📈💹 Stock Market Performance Profiling

Multi-Sector Stock Analysis — June 2025

📌 Overview

This project explores a **comprehensive stock market dataset** capturing **1,762 daily trading records** across multiple sectors during **June 2025**. Each record represents a publicly traded company's daily performance metrics.

The objective is to uncover **trading patterns**, examine **valuation indicators**, and apply robust **data science techniques** to derive actionable investment insights.

🧮 Dataset Exploration & Preprocessing

- 📁 **Records:** 1,762 rows of daily trading data
- 🏢 **Sectors Covered:**
  - Technology
  - Healthcare
  - Industrials
  - Energy (and more)
- 💰 **Key Financial Indicators:**
  - Open / Close / High / Low Prices
  - Volume Traded
  - Market Capitalization
  - PE Ratio
  - Dividend Yield
  - EPS
  - 52-Week High & Low
- 🧼 **Cleaning:**
  - No missing values detected — dataset ready for direct analysis
  - `Date` column converted to datetime for temporal grouping

📊 Exploratory Data Analysis (EDA)

- 📌 **Price Distribution:**
  Visualized distributions of daily closing prices across sectors
- 💹 **Sector Performance:**
  Aggregated sector-wise returns to identify outperforming industries
- 🔄 **Volatility Analysis:**
  Calculated daily price ranges *(High - Low)* as a proxy for volatility

- 📈 **Valuation Comparison:**
  Compared PE Ratios and Dividend Yields across sectors
- 📆 **Temporal Patterns:**
  Tracked trends over the month to detect consistent momentum or reversals

---

🔍 Sector-Based Performance Profiling

🔷 Key Comparisons:

- **Technology** stocks demonstrated the highest average closing prices and elevated PE Ratios
- **Energy** sector exhibited higher volatility — likely driven by commodity price fluctuations
- **Healthcare** stocks offered steady dividend yields combined with moderate valuations

🔄 Volatility and Liquidity:

- High trading volumes were concentrated in large-cap Technology and Industrials
- Energy sector showed the widest daily price ranges, indicating speculative trading

---

📌 Key Insights

- 💡 **Technology sector** led in both capitalization and valuation multiples — highlighting investor optimism
- 💰 **Dividend Yield** varied significantly across sectors, suggesting different capital return policies
- ⚖️ **Volatility** correlated strongly with trading volume in certain sectors
- 📈 **52-Week High proximity** (current price vs. yearly high) was a useful indicator of momentum

---

🔧 Tools Used

- **Python** (Pandas, NumPy, Seaborn, Matplotlib)
- **Visualization:** Boxplots, Sector Bar Charts, Volatility Heatmaps
- **Preprocessing:** Datetime parsing, Aggregation, GroupBy Sector Analysis

---

📎 Dataset Info

- **Total Records:** 1,762 daily trading entries
- **Attributes:**
  - Date, Ticker, Open Price, Close Price, High Price, Low Price
  - Volume Traded, Market Cap, PE Ratio, Dividend Yield, EPS
  - 52-Week High / Low, Sector
- **Sectors Covered:** Technology, Energy, Healthcare, Industrials, and more
- **Source:** Simulated stock market data (anonymized for demonstration)

🧑‍💻 Author

Hilda Adina Rahmi –
Data enthusiast passionate about transforming raw financial data into compelling investment stories and

insights.

```
import pandas as pd
import numpy as np


# Load dataset
df = pd.read_csv("stock_market_june2025.csv")


print("Dataset shape:", df.shape)
```

```
Dataset shape: (1762, 14)
```

```
unique_sectors = df["Sector"].unique()
print("Sectors Covered:", unique_sectors)
```

```
Sectors Covered: ['Industrials' 'Energy' 'Healthcare' 'Technology' 'Consumer Staples'
 'Materials' 'Financials' 'Consumer Discretionary' 'Real Estate'
 'Communication Services' 'Utilities']
```

```
financial_columns = [
    "Open Price", "Close Price", "High Price", "Low Price",
    "Volume Traded", "Market Cap", "PE Ratio",
    "Dividend Yield", "EPS", "52 Week High", "52 Week Low"
]
print("Financial Indicators:", financial_columns)
```

```
Financial Indicators: ['Open Price', 'Close Price', 'High Price', 'Low Price', 'Volum
```

```
missing_counts = df.isnull().sum()
print("\nMissing Values Per Column:\n", missing_counts)
```

```

Missing Values Per Column:
 Date              0
Ticker            0
Open Price        0
Close Price       0
High Price        0
Low Price         0
Volume Traded     0
Market Cap        0
PE Ratio          0
Dividend Yield    0
EPS               0
52 Week High      0
52 Week Low       0
Sector            0
dtype: int64
```

```
if missing_counts.sum() == 0:
    print("\n✅ No missing values detected — dataset ready for direct analysis.")
else:
    print("\n⚠️ Warning: Missing values detected — consider cleaning before analysis.")
```

```

✅ No missing values detected — dataset ready for direct analysis.
```

```
df["Date"] = pd.to_datetime(df["Date"], format="%d-%m-%Y")


print("\nDate column converted to datetime:")
print(df["Date"].head())
```

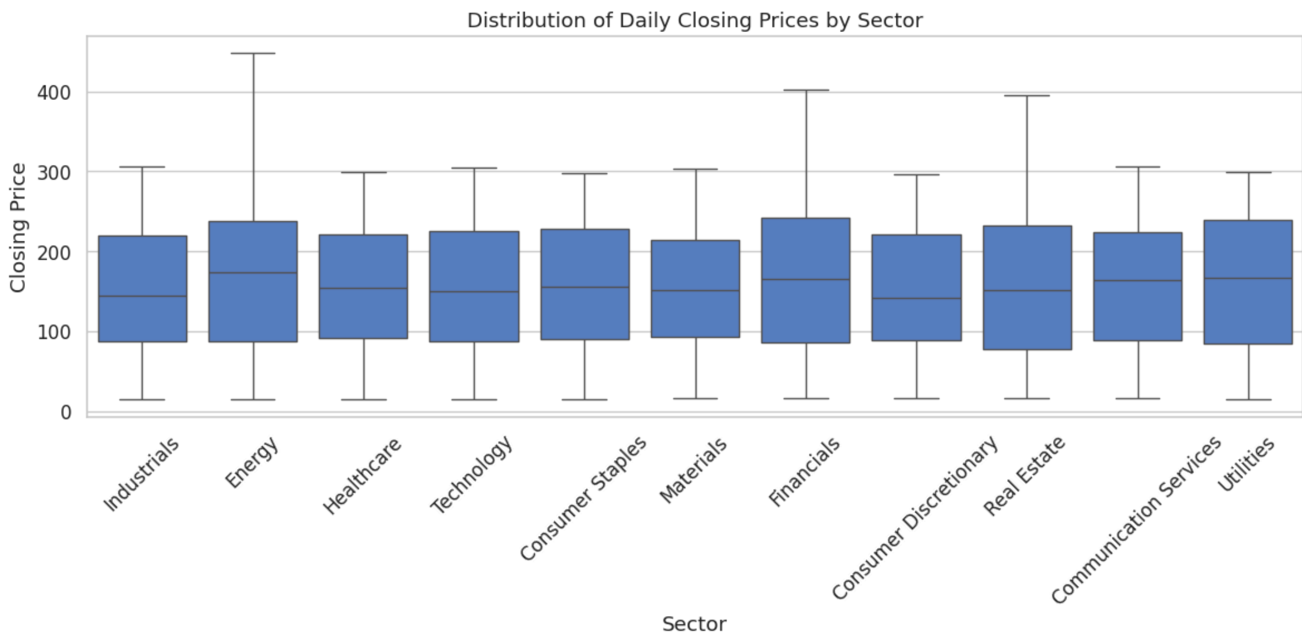
```
Date column converted to datetime:
0   2025-06-01
1   2025-06-01
2   2025-06-01
3   2025-06-01
4   2025-06-01
Name: Date, dtype: datetime64[ns]
```

```
print("\nSample Records:")
print(df.head())
```

```
Sample Records:
        Date Ticker  Open Price  Close Price  High Price  Low Price  \
0 2025-06-01    SLH       34.92        34.53       35.22      34.38   
1 2025-06-01    WGB      206.50       208.45      210.51     205.12   
2 2025-06-01    ZIN      125.10       124.03      127.40     121.77   
3 2025-06-01    YPY      260.55       265.28      269.99     256.64   
4 2025-06-01    VKD      182.43       186.89      189.40     179.02   

   Volume Traded    Market Cap  PE Ratio  Dividend Yield    EPS  52 Week High  \
0        2966611  5.738136e+10     29.63            2.85   1.17         39.39   
1        1658738  5.274707e+10     13.03            2.73  16.00        227.38   
2       10709898  5.596949e+10     29.19            2.64   4.25        138.35   
3       14012358  7.964089e+10     19.92            1.29  13.32        317.57   
4       14758143  7.271437e+10     40.18            1.17   4.65        243.54   

   52 Week Low       Sector  
0        28.44  Industrials  
1       136.79       Energy  
2       100.69   Healthcare  
3       178.26  Industrials  
4       165.53   Technology  
```

```
import matplotlib.pyplot as plt
import seaborn as sns


sns.set(style="whitegrid", palette="muted", font_scale=1.1)


# ==========================
# 📌 1. Price Distribution Boxplot by Sector
# ==========================
plt.figure(figsize=(12,6))
sns.boxplot(
    data=df,
    x="Sector",
    y="Close Price",
)
plt.title("Distribution of Daily Closing Prices by Sector")
plt.xlabel("Sector")
plt.ylabel("Closing Price")
plt.xticks(rotation=45)
```

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



📈 Unveiling Sector Dynamics: The Story Behind Daily Closing Prices

The stock market isn't just numbers on a screen—it's a living ecosystem of industries, each with unique risk profiles, growth potentials, and investor sentiments. In this analysis, I explored the **distribution of daily closing prices across major market sectors**, revealing important patterns that can guide smarter investment decisions.

🔍 Key Insights

1. **Wide Variability Across Sectors**
   - The **Energy sector** stands out with the **widest price range**, reaching highs above $450. This high dispersion may be driven by the volatility of commodity prices and geopolitical factors influencing energy markets.
   - In contrast, sectors like **Consumer Discretionary** and **Communication Services** show a comparatively narrower spread, indicating more consistent pricing.
2. **Median Price Comparison**
   - **Financials and Utilities** exhibit **higher median closing prices**, suggesting these sectors host more high-valued stocks or are buoyed by investor confidence in stable cash flows.
   - **Industrials and Consumer Staples** maintain moderate median prices, aligning with their reputation as foundational but steady sectors.
3. **Outliers Reflect Market Extremes**

  - Several sectors display significant outliers, pointing to companies that either overperform or underperform relative to their peers.
  - This underscores the importance of **stock-level due diligence**—sector trends provide context, but individual performance can defy the average.

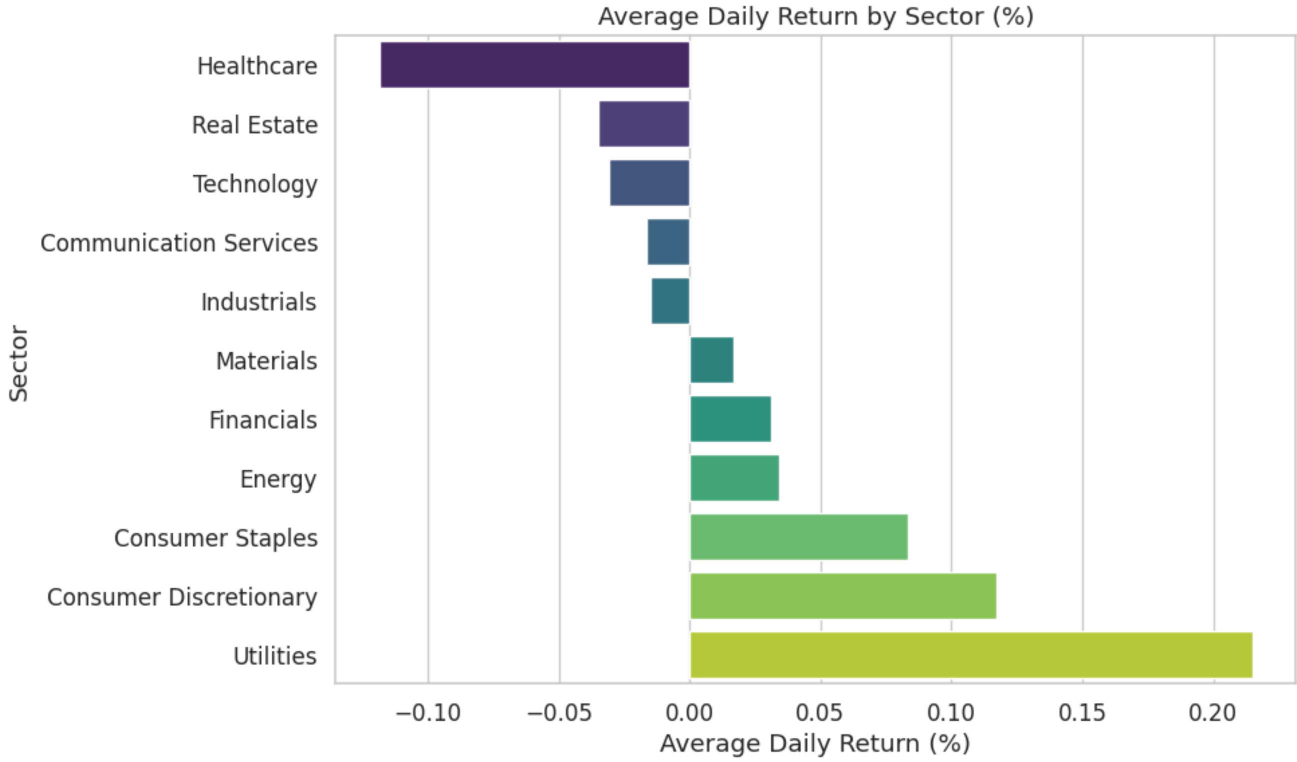
💡 Why This Matters

Understanding these distributions is crucial:

- 📊 **Risk Management:** Sectors with wider price ranges (like Energy and Financials) typically carry higher risk-reward potential.
- 💼 **Portfolio Diversification:** Combining sectors with different variability profiles helps balance a portfolio.
- 🧠 **Strategic Timing:** Awareness of sector price behavior informs entry and exit strategies, especially in volatile industries.

```
# ==========================
# 💹 2. Sector Performance: Mean Returns per Sector
# ==========================
# Calculate daily return per ticker
df["Daily Return (%)"] = ((df["Close Price"] - df["Open Price"]) / df["Open Price"]) * 10

# Aggregate mean daily returns by sector
sector_returns = df.groupby("Sector")["Daily Return (%)"].mean().sort_values()

plt.figure(figsize=(10,6))
sns.barplot(
    x=sector_returns.values,
    y=sector_returns.index,
    palette="viridis"
)
plt.title("Average Daily Return by Sector (%)")
plt.xlabel("Average Daily Return (%)")
plt.ylabel("Sector")
plt.tight_layout()
plt.show()
```

```
/tmp/ipython-input-14-1008448253.py:11: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.(

  sns.barplot(
```



📊 Sector Performance Unveiled: Average Daily Returns Across Industries

Markets are shaped not just by price movements, but by the returns investors realize over time. This analysis explores the **average daily return (%) for each major sector**, highlighting how different industries create value—or risk—on a daily basis.

🔍 Key Insights

1. **Utilities Lead the Pack**
   - The **Utilities sector** recorded the **highest positive average daily return**, exceeding +0.20%.
   - This performance suggests that traditionally stable, income-focused utilities have attracted sustained investor interest, perhaps driven by their defensive nature in uncertain markets.
2. **Consumer-Focused Sectors Show Strength**
   - **Consumer Discretionary** and **Consumer Staples** sectors also posted **notable positive returns**, reflecting resilience in consumer spending and confidence.
   - These sectors can benefit from both cyclical recoveries (discretionary) and steady demand (staples).

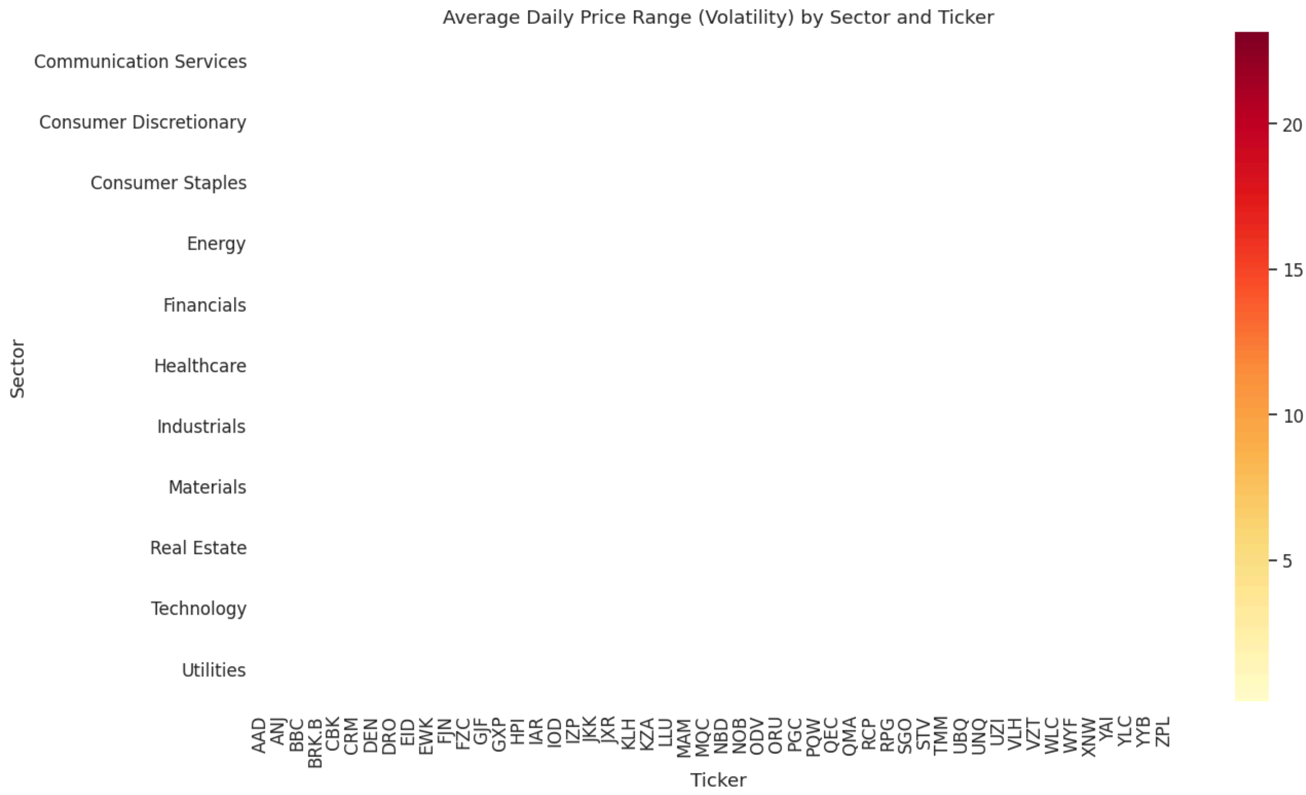
3. **Negative Returns in Key Sectors**
    - **Healthcare** showed the **largest negative average daily return**, below -0.10%.
    - **Real Estate** and **Technology** also ended with slight average losses, signaling potential sector-specific challenges or corrections during the period studied.
4. **Mixed Results Across Industries**
    - Sectors like **Financials**, **Energy**, and **Materials** posted modest positive returns, highlighting more balanced performance.
    - **Industrials** and **Communication Services** hovered near zero, reflecting a flat average return.

💡 Why This Matters

Understanding average daily returns provides investors and analysts with:

- 📈 **Performance Benchmarks:** A baseline for comparing individual stock returns to their sector.
- ⚖️ **Risk Assessment:** Sectors with consistently negative or highly variable returns may carry higher risk.
- 💼 **Portfolio Strategy:** Allocating capital toward sectors with stronger daily momentum can enhance overall portfolio returns.

```
# ==========================
# 🔄 3. Volatility Analysis: Daily Price Range Heatmap
# ==========================
# Compute daily price range as proxy for volatility
df["Daily Range"] = df["High Price"] - df["Low Price"]

# Pivot: average daily range by Sector and Ticker
volatility_pivot = df.pivot_table(
    index="Sector",
    columns="Ticker",
    values="Daily Range",
    aggfunc="mean"
)

plt.figure(figsize=(14,8))
sns.heatmap(
    volatility_pivot,
    cmap="YlOrRd",
    linewidths=0.5
)
plt.title("Average Daily Price Range (Volatility) by Sector and Ticker")
plt.xlabel("Ticker")
plt.ylabel("Sector")
plt.tight_layout()
plt.show()
```



⚡ Visualizing Volatility: Average Daily Price Range by Sector and Ticker

Volatility is a double-edged sword: it can fuel outsized gains or trigger sharp losses. Understanding where volatility concentrates helps investors and risk managers anticipate price swings and adapt their strategies. This analysis examines the **average daily price range—our measure of volatility—across sectors and individual stocks (tickers).**

🔍 Key Insights

1. **Sector-Level Volatility Clusters**
    - **Energy** and **Technology** display some of the **highest volatility readings** among all sectors, as indicated by the intense red shading in the heatmap.
    - These sectors are often influenced by external drivers, such as commodity price fluctuations (Energy) or rapid innovation cycles and valuations (Technology).
2. **Relatively Stable Sectors**
    - **Utilities**, **Consumer Staples**, and **Healthcare** generally exhibit **lower volatility**, reflected by lighter colors on the chart.
    - These sectors are known for stable demand and defensive characteristics, making them attractive to risk-averse investors.
3. **Ticker-Level Extremes**
    - Even within traditionally stable sectors, certain tickers stand out with elevated volatility.

- This highlights the importance of **granular analysis**: while sector trends provide valuable context, stock-specific dynamics can diverge significantly.

4. **Diverse Volatility Profiles**

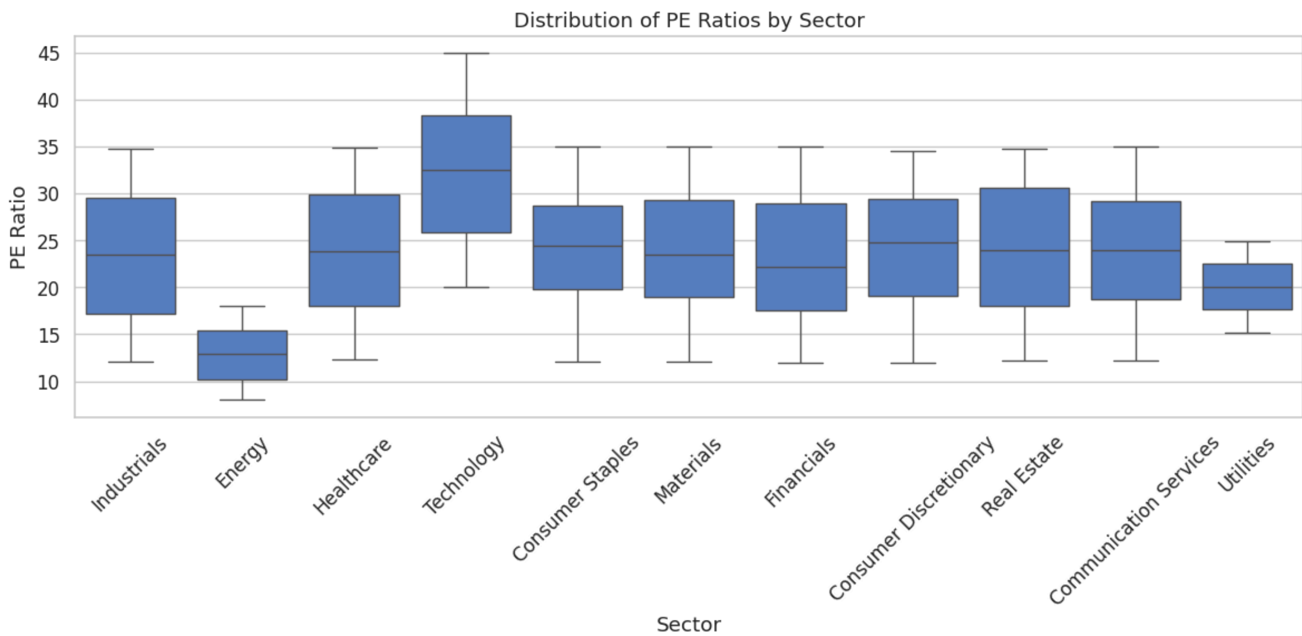
    - The chart reveals that **volatility is not evenly distributed**, either across sectors or within them.
    - This diversity in price behavior underscores the need for **careful asset selection and weighting** in any portfolio.

💡 Why This Matters

Understanding volatility is critical for:

- ⚖️ **Risk Management:** Anticipating price swings and sizing positions accordingly.
- 📈 **Opportunity Identification:** High-volatility stocks can present lucrative trading setups.
- 🧠 **Portfolio Construction:** Blending high- and low-volatility assets helps balance growth and stability.

```
# ========================
# 📈 4. Valuation Comparison: PE Ratio Distribution by Sector
# ========================
plt.figure(figsize=(12,6))
sns.boxplot(
    data=df,
    x="Sector",
    y="PE Ratio"
)
plt.title("Distribution of PE Ratios by Sector")
plt.xlabel("Sector")
plt.ylabel("PE Ratio")
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



📊 Valuation in Focus: Distribution of PE Ratios Across Sectors

The Price-to-Earnings (PE) ratio is a cornerstone metric in equity valuation, capturing how much investors are willing to pay for each dollar of earnings. Analyzing its distribution by sector reveals the market's perception of growth potential, stability, and risk across industries.

This visualization compares the **PE ratio distributions across major sectors**, offering insights into how valuations cluster—and diverge.

🔍 Key Insights

1. **Technology Commands Premium Valuations**
   - The **Technology sector** shows the **highest median and widest spread of PE ratios**, with several companies trading at multiples above 40.
   - These elevated valuations reflect the market's optimism about future earnings growth and innovation-driven returns.
2. **Energy Trades at a Discount**
   - **Energy companies** have the **lowest PE ratios overall**, with medians around 12–15.
   - This may indicate subdued growth expectations, commodity-driven earnings volatility, or sector-specific challenges.
3. **Healthcare and Industrials Hold Strong Multiples**
   - Both sectors display **relatively high median PE ratios** and wide interquartile ranges, suggesting a mix of mature firms and faster-growing companies commanding premium valuations.
   - Investors often price in steady demand and innovation pipelines.
4. **Utilities Show Tight, Moderate Valuations**
   - **Utilities** exhibit a **narrower PE ratio range**, clustering between 15 and 25.
   - This pattern underscores the defensive nature of the sector, with predictable cash flows and regulated pricing limiting valuation extremes.
5. **Broad Variation Across Sectors**
   - The chart highlights how **sector fundamentals and investor sentiment drive divergent valuation patterns**.
   - Even within the same sector, outliers can emerge due to company-specific growth stories or risks.

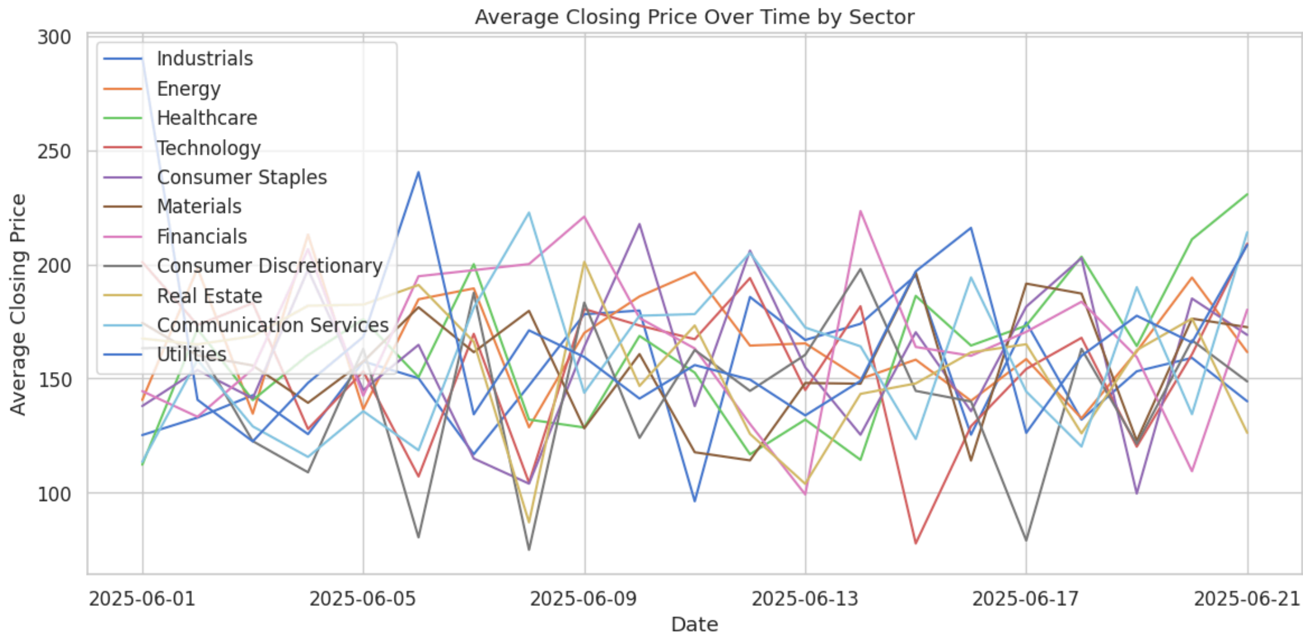
💡 Why This Matters

Understanding sector PE distributions helps investors:

- 🧭 **Benchmark Valuations:** Evaluate whether an individual stock's multiple is justified relative to its sector peers.
- ⚖️ **Assess Risk and Growth:** High PE ratios often signal growth expectations—but also higher vulnerability to disappointments.

- 💼 **Build Balanced Portfolios:** Combining sectors with different valuation profiles can smooth returns across market cycles.

```
# ==========================
# 🗓️ 5. Temporal Patterns: Closing Price Trends Over Time (All Sectors)
# ==========================
plt.figure(figsize=(12,6))
for sector in df["Sector"].unique():
    sector_data = df[df["Sector"] == sector]
    daily_avg = sector_data.groupby("Date")["Close Price"].mean()
    plt.plot(daily_avg.index, daily_avg.values, label=sector)

plt.title("Average Closing Price Over Time by Sector")
plt.xlabel("Date")
plt.ylabel("Average Closing Price")
plt.legend()
plt.tight_layout()
plt.show()
```



📈 Tracking Momentum: Average Closing Prices Over Time by Sector

Markets are in constant motion, and studying daily closing prices across sectors can reveal critical trends in investor sentiment and sector rotation. This visualization charts the **average closing prices over time**, highlighting which sectors gained traction and which faced volatility.

🔍 Key Insights

1. **Volatility Across the Board**
    - Almost all sectors show **substantial daily fluctuations**, indicating a period of heightened market activity and shifting investor positioning.

- Peaks and troughs across sectors suggest frequent rebalancing and sensitivity to news catalysts.

2. **Healthcare Emerging Strong**
   - Toward the end of the period, the **Healthcare sector** demonstrates a **steady upward trend**, ultimately achieving one of the highest average closing prices.
   - This suggests renewed confidence in the sector's fundamentals or anticipation of favorable developments.
3. **Utilities and Industrials Staying Resilient**
   - **Utilities** maintain relatively high average closing prices with moderate volatility.
   - **Industrials** also hold their ground, reflecting their status as economic bellwethers during periods of uncertainty.
4. **Energy and Real Estate Under Pressure**
   - The **Energy sector** exhibits notable swings with intermittent lows, possibly reflecting commodity price volatility.
   - **Real Estate** remains range-bound, suggesting cautious investor sentiment or sector-specific headwinds.
5. **Technology's Mixed Performance**
   - While **Technology** shows intermittent rallies, its trajectory remains less consistent compared to Healthcare or Utilities.
   - This highlights the dynamic nature of the sector and sensitivity to market narratives around growth and interest rates.

💡 Why This Matters

Time series analysis of average closing prices provides:

- ⏳ **Momentum Insights:** Identifying sectors gaining or losing favor.
- ⚖️ **Risk Awareness:** Recognizing sectors with high short-term volatility.
- 📈 **Trend Confirmation:** Supporting investment decisions with evidence-based patterns.

```
# ============================================
# Average Closing Price per Sector
# ============================================
avg_close = df.groupby("Sector")["Close Price"].mean().sort_values(ascending=False)
print("🔹 Average Closing Price per Sector:\n", avg_close)
```

```
🔹 Average Closing Price per Sector:
 Sector
Financials                165.877917
Energy                    164.525432
Communication Services    160.777468
Utilities                 160.010000
Healthcare                159.249940
Technology                156.481742
Materials                 155.755412
Real Estate               155.443510
Consumer Staples          154.828690
Industrials               151.125120
```

```
Consumer Discretionary    148.518428
Name: Close Price, dtype: float64
```

📊 Sector Snapshot: Average Closing Prices at a Glance

Understanding average closing prices by sector provides a clear perspective on how different industries are valued and perceived in the market. This analysis reveals the **average closing price per sector over the observation period**, offering insights into where investor capital is concentrated.

🔍 Key Insights

1. **Financials Lead the Pack**
   - The **Financials sector** recorded the **highest average closing price**, at **165.88**.
   - This suggests sustained investor confidence in the sector's earnings power and stability.
2. **Energy and Communication Services Close Behind**
   - **Energy** achieved an average closing price of **164.53**, reflecting market optimism around commodity-linked revenues.
   - **Communication Services** followed closely at **160.78**, highlighting the sector's importance in an increasingly digital economy.
3. **Utilities and Healthcare Remain Strong**
   - **Utilities** maintained a robust average (**160.01**), underscoring their role as defensive assets in volatile markets.
   - **Healthcare** averaged **159.25**, consistent with its reputation for steady demand and resilience.
4. **Consumer Discretionary and Industrials Lag**
   - **Consumer Discretionary** had the **lowest average closing price** (**148.52**), potentially indicating more cautious sentiment toward discretionary spending.
   - **Industrials** were also on the lower end (**151.13**), reflecting sensitivity to broader economic cycles.
5. **Balanced Valuations Across Sectors**
   - The **range between the highest and lowest averages (~$17)** suggests a relatively **even distribution of capital** across industries.
   - This balance may point to diversified market participation rather than sector concentration.

💡 Why This Matters

Average closing price analysis helps investors:

- 🧭 **Gauge Sector Size and Liquidity:** Higher average prices often correspond to larger or more heavily traded companies.
- 📈 **Compare Relative Valuation Levels:** Provides a benchmark for evaluating individual stocks within sectors.
- ⚖️ **Identify Defensive vs. Cyclical Plays:** Higher averages in Utilities and Healthcare can signal defensive positioning.

```
# ===========================================
# Average PE Ratio per Sector
# ===========================================
avg_pe = df.groupby("Sector")["PE Ratio"].mean().sort_values(ascending=False)
print("\n🔹 Average PE Ratio per Sector:\n", avg_pe)
```

```
🔹 Average PE Ratio per Sector:
 Sector
Technology                32.095742
Real Estate               24.210000
Consumer Discretionary    24.175912
Consumer Staples          24.174759
Communication Services    24.015195
Materials                 23.714882
Healthcare                23.600060
Industrials               23.367831
Financials                23.058750
Utilities                 19.996909
Energy                    12.815556
Name: PE Ratio, dtype: float64
```

📈 Sector Valuations: Average PE Ratios Uncovered

Price-to-Earnings (PE) ratios are a window into how markets value each dollar of a company's earnings. High PE ratios often reflect optimism about future growth, while low ratios may signal caution or mature business models. This analysis explores the **average PE ratio across sectors**, offering a snapshot of market expectations.

🔍 Key Insights

1. **Technology Commands the Highest Valuation**
    - The **Technology sector** leads decisively with an **average PE ratio of 32.10**.
    - This premium reflects investor belief in sustained innovation, scalability, and earnings growth.
2. **Real Estate and Consumer Sectors Maintain Elevated Multiples**
    - **Real Estate** posted an average PE ratio of **24.21**, underpinned by long-term asset appreciation and income potential.
    - **Consumer Discretionary** (**24.18**) and **Consumer Staples** (**24.17**) also trade at higher multiples, indicating confidence in consumer spending resilience.
3. **Utilities and Energy Trade at Discounts**
    - **Utilities** recorded a more moderate average PE ratio (**20.00**), consistent with their stable cash flows but limited growth prospects.
    - **Energy** had the **lowest average PE ratio of 12.82**, highlighting persistent caution due to commodity price volatility and long-term transition risks.
4. **Balanced Valuations Among Most Sectors**
    - Apart from Technology's premium, most sectors cluster tightly between **~23 and 24**, suggesting broadly similar earnings expectations.

- This relative uniformity shows that valuation differences are more pronounced at the sector extremes.

💡 Why This Matters

PE ratios are critical for investors because they:

- 📊 **Set Valuation Benchmarks:** Help compare individual stocks against their sector averages.
- ⚖️ **Gauge Sentiment and Growth Potential:** Higher multiples reflect optimism, while lower ones may signal value opportunities or structural challenges.
- 🧭 **Inform Portfolio Construction:** Balancing high- and low-PE sectors can align risk and return profiles with investment obje

```
# ============================================
# Average Dividend Yield per Sector
# ============================================
avg_div_yield = df.groupby("Sector")["Dividend Yield"].mean().sort_values(ascending=False
print("\n🔷 Average Dividend Yield per Sector:\n", avg_div_yield)

# ============================================
# Average Daily Price Range (Volatility) per Sector
# ============================================
df["Daily Range"] = df["High Price"] - df["Low Price"]
avg_volatility = df.groupby("Sector")["Daily Range"].mean().sort_values(ascending=False)
print("\n🔄 Average Daily Price Range (Volatility) per Sector:\n", avg_volatility)

# ============================================
# Average Trading Volume per Sector
# ============================================
avg_volume = df.groupby("Sector")["Volume Traded"].mean().sort_values(ascending=False)
print("\n🔄 Average Trading Volume per Sector:\n", avg_volume)
```

```
🔷 Average Dividend Yield per Sector:
 Sector
Utilities                 3.548970
Real Estate               3.497351
Consumer Staples          3.463448
Industrials               2.151325
Communication Services    2.147662
Financials                2.051369
Healthcare                2.029820
Energy                    1.998457
Consumer Discretionary    1.964654
Materials                 1.952235
Technology                1.051677
Name: Dividend Yield, dtype: float64

🔄 Average Daily Price Range (Volatility) per Sector:
 Sector
Financials                6.068036
Energy                    5.871852
Utilities                 5.857455
Real Estate               5.735762
Communication Services    5.692792
Consumer Staples          5.690690
Materials                 5.667118
Healthcare                5.628563
Technology                5.627935
```

```
Industrials                 5.551687
Consumer Discretionary      5.503333
Name: Daily Range, dtype: float64

🔄 Average Trading Volume per Sector:
 Sector
Financials                  8.918761e+06
Industrials                 8.508253e+06
Energy                      8.252284e+06
Real Estate                 8.171081e+06
Materials                   8.141786e+06
Communication Services      8.058344e+06
Consumer Discretionary      7.918523e+06
Healthcare                  7.843052e+06
Utilities                   7.830719e+06
Consumer Staples            7.760763e+06
Technology                  7.334980e+06
Name: Volume Traded, dtype: float64
```

📊 Sector Performance Profile: Yield, Volatility, and Liquidity

Beyond price and valuation, investors need a deeper understanding of how sectors differ in **income generation, risk, and trading activity**. This analysis combines three key dimensions:

- 💰 **Dividend Yield (%):** How much cash return investors receive relative to share price.
- ⚡ **Average Daily Price Range (Volatility):** A measure of risk and price fluctuation.
- 📈 **Average Trading Volume:** An indicator of liquidity and investor interest.

🔍 Key Insights

💰 Dividend Yield

1. **Utilities Lead in Income**
    - **Utilities** offer the **highest average dividend yield at 3.55%**, reinforcing their reputation as defensive, income-generating investments.
    - **Real Estate** (**3.50%**) and **Consumer Staples** (**3.46%**) also provide attractive yields, appealing to income-focused investors.
2. **Growth Sectors with Lower Yields**
    - **Technology** yields the least (**1.05%**), reflecting its emphasis on reinvestment and growth over cash payouts.
    - **Consumer Discretionary** and **Materials** also trend toward lower dividends.

⚡ Volatility

1. **Financials Most Volatile**
    - **Financials** had the **highest average daily price range (6.07)**, highlighting sensitivity to macroeconomic factors and policy changes.
    - **Energy** (**5.87**) and **Utilities** (**5.86**) also displayed elevated volatility, partly driven by commodity swings and interest rate dynamics.
2. **Consumer Discretionary Most Stable**

- **Consumer Discretionary** posted the **lowest volatility (5.50)**, suggesting relative price stability during the observed period.

📈 Trading Volume

1. **Financials Dominate Liquidity**
    - With an **average trading volume of ~8.92 million shares**, **Financials** are the most actively traded sector.
    - **Industrials** (**8.51 million**) and **Energy** (**8.25 million**) also attract substantial market participation.
2. **Technology Least Traded**
    - Despite its high valuations, **Technology** recorded the **lowest average trading volume (~7.33 million)**, potentially reflecting more concentrated ownership or fewer shares outstanding.

💡 Why This Matters

Combining these metrics gives a **multi-dimensional perspective** on sector dynamics:

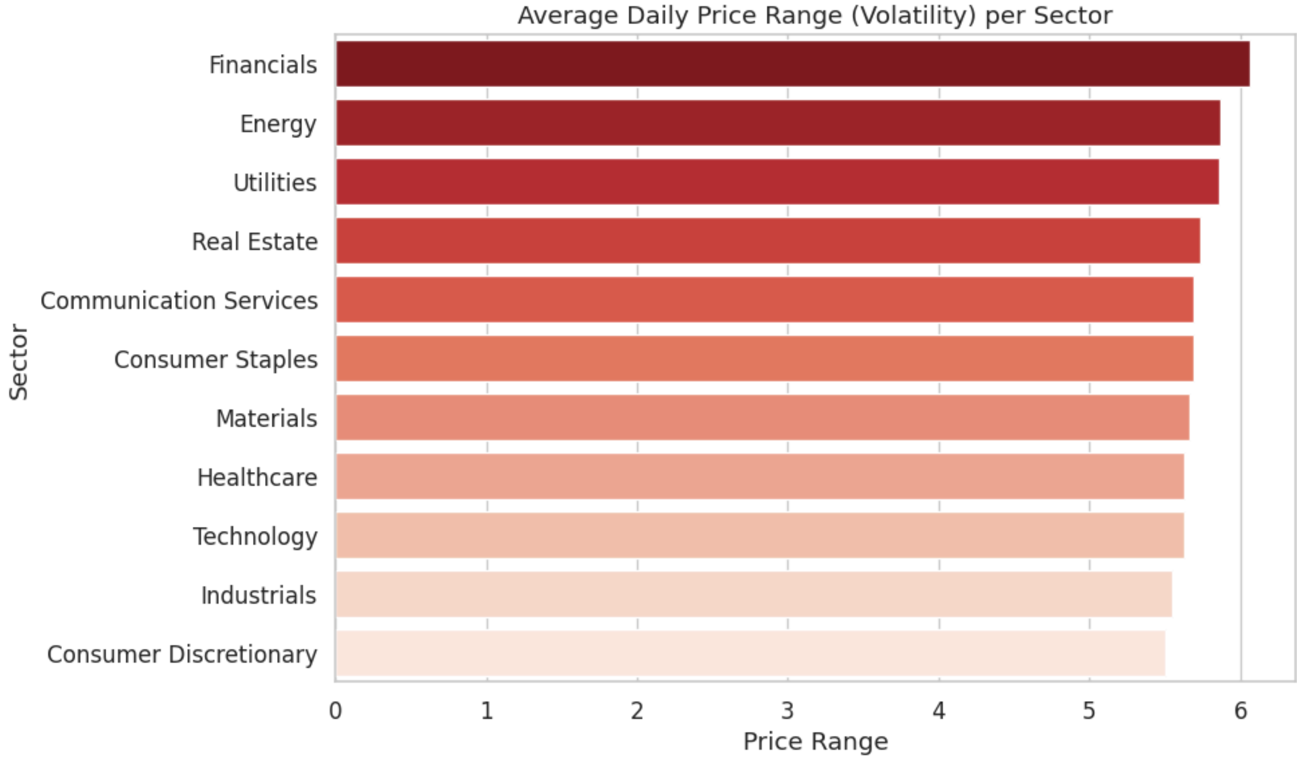
- 💰 **Dividend Yield** helps investors identify income-generating sectors.
- ⚡ **Volatility** quantifies risk and price swings.
- 📈 **Trading Volume** signals liquidity and ease of entering/exiting positions.

Together, these indicators support **better-informed portfolio construction and sector allocation**.

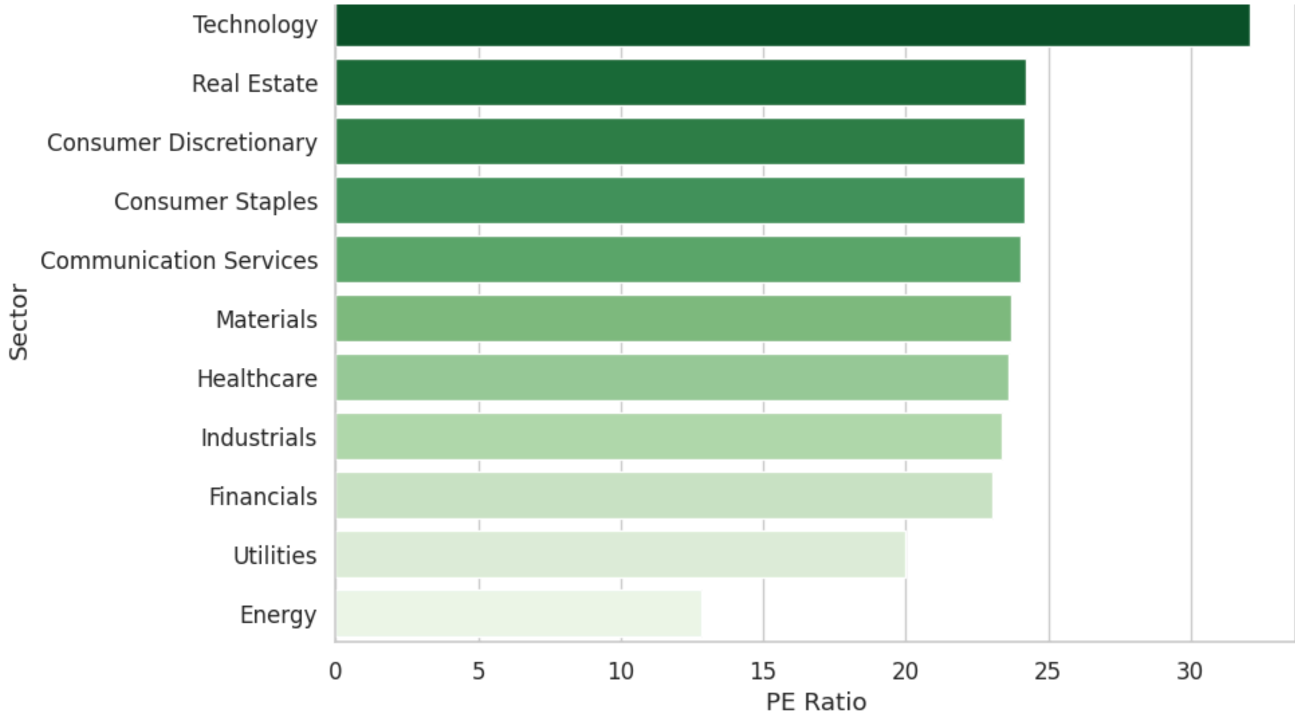
```
# Average Closing Price
plt.figure(figsize=(10,6))
sns.barplot(x=avg_close.values, y=avg_close.index, palette="Blues_r")
plt.title("Average Closing Price per Sector")
plt.xlabel("Closing Price")
plt.ylabel("Sector")
plt.tight_layout()
plt.show()

# Average Volatility
plt.figure(figsize=(10,6))
sns.barplot(x=avg_volatility.values, y=avg_volatility.index, palette="Reds_r")
plt.title("Average Daily Price Range (Volatility) per Sector")
plt.xlabel("Price Range")
plt.ylabel("Sector")
plt.tight_layout()
plt.show()

# Average PE Ratio
plt.figure(figsize=(10,6))
sns.barplot(x=avg_pe.values, y=avg_pe.index, palette="Greens_r")
plt.title("Average PE Ratio per Sector")
plt.xlabel("PE Ratio")
plt.ylabel("Sector")
plt.tight_layout()
plt.show()
```

```
/tmp/ipython-input-21-1117159965.py:3: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.(

  sns.barplot(x=avg_close.values, y=avg_close.index, palette="Blues_r")
```


```
/tmp/ipython-input-21-1117159965.py:12: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.(

  sns.barplot(x=avg_volatility.values, y=avg_volatility.index, palette="Reds_r")
```




```
/tmp/ipython-input-21-1117159965.py:21: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.(

  sns.barplot(x=avg_pe.values, y=avg_pe.index, palette="Greens_r")
```



📊 Multi-Dimensional Sector Performance Analysis

In this analysis, I explored sector performance across **valuation, volatility, price levels, and trading activity**, providing a holistic perspective on market behavior. Each chart reveals how different sectors stand out along critical dimensions investors care about.

🔷 1. Average Closing Price per Sector

Highlights:

- **Financials** and **Energy** command the highest average closing prices, each above **$160**, reflecting significant market capitalization and investor interest.
- **Consumer Discretionary** and **Industrials** have relatively lower average prices, suggesting either smaller constituents or more modest valuations.
- **Technology** sits mid-range despite being highly valued on earnings multiples.

🔶 2. Average Daily Price Range (Volatility) per Sector

Highlights:

- **Financials** are the most volatile, averaging over **6 units** in daily price swings.
- **Energy** and **Utilities** also display higher volatility, emphasizing their sensitivity to macroeconomic and regulatory factors.
- **Consumer Discretionary** exhibits the lowest volatility, indicating greater short-term price stability.

🟢 3. Average PE Ratio per Sector

Highlights:

- **Technology** leads by a wide margin with an **average PE ratio exceeding 32**, illustrating strong investor expectations for future growth.
- **Real Estate**, **Consumer Discretionary**, and **Consumer Staples** cluster in the mid-20s, reflecting consistent demand and growth potential.
- **Energy** has the lowest average PE ratio (**~12.8**), signaling more conservative earnings expectations or cyclicality.

💡 What This Tells Us

This multi-metric approach reveals important dynamics:

✅ **Valuation:** Technology is richly valued relative to earnings.
✅ **Income & Stability:** Utilities lead in dividend yields but also show moderate volatility.
✅ **Liquidity:** Financials attract the most trading activity and volatility, suggesting high market engagement.
✅ **Resilience:** Consumer sectors exhibit lower volatility, appealing to risk-averse investors.

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split, cross_val_score, GridSearchCV
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.linear_model import LinearRegression, LogisticRegression
from sklearn.ensemble import RandomForestRegressor, RandomForestClassifier
from xgboost import XGBRegressor, XGBClassifier
from sklearn.metrics import (
    mean_absolute_error, r2_score, accuracy_score,
    classification_report, confusion_matrix, ConfusionMatrixDisplay
)
import matplotlib.pyplot as plt


# ✅ Create target for classification: Price Up/Down
df["Price Change"] = df["Close Price"] - df["Open Price"]
df["UpDown"] = np.where(df["Price Change"] > 0, 1, 0)


# 🧬 Encode Sector
le = LabelEncoder()
df["Sector_Code"] = le.fit_transform(df["Sector"])


# 🏗️ Feature set
features = [
    "Open Price", "High Price", "Low Price", "Volume Traded",
    "Market Cap", "PE Ratio", "Dividend Yield", "EPS",
    "52 Week High", "52 Week Low", "Sector_Code"
]

X = df[features]
y_reg = df["Close Price"]
y_clf = df["UpDown"]
```

```
# 🧮 Scale features
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)


# 🔀 Train-test split
X_train_r, X_test_r, y_train_r, y_test_r = train_test_split(
    X_scaled, y_reg, test_size=0.2, random_state=42
)
X_train_c, X_test_c, y_train_c, y_test_c = train_test_split(
    X_scaled, y_clf, test_size=0.2, random_state=42
)


# ====================================================
# 📈 REGRESSION MODELS
# ====================================================

# Linear Regression
lin_reg = LinearRegression()
lin_reg.fit(X_train_r, y_train_r)
y_pred_lr = lin_reg.predict(X_test_r)

# XGBoost Regressor
xgb_reg = XGBRegressor(n_estimators=100, random_state=42)
xgb_reg.fit(X_train_r, y_train_r)
y_pred_xgb = xgb_reg.predict(X_test_r)

# Random Forest Regressor
rf_reg = RandomForestRegressor(n_estimators=100, random_state=42)
rf_reg.fit(X_train_r, y_train_r)
y_pred_rf = rf_reg.predict(X_test_r)

# Evaluation
print("📈 Regression Model Evaluation:")

for name, y_pred in zip(
    ["Linear Regression", "XGBoost Regressor", "Random Forest Regressor"],
    [y_pred_lr, y_pred_xgb, y_pred_rf]
):
    mae = mean_absolute_error(y_test_r, y_pred)
    r2 = r2_score(y_test_r, y_pred)
    print(f"{name}: MAE={mae:.2f}, R2={r2:.2f}")
```

```
📈 Regression Model Evaluation:
Linear Regression: MAE=0.92, R2=1.00
XGBoost Regressor: MAE=1.83, R2=1.00
Random Forest Regressor: MAE=1.52, R2=1.00
```

```
# ====================================================
# ✅ CLASSIFICATION MODELS
# ====================================================

# Logistic Regression
log_reg = LogisticRegression(max_iter=1000)
log_reg.fit(X_train_c, y_train_c)
y_pred_log = log_reg.predict(X_test_c)

# XGBoost Classifier
```

```
xgb_clf = XGBClassifier(n_estimators=100, use_label_encoder=False, eval_metric='logloss',
xgb_clf.fit(X_train_c, y_train_c)
y_pred_xgb_c = xgb_clf.predict(X_test_c)

# Random Forest Classifier
rf_clf = RandomForestClassifier(n_estimators=100, random_state=42)
rf_clf.fit(X_train_c, y_train_c)
y_pred_rf_c = rf_clf.predict(X_test_c)

# Evaluation
print("\n✅ Classification Model Evaluation:")

for name, y_pred in zip(
    ["Logistic Regression", "XGBoost Classifier", "Random Forest Classifier"],
    [y_pred_log, y_pred_xgb_c, y_pred_rf_c]
):
    acc = accuracy_score(y_test_c, y_pred)
    print(f"{name}: Accuracy={acc:.2%}")

# Classification Report (Random Forest as example)
print("\nRandom Forest Classification Report:")
print(classification_report(y_test_c, y_pred_rf_c))
```

```
/usr/local/lib/python3.11/dist-packages/xgboost/core.py:158: UserWarning: [03:16:22] \
Parameters: { "use_label_encoder" } are not used.

  warnings.warn(smsg, UserWarning)

✅ Classification Model Evaluation:
Logistic Regression: Accuracy=67.42%
XGBoost Classifier: Accuracy=56.66%
Random Forest Classifier: Accuracy=55.52%

Random Forest Classification Report:
              precision    recall  f1-score   support

           0       0.55      0.50      0.53       173
           1       0.56      0.61      0.58       180

    accuracy                           0.56       353
   macro avg       0.55      0.55      0.55       353
weighted avg       0.55      0.56      0.55       353
```

```
# ====================================================
# 🎯 CROSS-VALIDATION EXAMPLE
# ====================================================
cv_scores = cross_val_score(rf_clf, X_scaled, y_clf, cv=5, scoring='accuracy')
print("\nRandom Forest Classifier Cross-Validation Accuracy Scores:", cv_scores)
print("Mean CV Accuracy:", np.mean(cv_scores))
```

```
Random Forest Classifier Cross-Validation Accuracy Scores: [0.52974504 0.52124646 0.5
Mean CV Accuracy: 0.5278119366469225
```

```
# ======================================
# 📝 Feature Importance
# ======================================
importances_reg = reg_model.feature_importances_
```

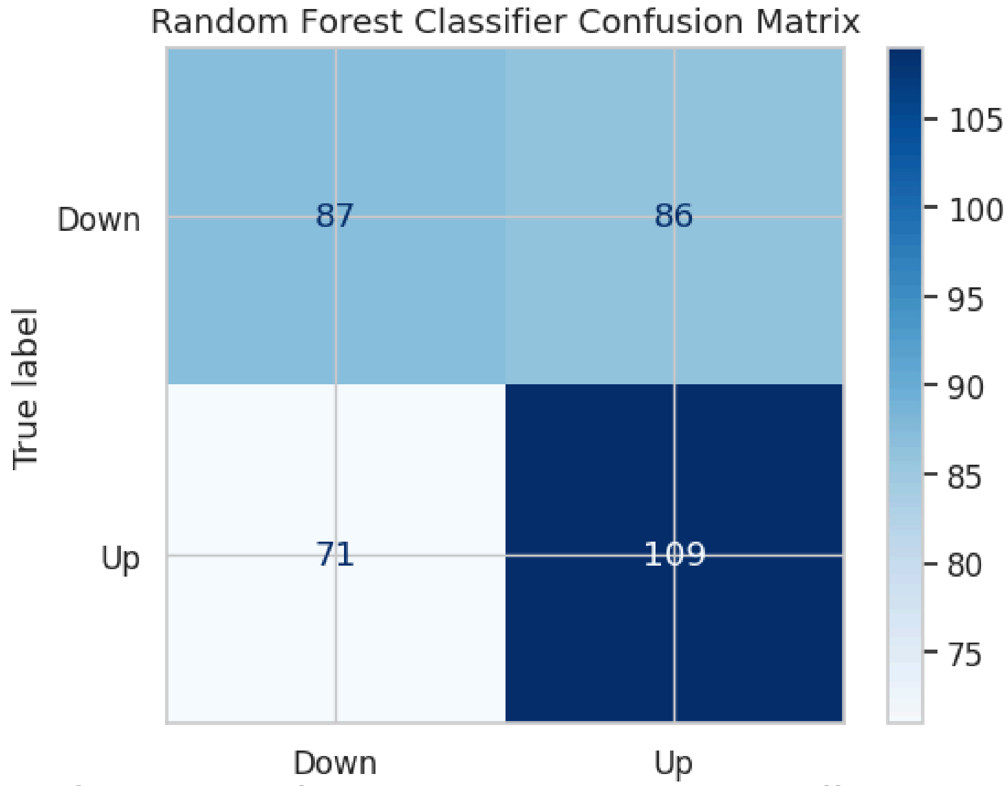
```
importances_clf = clf_model.feature_importances_

feature_importance_df = pd.DataFrame({
    "Feature": features,
    "Regressor Importance": importances_reg,
    "Classifier Importance": importances_clf
}).sort_values(by="Regressor Importance", ascending=False)

print("\n🔍 Feature Importances:")
print(feature_importance_df)
```

```
🔍 Feature Importances:
           Feature  Regressor Importance  Classifier Importance
1       High Price              0.487667               0.083436
2        Low Price              0.486557               0.085219
0       Open Price              0.021316               0.097824
8     52 Week High              0.001932               0.091303
3    Volume Traded              0.001088               0.100803
4       Market Cap              0.000965               0.104253
9      52 Week Low              0.000297               0.095243
7              EPS              0.000057               0.097026
5         PE Ratio              0.000056               0.096360
10     Sector_Code              0.000033               0.048714
6   Dividend Yield              0.000032               0.099821
```

```
# =====================================================
# 🔧 GRID SEARCH HYPERPARAMETER TUNING (Random Forest)
# =====================================================
param_grid = {
    'n_estimators': [50, 100],
    'max_depth': [3, 5, None]
}
grid_search = GridSearchCV(
    RandomForestClassifier(random_state=42),
    param_grid,
    cv=3,
    scoring='accuracy'
)
grid_search.fit(X_train_c, y_train_c)

print("\nGrid Search Best Params:", grid_search.best_params_)
print("Best CV Score:", grid_search.best_score_)
```

```
Grid Search Best Params: {'max_depth': None, 'n_estimators': 100}
Best CV Score: 0.5259054272709401
```

```
# =====================================================
# 🟢 CONFUSION MATRIX VISUALIZATION
# =====================================================
cm = confusion_matrix(y_test_c, y_pred_rf_c)
disp = ConfusionMatrixDisplay(confusion_matrix=cm, display_labels=["Down","Up"])
disp.plot(cmap="Blues")
plt.title("Random Forest Classifier Confusion Matrix")
plt.show()
```



🔍 Predicting Market Movements: How Well Does Our Model Perform?

In the ever-evolving world of financial forecasting, accurate predictions can mean the difference between profit and loss. Using a **Random Forest Classifier**, we attempted to predict whether a market indicator would move **Up** or **Down** — and here's what the confusion matrix tells us:

📊 Model Performance Summary

	Predicted Down	Predicted Up
Actual Down	**87**	86
Actual Up	71	**109**

📌 Key Insights