Unsupervised Learning: Trade&Ahead

Marks: 60

Problem Statement

Context

The stock market has consistently proven to be a good place to invest in and save for the future. There are a lot of compelling reasons to invest in stocks. It can help in fighting inflation, create wealth, and also provides some tax benefits. Good steady returns on investments over a long period of time can also grow a lot more than seems possible. Also, thanks to the power of compound interest, the earlier one starts investing, the larger the corpus one can have for retirement. Overall, investing in stocks can help meet life's financial aspirations.

It is important to maintain a diversified portfolio when investing in stocks in order to maximise earnings under any market condition. Having a diversified portfolio tends to yield higher returns and face lower risk by tempering potential losses when the market is down. It is often easy to get lost in a sea of financial metrics to analyze while determining the worth of a stock, and doing the same for a multitude of stocks to identify the right picks for an individual can be a tedious task. By doing a cluster analysis, one can identify stocks that exhibit similar characteristics and ones which exhibit minimum correlation. This will help investors better analyze stocks across different market segments and help protect against risks that could make the portfolio vulnerable to losses.

Objective

Trade&Ahead is a financial consultancy firm who provide their customers with personalized investment strategies. They have hired you as a Data Scientist and provided you with data comprising stock price and some financial indicators for a few companies listed under the New York Stock Exchange. They have assigned you the tasks of analyzing the data, grouping the stocks based on the attributes provided, and sharing insights about the characteristics of each group.

Data Dictionary

- Ticker Symbol: An abbreviation used to uniquely identify publicly traded shares

of a particular stock on a particular stock market
- Company: Name of the company
- GICS Sector: The specific economic sector assigned to a company by the Global Industry Classification Standard (GICS) that best defines its business operations
- GICS Sub Industry: The specific sub-industry group assigned to a company by the Global Industry Classification Standard (GICS) that best defines its business operations
- Current Price: Current stock price in dollars
- Price Change: Percentage change in the stock price in 13 weeks
- Volatility: Standard deviation of the stock price over the past 13 weeks
- ROE: A measure of financial performance calculated by dividing net income by shareholders' equity (shareholders' equity is equal to a company's assets minus its debt)
- Cash Ratio: The ratio of a company's total reserves of cash and cash equivalents to its total current liabilities
- Net Cash Flow: The difference between a company's cash inflows and outflows (in dollars)
- Net Income: Revenues minus expenses, interest, and taxes (in dollars)
- Earnings Per Share: Company's net profit divided by the number of common shares it has outstanding (in dollars)
- Estimated Shares Outstanding: Company's stock currently held by all its shareholders
- P/E Ratio: Ratio of the company's current stock price to the earnings per share
- P/B Ratio: Ratio of the company's stock price per share by its book value per share (book value of a company is the net difference between that company's total assets and total liabilities)

Importing necessary libraries

In [1]:
```
# Libraries to help with reading and manipulating data
import numpy as np
import pandas as pd

# Libraries to help with data visualization
import matplotlib.pyplot as plt
import seaborn as sns

sns.set_theme(style='darkgrid')

# Removes the limit for the number of displayed columns
pd.set_option("display.max_columns", None)
# Sets the limit for the number of displayed rows
pd.set_option("display.max_rows", 200)

# to scale the data using z-score
from sklearn.preprocessing import StandardScaler

# to compute distances
from scipy.spatial.distance import cdist, pdist

# to perform k-means clustering and compute silhouette scores
from sklearn.cluster import KMeans
from sklearn.metrics import silhouette_score

# to visualize the elbow curve and silhouette scores
from yellowbrick.cluster import KElbowVisualizer, SilhouetteVisualizer

# to perform hierarchical clustering, compute cophenetic correlation, and
from sklearn.cluster import AgglomerativeClustering
from scipy.cluster.hierarchy import dendrogram, linkage, cophenet

# to suppress warnings
import warnings
warnings.filterwarnings("ignore")
```

Loading the dataset

In [2]:
```
## Complete the code to import the data
data = pd.read_csv('stock_data.csv')
```

Overview of the Dataset

The initial steps to get an overview of any dataset is to:

- observe the first few rows of the dataset, to check whether the dataset has been loaded properly or not
- get information about the number of rows and columns in the dataset
- find out the data types of the columns to ensure that data is stored in the preferred format and the value of each property is as expected.
- check the statistical summary of the dataset to get an overview of the numerical columns of the data

Checking the shape of the dataset

In [3]:
```
# checking shape of the data
print(f"There are {data.shape[0]} rows and {data.shape[1]} columns") ## C
```

```
There are 340 rows and 15 columns
```

Displaying few rows of the dataset

In [4]:
```
# let's view a sample of the data
data.sample(n=10, random_state=1)
```

Out[4]:

	Ticker Symbol	Security	GICS Sector	GICS Sub Industry	Current Price	Price Change	Volatility
102	DVN	Devon Energy Corp.	Energy	Oil & Gas Exploration & Production	32.000000	-15.478079	2.923698
125	FB	Facebook	Information Technology	Internet Software & Services	104.660004	16.224320	1.320606
11	AIV	Apartment Investment & Mgmt	Real Estate	REITs	40.029999	7.578608	1.163334
248	PG	Procter & Gamble	Consumer Staples	Personal Products	79.410004	10.660538	0.806056
238	OXY	Occidental Petroleum	Energy	Oil & Gas Exploration & Production	67.610001	0.865287	1.589520
336	YUM	Yum! Brands Inc	Consumer Discretionary	Restaurants	52.516175	-8.698917	1.478877
112	EQT	EQT Corporation	Energy	Oil & Gas Exploration & Production	52.130001	-21.253771	2.364883
147	HAL	Halliburton Co.	Energy	Oil & Gas Equipment & Services	34.040001	-5.101751	1.966062
89	DFS	Discover Financial Services	Financials	Consumer Finance	53.619999	3.653584	1.159897
173	IVZ	Invesco Ltd.	Financials	Asset Management & Custody Banks	33.480000	7.067477	1.580839

Checking the data types of the columns for the dataset

In [5]:
```
# checking the column names and datatypes
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 340 entries, 0 to 339
Data columns (total 15 columns):
 #   Column                         Non-Null Count  Dtype  
---  ------                         --------------  -----  
 0   Ticker Symbol                  340 non-null    object 
 1   Security                       340 non-null    object 
 2   GICS Sector                    340 non-null    object 
 3   GICS Sub Industry              340 non-null    object 
 4   Current Price                  340 non-null    float64
 5   Price Change                   340 non-null    float64
 6   Volatility                     340 non-null    float64
 7   ROE                            340 non-null    int64  
 8   Cash Ratio                     340 non-null    int64  
 9   Net Cash Flow                  340 non-null    int64  
 10  Net Income                     340 non-null    int64  
 11  Earnings Per Share             340 non-null    float64
 12  Estimated Shares Outstanding   340 non-null    float64
 13  P/E Ratio                      340 non-null    float64
 14  P/B Ratio                      340 non-null    float64
dtypes: float64(7), int64(4), object(4)
memory usage: 40.0+ KB
```

Creating a copy of original data

In [6]:
```
# copying the data to another variable to avoid any changes to original d
df = data.copy()
```

Checking for duplicates and missing values

In [7]:
```
# checking for duplicate values
df.duplicated().sum() ## Complete the code to get total number of duplica
```

Out[7]:
```
0
```

In [8]:
```
# checking for missing values in the data
df.isnull().sum()
```

Out[8]:
```
Ticker Symbol                  0
Security                       0
GICS Sector                    0
GICS Sub Industry              0
Current Price                  0
Price Change                   0
Volatility                     0
ROE                            0
Cash Ratio                     0
Net Cash Flow                  0
Net Income                     0
Earnings Per Share             0
Estimated Shares Outstanding   0
P/E Ratio                      0
P/B Ratio                      0
dtype: int64
```

Statistical summary of the dataset

Let's check the statistical summary of the data.

In [9]:
```
df.describe(include='all').T
```

Out[9]:

	count	unique	top	freq	mean	std
Ticker Symbol	340	340	AAL	1	NaN	NaN
Security	340	340	American Airlines Group	1	NaN	NaN
GICS Sector	340	11	Industrials	53	NaN	NaN
GICS Sub Industry	340	104	Oil & Gas Exploration & Production	16	NaN	NaN
Current Price	340.0	NaN	NaN	NaN	80.862345	98.055086
Price Change	340.0	NaN	NaN	NaN	4.078194	12.006338
Volatility	340.0	NaN	NaN	NaN	1.525976	0.591798
ROE	340.0	NaN	NaN	NaN	39.597059	96.547538
Cash Ratio	340.0	NaN	NaN	NaN	70.023529	90.421331
Net Cash Flow	340.0	NaN	NaN	NaN	55537620.588235	1946365312.175789	-1
Net Income	340.0	NaN	NaN	NaN	1494384602.941176	3940150279.327937	-2
Earnings Per Share	340.0	NaN	NaN	NaN	2.776662	6.587779
Estimated Shares Outstanding	340.0	NaN	NaN	NaN	577028337.754029	845849595.417695
P/E Ratio	340.0	NaN	NaN	NaN	32.612563	44.348731
P/B Ratio	340.0	NaN	NaN	NaN	-1.718249	13.966912

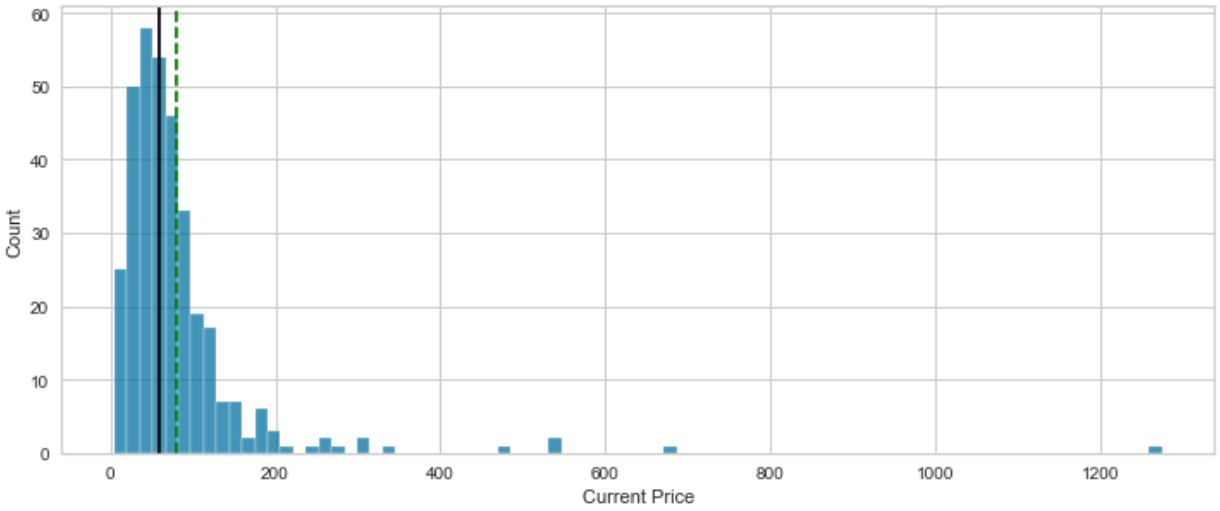
Exploratory Data Analysis

Univariate analysis

In [10]:
```
# function to plot a boxplot and a histogram along the same scale.


def histogram_boxplot(df, feature, figsize=(12, 7), kde=False, bins=None)
    """
    Boxplot and histogram combined

    data: dataframe
    feature: dataframe column
    figsize: size of figure (default (12,7))
    kde: whether to the show density curve (default False)
    bins: number of bins for histogram (default None)
    """
    f2, (ax_box2, ax_hist2) = plt.subplots(
        nrows=2,  # Number of rows of the subplot grid= 2
        sharex=True,  # x-axis will be shared among all subplots
        gridspec_kw={"height_ratios": (0.25, 0.75)},
        figsize=figsize,
    )  # creating the 2 subplots
    sns.boxplot(
        data=df, x=feature, ax=ax_box2, showmeans=True, color="violet"
    )  # boxplot will be created and a star will indicate the mean value
    sns.histplot(
        data=df, x=feature, kde=kde, ax=ax_hist2, bins=bins, palette="win
    ) if bins else sns.histplot(
        data=df, x=feature, kde=kde, ax=ax_hist2
    )  # For histogram
    ax_hist2.axvline(
        df[feature].mean(), color="green", linestyle="--"
    )  # Add mean to the histogram
    ax_hist2.axvline(
        df[feature].median(), color="black", linestyle="-"
    )  # Add median to the histogram
```

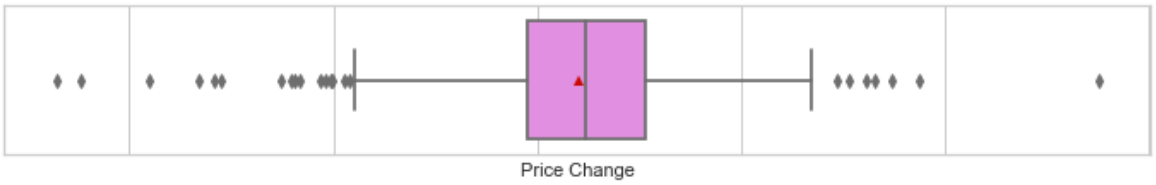
Current Price

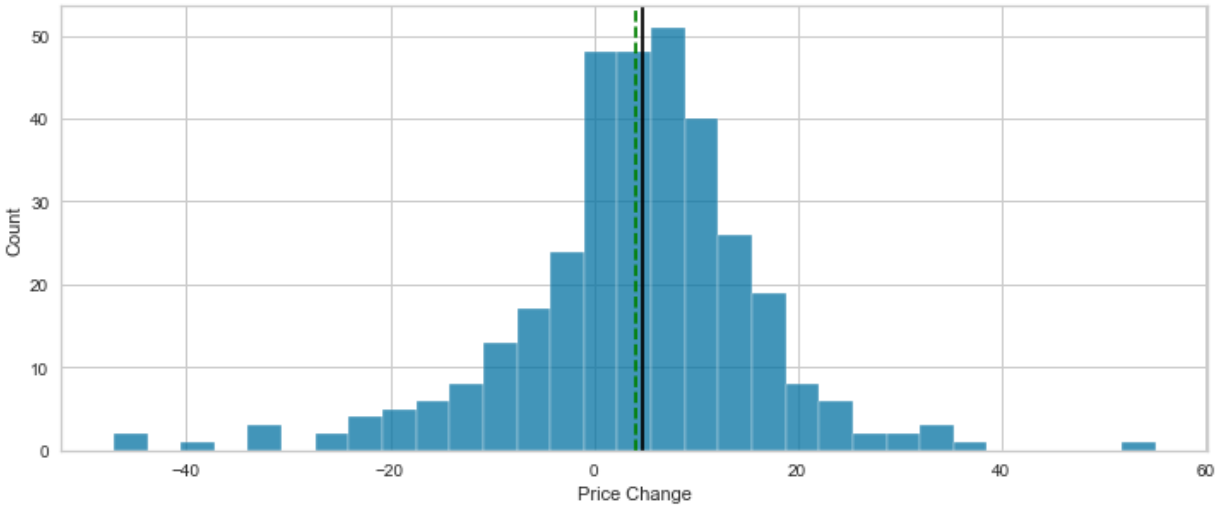
In [11]:
```
histogram_boxplot(df, 'Current Price')
```



Price Change

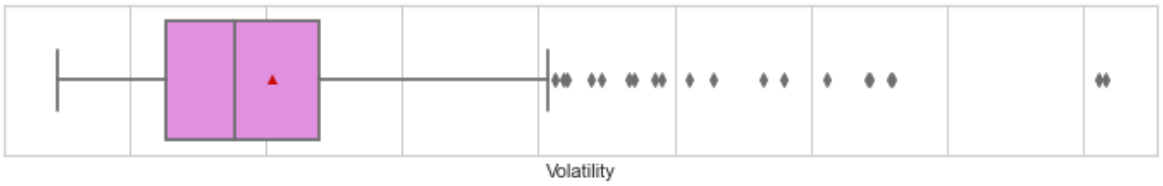
In [12]:
```
histogram_boxplot(df, 'Price Change') ## Complete the code to create hist
```

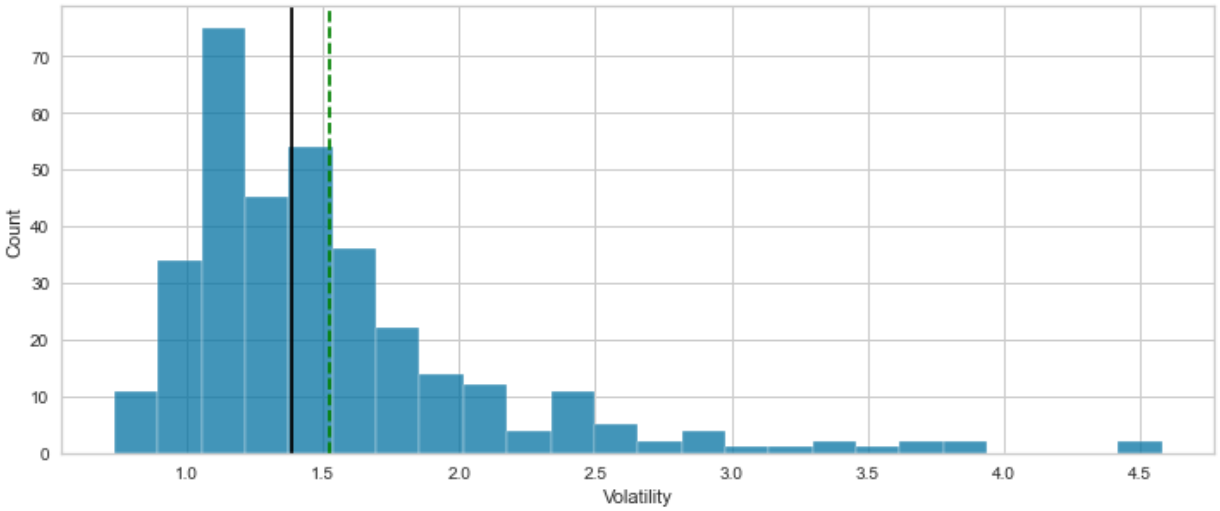




Volatility

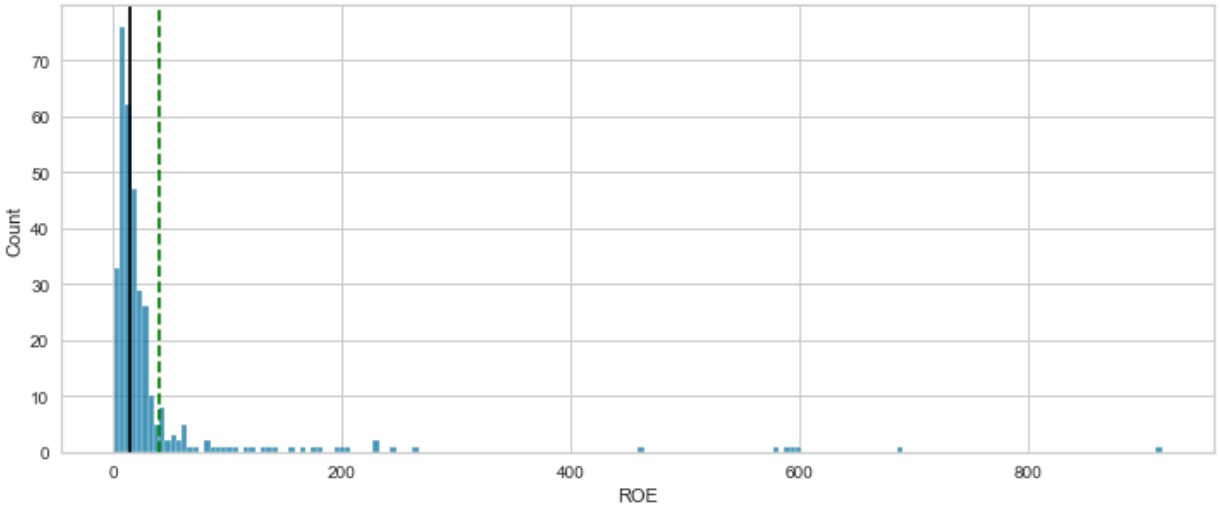
In [13]:
```
histogram_boxplot(df, 'Volatility') ## Complete the code to create histog
```





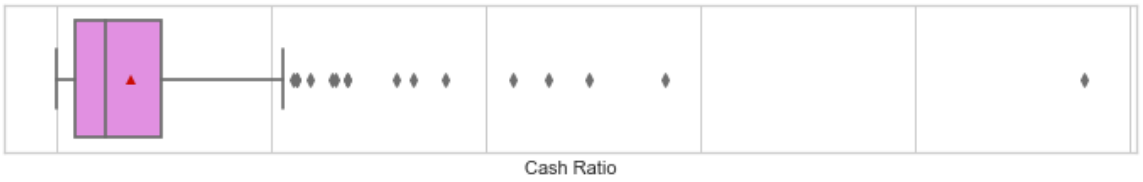
ROE

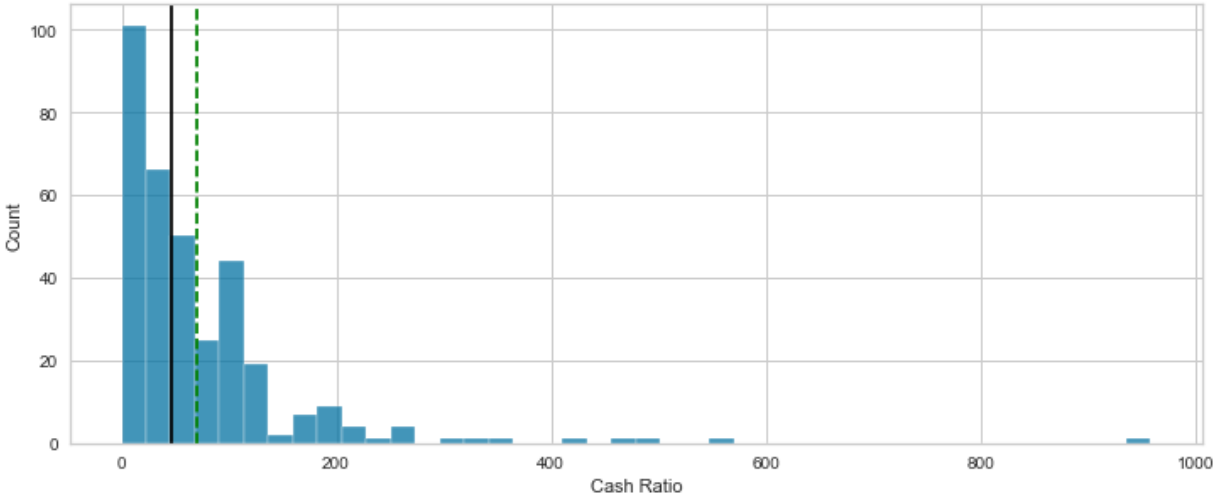
In [14]:
```
histogram_boxplot(df, 'ROE')  ## Complete the code to create histogram_bo
```



Cash Ratio

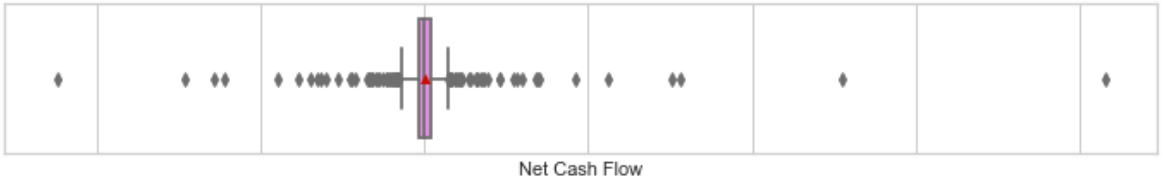
In [15]:
```
histogram_boxplot(df, 'Cash Ratio')  ## Complete the code to create histo
```

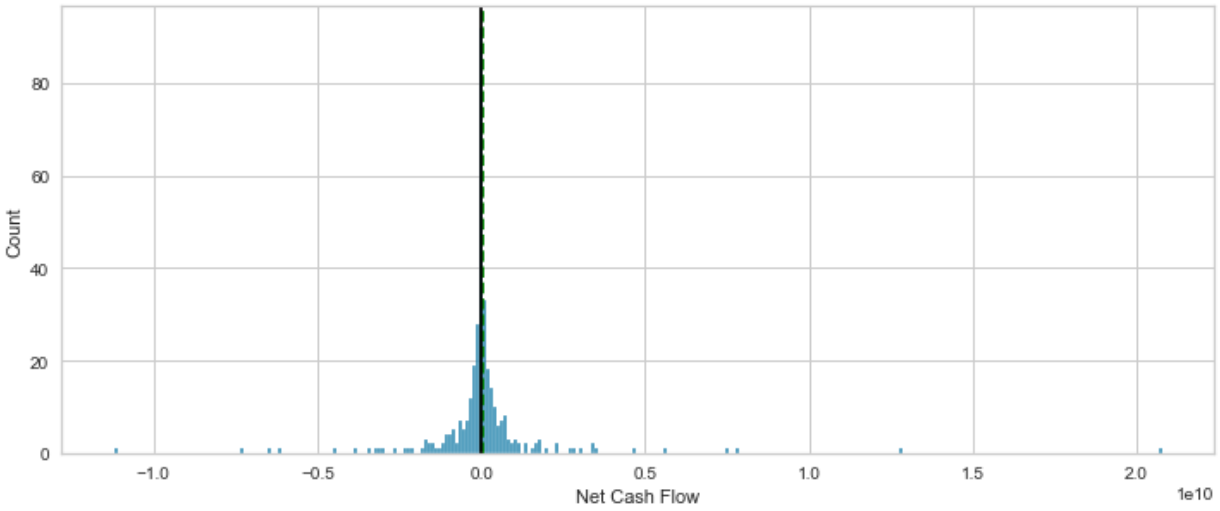




Net Cash Flow

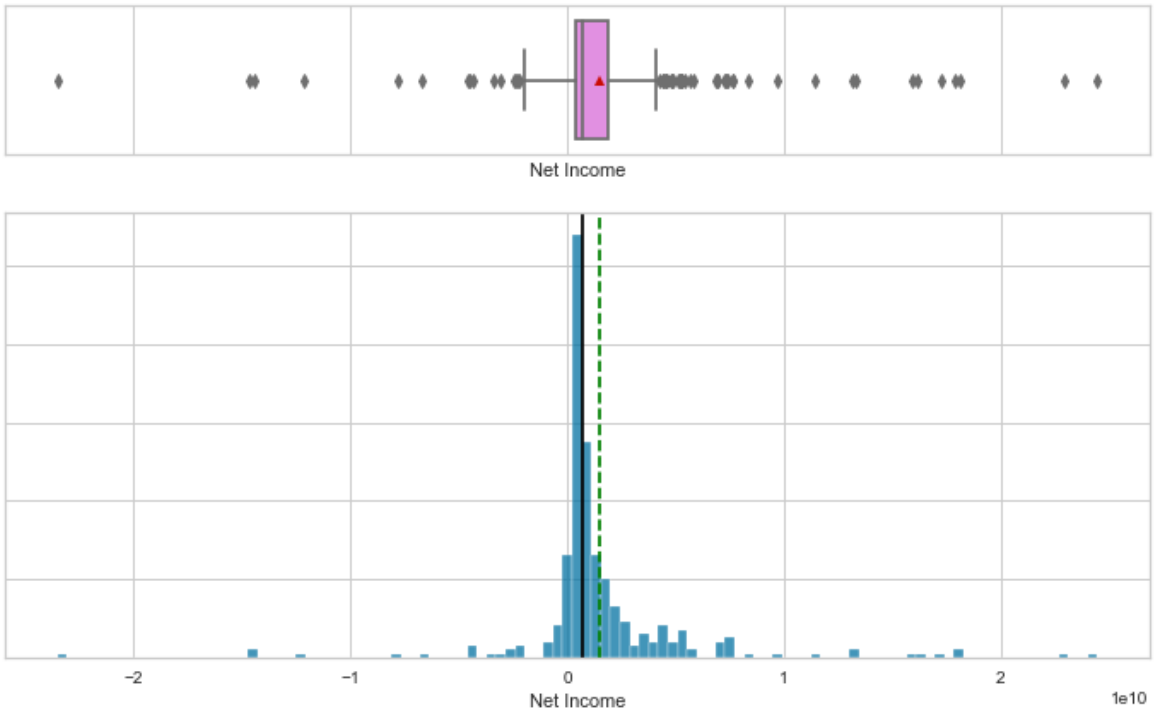
In [16]:
```
histogram_boxplot(df, 'Net Cash Flow')  ## Complete the code to create hi
```





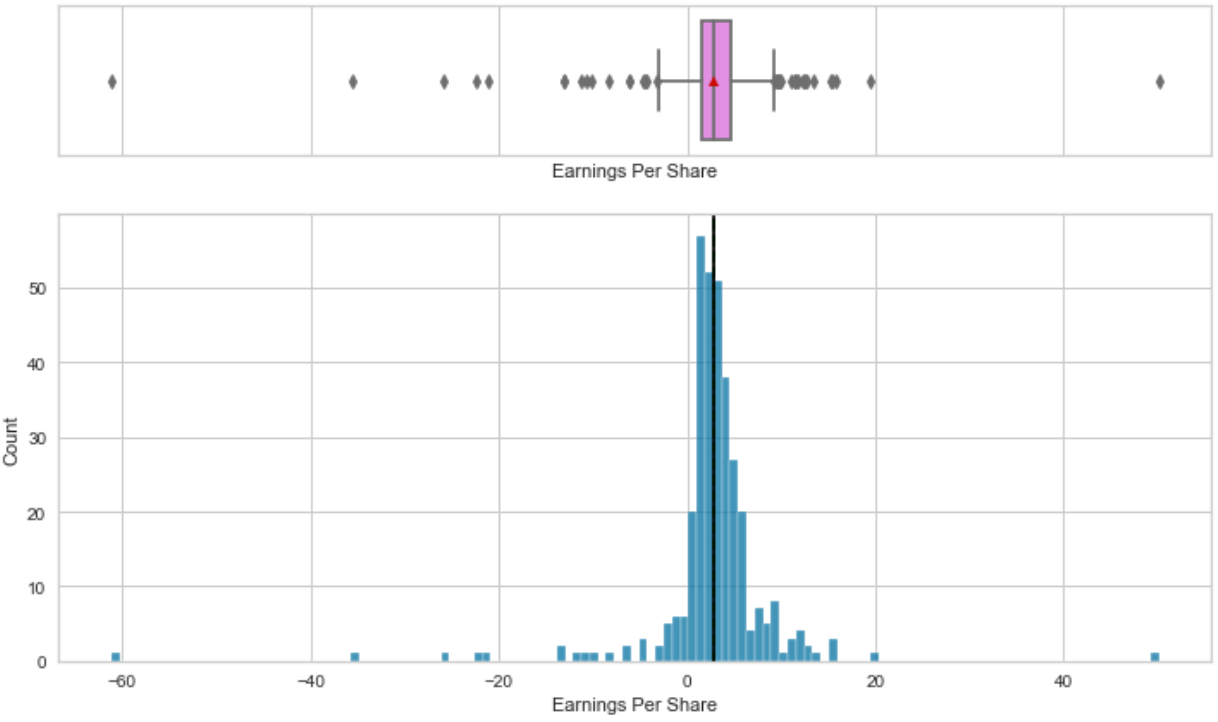
Net Income

In [17]:
```
histogram_boxplot(df, 'Net Income') ## Complete the code to create histog
```



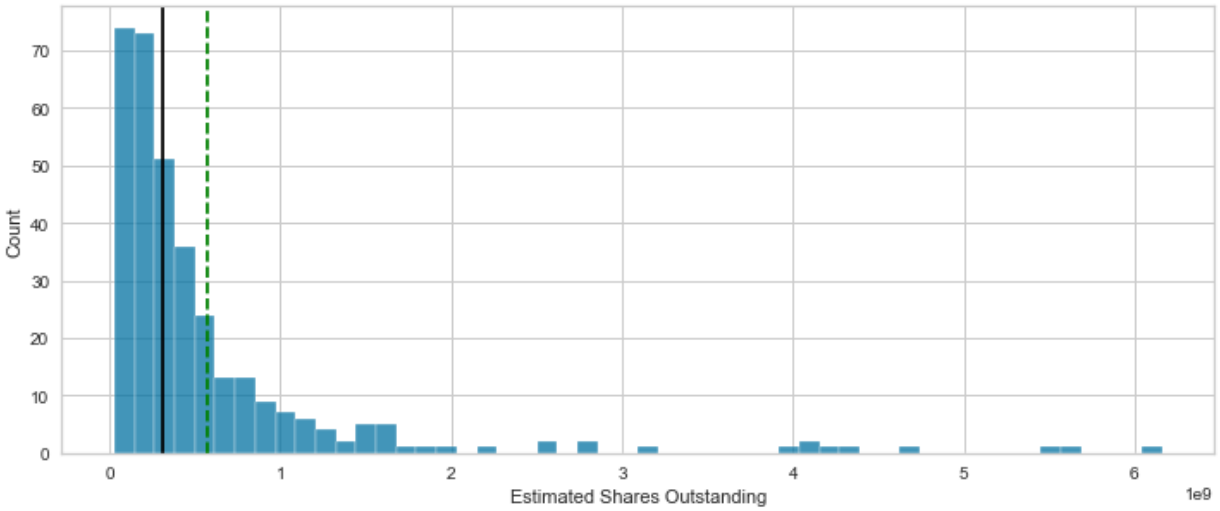
Earnings Per Share

In [19]:
```
histogram_boxplot(df, 'Earnings Per Share')  ## Complete the code to crea
```



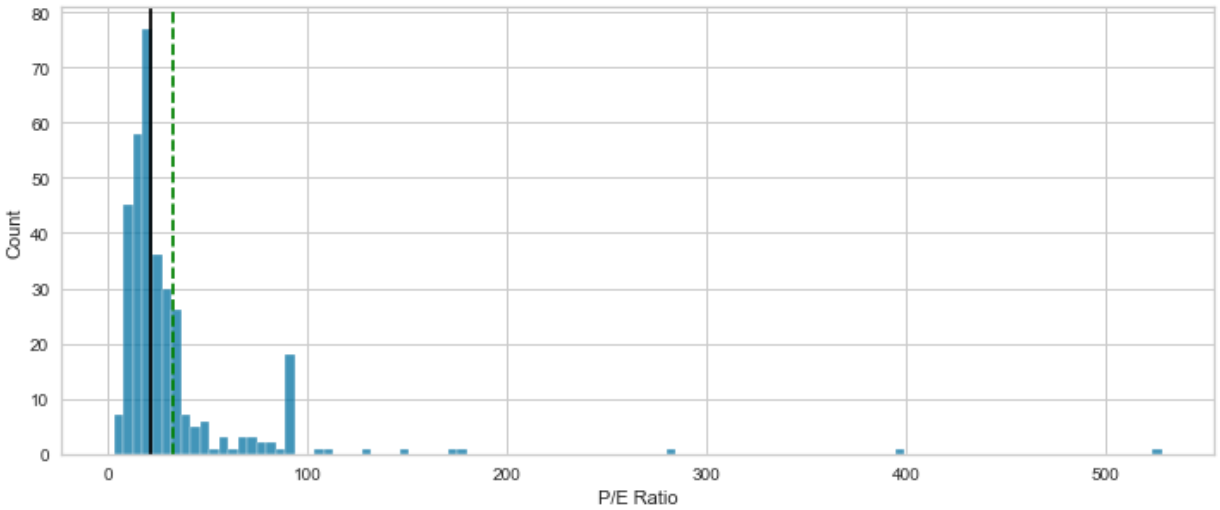
Estimated Shares Outstanding

In [20]:
```
histogram_boxplot(df, 'Estimated Shares Outstanding') ## Complete the cod
```



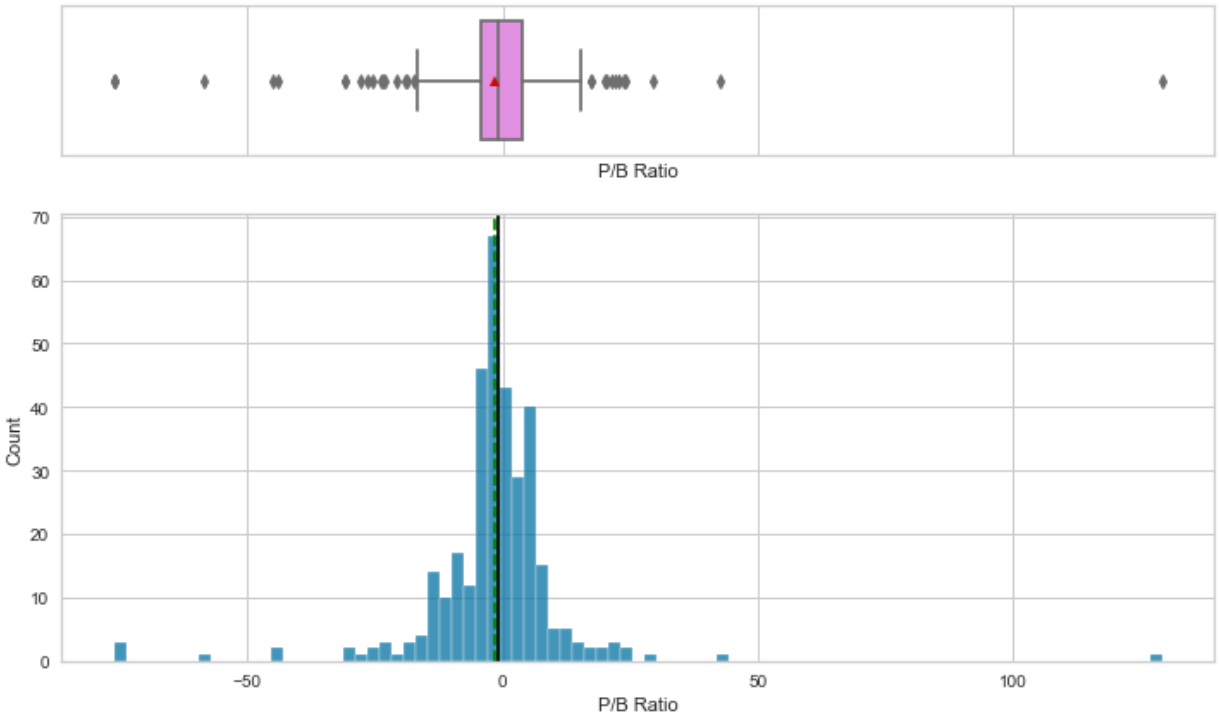
P/E Ratio

In [23]:
```
histogram_boxplot(df, 'P/E Ratio')  ## Complete the code to create histog
```



P/B Ratio

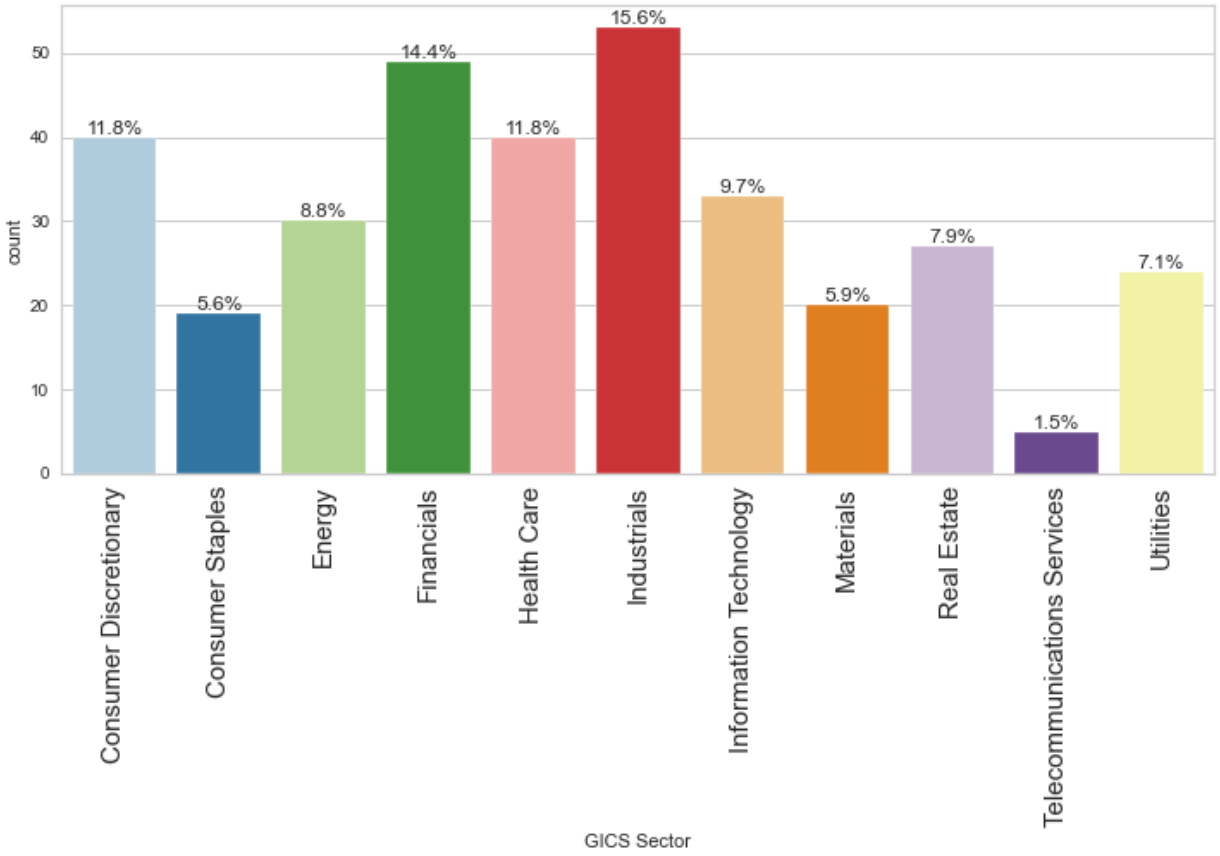
In [24]:
```
histogram_boxplot(df, 'P/B Ratio') ## Complete the code to create histogr
```


P/B Ratio
70
60
50
40
30
20
10
0
Count
−50
0
50
100
P/B Ratio

In [25]:
```
# function to create labeled barplots


def labeled_barplot(df, feature, perc=False, n=None):
    """
    Barplot with percentage at the top

    data: dataframe
    feature: dataframe column
    perc: whether to display percentages instead of count (default is Fal
    n: displays the top n category levels (default is None, i.e., display
    """

    total = len(df[feature])  # length of the column
    count = df[feature].nunique()
    if n is None:
        plt.figure(figsize=(count + 1, 5))
    else:
        plt.figure(figsize=(n + 1, 5))

    plt.xticks(rotation=90, fontsize=15)
    ax = sns.countplot(
        data=df,
        x=feature,
        palette="Paired",
        order=df[feature].value_counts().index[:n].sort_values(),
    )

    for p in ax.patches:
        if perc == True:
            label = "{:.1f}%".format(
                100 * p.get_height() / total
            )  # percentage of each class of the category
        else:
            label = p.get_height()  # count of each level of the category

        x = p.get_x() + p.get_width() / 2  # width of the plot
        y = p.get_height()  # height of the plot

        ax.annotate(
            label,
            (x, y),
            ha="center",
            va="center",
            size=12,
            xytext=(0, 5),
            textcoords="offset points",
        )  # annotate the percentage

    plt.show()  # show the plot
```

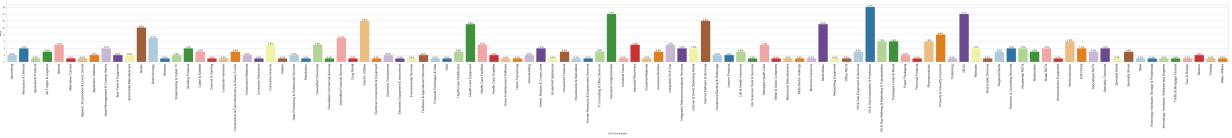
GICS Sector

In [26]:
```
labeled_barplot(df, 'GICS Sector', perc=True)
```



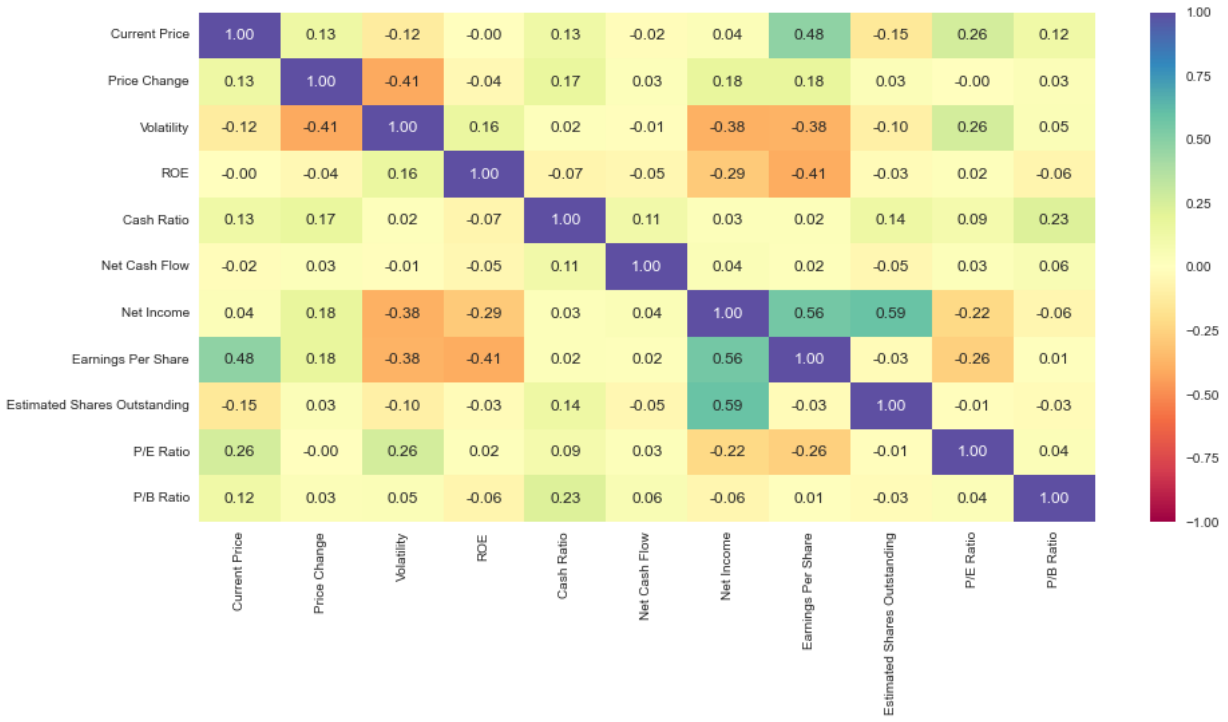
GICS Sub Industry

In [28]:
```
labeled_barplot(df, 'GICS Sub Industry', perc=True)  ## Complete the code
```



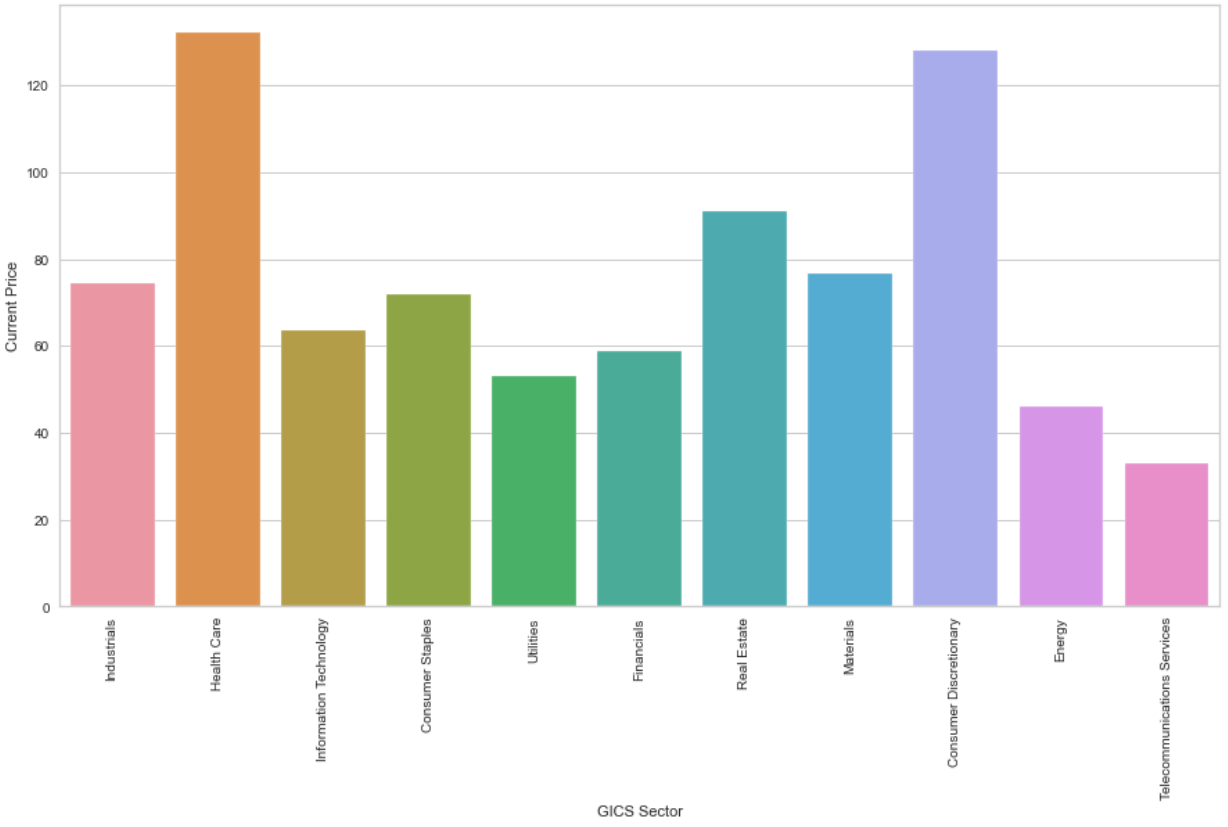
Bivariate Analysis

In [29]:
```
# correlation check
plt.figure(figsize=(15, 7))
sns.heatmap(
    df.corr(), annot=True, vmin=-1, vmax=1, fmt=".2f", cmap="Spectral"
)
plt.show()
```



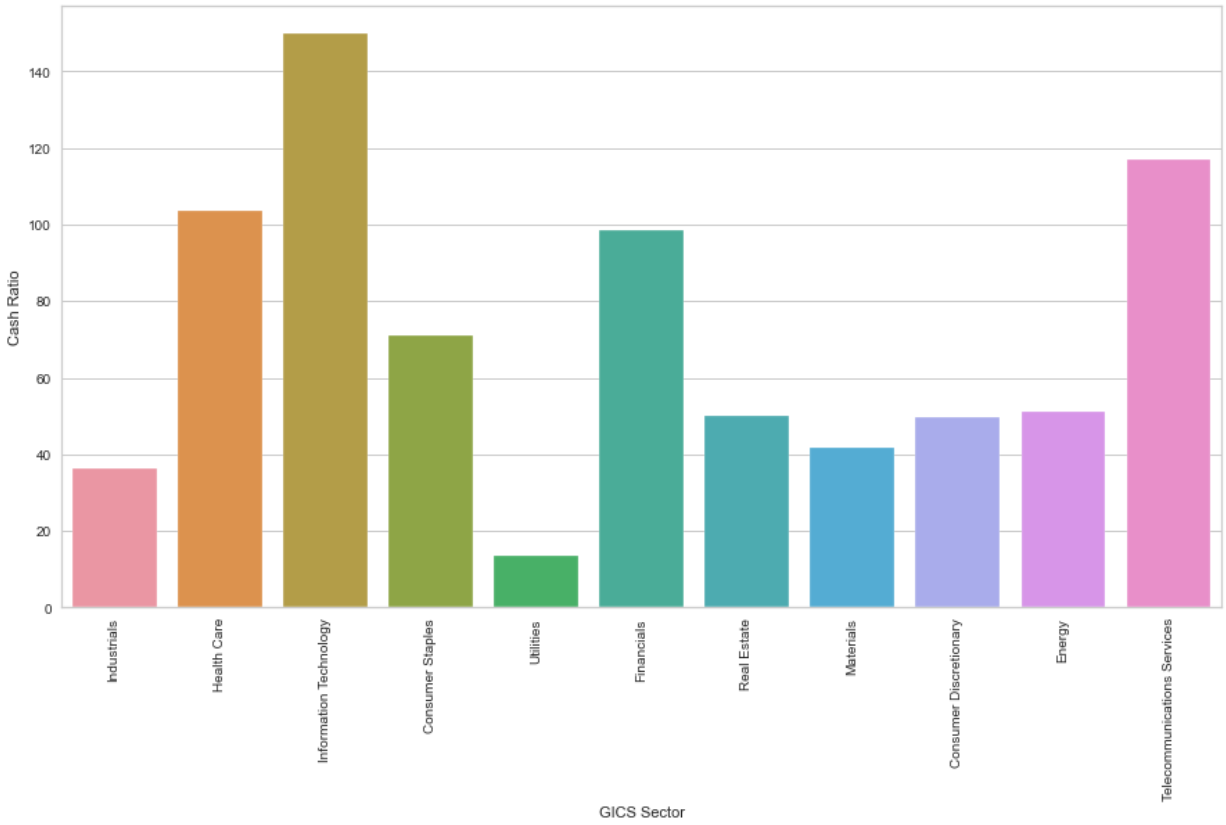
Let's check the stocks of which economic sector have seen the maximum price increase on average.

In [32]:
```
plt.figure(figsize=(15,8))
sns.barplot(data=df, x='GICS Sector', y='Current Price', ci=False)
plt.xticks(rotation=90)
plt.show()
```



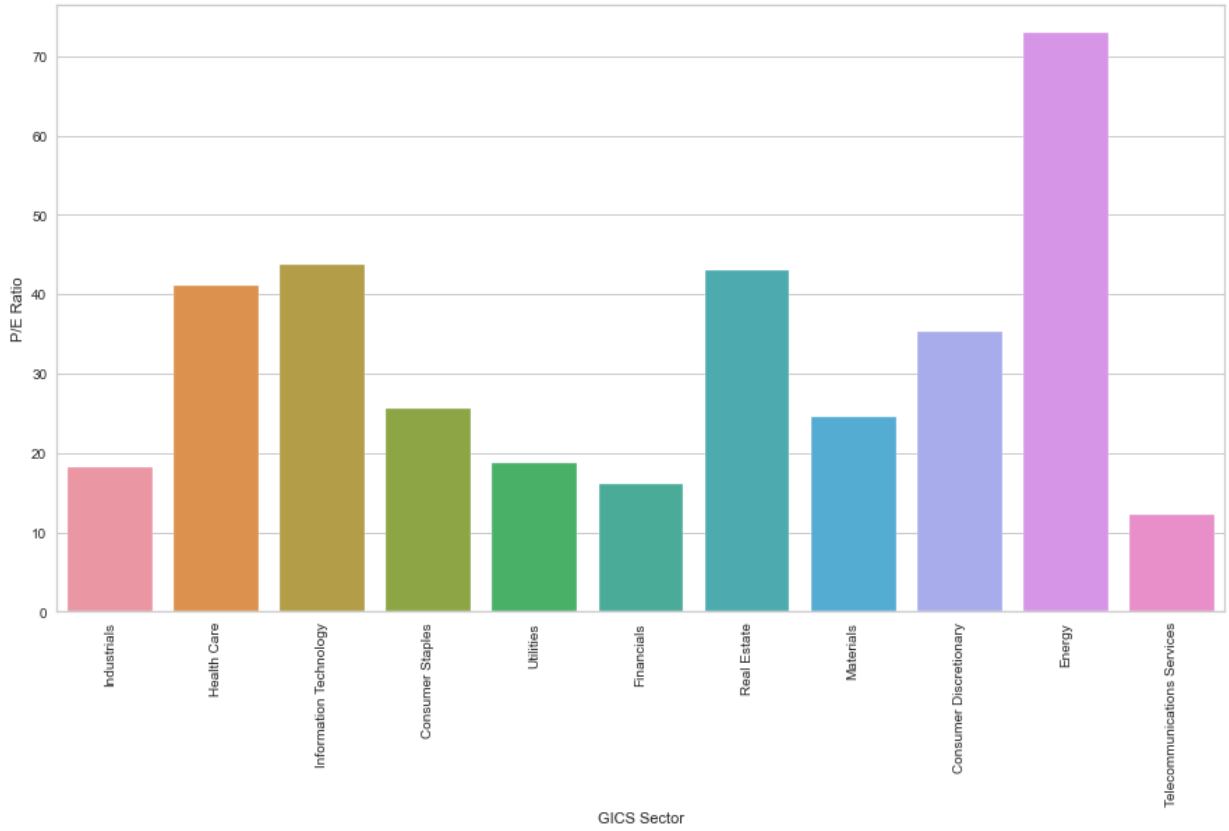
Cash ratio provides a measure of a company's ability to cover its short-term obligations using only cash and cash equivalents. Let's see how the average cash ratio varies across economic sectors.

In [33]:
```
plt.figure(figsize=(15,8))
sns.barplot(data=df, x='GICS Sector', y='Cash Ratio', ci=False)  ## Compl
plt.xticks(rotation=90)
plt.show()
```



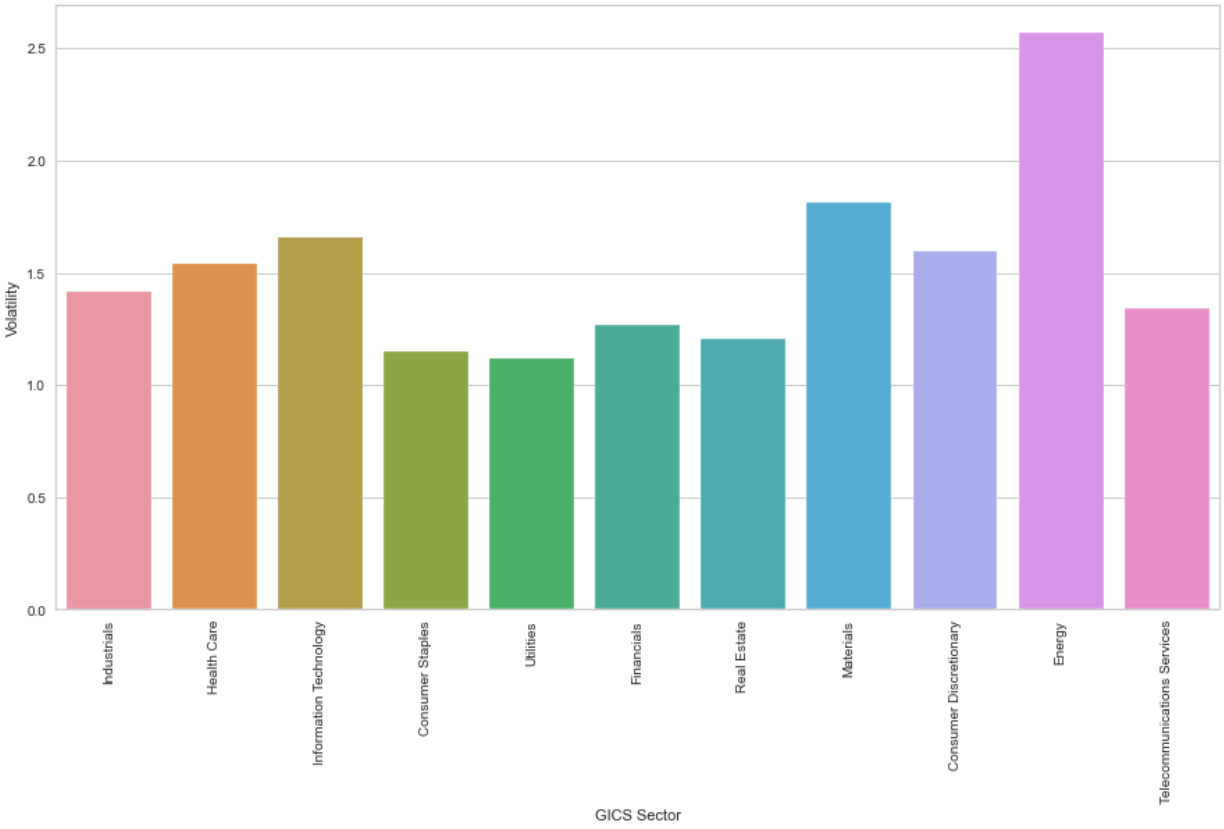
P/E ratios can help determine the relative value of a company's shares as they signify the amount of money an investor is willing to invest in a single share of a company per dollar of its earnings. Let's see how the P/E ratio varies, on average, across economic sectors.

In [34]:
```
plt.figure(figsize=(15,8))
sns.barplot(data=df, x='GICS Sector', y='P/E Ratio', ci=False)  ## Comple
plt.xticks(rotation=90)
plt.show()
```



Volatility accounts for the fluctuation in the stock price. A stock with high volatility will witness sharper price changes, making it a riskier investment. Let's see how volatility varies, on average, across economic sectors.

In [35]:
```
plt.figure(figsize=(15,8))
sns.barplot(data=df,  x='GICS Sector', y='Volatility', ci=False)  ## Comp
plt.xticks(rotation=90)
plt.show()
```

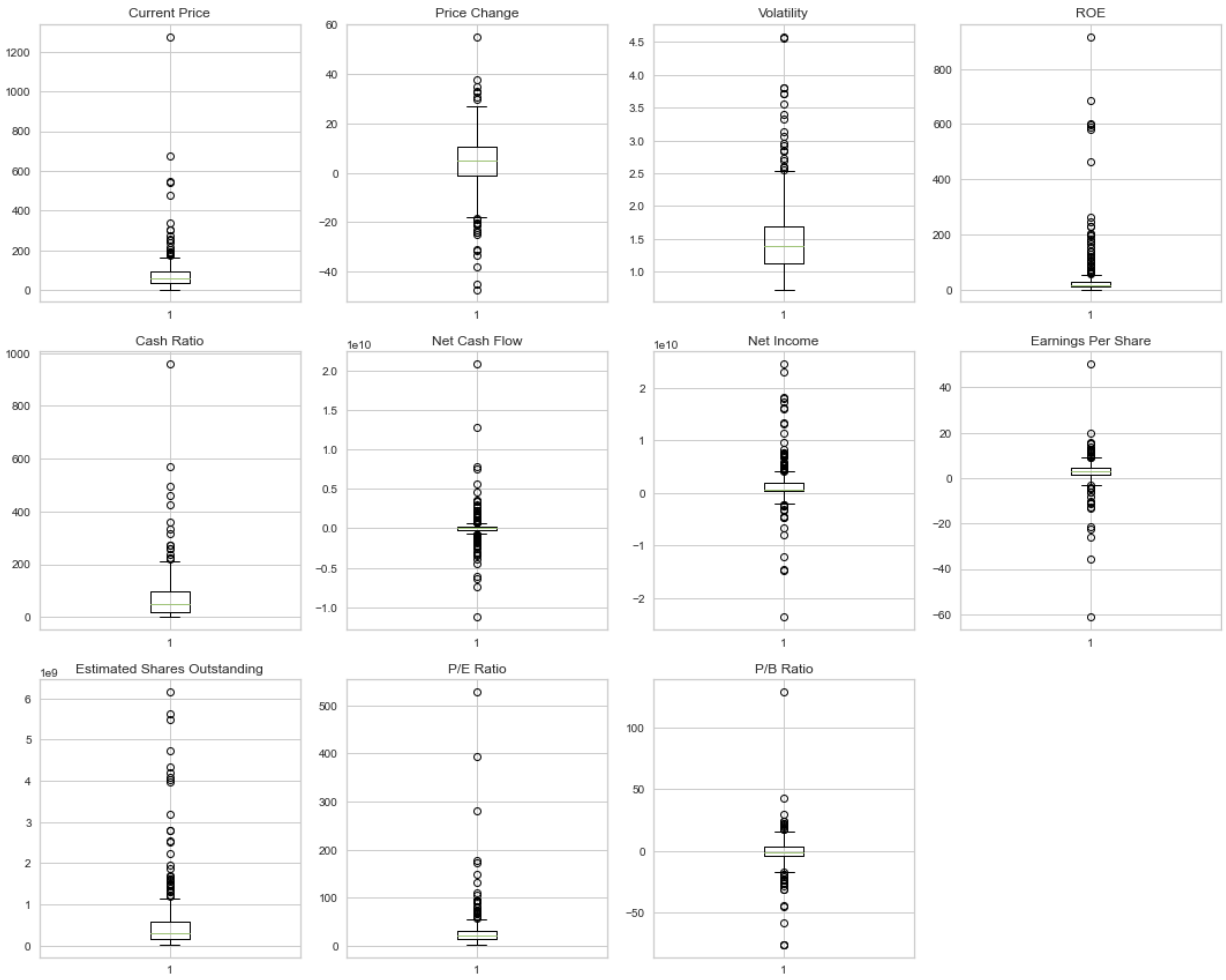


Data Preprocessing

Outlier Check

- Let's plot the boxplots of all numerical columns to check for outliers.

In [36]:
```
plt.figure(figsize=(15, 12))

numeric_columns = df.select_dtypes(include=np.number).columns.tolist()

for i, variable in enumerate(numeric_columns):
    plt.subplot(3, 4, i + 1)
    plt.boxplot(df[variable], whis=1.5)
    plt.tight_layout()
    plt.title(variable)

plt.show()
```



Scaling

- Let's scale the data before we proceed with clustering.

In [37]:
```
# scaling the data before clustering
scaler = StandardScaler()
subset = df.select_dtypes(include=np.number)
subset_scaled = scaler.fit_transform(subset)
```

In [38]:
```
# creating a dataframe of the scaled data
subset_scaled_df = pd.DataFrame(subset_scaled, columns=subset.columns)
```

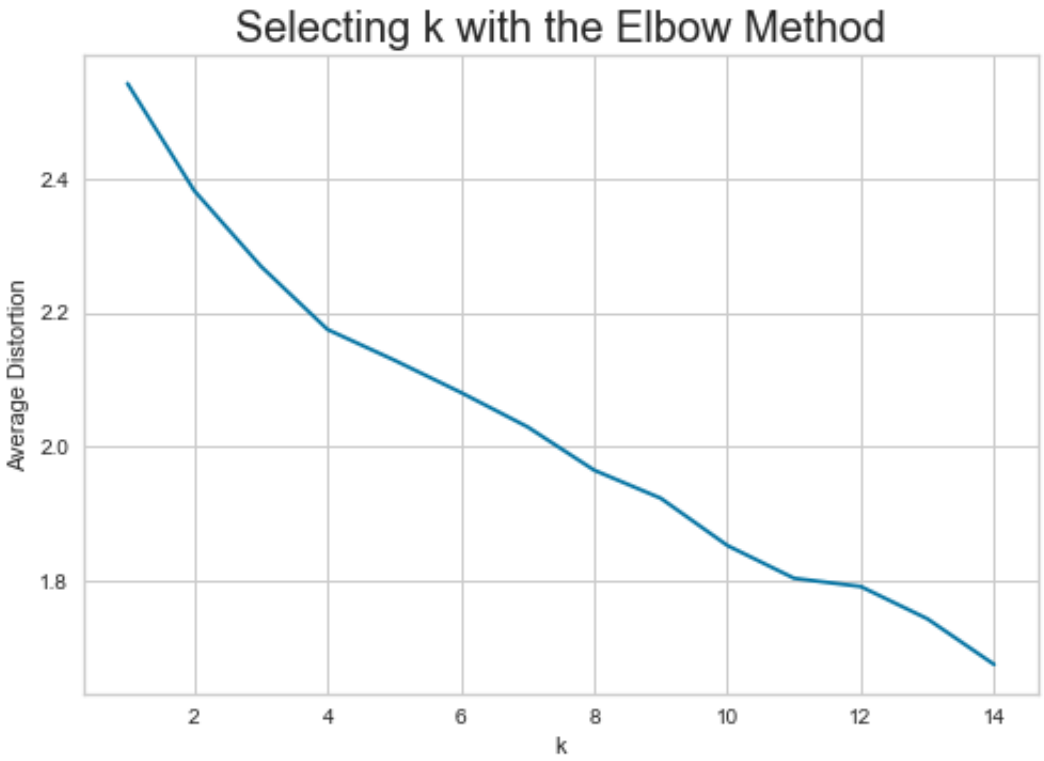
K-means Clustering

Checking Elbow Plot

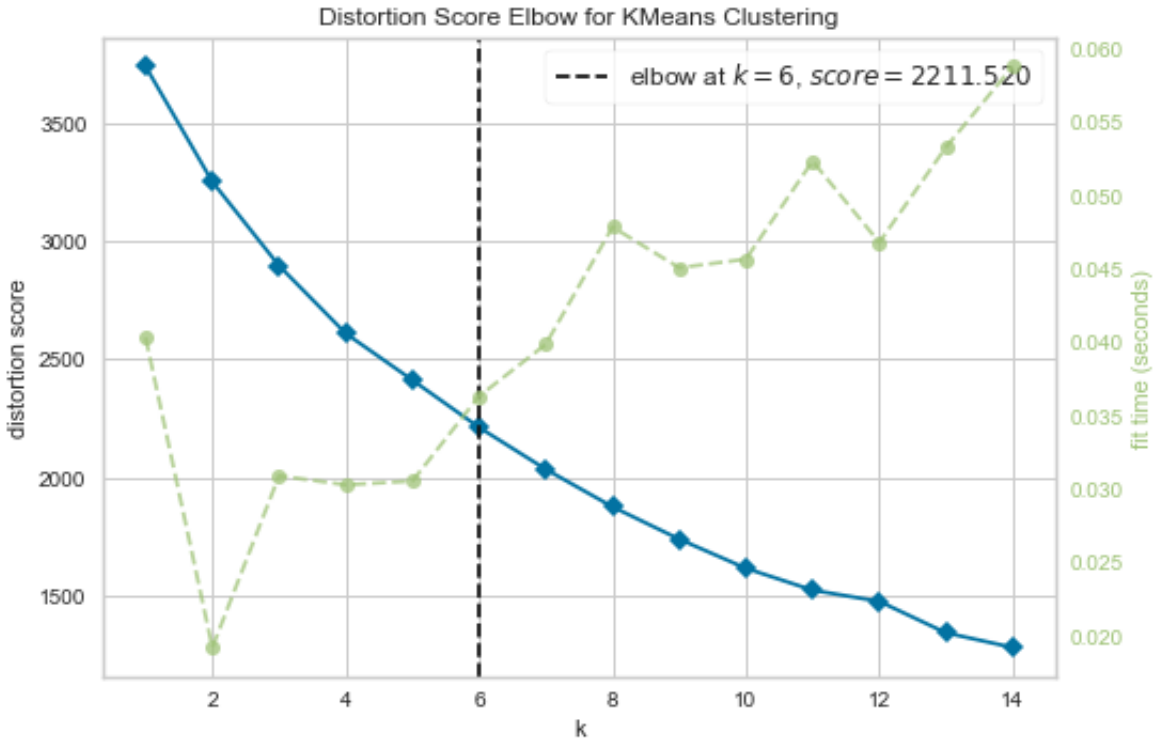
In [39]:
```
k_means_df = subset_scaled_df.copy()
```

In [40]:
```
clusters = range(1, 15)
meanDistortions = []

for k in clusters:
    model = KMeans(n_clusters=k, random_state=1)
    model.fit(subset_scaled_df)
    prediction = model.predict(k_means_df)
    distortion = (
        sum(np.min(cdist(k_means_df, model.cluster_centers_, "euclidean")
        / k_means_df.shape[0]
    )

    meanDistortions.append(distortion)

    print("Number of Clusters:", k, "\tAverage Distortion:", distortion)

plt.plot(clusters, meanDistortions, "bx-")
plt.xlabel("k")
plt.ylabel("Average Distortion")
plt.title("Selecting k with the Elbow Method", fontsize=20)
plt.show()
```

```
Number of Clusters: 1   Average Distortion: 2.5425069919221697
Number of Clusters: 2   Average Distortion: 2.382318498894466
Number of Clusters: 3   Average Distortion: 2.2692367155390745
Number of Clusters: 4   Average Distortion: 2.1745559827866363
Number of Clusters: 5   Average Distortion: 2.128799332840716
Number of Clusters: 6   Average Distortion: 2.080400099226289
Number of Clusters: 7   Average Distortion: 2.0289794220177395
Number of Clusters: 8   Average Distortion: 1.964144163389972
Number of Clusters: 9   Average Distortion: 1.9221492045198068
Number of Clusters: 10  Average Distortion: 1.8513913649973124
Number of Clusters: 11  Average Distortion: 1.8024134734578485
Number of Clusters: 12  Average Distortion: 1.7900931879652673
Number of Clusters: 13  Average Distortion: 1.7417609203336912
Number of Clusters: 14  Average Distortion: 1.673559857259703
```



In [41]:
```
model = KMeans(random_state=1)
visualizer = KElbowVisualizer(model, k=(1, 15), timings=True)
visualizer.fit(k_means_df)  # fit the data to the visualizer
visualizer.show()  # finalize and render figure
```

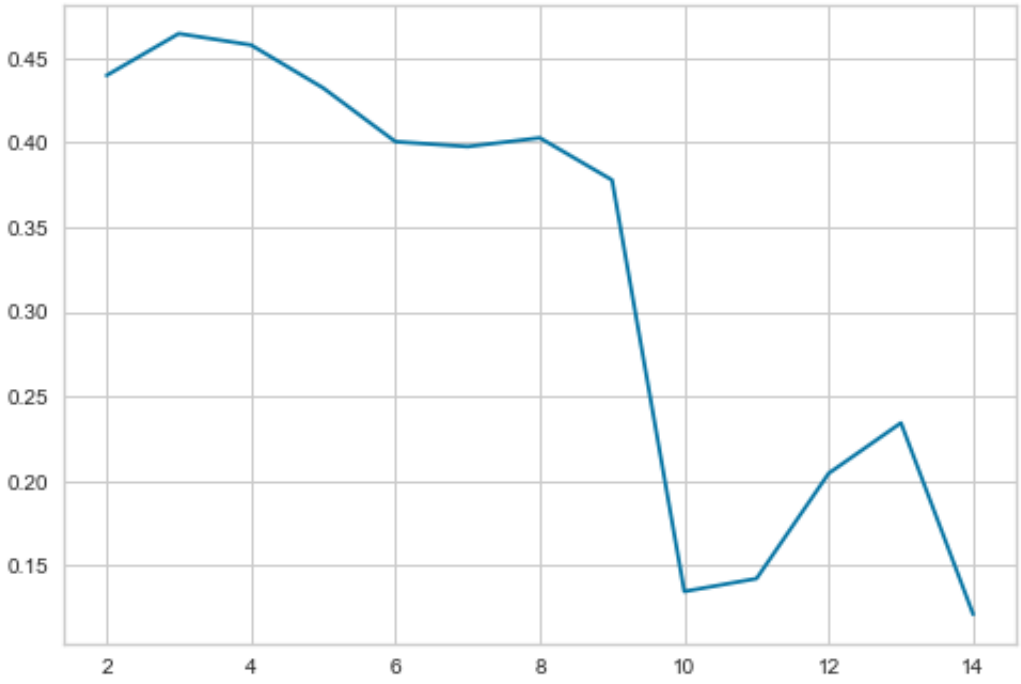


Out[41]:
```
<AxesSubplot:title={'center':'Distortion Score Elbow for KMeans Clusterin
g'}, xlabel='k', ylabel='distortion score'>
```

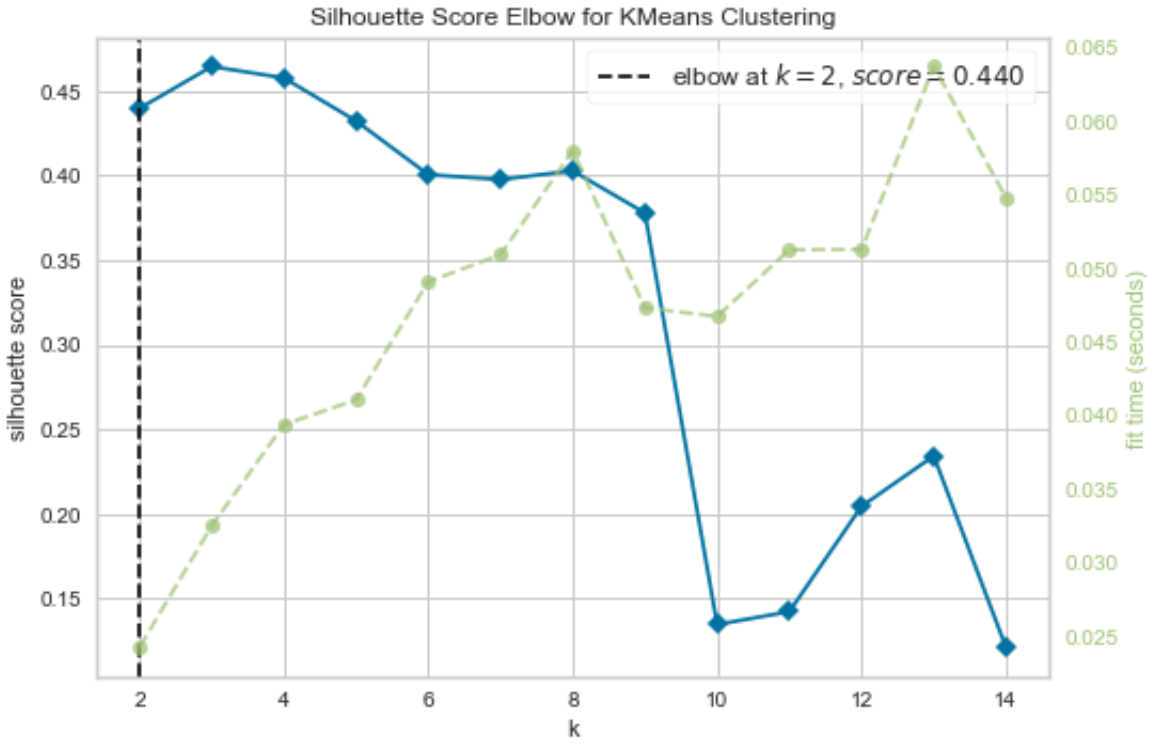
Let's check the silhouette scores

In [42]:
```
sil_score = []
cluster_list = range(2, 15)
for n_clusters in cluster_list:
    clusterer = KMeans(n_clusters=n_clusters, random_state=1)
    preds = clusterer.fit_predict((subset_scaled_df))
    score = silhouette_score(k_means_df, preds)
    sil_score.append(score)
    print("For n_clusters = {}, the silhouette score is {})".format(n_clu

plt.plot(cluster_list, sil_score)
plt.show()
```

```
For n_clusters = 2, the silhouette score is 0.43969639509980457)
For n_clusters = 3, the silhouette score is 0.4644405674779404)
For n_clusters = 4, the silhouette score is 0.4577225970476733)
For n_clusters = 5, the silhouette score is 0.43228336443659804)
For n_clusters = 6, the silhouette score is 0.4005422737213617)
For n_clusters = 7, the silhouette score is 0.3976335364987305)
For n_clusters = 8, the silhouette score is 0.40278401969450467)
For n_clusters = 9, the silhouette score is 0.3778585981433699)
For n_clusters = 10, the silhouette score is 0.13458938329968687)
For n_clusters = 11, the silhouette score is 0.1421832155528444)
For n_clusters = 12, the silhouette score is 0.2044669621527429)
For n_clusters = 13, the silhouette score is 0.23424874810104204)
For n_clusters = 14, the silhouette score is 0.12102526472829901)
```

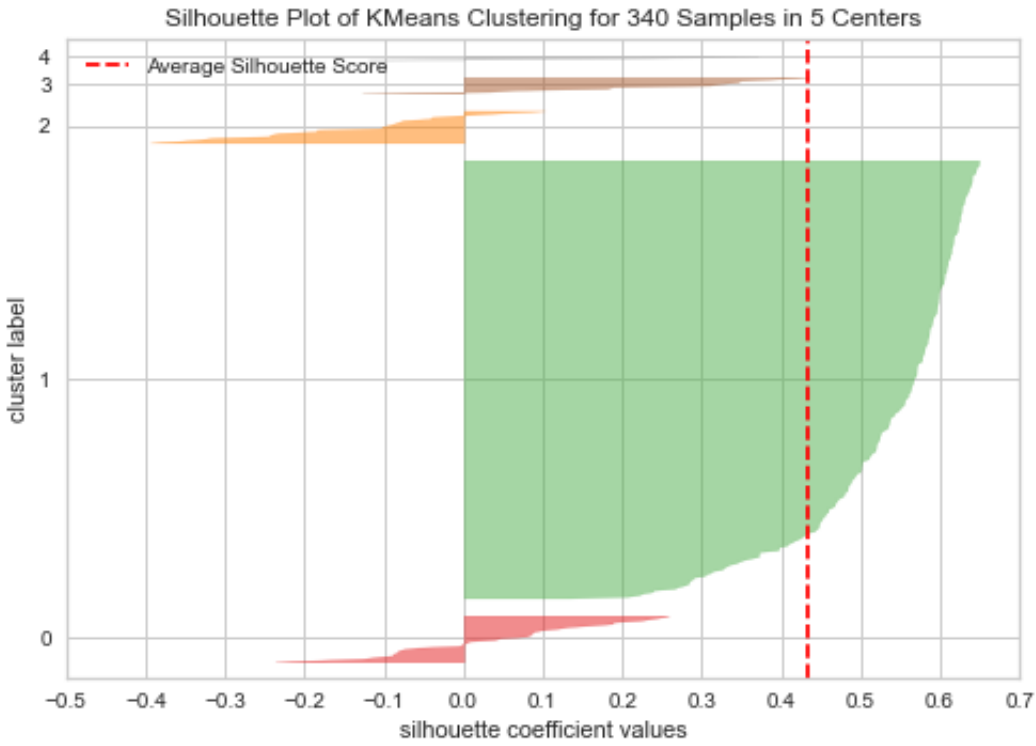


In [43]:
```
model = KMeans(random_state=1)
visualizer = KElbowVisualizer(model, k=(2, 15), metric="silhouette", timi
visualizer.fit(k_means_df)  # fit the data to the visualizer
visualizer.show()  # finalize and render figure
```



Out[43]:
```
<AxesSubplot:title={'center':'Silhouette Score Elbow for KMeans Clusterin
g'}, xlabel='k', ylabel='silhouette score'>
```

In [45]:
```
# finding optimal no. of clusters with silhouette coefficients
visualizer = SilhouetteVisualizer(KMeans(n_clusters=5, random_state=1))
visualizer.fit(k_means_df)
visualizer.show()
```



Out[45]:
```
<AxesSubplot:title={'center':'Silhouette Plot of KMeans Clustering for 34
0 Samples in 5 Centers'}, xlabel='silhouette coefficient values', ylabel=
'cluster label'>
```

Creating Final Model

In [46]:
```
# final K-means model
kmeans = KMeans(n_clusters=4, random_state=1)
kmeans.fit(k_means_df)
```

Out[46]:
```
KMeans(n_clusters=4, random_state=1)
```

In [47]:
```
# creating a copy of the original data
df1 = df.copy()

# adding kmeans cluster labels to the original and scaled dataframes
k_means_df["KM_segments"] = kmeans.labels_
df1["KM_segments"] = kmeans.labels_
```

Cluster Profiling

In [59]:
```
km_cluster_profile = df1.groupby("KM_segments").mean()  ## Complete the c
```

In [60]:
```
km_cluster_profile["count_in_each_segment"] = (
    df1.groupby("KM_segments")["Security"].count().values  ## Complete th
)
```

In [61]:
```
km_cluster_profile.style.highlight_max(color="lightgreen", axis=0)
```

Out[61]:

KM_segments	Current Price	Price Change	Volatility	ROE	Cash Ratio	Net Cash
0	72.399112	5.066225	1.388319	34.620939	53.000000	-14046223.82
1	50.517273	5.747586	1.130399	31.090909	75.909091	-1072272727.27
2	38.099260	-15.370329	2.910500	107.074074	50.037037	-159428481.48
3	234.170932	13.400685	1.729989	25.600000	277.640000	1554926560.000

In [62]:
```
## Complete the code to print the companies in each cluster
for cl in df1["KM_segments"].unique():
    print("In cluster {}, the following companies are present:".format(cl
    print(df1[df1["KM_segments"] == cl]["Security"].unique())
    print()
```

```
In cluster 0, the following companies are present:
['American Airlines Group' 'AbbVie' 'Abbott Laboratories'
 'Adobe Systems Inc' 'Archer-Daniels-Midland Co' 'Ameren Corp'
 'American Electric Power' 'AFLAC Inc'
 'American International Group, Inc.' 'Apartment Investment & Mgmt'
 'Assurant Inc' 'Arthur J. Gallagher & Co.' 'Akamai Technologies Inc'
 'Albemarle Corp' 'Alaska Air Group Inc' 'Allstate Corp' 'Allegion'
 'Applied Materials Inc' 'AMETEK Inc' 'Affiliated Managers Group Inc'
 'Ameriprise Financial' 'American Tower Corp A' 'AutoNation Inc'
```

```
       'Anthem Inc.' 'Aon plc' 'Amphenol Corp' 'Arconic Inc'
       'Activision Blizzard' 'AvalonBay Communities, Inc.' 'Broadcom'
       'American Water Works Company Inc' 'American Express Co' 'Boeing Company
      '
       'Baxter International Inc.' 'BB&T Corporation' 'Bard (C.R.) Inc.'
       'The Bank of New York Mellon Corp.' 'Ball Corp' 'Bristol-Myers Squibb'
       'Boston Scientific' 'BorgWarner' 'Boston Properties' 'Caterpillar Inc.'
       'Chubb Limited' 'CBRE Group' 'Crown Castle International Corp.'
       'Carnival Corp.' 'CF Industries Holdings Inc' 'Citizens Financial Group'
       'Church & Dwight' 'C. H. Robinson Worldwide' 'Charter Communications'
       'CIGNA Corp.' 'Cincinnati Financial' 'Colgate-Palmolive' 'Comerica Inc.'
       'CME Group Inc.' 'Cummins Inc.' 'CMS Energy' 'Centene Corporation'
       'CenterPoint Energy' 'Capital One Financial' 'The Cooper Companies'
       'CSX Corp.' 'CenturyLink Inc' 'Cognizant Technology Solutions'
       'Citrix Systems' 'CVS Health' 'Chevron Corp.' 'Dominion Resources'
       'Delta Air Lines' 'Du Pont (E.I.)' 'Deere & Co.'
       'Discover Financial Services' 'Quest Diagnostics' 'Danaher Corp.'
       'The Walt Disney Company' 'Discovery Communications-A'
       'Discovery Communications-C' 'Delphi Automotive' 'Digital Realty Trust'
       'Dun & Bradstreet' 'Dover Corp.' 'Dr Pepper Snapple Group' 'Duke Energy'
       'DaVita Inc.' 'eBay Inc.' 'Ecolab Inc.' 'Consolidated Edison'
       'Equifax Inc.' "Edison Int'l" 'Eastman Chemical' 'Equity Residential'
       'Eversource Energy' 'Essex Property Trust, Inc.' 'E*Trade'
       'Eaton Corporation' 'Entergy Corp.' 'Exelon Corp.' "Expeditors Int'l"
       'Expedia Inc.' 'Extra Space Storage' 'Fastenal Co'
       'Fortune Brands Home & Security' 'FirstEnergy Corp'
       'Fidelity National Information Services' 'Fiserv Inc' 'FLIR Systems'
       'Fluor Corp.' 'Flowserve Corporation' 'FMC Corporation'
       'Federal Realty Investment Trust' 'General Dynamics'
       'General Growth Properties Inc.' 'Corning Inc.' 'General Motors'
       'Genuine Parts' 'Garmin Ltd.' 'Goodyear Tire & Rubber'
       'Grainger (W.W.) Inc.' 'Hasbro Inc.' 'Huntington Bancshares'
       'HCA Holdings' 'Welltower Inc.' 'HCP Inc.' 'Hartford Financial Svc.Gp.'
       'Harley-Davidson' "Honeywell Int'l Inc." 'HP Inc.' 'Hormel Foods Corp.'
       'Henry Schein' 'Host Hotels & Resorts' 'The Hershey Company'
       'Humana Inc.' 'International Business Machines' 'IDEXX Laboratories'
       'Intl Flavors & Fragrances' 'International Paper' 'Interpublic Group'
       'Iron Mountain Incorporated' 'Illinois Tool Works' 'Invesco Ltd.'
       'J. B. Hunt Transport Services' 'Jacobs Engineering Group'
       'Juniper Networks' 'Kimco Realty' 'Kimberly-Clark' 'Kansas City Southern
      '
       'Leggett & Platt' 'Lennar Corp.' 'Laboratory Corp. of America Holding'
       'LKQ Corporation' 'L-3 Communications Holdings' 'Lilly (Eli) & Co.'
       'Lockheed Martin Corp.' 'Alliant Energy Corp' 'Leucadia National Corp.'
       'Southwest Airlines' 'Level 3 Communications' 'LyondellBasell'
       'Mastercard Inc.' 'Mid-America Apartments' 'Macerich' "Marriott Int'l."
       'Masco Corp.' 'Mattel Inc.' "Moody's Corp" 'Mondelez International'
       'MetLife Inc.' 'Mohawk Industries' 'Mead Johnson' 'McCormick & Co.'
       'Martin Marietta Materials' 'Marsh & McLennan' '3M Company'
       'Altria Group Inc' 'The Mosaic Company' 'Marathon Petroleum'
       'Merck & Co.' 'M&T Bank Corp.' 'Mettler Toledo' 'Mylan N.V.' 'Navient'
       'NASDAQ OMX Group' 'NextEra Energy' 'Newmont Mining Corp. (Hldg. Co.)'
       'Nielsen Holdings' 'Norfolk Southern Corp.' 'Northern Trust Corp.'
       'Nucor Corp.' 'Newell Brands' 'Realty Income Corporation' 'Omnicom Group
      '
       "O'Reilly Automotive" "People's United Financial" 'Pitney-Bowes'
       'PACCAR Inc.' 'PG&E Corp.' 'Public Serv. Enterprise Inc.' 'PepsiCo Inc.'
```

```
 'Principal Financial Group' 'Procter & Gamble' 'Progressive Corp.'
 'Pulte Homes Inc.' 'Philip Morris International' 'PNC Financial Services
'
 'Pentair Ltd.' 'Pinnacle West Capital' 'PPG Industries' 'PPL Corp.'
 'Prudential Financial' 'Phillips 66' 'Praxair Inc.' 'PayPal'
 'Ryder System' 'Royal Caribbean Cruises Ltd' 'Robert Half International'
 'Roper Industries' 'Republic Services Inc' 'SCANA Corp'
 'Charles Schwab Corporation' 'Spectra Energy Corp.' 'Sealed Air'
 'Sherwin-Williams' 'SL Green Realty' 'Scripps Networks Interactive Inc.'
 'Southern Co.' 'Simon Property Group Inc' 'S&P Global, Inc.'
 'Stericycle Inc' 'Sempra Energy' 'SunTrust Banks' 'State Street Corp.'
 'Skyworks Solutions' 'Synchrony Financial' 'Stryker Corp.'
 'Molson Coors Brewing Company' 'Tegna, Inc.' 'Torchmark Corp.'
 'Thermo Fisher Scientific' 'The Travelers Companies Inc.'
 'Tractor Supply Company' 'Tyson Foods' 'Tesoro Petroleum Co.'
 'Total System Services' 'Texas Instruments' 'Under Armour'
 'United Continental Holdings' 'UDR Inc' 'Universal Health Services, Inc.
'
 'United Health Group Inc.' 'Unum Group' 'Union Pacific'
 'United Parcel Service' 'United Technologies' 'Varian Medical Systems'
 'Valero Energy' 'Vulcan Materials' 'Vornado Realty Trust'
 'Verisk Analytics' 'Verisign Inc.' 'Ventas Inc' 'Wec Energy Group Inc'
 'Whirlpool Corp.' 'Waste Management Inc.' 'Western Union Co'
 'Weyerhaeuser Corp.' 'Wyndham Worldwide' 'Xcel Energy Inc' 'XL Capital'
 'Dentsply Sirona' 'Xerox Corp.' 'Xylem Inc.' 'Yum! Brands Inc'
 'Zimmer Biomet Holdings' 'Zions Bancorp' 'Zoetis']

In cluster 3, the following companies are present:
['Analog Devices, Inc.' 'Alliance Data Systems' 'Alexion Pharmaceuticals'
 'Amgen Inc' 'Amazon.com Inc' 'Bank of America Corp' 'BIOGEN IDEC Inc.'
 'Celgene Corp.' 'Chipotle Mexican Grill' 'Equinix' 'Edwards Lifesciences
'
 'Facebook' 'First Solar Inc' 'Frontier Communications' 'Halliburton Co.'
 'Intuitive Surgical Inc.' "McDonald's Corp." 'Monster Beverage'
 'Priceline.com Inc' 'Regeneron' 'TripAdvisor'
 'Vertex Pharmaceuticals Inc' 'Waters Corporation' 'Wynn Resorts Ltd'
 'Yahoo Inc.']

In cluster 2, the following companies are present:
['Apache Corporation' 'Anadarko Petroleum Corp' 'Baker Hughes Inc'
 'Chesapeake Energy' 'Cabot Oil & Gas' 'Concho Resources'
 'Devon Energy Corp.' 'EOG Resources' 'EQT Corporation'
 'Freeport-McMoran Cp & Gld' 'Hess Corporation'
 'Hewlett Packard Enterprise' 'Kinder Morgan' 'Marathon Oil Corp.'
 'Murphy Oil' 'Noble Energy Inc' 'Netflix Inc.' 'Newfield Exploration Co'
 'National Oilwell Varco Inc.' 'ONEOK' 'Occidental Petroleum'
 'Quanta Services Inc.' 'Range Resources Corp.' 'Southwestern Energy'
 'Teradata Corp.' 'Williams Cos.' 'Cimarex Energy']

In cluster 1, the following companies are present:
['Citigroup Inc.' 'Ford Motor' 'Gilead Sciences' 'Intel Corp.'
 'JPMorgan Chase & Co.' 'Coca Cola Company' 'Pfizer Inc.' 'AT&T Inc'
 'Verizon Communications' 'Wells Fargo' 'Exxon Mobil Corp.']
```

In [63]:
```
df1.groupby(["KM_segments", "GICS Sector"])['Security'].count()
```

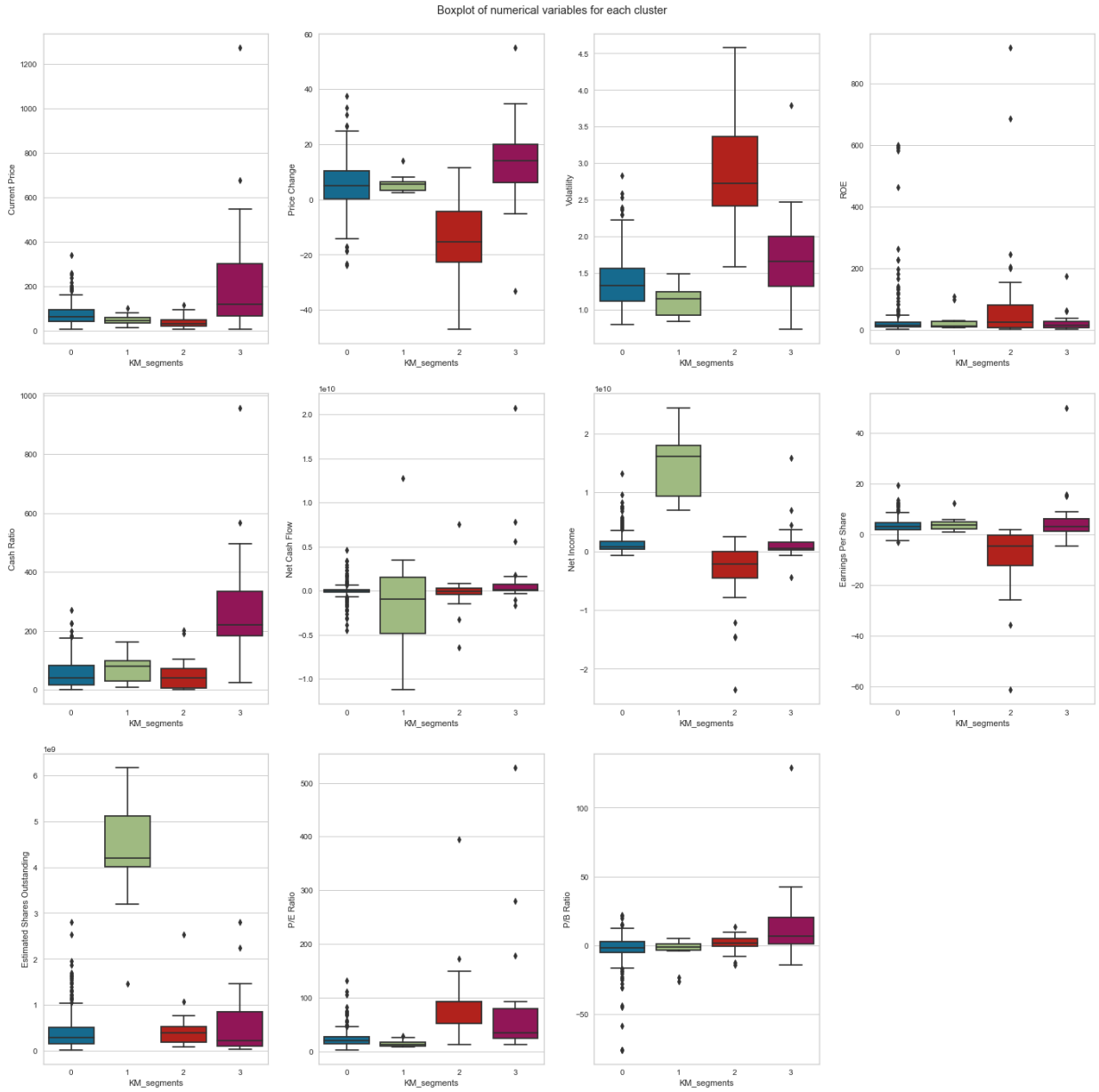
Out[63]:

KM_segments	GICS Sector	Security
0	Consumer Discretionary	33
	Consumer Staples	17
	Energy	6
	Financials	45
	Health Care	29
	Industrials	52
	Information Technology	24
	Materials	19
	Real Estate	26
	Telecommunications Services	2
	Utilities	24
1	Consumer Discretionary	1
	Consumer Staples	1
	Energy	1
	Financials	3
	Health Care	2
	Information Technology	1
	Telecommunications Services	2
2	Energy	22
	Industrials	1
	Information Technology	3
	Materials	1
3	Consumer Discretionary	6
	Consumer Staples	1
	Energy	1
	Financials	1
	Health Care	9
	Information Technology	5
	Real Estate	1
	Telecommunications Services	1

Name: Security, dtype: int64

In [64]:

```
plt.figure(figsize=(20, 20))
plt.suptitle("Boxplot of numerical variables for each cluster")

# selecting numerical columns
num_col = df.select_dtypes(include=np.number).columns.tolist()

for i, variable in enumerate(num_col):
    plt.subplot(3, 4, i + 1)
    sns.boxplot(data=df1, x="KM_segments", y=variable)

plt.tight_layout(pad=2.0)
```



Insights

-

Hierarchical Clustering

Computing Cophenetic Correlation

In [69]:
```
hc_df = subset_scaled_df.copy()
```

In [70]:
```
# list of distance metrics
distance_metrics = ["euclidean", "chebyshev", "mahalanobis", "cityblock"]

# list of linkage methods
linkage_methods = ["single", "complete", "average", "weighted"] ## Comple

high_cophenet_corr = 0
high_dm_lm = [0, 0]

for dm in distance_metrics:
    for lm in linkage_methods:
        Z = linkage(hc_df, metric=dm, method=lm)
        c, coph_dists = cophenet(Z, pdist(hc_df))
        print(
            "Cophenetic correlation for {} distance and {} linkage is {}.
                dm.capitalize(), lm, c
            )
        )
        if high_cophenet_corr < c:
            high_cophenet_corr = c
            high_dm_lm[0] = dm
            high_dm_lm[1] = lm

# printing the combination of distance metric and linkage method with the
print('*'*100)
print(
    "Highest cophenetic correlation is {}, which is obtained with {} dist
        high_cophenet_corr, high_dm_lm[0].capitalize(), high_dm_lm[1]
    )
)
```

```
Cophenetic correlation for Euclidean distance and single linkage is 0.923
2271494002922.
Cophenetic correlation for Euclidean distance and complete linkage is 0.7
873280186580672.
Cophenetic correlation for Euclidean distance and average linkage is 0.94
22540609560814.
Cophenetic correlation for Euclidean distance and weighted linkage is 0.8
693784298129404.
Cophenetic correlation for Chebyshev distance and single linkage is 0.906
2538164750717.
Cophenetic correlation for Chebyshev distance and complete linkage is 0.5
98891419111242.
Cophenetic correlation for Chebyshev distance and average linkage is 0.93
38265528030499.
Cophenetic correlation for Chebyshev distance and weighted linkage is 0.9
127355892367.
Cophenetic correlation for Mahalanobis distance and single linkage is 0.9
259195530524591.
Cophenetic correlation for Mahalanobis distance and complete linkage is 0
.7925307202850002.
Cophenetic correlation for Mahalanobis distance and average linkage is 0.
9247324030159736.
Cophenetic correlation for Mahalanobis distance and weighted linkage is 0
.8708317490180428.
Cophenetic correlation for Cityblock distance and single linkage is 0.933
4186366528574.
Cophenetic correlation for Cityblock distance and complete linkage is 0.7
375328863205818.
Cophenetic correlation for Cityblock distance and average linkage is 0.93
02145048594667.
Cophenetic correlation for Cityblock distance and weighted linkage is 0.7
31045513520281.
**************************************************************************
**************************
Highest cophenetic correlation is 0.9422540609560814, which is obtained w
ith Euclidean distance and average linkage.
```

Let's explore different linkage methods with Euclidean distance only.

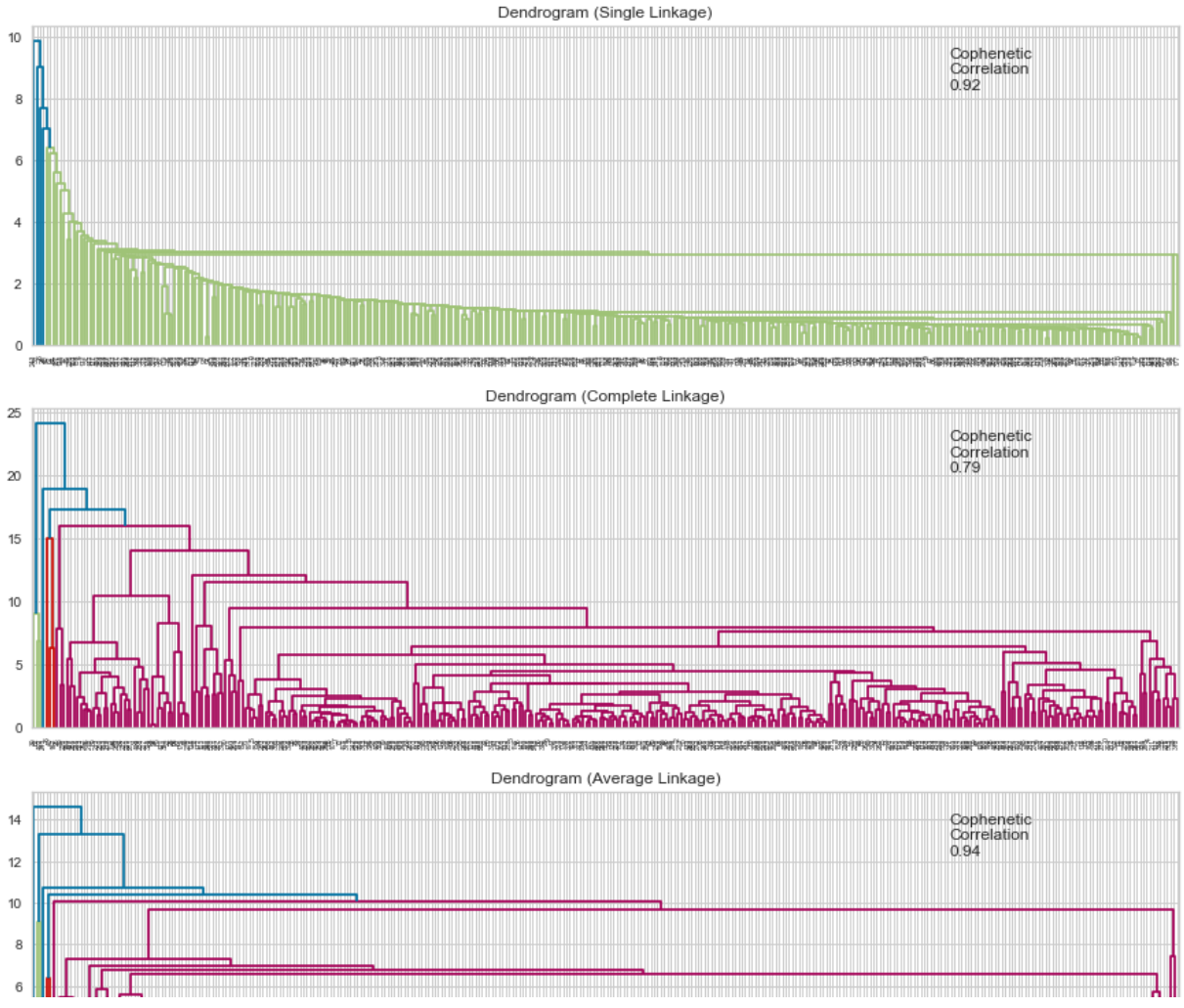
In [71]:
```
# list of linkage methods
linkage_methods = ["single", "complete", "average", "centroid", "ward", "

high_cophenet_corr = 0
high_dm_lm = [0, 0]

for lm in linkage_methods:
    Z = linkage(hc_df, metric="euclidean", method=lm)
    c, coph_dists = cophenet(Z, pdist(hc_df))
    print("Cophenetic correlation for {} linkage is {}.".format(lm, c))
    if high_cophenet_corr < c:
        high_cophenet_corr = c
        high_dm_lm[0] = "euclidean"
        high_dm_lm[1] = lm

# printing the combination of distance metric and linkage method with the
print('*'*100)
print(
    "Highest cophenetic correlation is {}, which is obtained with {} link
        high_cophenet_corr, high_dm_lm[1]
    )
)
```

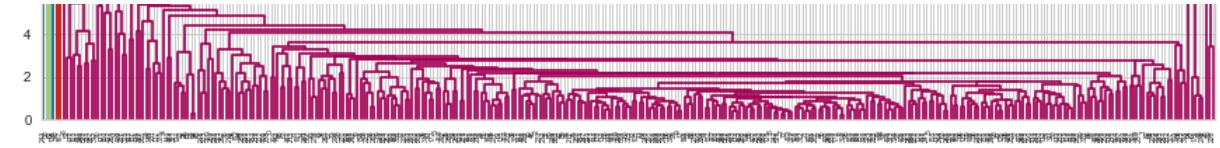
```
Cophenetic correlation for single linkage is 0.9232271494002922.
Cophenetic correlation for complete linkage is 0.7873280186580672.
Cophenetic correlation for average linkage is 0.9422540609560814.
Cophenetic correlation for centroid linkage is 0.9314012446828154.
Cophenetic correlation for ward linkage is 0.7101180299865353.
Cophenetic correlation for weighted linkage is 0.8693784298129404.
****************************************************************************
**************************
Highest cophenetic correlation is 0.9422540609560814, which is obtained w
ith average linkage.
```

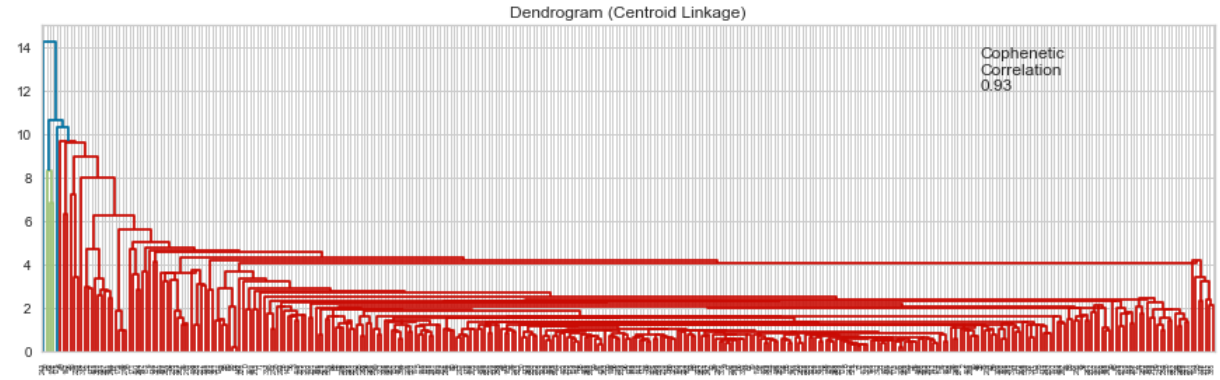
Let's view the dendrograms for the different linkage methods with Euclidean distance.

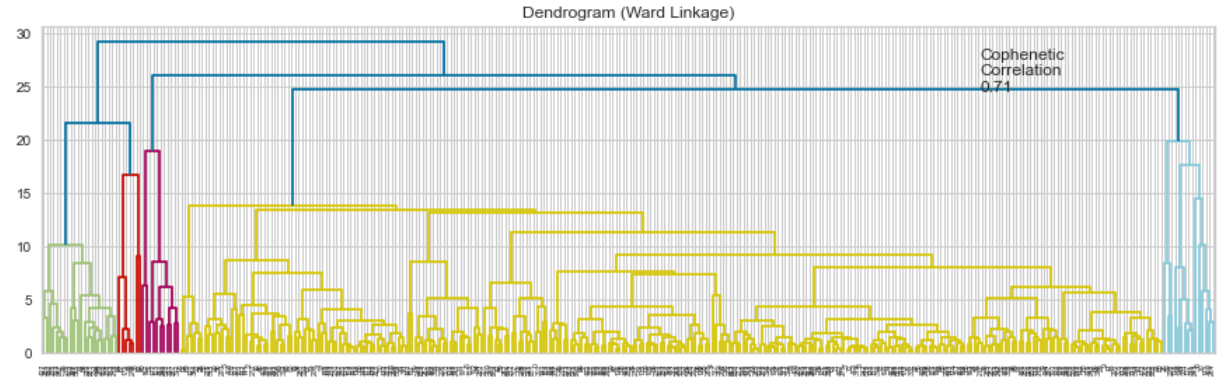
Checking Dendrograms

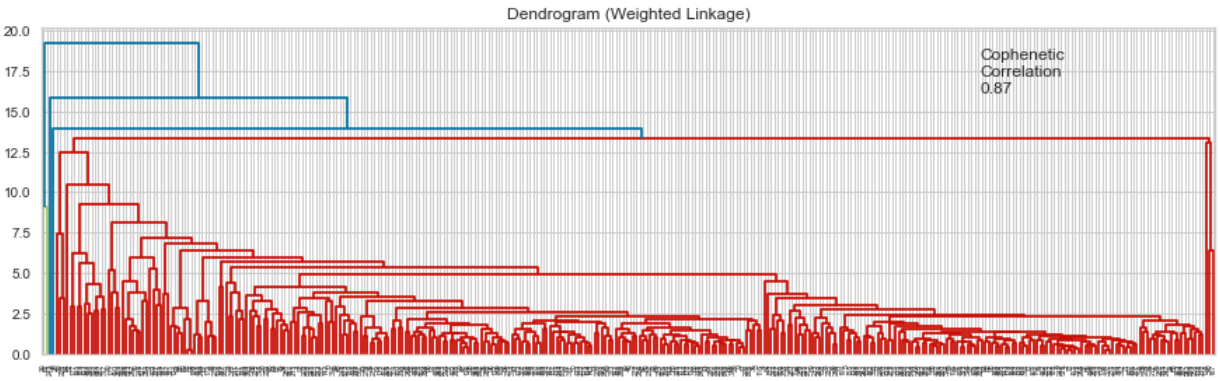
In [72]:
```
# list of linkage methods
linkage_methods = ["single", "complete", "average", "centroid", "ward", "

# lists to save results of cophenetic correlation calculation
compare_cols = ["Linkage", "Cophenetic Coefficient"]
compare = []

# to create a subplot image
fig, axs = plt.subplots(len(linkage_methods), 1, figsize=(15, 30))

# We will enumerate through the list of linkage methods above
# For each linkage method, we will plot the dendrogram and calculate the
for i, method in enumerate(linkage_methods):
    Z = linkage(hc_df, metric="euclidean", method=method)

    dendrogram(Z, ax=axs[i])
    axs[i].set_title(f"Dendrogram ({method.capitalize()} Linkage)")

    coph_corr, coph_dist = cophenet(Z, pdist(hc_df))
    axs[i].annotate(
        f"Cophenetic\nCorrelation\n{coph_corr:0.2f}",
        (0.80, 0.80),
        xycoords="axes fraction",
    )

    compare.append([method, coph_corr])
```











In [73]:
```
# create and print a dataframe to compare cophenetic correlations for dif
df_cc = pd.DataFrame(compare, columns=compare_cols)
df_cc = df_cc.sort_values(by="Cophenetic Coefficient")
df_cc
```

Out[73]:

	Linkage	Cophenetic Coefficient
4	ward	0.710118
1	complete	0.787328
5	weighted	0.869378
0	single	0.923227
3	centroid	0.931401
2	average	0.942254

Creating model using sklearn

In [74]:
```
HCmodel = AgglomerativeClustering(n_clusters=4, affinity='euclidean', lin
HCmodel.fit(hc_df)
```

Out[74]:
```
AgglomerativeClustering(n_clusters=4)
```

In [75]:
```
# creating a copy of the original data
df2 = df.copy()

# adding hierarchical cluster labels to the original and scaled dataframe
hc_df["HC_segments"] = HCmodel.labels_
df2["HC_segments"] = HCmodel.labels_
```

Cluster Profiling

In [76]:
```
hc_cluster_profile = df2.groupby("HC_segments").mean()  ## Complete the c
```

In [77]:
```
hc_cluster_profile["count_in_each_segment"] = (
    df2.groupby("HC_segments")["Security"].count().values  ## Complete th
)
```

In [78]:
```
hc_cluster_profile.style.highlight_max(color="lightgreen", axis=0)
```

Out[78]:

HC_segments	Current Price	Price Change	Volatility	ROE	Cash Ratio	Net Cash F
0	48.006208	-11.263107	2.590247	196.551724	40.275862	-495901724.137
1	326.198218	10.563242	1.642560	14.400000	309.466667	288850666.666
2	42.848182	6.270446	1.123547	22.727273	71.454545	558636363.636
3	72.760400	5.213307	1.427078	25.603509	60.392982	79951512.280

In [79]:
```
## Complete the code to print the companies in each cluster
for cl in df2["HC_segments"].unique():
    print("In cluster {}, the following companies are present:".format(cl
    print(df2[df2["HC_segments"] == cl]["Security"].unique())
    print()
```

```
In cluster 3, the following companies are present:
['American Airlines Group' 'AbbVie' 'Abbott Laboratories'
 'Adobe Systems Inc' 'Analog Devices, Inc.' 'Archer-Daniels-Midland Co'
 'Ameren Corp' 'American Electric Power' 'AFLAC Inc'
 'American International Group, Inc.' 'Apartment Investment & Mgmt'
 'Assurant Inc' 'Arthur J. Gallagher & Co.' 'Akamai Technologies Inc'
 'Albemarle Corp' 'Alaska Air Group Inc' 'Allstate Corp'
 'Applied Materials Inc' 'AMETEK Inc' 'Affiliated Managers Group Inc'
 'Ameriprise Financial' 'American Tower Corp A' 'AutoNation Inc'
 'Anthem Inc.' 'Aon plc' 'Amphenol Corp' 'Arconic Inc'
```

```
       'Activision Blizzard' 'AvalonBay Communities, Inc.' 'Broadcom'
       'American Water Works Company Inc' 'American Express Co' 'Boeing Company
      '
       'Baxter International Inc.' 'BB&T Corporation' 'Bard (C.R.) Inc.'
       'BIOGEN IDEC Inc.' 'The Bank of New York Mellon Corp.' 'Ball Corp'
       'Bristol-Myers Squibb' 'Boston Scientific' 'BorgWarner'
       'Boston Properties' 'Caterpillar Inc.' 'Chubb Limited' 'CBRE Group'
       'Crown Castle International Corp.' 'Carnival Corp.' 'Celgene Corp.'
       'CF Industries Holdings Inc' 'Citizens Financial Group' 'Church & Dwight
      '
       'C. H. Robinson Worldwide' 'CIGNA Corp.' 'Cincinnati Financial'
       'Comerica Inc.' 'CME Group Inc.' 'Cummins Inc.' 'CMS Energy'
       'Centene Corporation' 'CenterPoint Energy' 'Capital One Financial'
       'The Cooper Companies' 'CSX Corp.' 'CenturyLink Inc'
       'Cognizant Technology Solutions' 'Citrix Systems' 'CVS Health'
       'Chevron Corp.' 'Dominion Resources' 'Delta Air Lines' 'Du Pont (E.I.)'
       'Deere & Co.' 'Discover Financial Services' 'Quest Diagnostics'
       'Danaher Corp.' 'The Walt Disney Company' 'Discovery Communications-A'
       'Discovery Communications-C' 'Delphi Automotive' 'Digital Realty Trust'
       'Dun & Bradstreet' 'Dover Corp.' 'Dr Pepper Snapple Group' 'Duke Energy'
       'DaVita Inc.' 'eBay Inc.' 'Ecolab Inc.' 'Consolidated Edison'
       'Equifax Inc.' "Edison Int'l" 'Eastman Chemical' 'Equity Residential'
       'EQT Corporation' 'Eversource Energy' 'Essex Property Trust, Inc.'
       'E*Trade' 'Eaton Corporation' 'Entergy Corp.' 'Edwards Lifesciences'
       'Exelon Corp.' "Expeditors Int'l" 'Expedia Inc.' 'Extra Space Storage'
       'Fastenal Co' 'Fortune Brands Home & Security' 'FirstEnergy Corp'
       'Fidelity National Information Services' 'Fiserv Inc' 'FLIR Systems'
       'Fluor Corp.' 'Flowserve Corporation' 'FMC Corporation'
       'Federal Realty Investment Trust' 'First Solar Inc' 'General Dynamics'
       'General Growth Properties Inc.' 'Gilead Sciences' 'Corning Inc.'
       'General Motors' 'Genuine Parts' 'Garmin Ltd.' 'Goodyear Tire & Rubber'
       'Grainger (W.W.) Inc.' 'Halliburton Co.' 'Hasbro Inc.'
       'Huntington Bancshares' 'HCA Holdings' 'Welltower Inc.' 'HCP Inc.'
       'Hartford Financial Svc.Gp.' 'Harley-Davidson' "Honeywell Int'l Inc."
       'Hewlett Packard Enterprise' 'HP Inc.' 'Hormel Foods Corp.'
       'Henry Schein' 'Host Hotels & Resorts' 'The Hershey Company'
       'Humana Inc.' 'International Business Machines' 'IDEXX Laboratories'
       'Intl Flavors & Fragrances' 'International Paper' 'Interpublic Group'
       'Iron Mountain Incorporated' 'Illinois Tool Works' 'Invesco Ltd.'
       'J. B. Hunt Transport Services' 'Jacobs Engineering Group'
       'Juniper Networks' 'Kimco Realty' 'Kansas City Southern'
       'Leggett & Platt' 'Lennar Corp.' 'Laboratory Corp. of America Holding'
       'LKQ Corporation' 'L-3 Communications Holdings' 'Lilly (Eli) & Co.'
       'Lockheed Martin Corp.' 'Alliant Energy Corp' 'Leucadia National Corp.'
       'Southwest Airlines' 'Level 3 Communications' 'LyondellBasell'
       'Mastercard Inc.' 'Mid-America Apartments' 'Macerich' "Marriott Int'l."
       'Masco Corp.' 'Mattel Inc.' "McDonald's Corp." "Moody's Corp"
       'Mondelez International' 'MetLife Inc.' 'Mohawk Industries'
       'Mead Johnson' 'McCormick & Co.' 'Martin Marietta Materials'
       'Marsh & McLennan' '3M Company' 'Altria Group Inc' 'The Mosaic Company'
       'Marathon Petroleum' 'Merck & Co.' 'M&T Bank Corp.' 'Mettler Toledo'
       'Mylan N.V.' 'Navient' 'NASDAQ OMX Group' 'NextEra Energy'
       'Newmont Mining Corp. (Hldg. Co.)' 'Nielsen Holdings'
       'Norfolk Southern Corp.' 'Northern Trust Corp.' 'Nucor Corp.'
       'Newell Brands' 'Realty Income Corporation' 'Omnicom Group'
       "O'Reilly Automotive" "People's United Financial" 'Pitney-Bowes'
       'PACCAR Inc.' 'PG&E Corp.' 'Public Serv. Enterprise Inc.' 'PepsiCo Inc.'
```

```
 'Principal Financial Group' 'Procter & Gamble' 'Progressive Corp.'
 'Pulte Homes Inc.' 'Philip Morris International' 'PNC Financial Services
'
 'Pentair Ltd.' 'Pinnacle West Capital' 'PPG Industries' 'PPL Corp.'
 'Prudential Financial' 'Phillips 66' 'Quanta Services Inc.'
 'Praxair Inc.' 'PayPal' 'Ryder System' 'Royal Caribbean Cruises Ltd'
 'Robert Half International' 'Roper Industries' 'Republic Services Inc'
 'SCANA Corp' 'Charles Schwab Corporation' 'Sealed Air' 'Sherwin-Williams
'
 'SL Green Realty' 'Scripps Networks Interactive Inc.' 'Southern Co.'
 'Simon Property Group Inc' 'Stericycle Inc' 'Sempra Energy'
 'SunTrust Banks' 'State Street Corp.' 'Skyworks Solutions'
 'Synchrony Financial' 'Stryker Corp.' 'Molson Coors Brewing Company'
 'Tegna, Inc.' 'Torchmark Corp.' 'Thermo Fisher Scientific' 'TripAdvisor'
 'The Travelers Companies Inc.' 'Tractor Supply Company' 'Tyson Foods'
 'Tesoro Petroleum Co.' 'Total System Services' 'Texas Instruments'
 'Under Armour' 'United Continental Holdings' 'UDR Inc'
 'Universal Health Services, Inc.' 'United Health Group Inc.' 'Unum Group
'
 'Union Pacific' 'United Parcel Service' 'United Technologies'
 'Varian Medical Systems' 'Valero Energy' 'Vulcan Materials'
 'Vornado Realty Trust' 'Verisk Analytics' 'Verisign Inc.'
 'Vertex Pharmaceuticals Inc' 'Ventas Inc' 'Wec Energy Group Inc'
 'Whirlpool Corp.' 'Waste Management Inc.' 'Western Union Co'
 'Weyerhaeuser Corp.' 'Wyndham Worldwide' 'Wynn Resorts Ltd'
 'Xcel Energy Inc' 'XL Capital' 'Dentsply Sirona' 'Xerox Corp.'
 'Xylem Inc.' 'Yum! Brands Inc' 'Zimmer Biomet Holdings' 'Zions Bancorp'
 'Zoetis']

In cluster 1, the following companies are present:
['Alliance Data Systems' 'Alexion Pharmaceuticals' 'Amgen Inc'
 'Amazon.com Inc' 'Chipotle Mexican Grill' 'Equinix' 'Facebook'
 'Frontier Communications' 'Intuitive Surgical Inc.' 'Monster Beverage'
 'Netflix Inc.' 'Priceline.com Inc' 'Regeneron' 'Waters Corporation'
 'Yahoo Inc.']

In cluster 0, the following companies are present:
['Allegion' 'Apache Corporation' 'Anadarko Petroleum Corp'
 'Baker Hughes Inc' 'Chesapeake Energy' 'Charter Communications'
 'Colgate-Palmolive' 'Cabot Oil & Gas' 'Concho Resources'
 'Devon Energy Corp.' 'EOG Resources' 'Freeport-McMoran Cp & Gld'
 'Hess Corporation' 'Kimberly-Clark' 'Kinder Morgan' 'Marathon Oil Corp.'
 'Murphy Oil' 'Noble Energy Inc' 'Newfield Exploration Co'
 'National Oilwell Varco Inc.' 'ONEOK' 'Occidental Petroleum'
 'Range Resources Corp.' 'Spectra Energy Corp.' 'S&P Global, Inc.'
 'Southwestern Energy' 'Teradata Corp.' 'Williams Cos.' 'Cimarex Energy']

In cluster 2, the following companies are present:
['Bank of America Corp' 'Citigroup Inc.' 'Ford Motor' 'Intel Corp.'
 'JPMorgan Chase & Co.' 'Coca Cola Company' 'Pfizer Inc.' 'AT&T Inc'
 'Verizon Communications' 'Wells Fargo' 'Exxon Mobil Corp.']
```

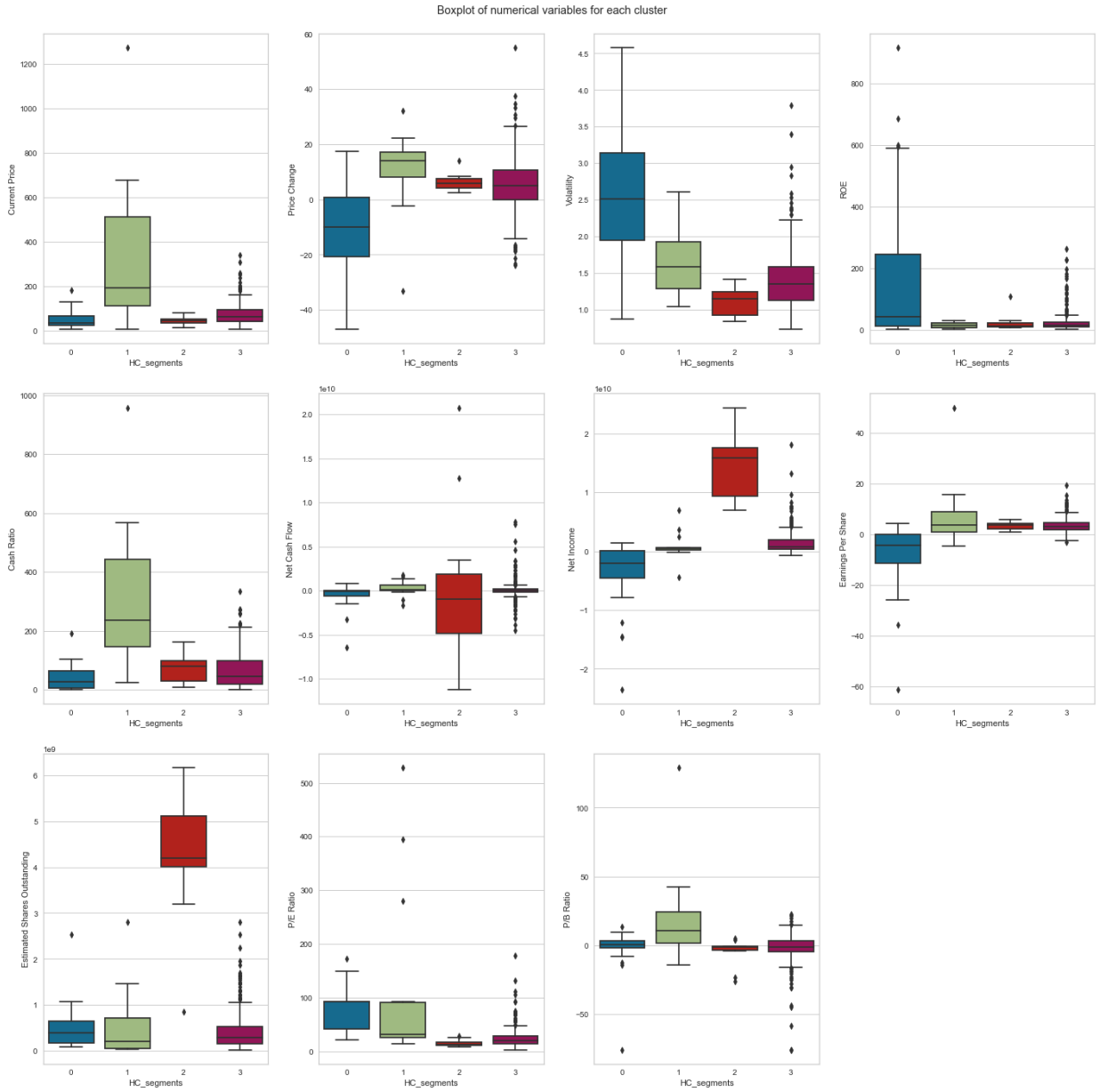
In [80]:
```
df2.groupby(["HC_segments", "GICS Sector"])['Security'].count()
```

Out[80]:

```
HC_segments  GICS Sector
0            Consumer Discretionary          1
             Consumer Staples                2
             Energy                         22
             Financials                      1
             Industrials                     1
             Information Technology          1
             Materials                       1
1            Consumer Discretionary          3
             Consumer Staples                1
             Health Care                     5
             Information Technology          4
             Real Estate                     1
             Telecommunications Services     1
2            Consumer Discretionary          1
             Consumer Staples                1
             Energy                          1
             Financials                      4
             Health Care                     1
             Information Technology          1
             Telecommunications Services     2
3            Consumer Discretionary         35
             Consumer Staples               15
             Energy                          7
             Financials                     44
             Health Care                    34
             Industrials                    52
             Information Technology         27
             Materials                      19
             Real Estate                    26
             Telecommunications Services     2
             Utilities                      24
Name: Security, dtype: int64
```

In [81]:

```
plt.figure(figsize=(20, 20))
plt.suptitle("Boxplot of numerical variables for each cluster")

for i, variable in enumerate(num_col):
    plt.subplot(3, 4, i + 1)
    sns.boxplot(data=df2, x="HC_segments", y=variable)

plt.tight_layout(pad=2.0)
```



K-means vs Hierarchical Clustering

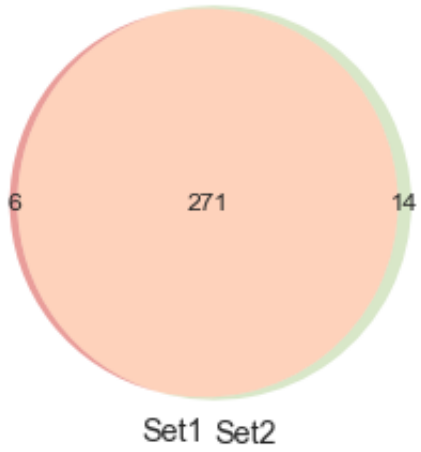
You compare several things, like:

- Which clustering technique took less time for execution?
- Which clustering technique gave you more distinct clusters, or are they the same?
- How many observations are there in the similar clusters of both algorithms?
- How many clusters are obtained as the appropriate number of clusters from both algorithms?

You can also mention any differences or similarities you obtained in the cluster profiles from both the clustering techniques.

In [83]:
```
df1_comp=df1[["Ticker Symbol", "KM_segments" ]].copy()
df2_comp=df2[["Ticker Symbol", "HC_segments" ]].copy()
```

In [88]:
```
from matplotlib_venn import venn2

KC_0 = df1_comp[(df1_comp["KM_segments"] == 0)]
HC_3 = df2_comp[(df2_comp["HC_segments"] == 3)]

plt.figure(figsize=(4,4))
set1 = set(KC_0["Ticker Symbol"])
set2 = set(HC_3["Ticker Symbol"])

venn2([set1, set2], ('Set1', 'Set2'))
plt.show()
```



In [87]:
```
!pip install matplotlib-venn
```

```
Collecting matplotlib-venn
  Downloading matplotlib-venn-0.11.9.tar.gz (30 kB)
  Installing build dependencies ... done
  Getting requirements to build wheel ... done
  Installing backend dependencies ... done
    Preparing wheel metadata ... done
Requirement already satisfied: scipy in /Users/Moafdhal/opt/anaconda3/lib
/python3.9/site-packages (from matplotlib-venn) (1.7.3)
Requirement already satisfied: matplotlib in /Users/Moafdhal/opt/anaconda
3/lib/python3.9/site-packages (from matplotlib-venn) (3.5.1)
Requirement already satisfied: numpy in /Users/Moafdhal/opt/anaconda3/lib
/python3.9/site-packages (from matplotlib-venn) (1.21.5)
Requirement already satisfied: cycler>=0.10 in /Users/Moafdhal/opt/anacon
da3/lib/python3.9/site-packages (from matplotlib->matplotlib-venn) (0.11.
0)
Requirement already satisfied: pyparsing>=2.2.1 in /Users/Moafdhal/opt/an
aconda3/lib/python3.9/site-packages (from matplotlib->matplotlib-venn) (3
.0.4)
Requirement already satisfied: pillow>=6.2.0 in /Users/Moafdhal/opt/anaco
nda3/lib/python3.9/site-packages (from matplotlib->matplotlib-venn) (9.0.
1)
Requirement already satisfied: packaging>=20.0 in /Users/Moafdhal/opt/ana
conda3/lib/python3.9/site-packages (from matplotlib->matplotlib-venn) (21
.3)
Requirement already satisfied: kiwisolver>=1.0.1 in /Users/Moafdhal/opt/a
naconda3/lib/python3.9/site-packages (from matplotlib->matplotlib-venn) (
1.3.2)
Requirement already satisfied: python-dateutil>=2.7 in /Users/Moafdhal/op
t/anaconda3/lib/python3.9/site-packages (from matplotlib->matplotlib-venn
) (2.8.2)
Requirement already satisfied: fonttools>=4.22.0 in /Users/Moafdhal/opt/a
naconda3/lib/python3.9/site-packages (from matplotlib->matplotlib-venn) (
4.25.0)
Requirement already satisfied: six>=1.5 in /Users/Moafdhal/opt/anaconda3/
lib/python3.9/site-packages (from python-dateutil>=2.7->matplotlib->matpl
otlib-venn) (1.16.0)
Building wheels for collected packages: matplotlib-venn
  Building wheel for matplotlib-venn (PEP 517) ... done
  Created wheel for matplotlib-venn: filename=matplotlib_venn-0.11.9-py3-
none-any.whl size=32999 sha256=0ebd2e3e84a4466408dd064855a1005f9df3415449
ed96d90e71640a26676d6a
  Stored in directory: /Users/Moafdhal/Library/Caches/pip/wheels/25/de/d4
/29dfc5d4520b956df7bc54a8464ad053042918bc525e88bf66
Successfully built matplotlib-venn
Installing collected packages: matplotlib-venn
Successfully installed matplotlib-venn-0.11.9
```

Actionable Insights and Recommendations

-
