Stock Price Trend Prediction

Using Machine Learning

Presented by :

Shon Khundiashvili

Netanel Yomtovian

Abut the project

Why did we choose this topic?

First of all, due to the corona virus spread the stock prices change everyday dramatically.

We like this topic and interested about stocks and its prices. In conclusion, we decided to investigate what are the reasons for the change of stock prices and what can effect the price of the stock.

We personally invest in the capital market and we have always been interested in knowing what are the characteristics through which the stock price trend can be predicted.

Crawling

.Using **Beautiful Soup** and **Selenuim** library

Step 1

Crawling the list of all stocks in Nasdaq with their Sector names

- We created two arrays with the name 'Symbols' and 'Sectors' and appending to them each stock data.

```
letters = ['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J', 'K', 'L', 'M', 'N', 'O', 'P',
           'Q', 'R', 'S', 'T', 'U', 'V', 'W', 'X', 'Y', 'Z']

for letter in letters:
    New_Stock_List_Url = Stock_List_Url + letter
    response  = requests.get(New_Stock_List_Url)
    soup = BeautifulSoup(response.content, 'html.parser')

    odd_rows = soup.find_all('tr', attrs = {"class" : 'ts1'})
    even_rows = soup.find_all('tr' , attrs = {"class" : 'ts0'})

    for item in odd_rows:
        stock = item.find_all('td')
        Symbols.append(stock[1].get_text())

    for item in even_rows:
        stock = item.find_all('td')
        Symbols.append(stock[1].get_text())

    time.sleep(5)
```

Step 2

Crawlig Data for 15 columns:

- **Avg_50** - the average stock price in 50 last days.
- **Avg_200** - the average stock price in 200 last days.
- **Week_52_High** - Highest price in last 50 days.
- **Week_52_Low** - Lowest price in last 50 days
- **Perf_Month** - the changes price value in last month (%)
- **Perf_Quarter** - the changes price value in last quarter (%)
- **Perf_Half_Year** - the changes price value in last half year (%)
- **Perf_Year** - the changes price value in last year (%)

```
for symbol in Symbols:
    New_Finviz_Url = Finviz_Url + symbol
    response = requests.get(New_Finviz_Url, headers = user_agent)
    soup = BeautifulSoup(response.content, 'html.parser')

    trs = soup.find_all('tr', attrs = {'class': 'table-dark-row'})

    tds = trs[0].find_all('td')
    try:
        Perf_Week.append(tds[11].get_text())
    except:
        Perf_Week.append(np.nan)

    tds = trs[1].find_all('td')
    try:
        Perf_Month.append(tds[11].get_text())
    except:
        Perf_Month.append(np.nan)

    tds = trs[2].find_all('td')
    try:
        Perf_Quarter.append(tds[11].get_text())
    except:
        Perf_Quarter.append(np.nan)

    tds = trs[3].find_all('td')
    try:
        Perf_Half_Year.append(tds[11].get_text())
    except:
        Perf_Half_Year.append(np.nan)

    tds = trs[4].find_all('td')
    try:
        Perf_Year.append(tds[11].get_text())
    except:
        Perf_Year.append(np.nan)

    tds = trs[4].find_all('td')
    try:
        Analyst_Mean_Target_Price.append(tds[9].get_text())
```

Step 2

- **Perf_Week** - The changes price value in last year (%)
- **Avg_3_Month** –The average price in last 3 month.
- **Vol_Avg_10_Day** – The volume average in last 10 days
- **Vol_Avg_3_Months** - The volume average in last 3 month
- **Current_Stock_Price** – The current stock price
- **Held_By_Institusion** - Percentage of the stock held by the institutions
- **Analyst_Mean_Target_Price** - Target price presented by analysts for the coming year

```
iver = webdriver.Chrome(PATH)
Q_Net = [[], []]
i=1
url='https://finviz.com/quote.ashx?t='
for symbol in Symbols:
    driver.get(url + symbol)
    print("menaya:" ,i)
    i+=1
    try:
        time.sleep(2)
        driver.find_element(By.CSS_SELECTOR,'#statements > table.fullview-links > tbody > tr > td:nth-child(2) >
        time.sleep(2)
        try:
            q1=driver.find_element(By.XPATH,'//*[@id="statements"]/table[2]/tbody/tr[15]/td[3]').text
            print(q1)
            Q_Net[0].append(q1)
        except:
            print('helem')
            Q_Net[0].append(np.nan)
        try:
            q2=driver.find_element(By.XPATH,'//*[@id="statements"]/table[2]/tbody/tr[15]/td[4]').text
            print(q2)
            Q_Net[1].append(q2)
        except:
            print('helem')
            Q_Net[1].append(np.nan)
    except:
        print('HELEM')
        Q_Net[0].append(np.nan)
        Q_Net[1].append(np.nan)
```

OUR DATA FRAME

In total we collected 5399 stocks

```
In [3]:  1 df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5399 entries, 0 to 5398
Data columns (total 18 columns):
 #   Column                      Non-Null Count  Dtype  
---  ------                      --------------  -----  
 0   Symbols                     5399 non-null   object 
 1   Avg_50                      3785 non-null   object 
 2   Avg_200                     3785 non-null   object 
 3   Week_52_High                3811 non-null   object 
 4   Week_52_Low                 3811 non-null   object 
 5   Vol_Avg_10_Day              3762 non-null   object 
 6   Vol_Avg_3_Months            3779 non-null   object 
 7   Held_By_Institusion(%)      3409 non-null   object 
 8   Perf_Week(%)                3186 non-null   object 
 9   Perf_Month(%)               3186 non-null   object 
 10  Perf_Quarter(%)             3186 non-null   object 
 11  Perf_Half_Year(%)           3186 non-null   object 
 12  Perf_Year(%)                3186 non-null   object 
 13  Analyst_Mean_Target_Price   3186 non-null   object 
 14  Current_Stock_Price         3186 non-null   float64
 15  Last_Q_Net                  2582 non-null   object 
 16  Prev_Q_Net                  2576 non-null   object 
 17  Sector                      3647 non-null   object 
dtypes: float64(1), object(17)
memory usage: 759.4+ KB
```

External Resources

- **Yahoo finance**/moc.oohay.ecnanfi//:sptth -
- **Finviz** /moc.zivnfi//:sptth -
- **ADVFN** psa.qadsan/qadsan/moc.nfvda.www//:sptth -

- **Stock market**

The stock market or equity market is the market for the trading of company stock (shares).

Share / Stock

Shares represent a fraction of ownership in a business. The common feature of all these is equity participation. Different classes of shares have different voting rights.

What is stock value per share?

The market value per share is **a company's current stock price**, and it reflects a value that market participants are willing to pay for its common share.

Handling Missing Data

- Understanding the data: Data Types, missing data, mean, unique, count, etc...

- Replacing all " ," --> "" "%" --> "" . In order to convert every data types to float.

```
In [5]:
# replace all chars: '%' ','

df.replace({'%':'', ',':''},regex=True, inplace=True)
```

- The columns "Vol_Avg_10_Day" and "Vol_Avg_3_month" included the signs "k" (thousand) "m" (milions). In order to deal with this case we found all the cells that Including 'k' or 'm'. With the sign 'k' we multiplied by thousand and with 'm' we multiplied by million.

```
# we want to take care on k(thousnd) and m(milions)
#Vol_Avg_10_Day:

for index in range(5399):
    if("k" in df.iloc[index,5]):
        df.iloc[index,5] = float(df.iloc[index,5].split('k')[0])*1000
    elif("M" in df.iloc[index,5]):
        df.iloc[index,5] = float(df.iloc[index,5].split('M')[0])*1000000
```

- Filling all the cells with Nan by the median of specific columns:

```
# fill all nan cells in median
df.fillna(df.median().round(2), inplace = True)
```

- Dividing all the columns that were presented with percent sign by 100

```
# devide all cols in 100
df[cols] = df[cols].div(100)
df.info()
```

- We removed all the rows that all of them contained 'Nan'

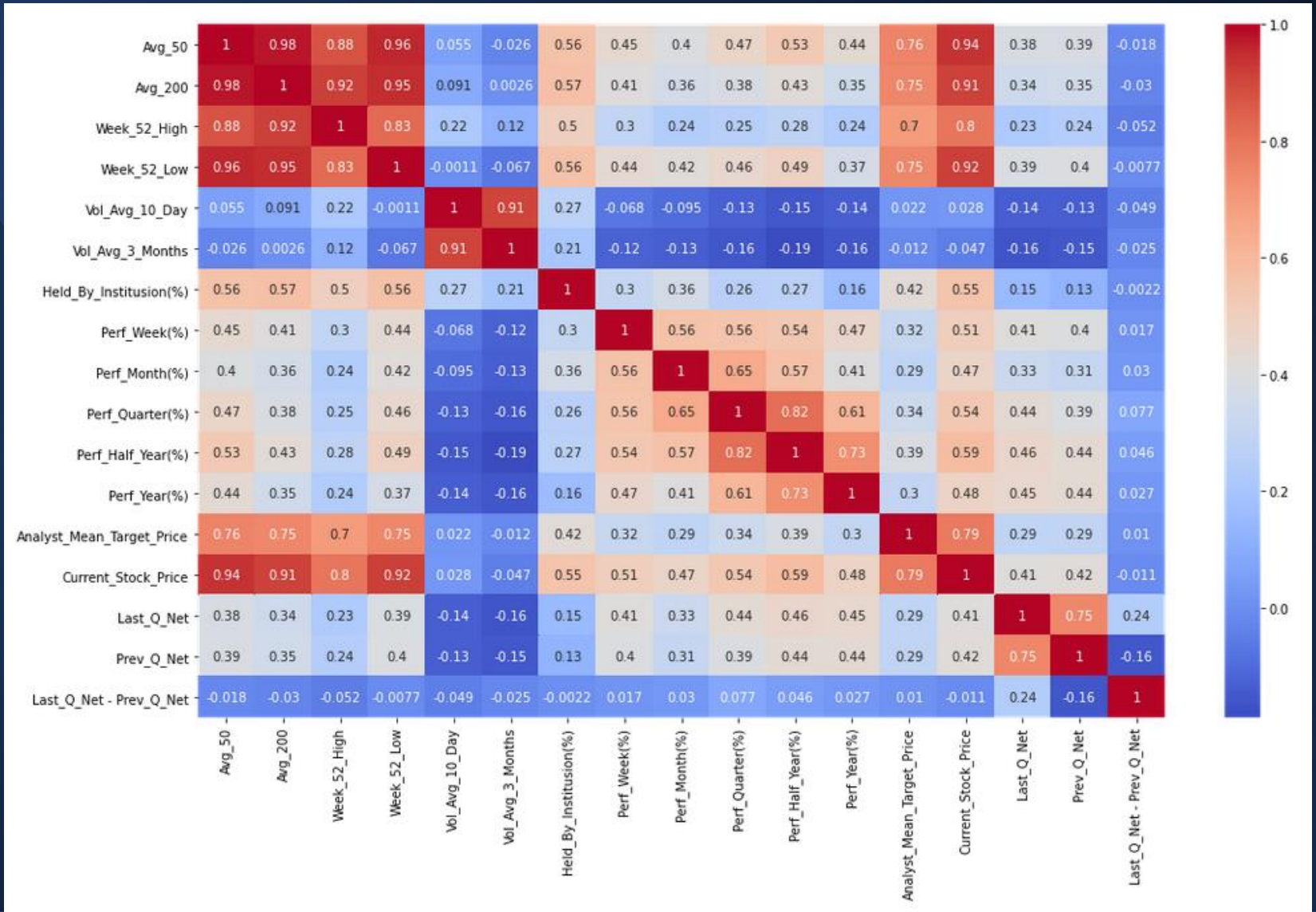
```
# remove rows with nan in all cols
df.dropna(how = 'all', inplace = True)
```

- We have added a new column to the dataframe to indicate what are the changes in the stock price compared to the average over the last 200 days

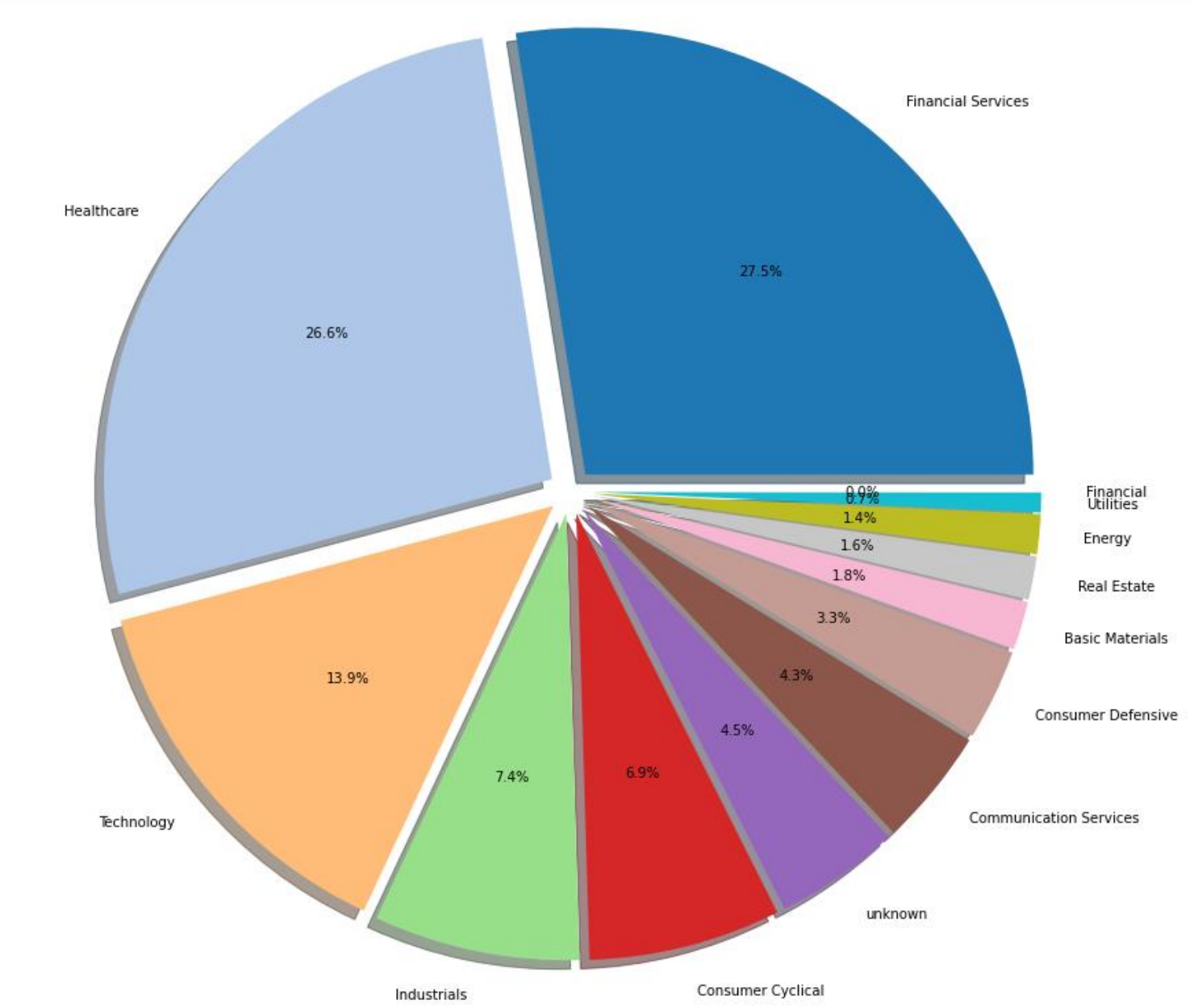
```
# We read in the internet that this is an important indicator if the price of the stock will grow or not!
df['Current_Stock_Price/Avg_50'] = df['Current_Stock_Price'] / df['Avg_50']
```


Visualizations

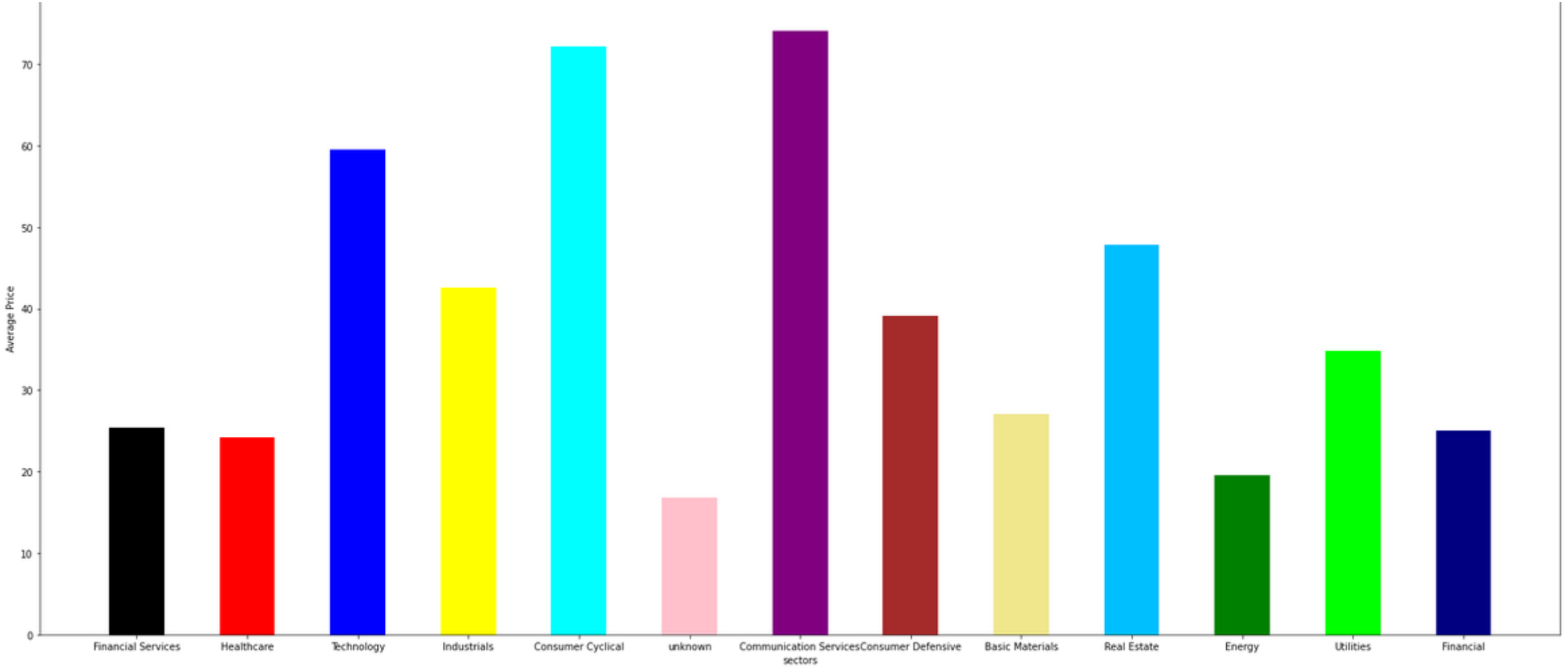
The general correlation between all our columns and rows in data frame.



Dividing our pie plots by sectors

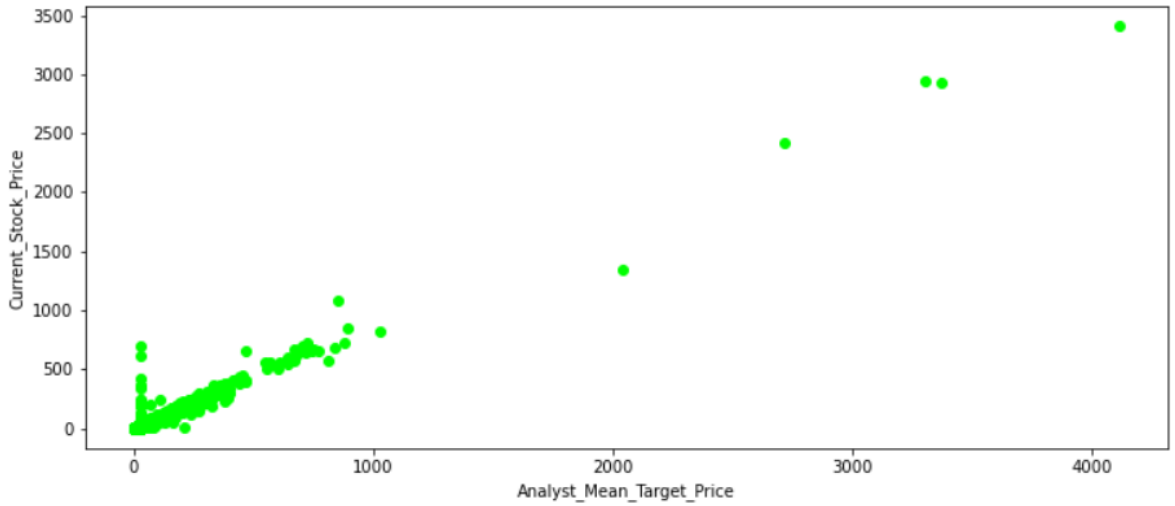


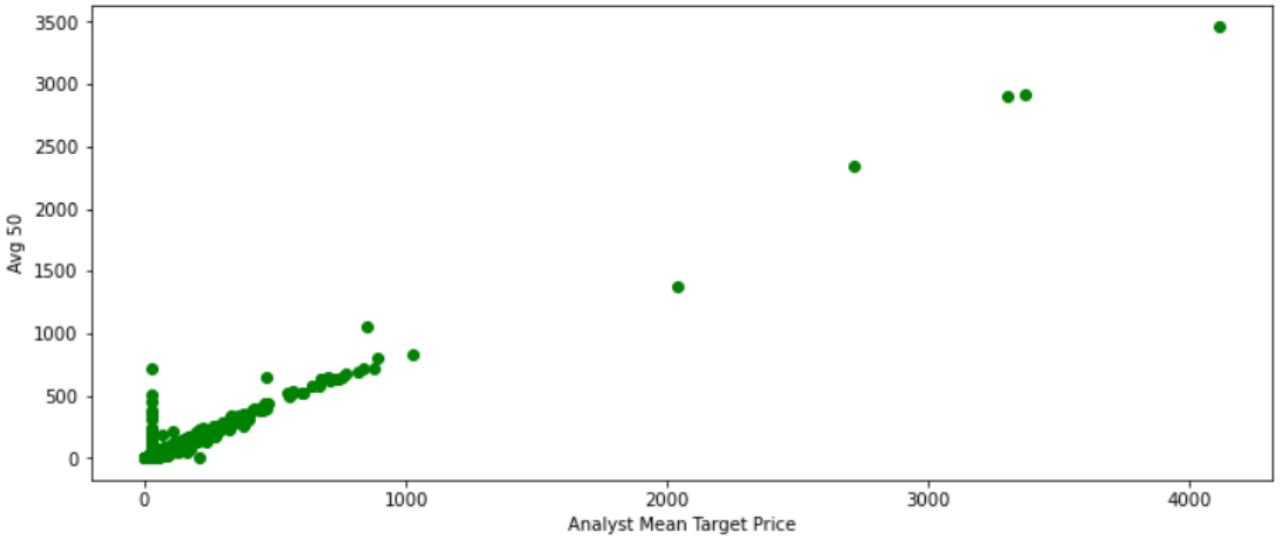
Visualizing the increase the average of all prices in each sector



We can see have strong correlation between the 'Analyst mean target price' and 'Current stock price'

As the analyst mean target price increases, the price of the stock increases as well



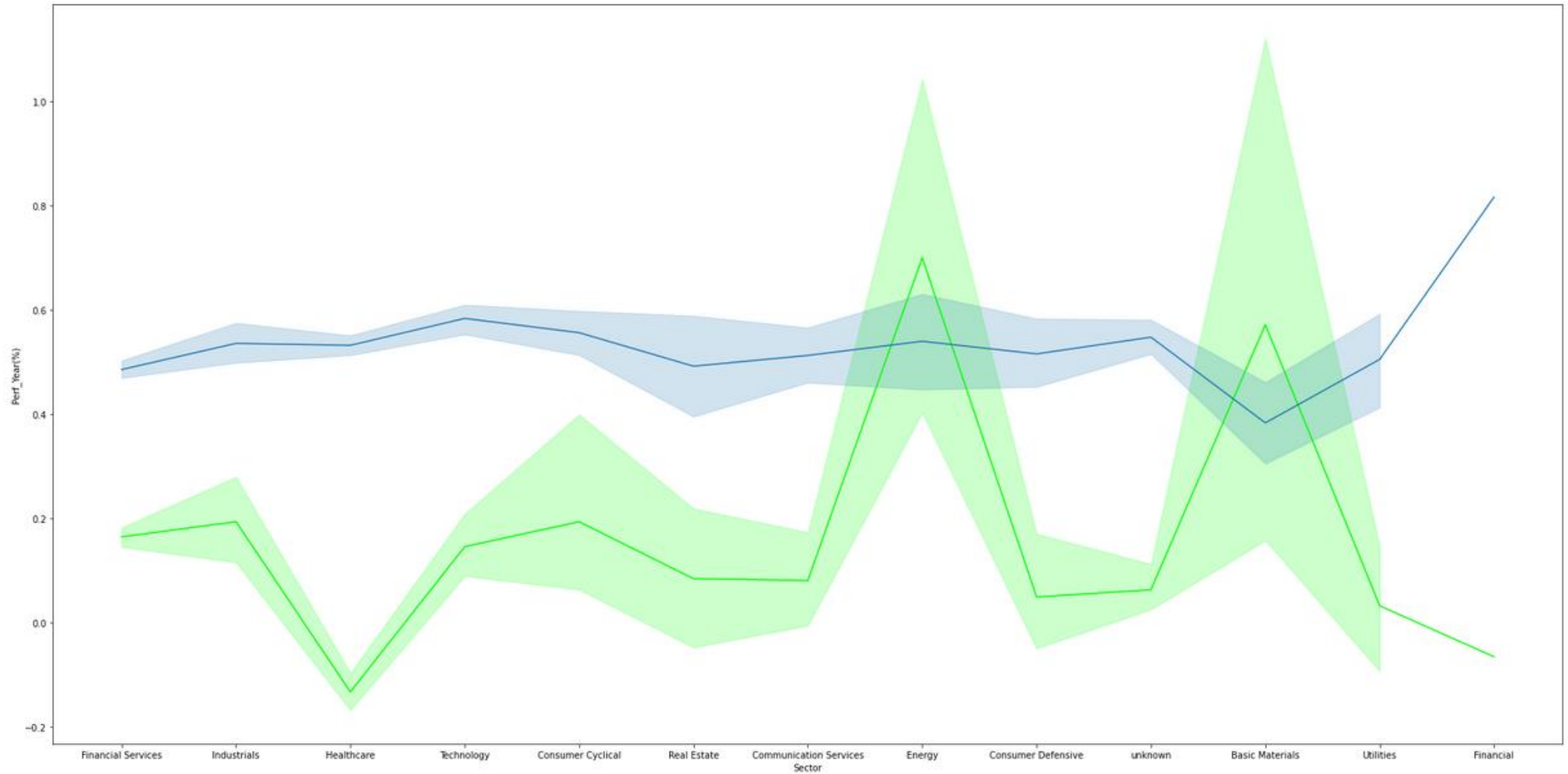


Explanation:

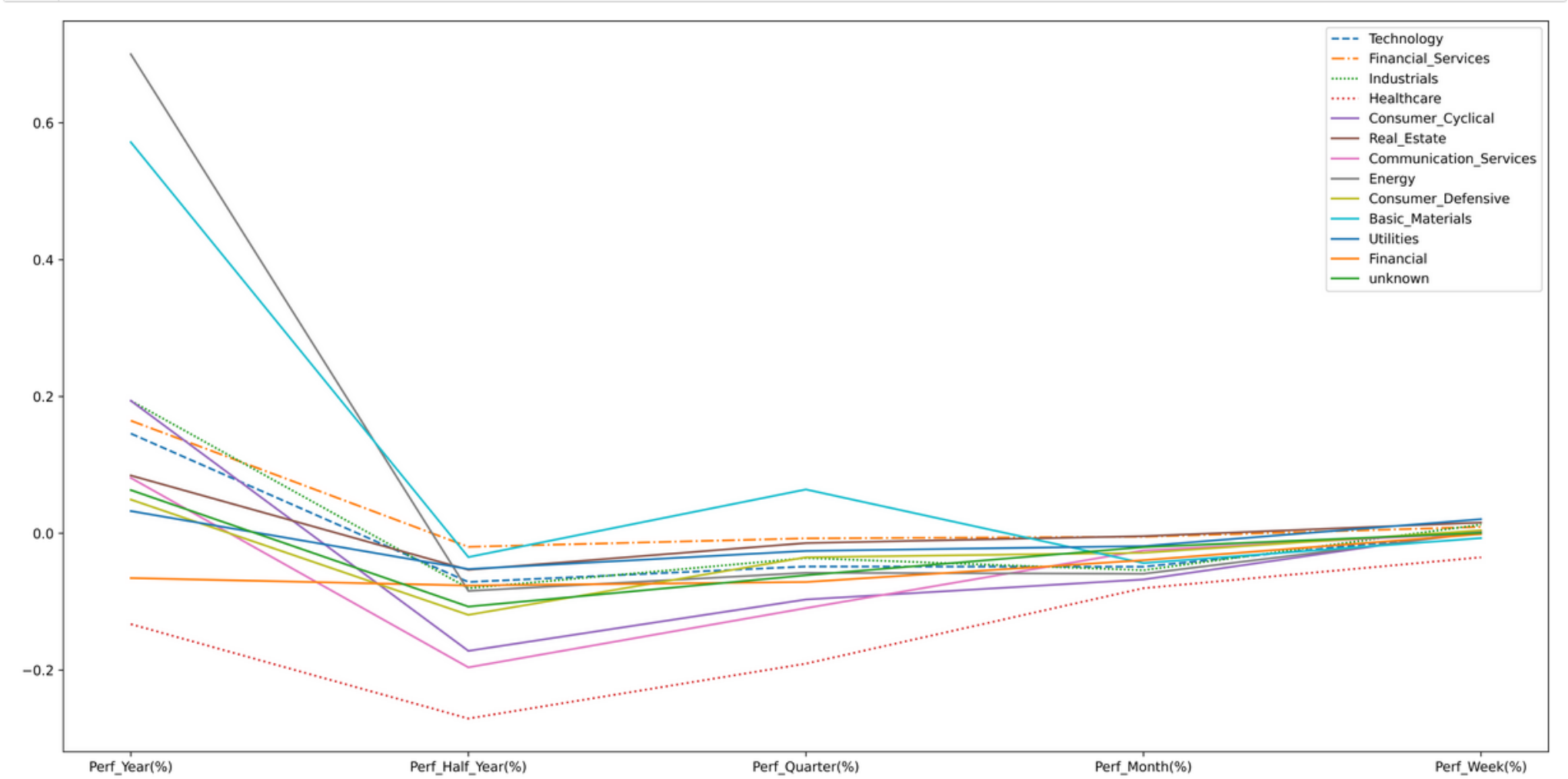
As the recommendation of the stock is high, more people would be interested to buy this stock. in conclusion, from this reason the average of the stock is getting higher.

We wanted to check if there is a connection between the percentage of 'Held_By Institutions' and the changes of the stock for the last year.

it is easy to see that there is no connection between two



From this plot graph it is visible that all the sectors have dramatically decreased due to the corona virus new wave in the last year.



Important to mention: that there are events beyond our control that can cause drastic changes in the entire space of the stock market


Machine Learning

Our label is "Predict_Inc_Dec" column, as our research question refers to whether the trend of the price of the stock will increase or decrease.

- We chose to apply KNN and Decision Tree - Supervised Learning for classification problems – categoric label .

- Before the feature engineering, our label column contained more '0' than '1' and we normalized it. after the normalization the accuracy score is 76%

```
In [14]:
y_pred = knn(X_train, y_train, X_test, y_test)
print("%.6f" % precision_score(y_test, y_pred, average='macro'))
print("%.6f" % recall_score(y_test, y_pred, average='macro'))
print("%.6f" % f1_score(y_test, y_pred, average='macro'))
print("%.6f" % accuracy_score(y_test, y_pred))
confusion_matrix(y_test, y_pred)
```

```
0.720094
0.656682
0.670625
0.760487


array([[474,  46],
       [131,  88]], dtype=int64)
```

Steps:

- We added a new label by the name “Predict_Inc_Dec”. The column contains numbers ‘1’ or ‘0’. According to the formula we checked the updated price of the stock with a previous price, if the updated price Is higher than the previous price filling label with value 1, else with 0.

- For the model section we used two kind of machine learnings:

The first one is KNN with the score of 76% and the second one is **Decision Tree** with the score 70%

KNN Model

```
[14]:
y_pred = knn(X_train, y_train, X_test, y_test)
print("%.6f" % precision_score(y_test, y_pred, average='macro'))
print("%.6f" % recall_score(y_test, y_pred, average='macro'))
print("%.6f" % f1_score(y_test, y_pred, average='macro'))
print("%.6f" % accuracy_score(y_test, y_pred))
confusion_matrix(y_test, y_pred)
```

```
0.720094
0.656682
0.670625
0.760487


array([[474,  46],
       [131,  88]], dtype=int64)
```

Decision model

```
y_pred = decision_tree(X_train, y_train, X_test, y_test)
print("%.6f" % precision_score(y_test, y_pred,average='macro'))
print("%.6f" % recall_score(y_test, y_pred,average='macro'))
print("%.6f" % f1_score(y_test, y_pred,average='macro'))
print("%.6f" % accuracy_score(y_test, y_pred))
confusion_matrix(y_test, y_pred)
```

```
0.704283
0.702094
0.703158
0.753721


array([[431,  89],
       [ 93, 126]], dtype=int64)
```


Thank you ☺