color='green', label=
sns.histplot(merged_df['social_score'], kde=True, color='blue', label='Socia
sns.histplot(merged_df['governance_score'], kde=True, color='red', label='Go
sns.histplot(merged_df['total_score'], kde=True, color='purple', label='Tota
plt.title('Distribution of ESG Scores Across Companies')
plt.xlabel('Score')
plt.ylabel('Frequency')
plt.legend()
plt.show()
```



Description

This plot highlights the distribution of Environmental, Social, Governance (E/S/G) scores and the total ESG score across companies. A clear bimodal pattern in the total ESG score (purple) suggests two main groups of companies—one with lower scores around 400-500 and another with higher scores around 1000-1200. The Environmental score (green) shows the widest spread, indicating greater variability, while Social and Governance scores (blue and red) are more concentrated. This plot provides a snapshot of ESG performance distribution, revealing key trends in company behavior across these metrics, giving us general understanding about ESG score in our datasets.

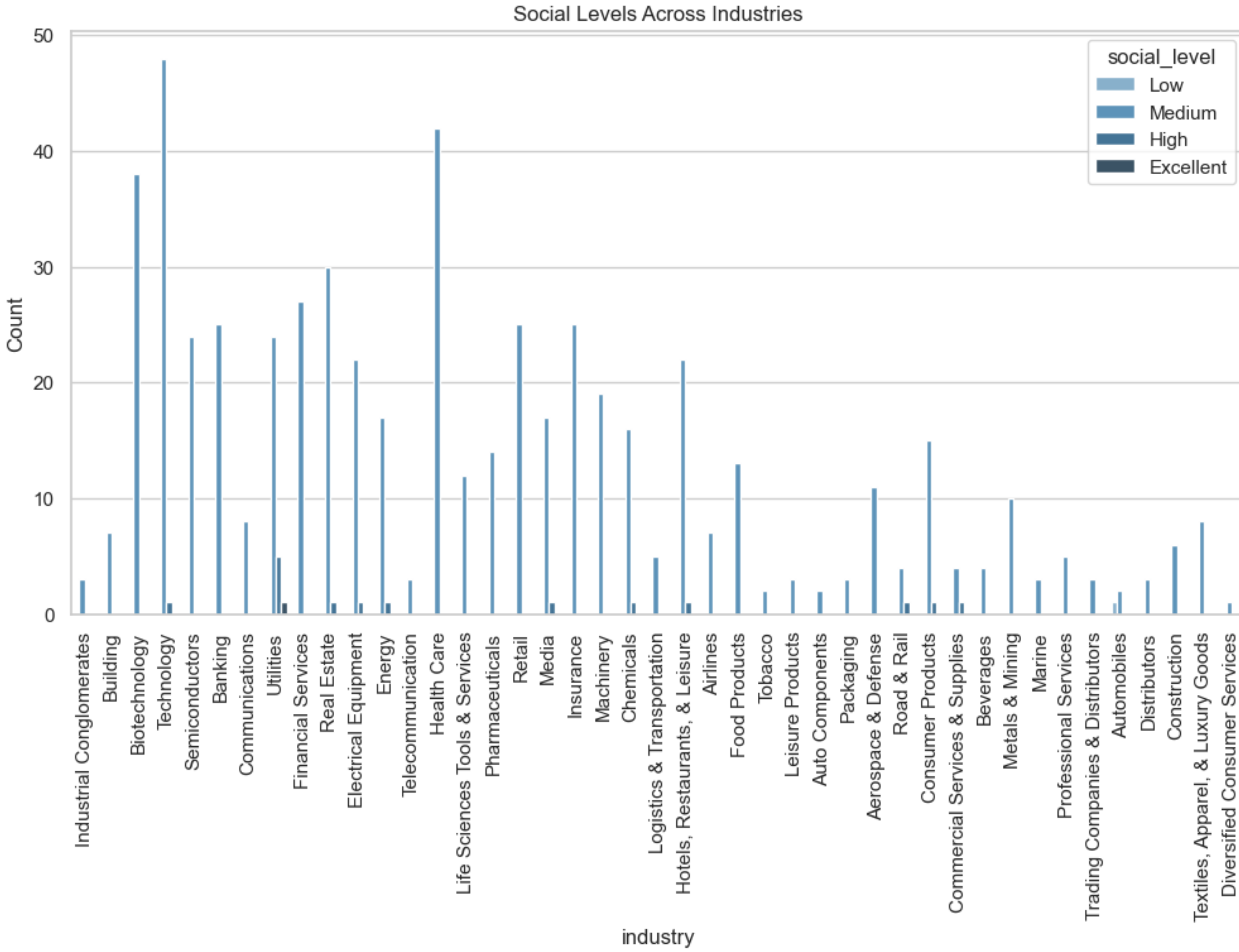
In [17]:
```
# 2. ESG Levels by Industry (Count Plots)

# Count plot for Environmental Level
plt.figure(figsize=(12, 6))
sns.countplot(data=merged_df, x='industry', hue='environment_level', palette
plt.title('Environmental Levels Across Industries')
plt.xticks(rotation=90)
plt.ylabel('Count')
plt.show()
```



Description

The plot shows distribution of Environmental levels for each industry. In terms of coloration, the darker the color is, the higher Environmental level is indicated. As indicated by the plot, the distribution of companies in each Environmental level are relatively wide spreaded, in which most of the companies has "Medium" or "High" Environmental levels, but the population with "High" environmental level are not extremely low. Overall, this plot provides an overview about the Environmental level across industries.

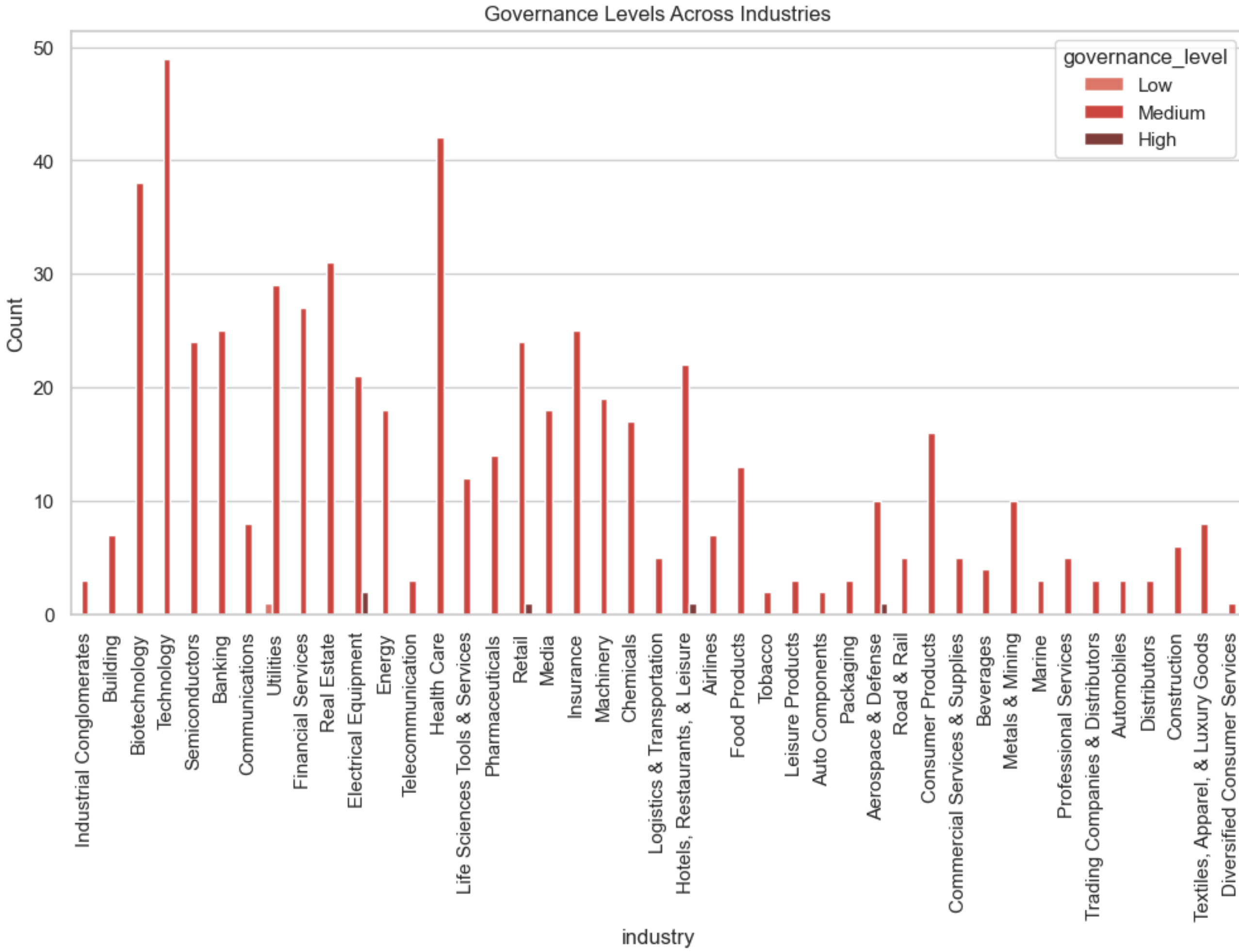
In [18]:
```
# Count plot for Social Level
plt.figure(figsize=(12, 6))
sns.countplot(data=merged_df, x='industry', hue='social_level', palette='Blu
plt.title('Social Levels Across Industries')
plt.xticks(rotation=90)
plt.ylabel('Count')
plt.show()
```



Description

The plot shows distribution of Social levels for each industry. In terms of coloration, the darker the color is, the higher Social level is indicated. As indicated by the graph, most of the companies in the population has "Medium" or "High" social levels and just a few companies (less then 10) has "Excellent" or "Low" Social levels. Overall, this plot provides some idea about the Social level for all industries.

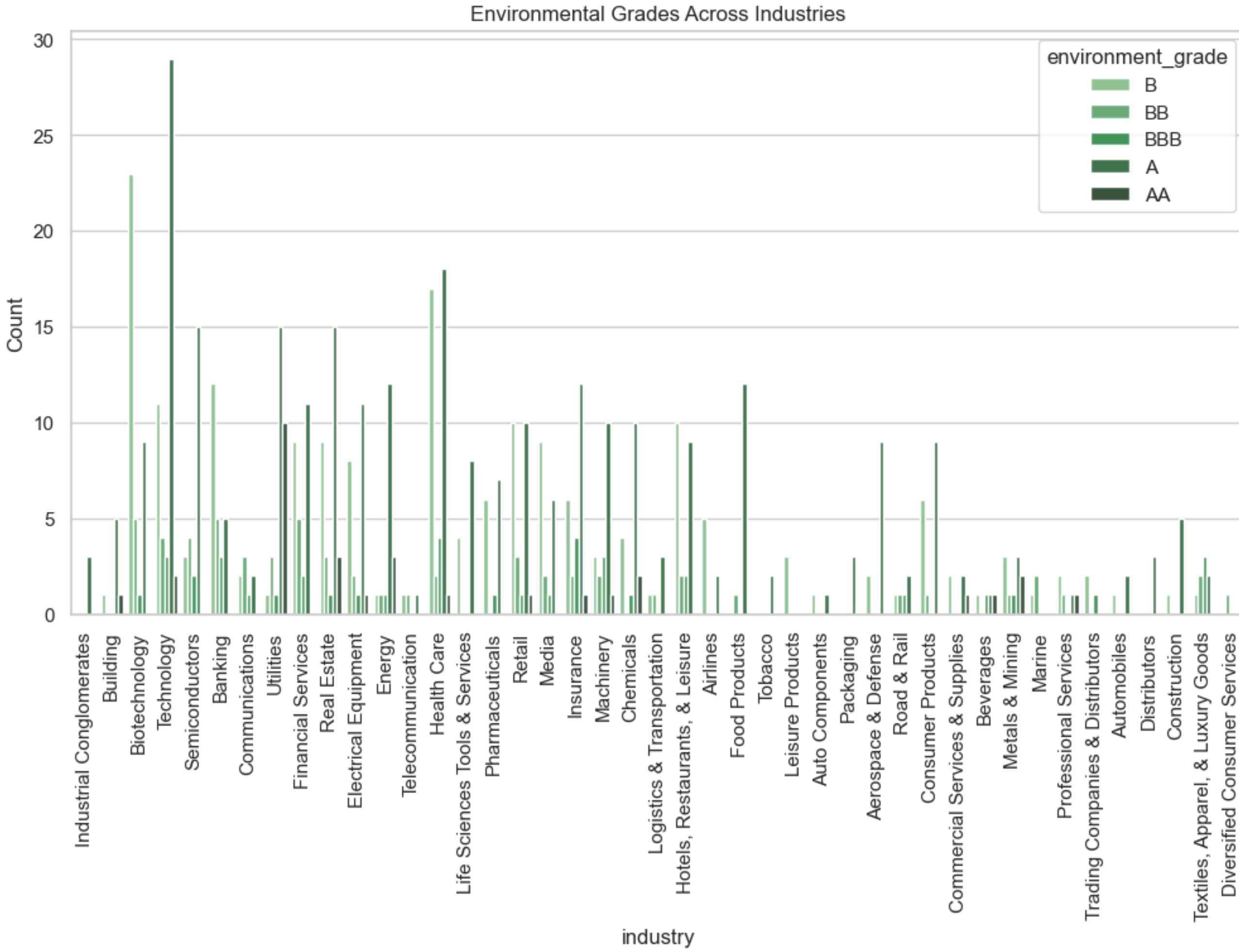
In [19]:
```
# Count plot for Governance Level
plt.figure(figsize=(12, 6))
sns.countplot(data=merged_df, x='industry', hue='governance_level', palette=
plt.title('Governance Levels Across Industries')
plt.xticks(rotation=90)
plt.ylabel('Count')
plt.show()
```



Description

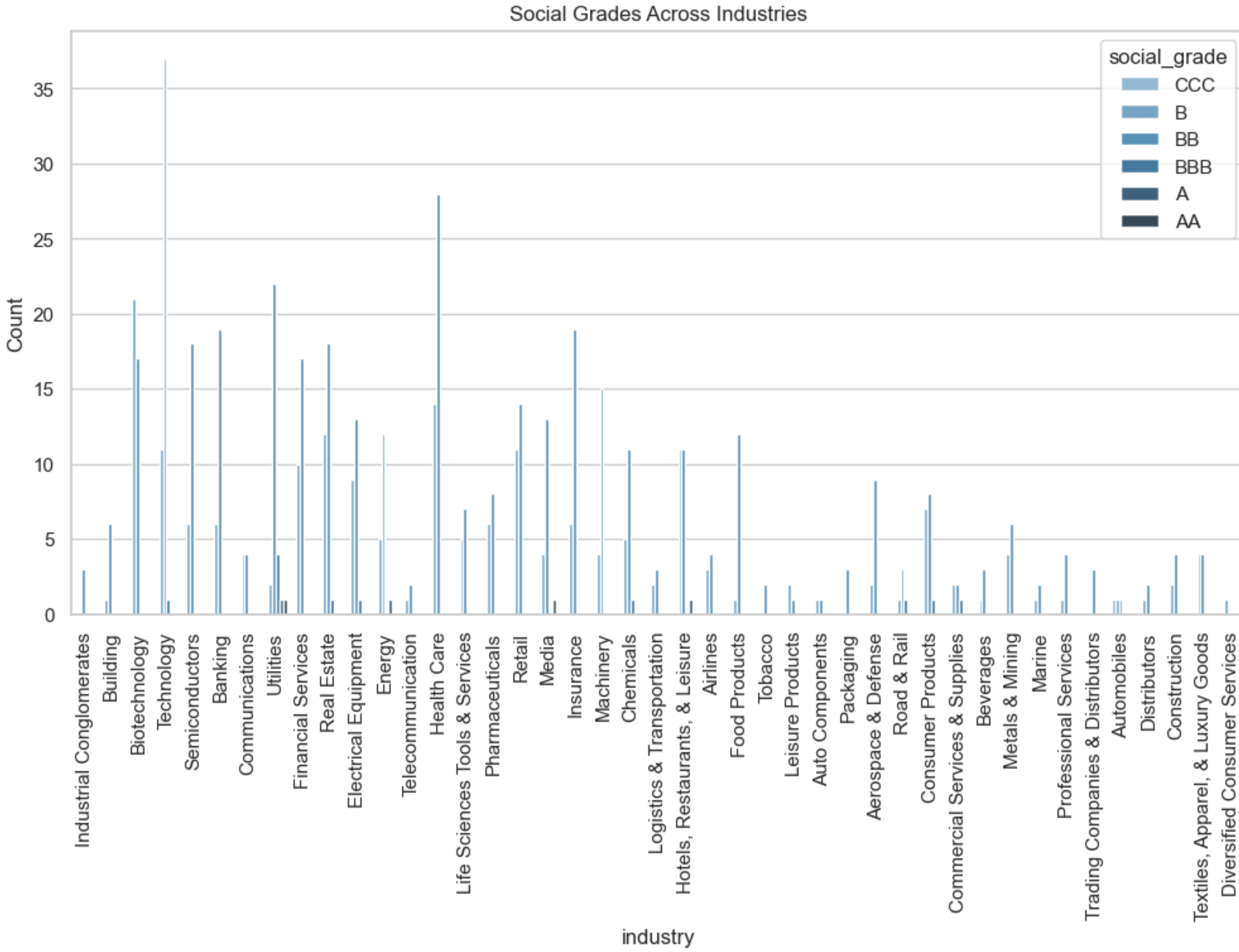
The plot shows distribution of Governance levels for each industry. In terms of coloration, the darker the color is, the higher Governance level is indicated. As indicated by the graph, most of the companies in the population has "Medium" Governance levels and just 5 has "High" and only 1 "Low" Governance levels, which indicates more concentrated Governance Level comparing to Social or Environmental Levels.

In [20]:
```
# 3. ESG Grades by Industry (Bar Plots)

# Bar plot for Environmental Grade
plt.figure(figsize=(12, 6))
sns.countplot(data=merged_df, x='industry', hue='environment_grade', palette
plt.title('Environmental Grades Across Industries')
plt.xticks(rotation=90)
plt.ylabel('Count')
plt.show()
```

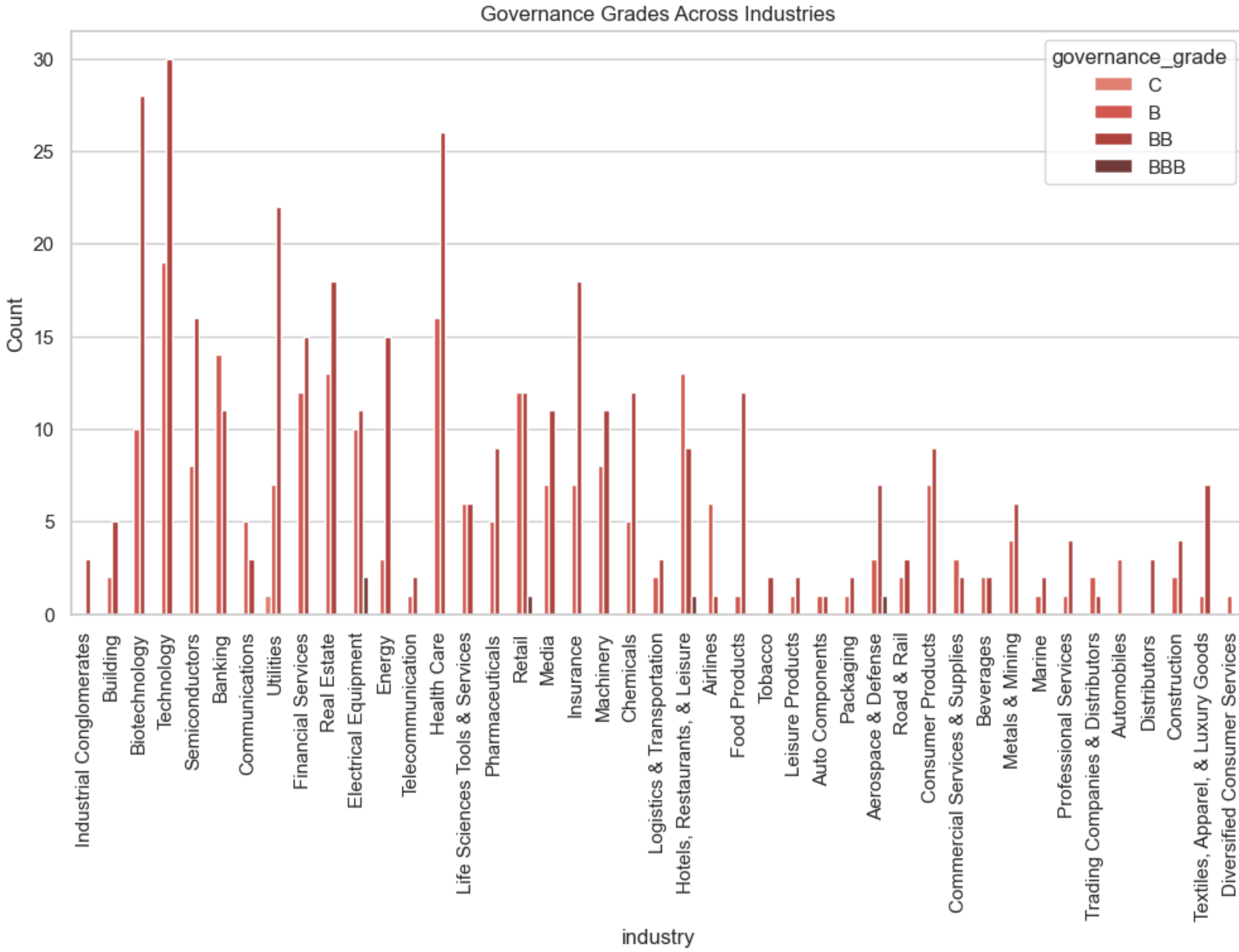


In [21]:

```
# Bar plot for Social Grade
plt.figure(figsize=(12, 6))
sns.countplot(data=merged_df, x='industry', hue='social_grade', palette='Blu
plt.title('Social Grades Across Industries')
plt.xticks(rotation=90)
plt.ylabel('Count')
plt.show()
```



In [22]:
```
# Bar plot for Governance Grade
plt.figure(figsize=(12, 6))
sns.countplot(data=merged_df, x='industry', hue='governance_grade', palette=
plt.title('Governance Grades Across Industries')
plt.xticks(rotation=90)
plt.ylabel('Count')
plt.show()
```



Description

The bar plots showing ESG grades follow a similar pattern to those with ESG levels but provide more detailed ratings. We are keeping these plots for potential use in future research, in case the ESG grades become relevant for further analysis.

---

part two - exploring different average environmental, social, governance, and total ESG scores by industry

In [23]:
```
fig, axes = plt.subplots(3, 1, figsize=(8, 10))

# Scatter plot: Percentage Change vs. Total ESG Score (with legend)
sns.scatterplot(data=merged_df, x='total_score', y='percentage_change', hue=
axes[0].set_title("Percentage Change vs Total ESG Score", fontsize=12)
axes[0].set_xlabel("Total ESG Score", fontsize=10)
axes[0].set_ylabel("Percentage Change in Stock Price", fontsize=10)
axes[0].legend(bbox_to_anchor=(1.05, 1), loc='upper left', fontsize='small')

# Scatter plot: Volatility vs. Total ESG Score (no legend)
sns.scatterplot(data=merged_df, x='total_score', y='volatility', hue='indust
axes[1].set_title("Volatility vs Total ESG Score", fontsize=12)
axes[1].set_xlabel("Total ESG Score", fontsize=10)
```

```
axes[1].set_ylabel("Stock Volatility", fontsize=10)

# Scatter plot: Cumulative Return vs. Total ESG Score (no legend)
sns.scatterplot(data=merged_df, x='total_score', y='cumulative_return', hue=
axes[2].set_title("Cumulative Return vs Total ESG Score", fontsize=12)
axes[2].set_xlabel("Total ESG Score", fontsize=10)
axes[2].set_ylabel("Cumulative Return", fontsize=10)


plt.tight_layout()
plt.subplots_adjust(hspace=0.5)
plt.show()
```

```
C:\Users\anush\AppData\Local\Temp\ipykernel_5676\2108959155.py:23: UserWarni
ng: Tight layout not applied. tight_layout cannot make axes height small eno
ugh to accommodate all axes decorations.
  plt.tight_layout()
```

Description

1. Percentage Change vs Total ESG Score: The first plot shows the relationship

between Total ESG Score and percentage change in stock price. The data points are widely scattered, indicating no clear correlation between ESG scores and drastic stock price changes. However, companies with higher ESG scores tend to have less extreme changes in stock prices, suggesting stability.

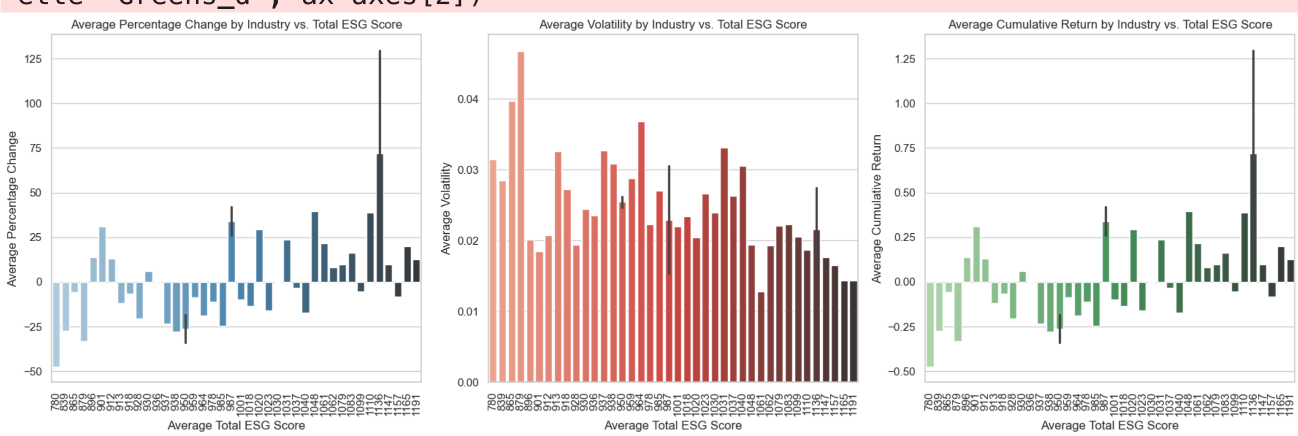
2. Volatility vs Total ESG Score: The second plot displays the stock volatility as a function of Total ESG Score. Overall, companies with higher ESG scores appear to have lower volatility, reflecting that higher ESG performance might be associated with less risky, more stable stock behavior.

3. Cumulative Return vs Total ESG Score: The third plot visualizes the cumulative return relative to Total ESG Score. This plot is very similar to the first plot with Percentage Change vs Total ESG Score, suggesting Cumulative Return can be removed as a potential influential variable.

In [24]:
```
# 2. Industry-wise Average ESG Scores vs. Stock Performance

# Grouping by Industry to get average values
industry_avg = merged_df.groupby('industry').agg({
    'environment_score': 'mean',
    'social_score': 'mean',
    'governance_score': 'mean',
    'total_score': lambda x: round(x.mean()),  # Rounding the average total_
    'percentage_change': 'mean',
    'volatility': 'mean',
    'cumulative_return': 'mean'
}).reset_index()
fig, axes = plt.subplots(1, 3, figsize=(18, 6))  # Adjust the figsize to mak

# Plotting Average Percentage Change by Industry vs. Total ESG Score
sns.barplot(data=industry_avg, x='total_score', y='percentage_change', palet
axes[0].set_title("Average Percentage Change by Industry vs. Total ESG Score
axes[0].set_xlabel("Average Total ESG Score")
axes[0].set_ylabel("Average Percentage Change")
axes[0].tick_params(axis='x', rotation=90)

# Plotting Average Volatility by Industry vs. Total ESG Score
sns.barplot(data=industry_avg, x='total_score', y='volatility', palette='Rec
axes[1].set_title("Average Volatility by Industry vs. Total ESG Score")
axes[1].set_xlabel("Average Total ESG Score")
axes[1].set_ylabel("Average Volatility")
axes[1].tick_params(axis='x', rotation=90)

# Plotting Average Cumulative Return by Industry vs. Total ESG Score
sns.barplot(data=industry_avg, x='total_score', y='cumulative_return', palet
axes[2].set_title("Average Cumulative Return by Industry vs. Total ESG Score
axes[2].set_xlabel("Average Total ESG Score")
axes[2].set_ylabel("Average Cumulative Return")
axes[2].tick_params(axis='x', rotation=90)
```

```python
# Adjust layout to prevent overlapping
plt.tight_layout()

# Show the plots
plt.show()
```

```
C:\Users\anush\AppData\Local\Temp\ipykernel_5676\924826018.py:16: FutureWarn
ing:

Passing `palette` without assigning `hue` is deprecated and will be removed
in v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the
same effect.

  sns.barplot(data=industry_avg, x='total_score', y='percentage_change', pal
ette='Blues_d', ax=axes[0])
C:\Users\anush\AppData\Local\Temp\ipykernel_5676\924826018.py:23: FutureWarn
ing:

Passing `palette` without assigning `hue` is deprecated and will be removed
in v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the
same effect.

  sns.barplot(data=industry_avg, x='total_score', y='volatility', palette='R
eds_d', ax=axes[1])
C:\Users\anush\AppData\Local\Temp\ipykernel_5676\924826018.py:30: FutureWarn
ing:

Passing `palette` without assigning `hue` is deprecated and will be removed
in v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the
same effect.

  sns.barplot(data=industry_avg, x='total_score', y='cumulative_return', pal
ette='Greens_d', ax=axes[2])
```



Description

1. Average Percentage Change by Industry vs. Total ESG Score: This plot shows the average percentage change in stock price for each industry relative to its average Total ESG Score. Industries with lower ESG scores tend to exhibit more negative

price changes, while those with higher ESG scores generally show more positive price changes.

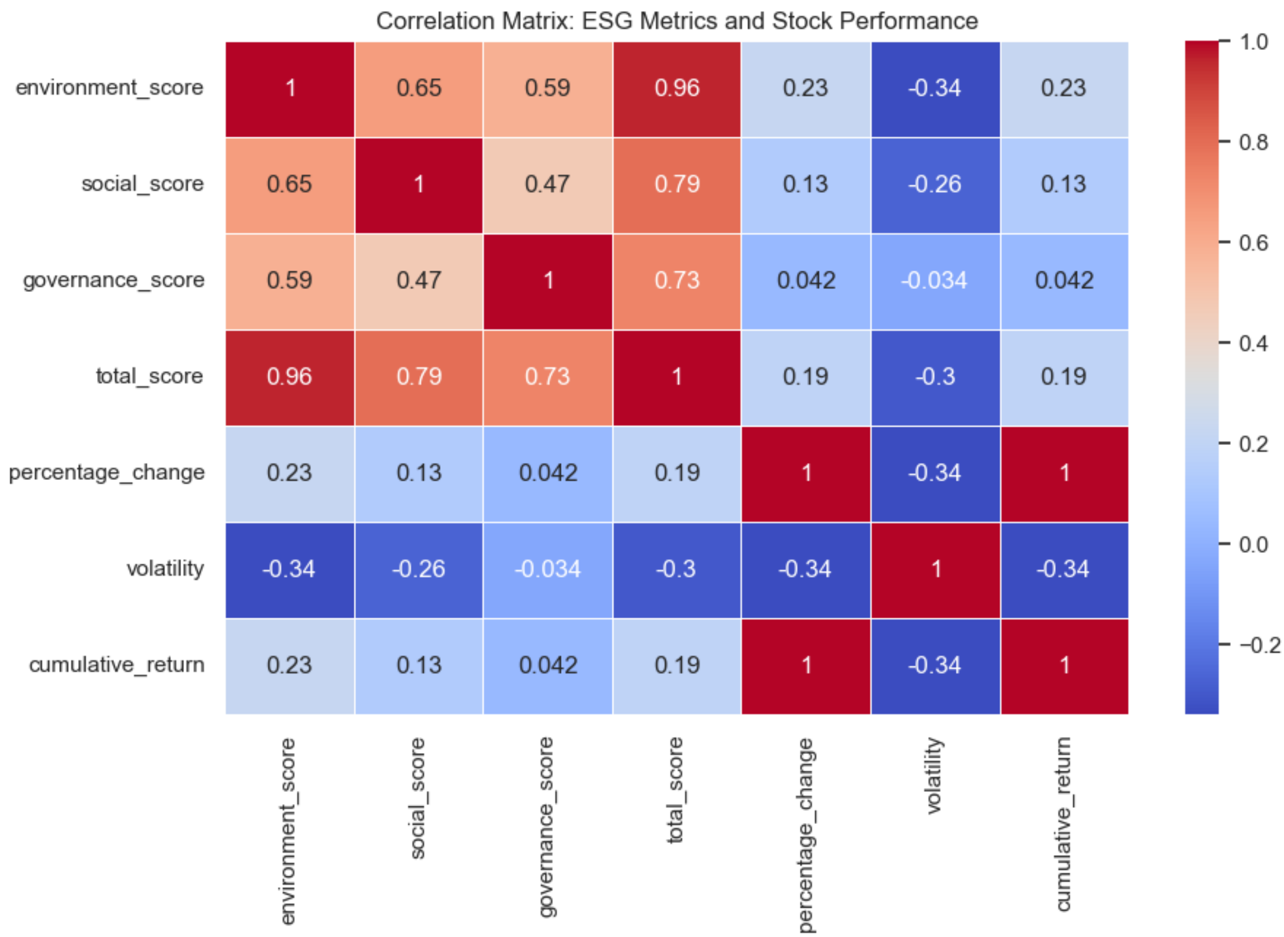
2. Average Volatility by Industry vs. Total ESG Score: The second plot presents the average stock volatility for each industry compared to its Total ESG Score. The plot illustrates a decreasing pattern, in which when industries have lower ESG scores usually exhibit greater volatility, suggesting that companies with lower ESG performance may experience more unpredictable stock behavior. In contrast, higher ESG scores are associated with relatively lower volatility.

3. Average Cumulative Return by Industry vs. Total ESG Score: This plot visualizes the relationship between cumulative stock returns and Total ESG Score by industry. Trends illustrated in this plot is similar to the first plot, again, indicating Cumulative Return might not be a significant variable if we already included Percentage Change in our model.

In [25]:

```
# Summary Statistics and Correlation Analysis
# Summary statistics for ESG scores and stock performance variables
summary_stats = merged_df[['environment_score', 'social_score', 'governance_
                           'percentage_change', 'volatility', 'cumulative_re
print(summary_stats)

# Correlation matrix between ESG scores and stock performance variables
correlation_matrix = merged_df[['environment_score', 'social_score', 'goverr
                                'percentage_change', 'volatility', 'cumulati

plt.figure(figsize=(10, 6))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', linewidths=0.5)
plt.title('Correlation Matrix: ESG Metrics and Stock Performance')
plt.show()
```

```
       environment_score  social_score  governance_score  total_score  \
count         599.000000    599.000000        599.000000   599.000000   
mean          419.814691    297.061770        280.053422   996.929883   
std           141.996964     56.412224         47.608533   215.688169   
min           200.000000    160.000000         75.000000   600.000000   
25%           260.000000    259.000000        235.000000   796.000000   
50%           500.000000    303.000000        300.000000  1081.000000   
75%           525.000000    324.500000        310.000000  1152.500000   
max           719.000000    667.000000        475.000000  1536.000000   

       percentage_change  volatility  cumulative_return  
count         599.000000  599.000000         599.000000  
mean            1.186775    0.026581           0.011868  
std            58.411843    0.018022           0.584118  
min           -99.798043    0.009960          -0.997980  
25%           -29.766038    0.017021          -0.297660  
50%             1.697562    0.020945           0.016976  
75%            23.915640    0.029647           0.239156  
max           737.499983    0.192400           7.375000  
```



Description

This correlation matrix visualizes the relationships between ESG (Environmental, Social, and Governance) metrics and stock performance indicators (percentage change, volatility, and cumulative return). The color intensity represents the strength of the correlation, with red indicating positive correlation and blue indicating negative correlation.

1. Strong Positive Correlations Among ESG Scores:
   The matrix shows strong positive correlations between the environmental, social, governance, and total ESG scores. For example, the environmental score is highly correlated with the total ESG score (0.96), indicating that companies with higher environmental scores tend to have higher overall ESG performance.

2. ESG Scores and Stock Performance:
   Percentage Change: ESG scores have weak positive correlations with stock price percentage changes, with environmental and total scores showing the strongest relationships (0.23 and 0.20, respectively).
   Volatility: All ESG scores are negatively correlated with volatility. The environmental score (-0.34) and total ESG score (-0.30) show the strongest negative correlations, suggesting that higher ESG performance is associated with lower stock price volatility.
   Cumulative Return: ESG scores have weak positive correlations with cumulative return, with environmental and total scores showing the most significant relationships (0.23 and 0.20), implying that higher ESG scores may be linked to better long-term returns.

This matrix provides a comprehensive overview of how ESG performance metrics relate to stock performance, highlighting the role ESG scores may play in stabilizing stock behavior and influencing returns

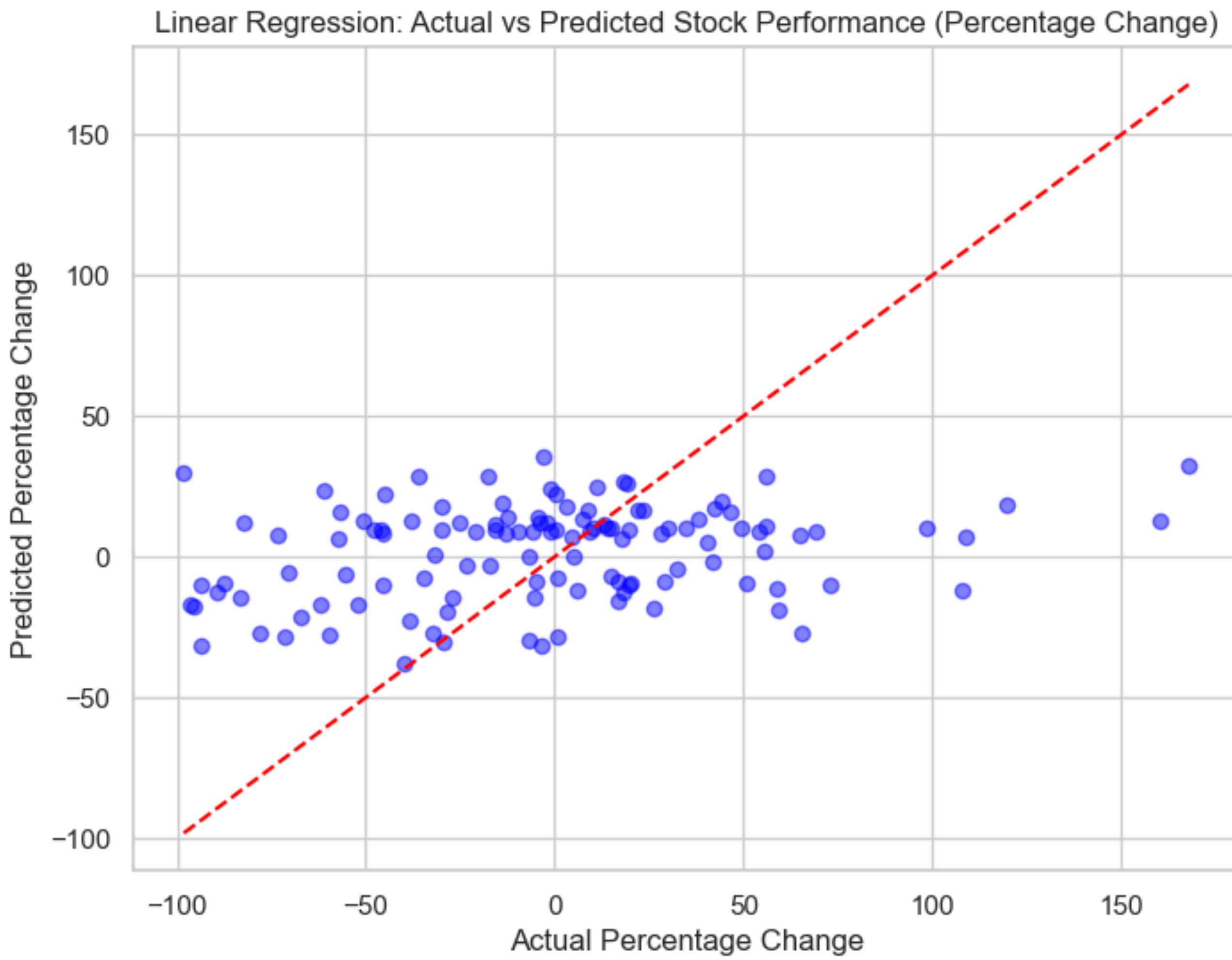
---

In [26]:

```
# Linear Regression - Predicting Stock Performance (percentage_change) basec

# Define X (independent variables) and y (dependent variable)
X = merged_df[['environment_score', 'social_score', 'governance_score', 'tot
y = merged_df['percentage_change']

# Train-test split (80% training, 20% testing)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, rar

# Initialize and fit linear regression model
lin_reg = LinearRegression()
lin_reg.fit(X_train, y_train)

# Predict on test set
y_pred = lin_reg.predict(X_test)

# Evaluate the model
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

print(f"Mean Squared Error: {mse}")
print(f"R-squared: {r2}")
```

```
# Plotting actual vs predicted percentage changes
plt.figure(figsize=(8, 6))
plt.scatter(y_test, y_pred, color='blue', alpha=0.5)
plt.plot([min(y_test), max(y_test)], [min(y_test), max(y_test)], color='red'
plt.title('Linear Regression: Actual vs Predicted Stock Performance (Percent
plt.xlabel('Actual Percentage Change')
plt.ylabel('Predicted Percentage Change')
plt.show()

# Step 3 (Optional): Logistic Regression - Binary Stock Increase/Decrease

# Create a binary variable for stock increase (1 if percentage_change > 0, e
merged_df['stock_increase'] = np.where(merged_df['percentage_change'] > 0, 1

# Define X and y for logistic regression
X = merged_df[['environment_score', 'social_score', 'governance_score', 'tot
y = merged_df['stock_increase']

# Train-test split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, ran

# Initialize and fit logistic regression model
log_reg = LogisticRegression()
log_reg.fit(X_train, y_train)

# Predict on test set
y_pred = log_reg.predict(X_test)

# Evaluate the model
accuracy = accuracy_score(y_test, y_pred)
print(f"Accuracy: {accuracy}")

# Confusion matrix and classification report (optional)
from sklearn.metrics import confusion_matrix, classification_report
print(confusion_matrix(y_test, y_pred))
print(classification_report(y_test, y_pred))

# Step 4: Summary of Linear Regression Results Using statsmodels (OLS)

# Adding constant for intercept
X_train_const = sm.add_constant(X_train)

# Fit the model using OLS (Ordinary Least Squares)
model = sm.OLS(y_train, X_train_const)
results = model.fit()

# Summary of the regression model
print(results.summary())
```

```
Mean Squared Error: 2502.9090371978673
R-squared: 0.0602998506827328
```




```
Accuracy: 0.5833333333333334
[[29 33]
 [17 41]]
              precision    recall  f1-score   support

           0       0.63      0.47      0.54        62
           1       0.55      0.71      0.62        58

    accuracy                           0.58       120
   macro avg       0.59      0.59      0.58       120
weighted avg       0.59      0.58      0.58       120

                            OLS Regression Results
=============================================================================
==
Dep. Variable:         stock_increase   R-squared:                       0.0
75
Model:                            OLS   Adj. R-squared:                  0.0
70
Method:                 Least Squares   F-statistic:                     12.
90
Date:                Tue, 22 Oct 2024   Prob (F-statistic):           4.07e-
08
Time:                        16:29:53   Log-Likelihood:              -328.
13
```

```
No. Observations:                 479   AIC:                             66
4.3
Df Residuals:                     475   BIC:                             68
0.9
Df Model:                           3
Covariance Type:            nonrobust
==============================================================================
=========
                      coef    std err          t      P>|t|      [0.025
0.975]
------------------------------------------------------------------------------
---------
const               0.6356      0.155      4.106      0.000       0.331
0.940
environment_score   0.0014      0.000      4.202      0.000       0.001
0.002
social_score        0.0005      0.000      1.141      0.254      -0.000
0.001
governance_score   -0.0021      0.000     -4.462      0.000      -0.003
-0.001
total_score        -0.0002      0.000     -1.676      0.094      -0.001
4.19e-05
==============================================================================
==
Omnibus:                     2535.277   Durbin-Watson:                   1.8
41
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               60.8
27
Skew:                          -0.120   Prob(JB):                      6.19e-
14
Kurtosis:                       1.271   Cond. No.                      2.17e+
16
==============================================================================
==

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is corre
ctly specified.
[2] The smallest eigenvalue is 1.43e-24. This might indicate that there are
strong multicollinearity problems or that the design matrix is singular.
```

Description

We are building two models to predict stock performance based on ESG (Environmental, Social, and Governance) scores:

Linear Regression: With a low R-squared value of 0.052 and a high MSE of 2494.81, the model shows that ESG scores have limited predictive power for percentage change in stock prices. The model struggles to explain the majority of stock price variability.

Logistic Regression: The model's accuracy of 57% and precision/recall metrics show that ESG scores alone do not effectively predict whether a stock will increase or decrease in value.

The perfornace of both Linear Regression and Logistic Regression indicates we might need to explore other regression methods in interepreting companies' ESG measurements and their stock performances.

In [ ]:

```
#Stock price changes over time of sampled companies
data_with_sp500 = yf.download(list(sample_companies['ticker']) + ['^GSPC'],

# Stock price changes over time of sampled companies (WITHOUT S&P 500)
data_without_sp500 = yf.download(list(sample_companies['ticker']), start='20

# Create subplots with 1 row and 2 columns
fig, axes = plt.subplots(1, 2, figsize=(20, 8))  # Adjust the figsize to mak

# Plot WITH S&P 500
colors_with_sp500 = sns.color_palette('husl', len(data_with_sp500.columns))
for i, ticker in enumerate(data_with_sp500.columns):
    axes[0].plot(data_with_sp500.index, data_with_sp500[ticker], label=ticke
axes[0].set_xlabel('Date')
axes[0].set_ylabel('Adjusted Closing Price (USD)')
axes[0].set_title('Stock Price Changes Over Time for Selected Companies (Inc
axes[0].legend(loc='upper left', fontsize='small')
axes[0].tick_params(axis='x', rotation=45)

# Plot WITHOUT S&P 500
colors_without_sp500 = sns.color_palette('husl', len(data_without_sp500.colu
for i, ticker in enumerate(data_without_sp500.columns):
    axes[1].plot(data_without_sp500.index, data_without_sp500[ticker], label
axes[1].set_xlabel('Date')
axes[1].set_ylabel('Adjusted Closing Price (USD)')
axes[1].set_title('Stock Price Changes Over Time for Selected Companies (W/O
axes[1].legend(loc='upper left', fontsize='small')
axes[1].tick_params(axis='x', rotation=45)

# Adjust layout to prevent overlapping
plt.tight_layout()

# Show both plots side by side
plt.show()
```

Description

These side-by-side line plots illustrate the stock price changes over time for selected companies, both with and without the inclusion of the S&P 500 index (^GSPC), from February 2021 to December 2022.

Left Plot (Including S&P 500): This plot shows the adjusted closing prices for the selected companies alongside the S&P 500 index. The S&P 500 (pink line) dominates the chart due to its much higher adjusted closing price compared to individual companies. This makes the comparison between the companies harder to interpret visually because the large scale of the S&P 500 price compresses the other stock prices.

Right Plot (Without S&P 500): This plot excludes the S&P 500, making the individual stock prices of the selected companies easier to differentiate. With the S&P 500 removed, we can observe the individual trends more clearly, including which companies experienced significant growth or fluctuations during the observed period. Notable trends include some companies showing relatively stable prices, while others display more pronounced volatility.

Questions for Reviewers:

1. Does it seem like we have enough columns in merged_df and sample_companies to satisfy the complexity requirement for the project? Does our research question also seem complex enough given our EDA?

2. Is there any advice or recommended steps to follow in creating our sample_companies dataset from the population? If not, is our method of sampling acceptable? We wanted to get some feedback before proceeding with a lot of analyses for sample_companies, so the bulk of our EDA involves merged_df (as a refresher - we have one big dataset, merged_df with 600 companies that we did some EDA with, but we also want to include a smaller sample dataset so we can look at some individual companies as well. the purpose of this smaller dataset would be then to observe trends on a company-level, and study the metrics of our research question on this smaller scale). Is our sample size of 30 also acceptable?

3. How many visualizations (or like data analysis chunks) are recommended for the final project? (ballpark range would be helpful, or if there's any better way of quantifying, or will we get to look at some examples of a completed project later?)

4. Do the visualizations we currently have seem like they're on the right path for the final phases? Is there anything glaringly obvious that should be changed or labeled better?

5. Regarding the visualizations and stuff we have made for our EDA so far: should we further explore these specific visualizations more in depth? Or should we expand our DA to other variables in the datasets that we maybe haven't used yet?

6. We were looking at the guidelines for phase III as well - are you able to give us an example of a 'sample analysis' that we should aim to present in that part, based on the EDA that we currently have done? We are little confused about what that entails.

minor note - this group consists of 1 non-native English speaker :)