Building the optimal Investment Portfolio
Big Data Analysis Midterm Project

Jashaina Thomas & Sarah Lu

April 7, 2020

Contents

1 Executive Summary

Anyone interested in Finance has come across literature related to Modern Portfolio Theory and the relationship between returns and risk. The following is an implementation based on data from the securities in the New York Stock Exchange dataset found on Kaggle.com. The quality of the data is not our main concern in this case. Based on the data, we predict stock returns and construct optimal portfolios based on desired levels of risk for return. The results that we will show are relevant to understand optimal portfolio theory and the Markowitz Model but the model we obtain shouldn't be taken to make any investment decisions yet.

2 Introduction

One assumption in investing is that a higher degree of risk means a higher potential return. Conversely, investors who take on a low degree of risk have a low potential return. According to Markowitz's theory, there is an optimal portfolio that could be designed with a perfect balance between risk and return. The optimal portfolio does not simply include securities with the highest potential returns or low-risk securities. The optimal portfolio aims to balance securities with the greatest potential returns with an acceptable degree of risk or securities with the lowest degree of risk for a given level of potential return. In this report we set out to construct the optimal investment portfolios based on an investors chosen level of risk.

In the next section we will take a through look at the data used in this work. After that we will share the inspiration and methodology which we used as guides for our implementation and analysis. We will then include a brief analysis and results ending with by a conclusion. MATlab was used to implement this analysis. The MATlab code used to implement models and references can be found in the appendix at the end of this report.

3 Data

The dataset we are working with is from kaggle.com; kaggle.com/dgawlik/nyse provided us with financial data for SP 500 companies. We used two of the four files available to us. The prices.csv file gives daily trading prices from 2010 to the end 2016, and date range is shorter for companies new on stock market. The prices-split-adjusted.csv file is similar to the prices.csv, except this one includes adjustments for stock splits. The securities.csv file gives general description of each company with division on sectors. The fundamentals.csv file provides popular fundamental indicators extracted from annual SEC 10K fill- ings to derive popular fundamental indicators. In this project, we are working primarily with the prices.csv and the securities.csv files.

3.1 Description

The price.csv file contains 851,246 trading data and the securities.csv file contains 505 data about company information. There are 7 variables used in our study, described as follows:

Name	Description
Date	Date and time of the trading information
Symbol	Unique stock symbol that identify each stock
Open	Last price anyone paid for a stock during the business day
Close	Price from the first transaction of a business day
Low	Lowest traded price for a stock during the business day
High	Highest traded price for a stock during the business day
Volume	Total quantity of shares traded for a stock
GICS Sector	Industry in which the company operates.

3.2 Data Exploration

Among the 505 companies, we randomly chose one stock RHT (Reliq Health Technologies Inc) to take a look at to begin our analysis. We plotted the day to day high-low and open-close graphs below in figure 1 to simply visualize and analyze the price changes over time for this stock.

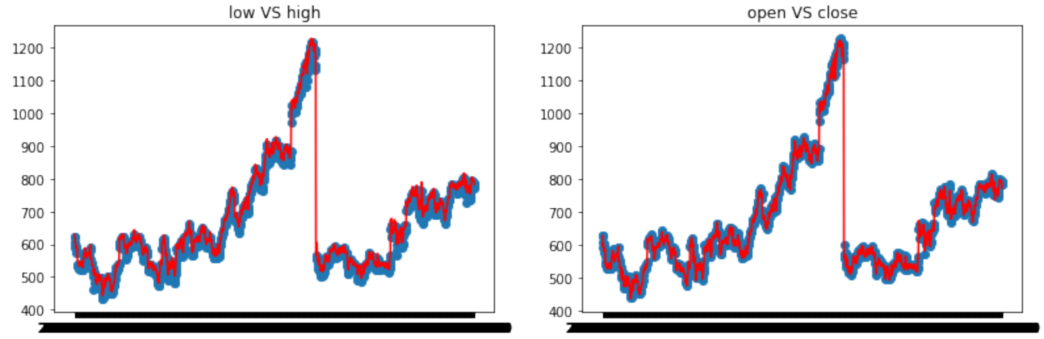


Figure 1: Reliq Health Technologies Inc.

Because we are performing the analysis from the perspective of investors, we decided to use the closing price in our study. We plotted the distribution of observations using the seaborn.distplot function in figure 2 below.

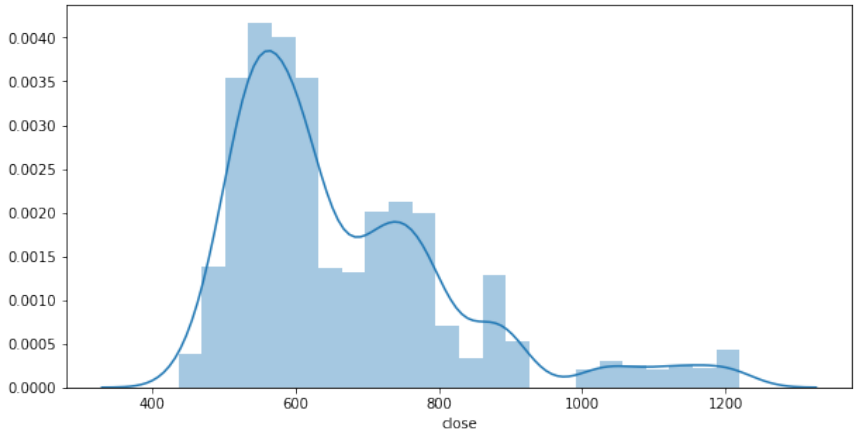


Figure 2: Reliq Health Technologies Inc. Closing Price

Now let us take a look at all of our stock data by industry in figure 3. Each stock is plotted below in a "sector" generally representative of the overall sectors of the economy. Here, we can observe that the Consumer Discretionary sector is the most prevalent in our data while Telecommunication Services makes up the smallest amount of our data. In the methodology section we will take a look at the correlation matrix of these Sectors based on our data. The higher the correlation of certain industries, the more likely it is that the stocks within those sectors will move in the same direction. This concept is essential to building an optimal investment portfolio as our goal is to to build a well-diversified portfolio.

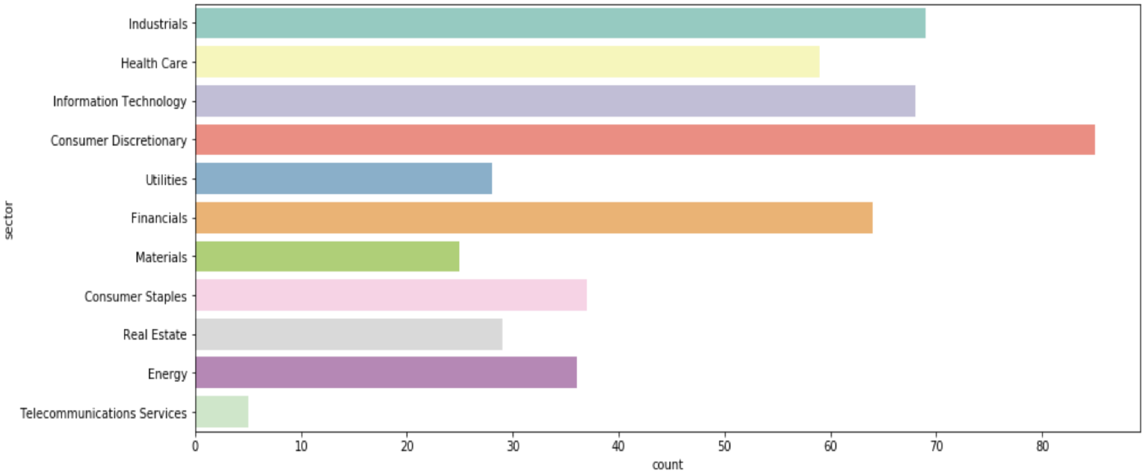


Figure 3: Stock Frequency by Sector

4 Methodology

4.1 Prediction Using Local Linear Regression Model

The first task we set out to do is predict stock return. We did this by running a local linear regression on every five days of data to develop a simple model to fit our non-linear data. Each stocks return is predicted based on its daily opening, closing, high, and low prices along with its volume and sector. Linear regression attempts to model the linear relationship between these variables and stock price by fitting a linear equation to the data set. We used the built-in linear regression model from the scikit-learn to predict the 5-day ahead stock price, (i.e. using price on 01/01/2013 to predict price on 01/06/2013). After applying train-test split for cross validation, there are 1,405 training data points and 352 testing data points, with dates ranging from the year 2010 to 2016.

4.2 Optimal Portfolio Theory and Markowitz Efficiency Frontier

Modern Portfolio Theory and the relationship between returns and risk was best explained by Harry Markowitz back in 1952. We model our assets by their expected return, $E[R]$ and their risk, which is expressed by their standard deviation, σ. Our investment decisions are expressed by investing 100% of our wealth in assets, where each particular investment represents a proportion of our total wealth. That is, we invest w_i in $asset_i$ for every i, and we always maintain $\sum_{i=1}^{n} w_i = 1$ as a condition of being fully invested.

A portfolio is constructed by investing in different assets. We can express the return and risk of our portfolio by the following equations:

1) $E[R_p] = \sum_{i=1}^{n} w_i E[R_i]$
2) $\sigma^2(R_p) = \sum_{i=1}^{n} w_i^2 \sigma^2(R_i) + \sum_i \sum_{j \neq i} w_i w_j \sigma(R_i) \sigma(R_j) \rho_{ij}$

Returns are dependent on the investment combinations that make up the portfolio. An efficient portfolio is one that maximizes return for a given level of risk. The task at hand is to select the weights (w_i) adequately to accomplish this. Our goal is to construct the efficient frontier given our data. The efficient frontier is the set of optimal portfolios that offer the highest expected return for a defined level of risk or the lowest risk for a given level of expected return. It rates portfolios (investments) on a scale of return (y-axis) versus risk (x-axis)

Below we have plotted the correlation matrix of our Sectors. The higher the correlation, the more likely that the stocks are moving in the same direction. To put it simply, if two stocks are highly correlated, they are likely to increase or decrease in price together. When building a diversified portfolio, investors seek negatively correlated stocks. Doing so reduces the risk of catastrophic losses in the portfolio and helps the investor sleep better at night. Assume the portfolio consists of two stocks and they are negatively correlated. This implies that when the price of one performs worse than usual, the other will likely do better than usual. However, risk takers would love to seek for positively correlated stocks for higher expected return, and of course, higher risk.

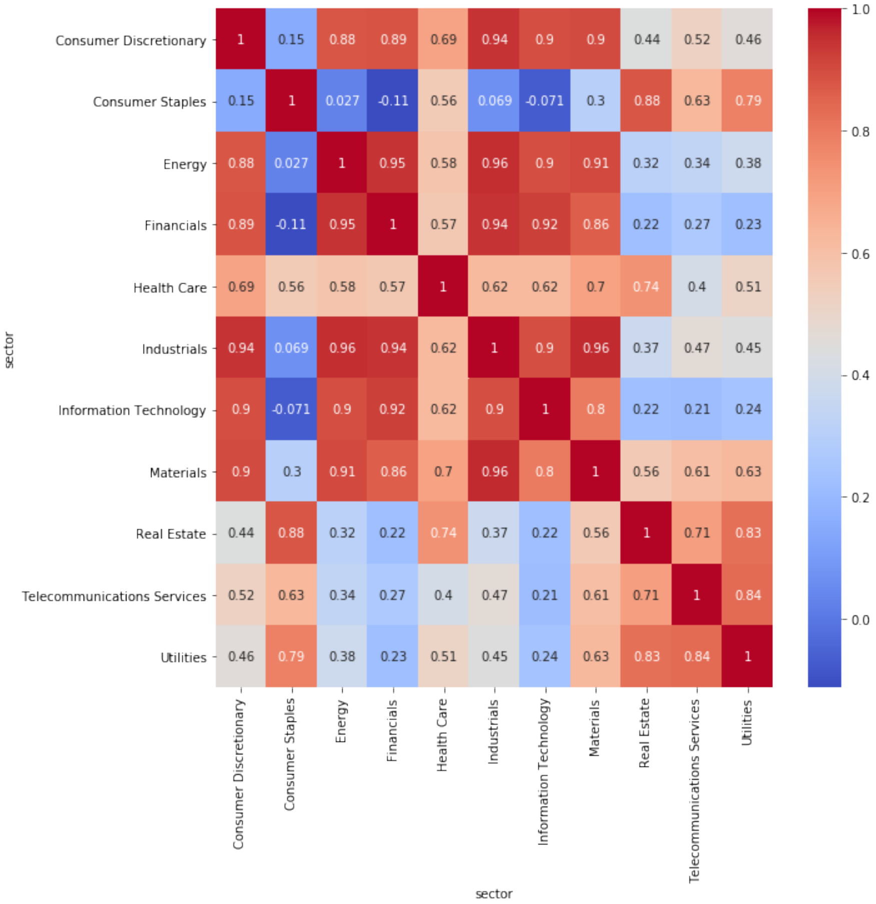


Figure 4: Correlation of Sectors

Sharpe ratio is often used to describe how good our portfolio is. The higher the sharpe ratio, the better the portfolio. A sharpe ratio more than 1 is acceptable to investors. As a matter of fact, we will only choose the sectors that have sharpe ratio more than 1. By using correlation matrix and assigning random weights, we can get portfolio variances and sharpe ratios. The general formula for the variance of a portfolio is:

$$\sigma_p^2 = \sum_{i=1}^{n} \sum_{j=1}^{n} w_i w_j \operatorname{Cov}(r_i, r_j)$$

To calculate the variance, we take the covariance matrix multiply with the weighted matrix and the transposed of the weighted matrix:

$$\sigma_p^2 = \begin{pmatrix} w_1 & w_2 & w_3 & w_4 \end{pmatrix} \begin{pmatrix} \sigma_{11} & \sigma_{12} & \sigma_{13} & \sigma_{14} \\ \sigma_{21} & \sigma_{22} & \sigma_{23} & \sigma_{24} \\ \sigma_{31} & \sigma_{32} & \sigma_{33} & \sigma_{34} \\ \sigma_{41} & \sigma_{42} & \sigma_{43} & \sigma_{44} \end{pmatrix} \begin{pmatrix} w_1 \\ w_2 \\ w_3 \\ w_4 \end{pmatrix}$$

The Sharpe ratio is calculated by subtracting the risk-free rate from the return of the portfolio and dividing that result by the standard deviation of the portfolio's excess return:

$$\textit{Sharpe Ratio} = \frac{R_p - R_f}{\sigma_p}$$

where:

R_p = return of portfolio

R_f = risk-free rate

σ_p = standard deviation of the portfolio's excess return

5 Analysis & Results

5.1 Prediction Model Evaluation Metrics

The following indicators evaluate the performance of our local linear regression prediction model:

R^2 score	0.9763529901583069
Mean Absolute Error	1.6923804307829216
Mean Squared Error	5.1397782459967924
Root Mean Squared Error	2.267107903474555

An R-squared score of 0.976 suggests that approximately 97.6% of the variation in the dependent variable, which is closing stock price, is explained by the variables we included in our regression model. These indicators suggest that our model gives an accurate prediction of the 5-day forward stock prices we set out to predict. This is very good but is also sometimes sign of a "too good" or overly fitted model.

5.2 Efficiency Froniter

We stated before, that the efficient frontier is the set of optimal portfolios that offer the highest expected return for a defined level of risk or the lowest risk for a given level of expected return. Portfolios that lie below the efficient frontier are sub-optimal because they do not provide enough return for the level of risk. Portfolios that cluster to the right of the efficient frontier are sub-optimal because they have a higher level of risk for the defined rate of return. We constructed the efficiency frontier based on our data in figure 5 shown on the next page. After running 2000 simulations, in MATlab we finally plot the results. The curved line that is darkest in color makes up our efficiency frontier:

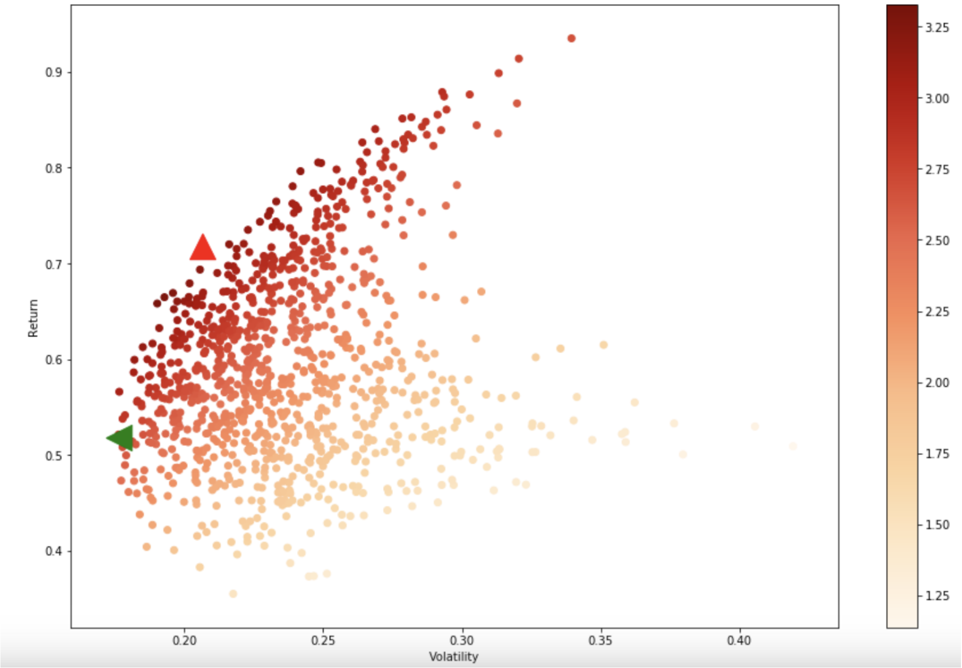


Figure 5: Efficiency Frontier

We see the green pointer near optimal portfolios for risk averse investors while risk takers would not mind investing in portfolios near or above the red pointer with slightly more risk for the possibility of greater return.

Based on the efficiency frontier here, the highest Sharpe portfolio has about 0.35 volatility, with a 3.327 Sharpe. We can also see observe that the overall safest portfolio is 0.176 risk with a Sharpe ratio of around 2.

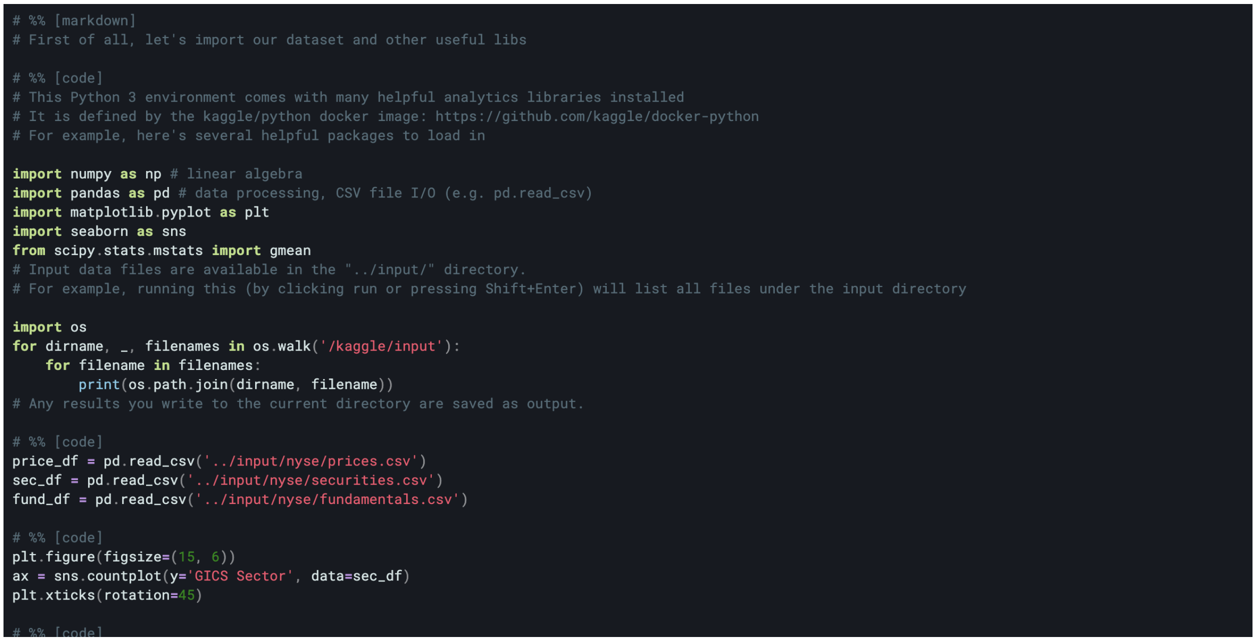
6 Conclusion

Modern Portfolio Theory states that adding assets to a diversified portfolio which have low correlations can decrease portfolio risk without sacrificing return which we have shown using the Markowitz Efficient Frontier. The optimal portfolio refers to the one portfolio on the Efficient Frontier with the highest return-to-risk combination given the specific investor's tolerance for risk. A shortcoming of Markowitz model is that it is based only on historical data. One can essentially reduce investment risk through optimal portfolio diversification using a quantitative methods which take into account time, historical data, current industry trends, and other relevant factors.

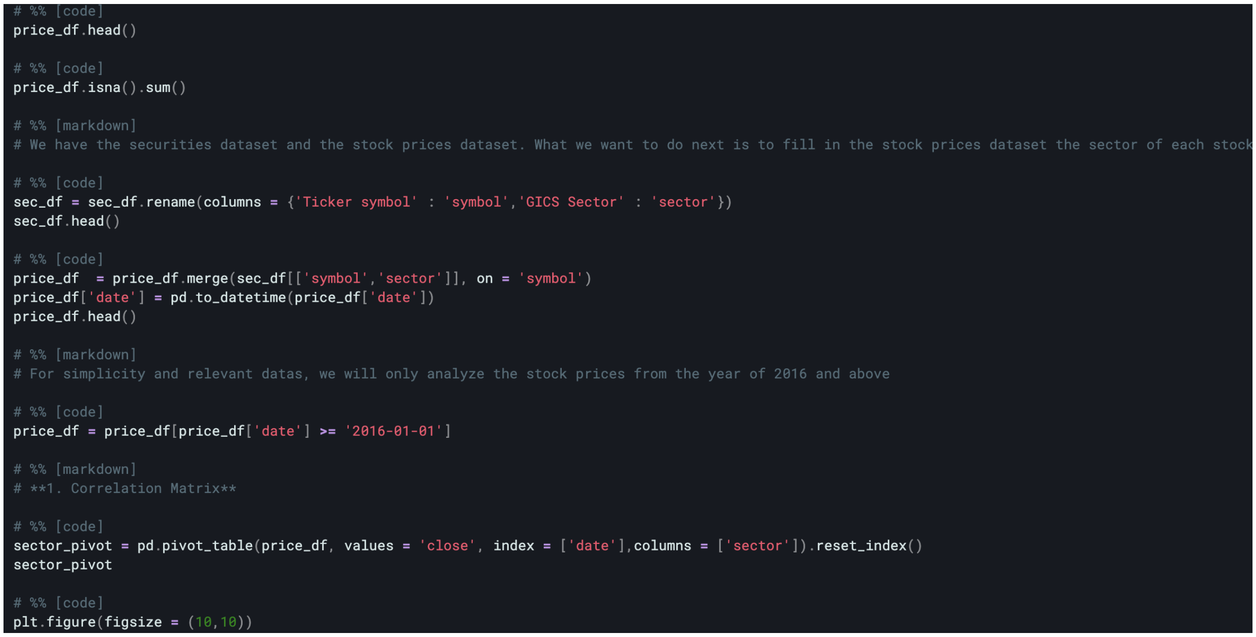
We are aware that our model has some limitations based on this and other shortcomings of our process. First, we are only predicting prices for one stock at a time, so the performance of our model may vary when applied to different or multiple stocks. In future studies, we may want to run the model on all stocks in our data set and find the stock that can be best predicted to invest in. Second, we used the 5-day historical prices to predict future prices. Performance of our model could be increased if we change or extend the time window. Third, as mentioned before, in this project we used historical information (stock price) as our only independent variable, but we know stock prices are influenced by a lot of other factors, such as the overall economy, industry trends, or financial stress. Therefore, in future studies, we would like to include more relevant independent variables, such as macroeconomic indicators, industry volatility, and financial ratios. This would improve our process overall by including more than historical data in the prediction stage as all else is ignored in calculating the efficient frontier. We could also improve this process by increasing the quality of our data. This can be done by accessing more recent or automatically updated data. Something to consider further is other prediction models for comparison. Implementing some form of neural network could lead to a more flexible and accurate prediction model. Considering and including further methods of analysis could also be a major improvement. Other methods, such as sentiment analysis, could also be included for better prediction accuracy. Including sentiment analysis based on news current headlines or social network posts could lead to identifying and using new industry indicators, especially at a time of crisis like now (COVID-19).

7 Appendix

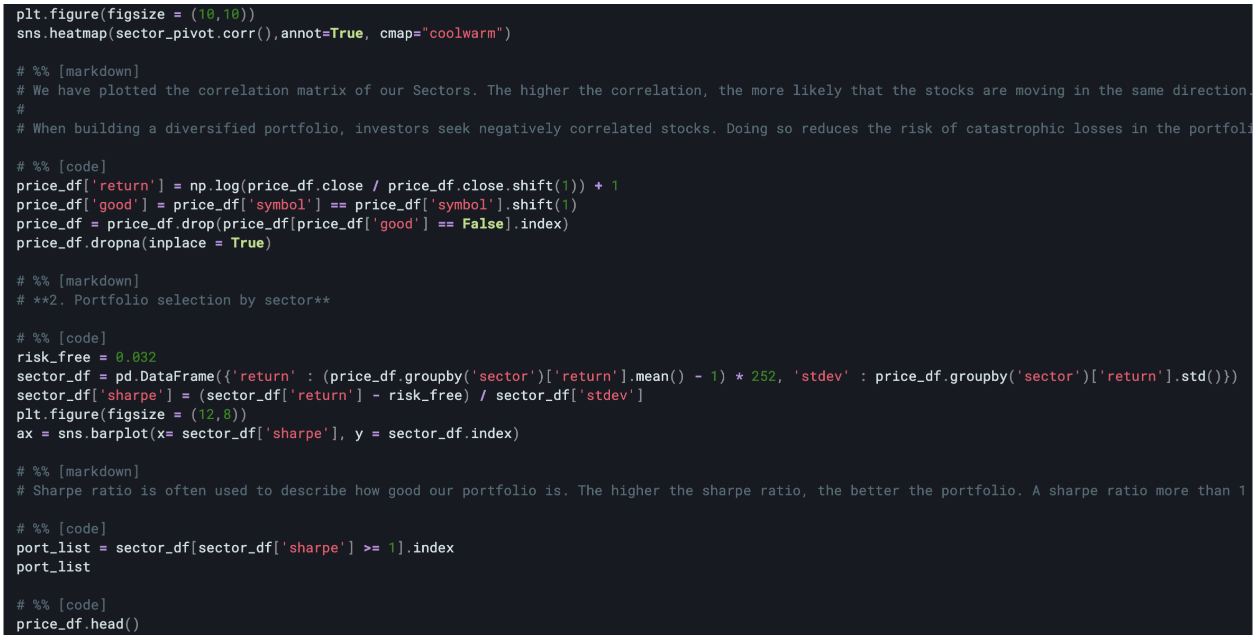
7.1 MATlab Code



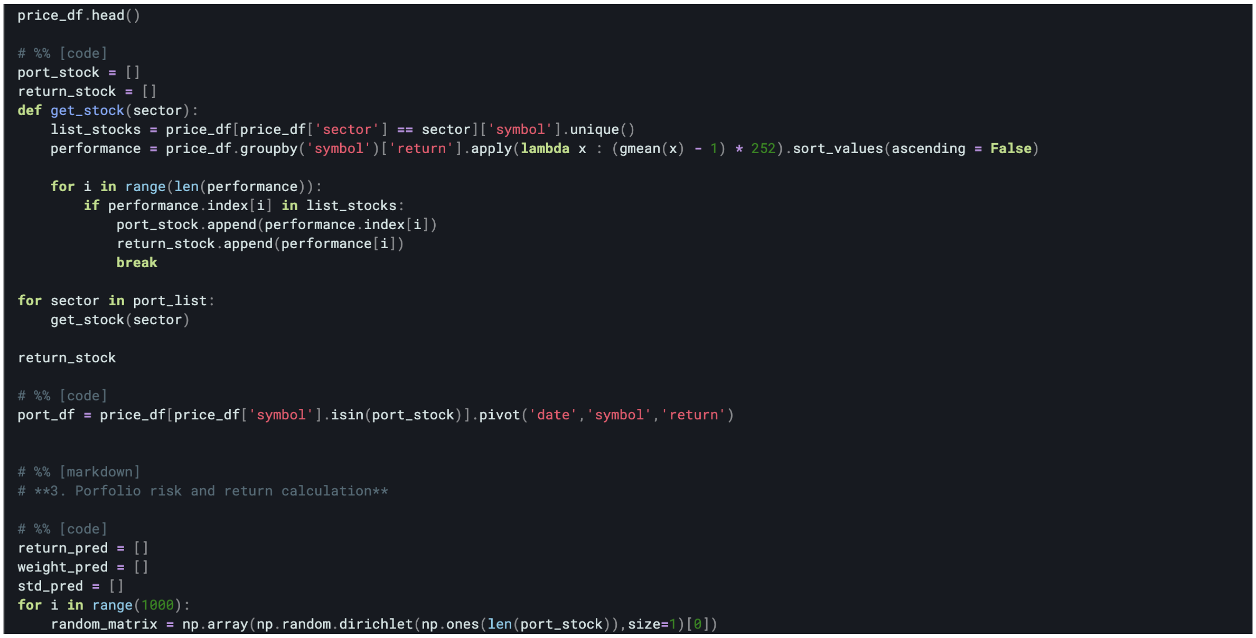
```
# %% [markdown]
# First of all, let's import our dataset and other useful libs

# %% [code]
# This Python 3 environment comes with many helpful analytics libraries installed
# It is defined by the kaggle/python docker image: https://github.com/kaggle/docker-python
# For example, here's several helpful packages to load in

import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
import matplotlib.pyplot as plt
import seaborn as sns
from scipy.stats.mstats import gmean
# Input data files are available in the "../input/" directory.
# For example, running this (by clicking run or pressing Shift+Enter) will list all files under the input directory

import os
for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))
# Any results you write to the current directory are saved as output.

# %% [code]
price_df = pd.read_csv('../input/nyse/prices.csv')
sec_df = pd.read_csv('../input/nyse/securities.csv')
fund_df = pd.read_csv('../input/nyse/fundamentals.csv')

# %% [code]
plt.figure(figsize=(15, 6))
ax = sns.countplot(y='GICS Sector', data=sec_df)
plt.xticks(rotation=45)

# %% [code]
```



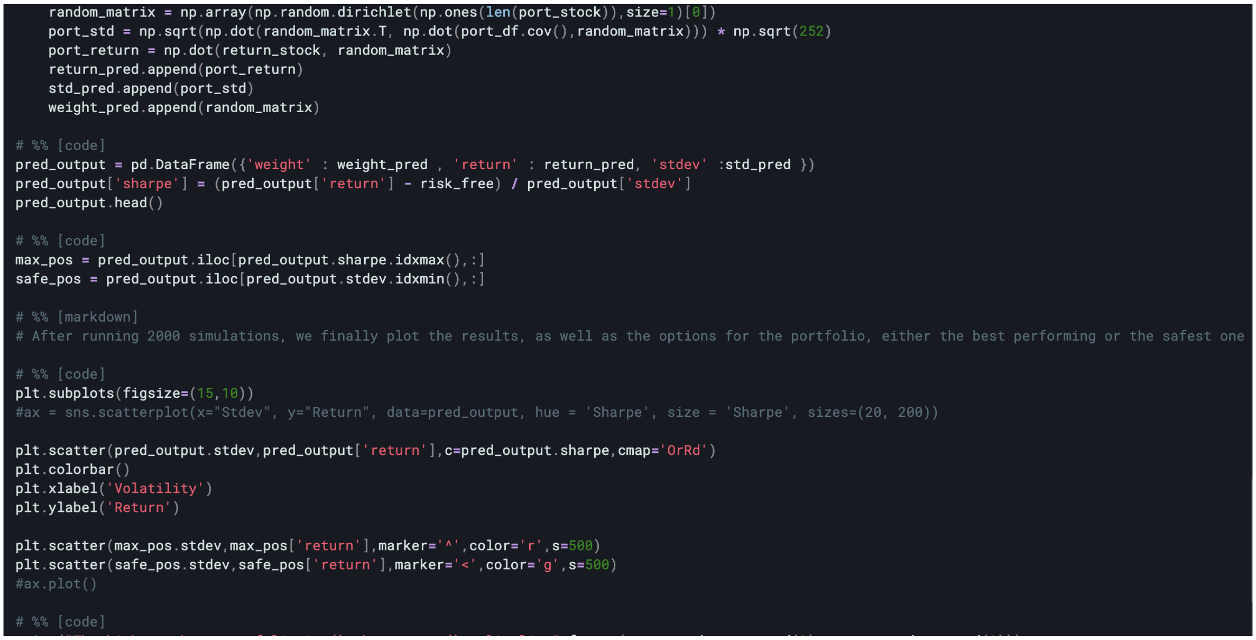
```
# %% [code]
price_df.head()

# %% [code]
price_df.isna().sum()

# %% [markdown]
# We have the securities dataset and the stock prices dataset. What we want to do next is to fill in the stock prices dataset the sector of each stock

# %% [code]
sec_df = sec_df.rename(columns = {'Ticker symbol' : 'symbol','GICS Sector' : 'sector'})
sec_df.head()

# %% [code]
price_df  = price_df.merge(sec_df[['symbol','sector']], on = 'symbol')
price_df['date'] = pd.to_datetime(price_df['date'])
price_df.head()

# %% [markdown]
# For simplicity and relevant datas, we will only analyze the stock prices from the year of 2016 and above

# %% [code]
price_df = price_df[price_df['date'] >= '2016-01-01']

# %% [markdown]
# **1. Correlation Matrix**

# %% [code]
sector_pivot = pd.pivot_table(price_df, values = 'close', index = ['date'],columns = ['sector']).reset_index()
sector_pivot

# %% [code]
plt.figure(figsize = (10,10))
```



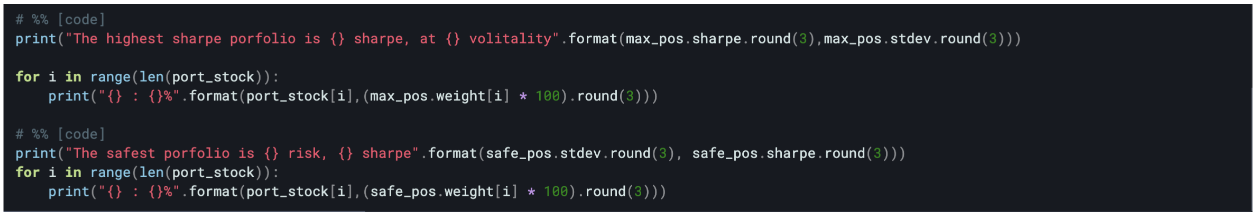
```
plt.figure(figsize = (10,10))
sns.heatmap(sector_pivot.corr(),annot=True, cmap="coolwarm")

# %% [markdown]
# We have plotted the correlation matrix of our Sectors. The higher the correlation, the more likely that the stocks are moving in the same direction.
#
# When building a diversified portfolio, investors seek negatively correlated stocks. Doing so reduces the risk of catastrophic losses in the portfoli

# %% [code]
price_df['return'] = np.log(price_df.close / price_df.close.shift(1)) + 1
price_df['good'] = price_df['symbol'] == price_df['symbol'].shift(1)
price_df = price_df.drop(price_df[price_df['good'] == False].index)
price_df.dropna(inplace = True)

# %% [markdown]
# **2. Portfolio selection by sector**

# %% [code]
risk_free = 0.032
sector_df = pd.DataFrame({'return' : (price_df.groupby('sector')['return'].mean() - 1) * 252, 'stdev' : price_df.groupby('sector')['return'].std()})
sector_df['sharpe'] = (sector_df['return'] - risk_free) / sector_df['stdev']
plt.figure(figsize = (12,8))
ax = sns.barplot(x= sector_df['sharpe'], y = sector_df.index)

# %% [markdown]
# Sharpe ratio is often used to describe how good our portfolio is. The higher the sharpe ratio, the better the portfolio. A sharpe ratio more than 1

# %% [code]
port_list = sector_df[sector_df['sharpe'] >= 1].index
port_list

# %% [code]
price_df.head()
```



```
price_df.head()

# %% [code]
port_stock = []
return_stock = []
def get_stock(sector):
    list_stocks = price_df[price_df['sector'] == sector]['symbol'].unique()
    performance = price_df.groupby('symbol')['return'].apply(lambda x : (gmean(x) - 1) * 252).sort_values(ascending = False)

    for i in range(len(performance)):
        if performance.index[i] in list_stocks:
            port_stock.append(performance.index[i])
            return_stock.append(performance[i])
            break

for sector in port_list:
    get_stock(sector)

return_stock

# %% [code]
port_df = price_df[price_df['symbol'].isin(port_stock)].pivot('date','symbol','return')


# %% [markdown]
# **3. Porfolio risk and return calculation**

# %% [code]
return_pred = []
weight_pred = []
std_pred = []
for i in range(1000):
    random_matrix = np.array(np.random.dirichlet(np.ones(len(port_stock)),size=1)[0])
```



```
    random_matrix = np.array(np.random.dirichlet(np.ones(len(port_stock)),size=1)[0])
    port_std = np.sqrt(np.dot(random_matrix.T, np.dot(port_df.cov(),random_matrix))) * np.sqrt(252)
    port_return = np.dot(return_stock, random_matrix)
    return_pred.append(port_return)
    std_pred.append(port_std)
    weight_pred.append(random_matrix)

# %% [code]
pred_output = pd.DataFrame({'weight' : weight_pred , 'return' : return_pred, 'stdev' :std_pred })
pred_output['sharpe'] = (pred_output['return'] - risk_free) / pred_output['stdev']
pred_output.head()

# %% [code]
max_pos = pred_output.iloc[pred_output.sharpe.idxmax(),:]
safe_pos = pred_output.iloc[pred_output.stdev.idxmin(),:]

# %% [markdown]
# After running 2000 simulations, we finally plot the results, as well as the options for the portfolio, either the best performing or the safest one

# %% [code]
plt.subplots(figsize=(15,10))
#ax = sns.scatterplot(x="Stdev", y="Return", data=pred_output, hue = 'Sharpe', size = 'Sharpe', sizes=(20, 200))

plt.scatter(pred_output.stdev,pred_output['return'],c=pred_output.sharpe,cmap='OrRd')
plt.colorbar()
plt.xlabel('Volatility')
plt.ylabel('Return')

plt.scatter(max_pos.stdev,max_pos['return'],marker='^',color='r',s=500)
plt.scatter(safe_pos.stdev,safe_pos['return'],marker='<',color='g',s=500)
#ax.plot()

# %% [code]
```



```
# %% [code]
print("The highest sharpe porfolio is {} sharpe, at {} volitality".format(max_pos.sharpe.round(3),max_pos.stdev.round(3)))

for i in range(len(port_stock)):
    print("{} : {}%".format(port_stock[i],(max_pos.weight[i] * 100).round(3)))

# %% [code]
print("The safest porfolio is {} risk, {} sharpe".format(safe_pos.stdev.round(3), safe_pos.sharpe.round(3)))
for i in range(len(port_stock)):
    print("{} : {}%".format(port_stock[i],(safe_pos.weight[i] * 100).round(3)))
```

7.2 References

https://www.kaggle.com/dgawlik/nyse

https://onlinelibrary.wiley.com/doi/full/10.1111/j.1540-6261.1952.tb01525.x