Monte Carlo Approach for Effective Stock Optimization Based on Modern Portfolio Theory

Md Muhtasim Billah

Abstract

"Buy low, sell high" - this idea has always been at the core of stock market finance. However, the challenges for implementing this in the real world are plenty. Over the decades, a myriad of concepts and theories, driven by economics and statistics, were proposed and they were aided by modern computational tools to make predictions for the future. However, methodologies for getting a complete grasp of such ever-illusive events in terms of forecasting, although not perfected, has improved. The modern portfolio theory (MPT) is a cencept that constructs diversified portfolios to maximizes the inverstors' returns without incurring unacceptable levels of risk. On the other hand, Monte Carlo (MC) methods have gained popularity in many scientific fields and they have extended their reach to solve financial problems as well. In this work, a methodology is presented where MC simulation, coupled with the MPT, can estimate the most effective way for the diversification of stock portfolio that would essentially maximize the return on investments.

1. Introduction

Monte Carlo (MC) simulation is primarily based on repeated random sampling. The underlying concept of MC simulation is to use randomness to solve problems that might be deterministic in principle. Monte Carlo simulation is one of the most popular techniques to draw inferences about a population without knowing the true underlying distribution. This sampling technique becomes handy especially when one doesn't have the luxury to repeatedly sample from the original population. Applications of Monte Carlo simulation range from solving problems in theoretical physics to predicting trends in financial investments.

The modern portfolio theory (MPT) is a practical method for selecting investments in order to maximize the investors' overall returns within an acceptable level of risk. American economist Harry Markowitz pioneered this theory and was later awarded a Nobel Prize in 1952 for his groundbreaking idea. [1] The key component of the MPT is diversification. Most investments are either high risk and high return or low risk and low return. MPT argues that investors could achieve their best results by choosing an optimal mix of the two based on an assessment of their individual tolerance to risk. The MPT also assumes that investors are risk-averse, meaning they prefer a less risky portfolio to a riskier one for a given level of return. As a practical matter, risk aversion implies that most people should invest in multiple asset classes. This idea is also known as "not putting all of your eggs in one basket" as popularized by American investor and philanthropist Warren Buffet.

The modern portfolio theory also implies that any given investment's risk and return characteristics should not be viewed alone but should be evaluated by how it affects the overall portfolio's risk and return. Based on statistical measures such as variance and correlation, a single investment's performance is expected to be less important than how it impacts the entire portfolio. This idea can be useful for investors trying to construct efficient and diversified portfolios. As an alternative, starting with a desired level of expected return, the investor can construct a portfolio with the lowest possible risk that is capable of producing that return. Moreover, investors who are more concerned with downside risk might prefer the post-modern portfolio theory (PMPT) to MPT.

Thw primary objective of the current work is to demonstrate a systematic way of utilizing the MPT and MC simulation together on data drawn from actual stock market. For this work, 10 of the most popular companies enlisted in Fortune 500 were randomly chosen which represent different industries such as technology, e-commerce, enetertainment and retail etc. The goal is to build a diversified portfolio based on these companies so that an investor can maximize the return on his investments.

2. Datasets

For building an example investor portfolio, 10 companies (assets) are considered for this work which are (alphabetically) Apple, Amazon, Facebook, Google, Intel, Microsoft, Netflix, Oracle, Tesla and Walmart. The stock price information was collected from the Yahoo Finance [2] which is an open data source. The stock ticker symbols for these companies are provided in Table 1.

Asset	Apple	Amazon	Facebook	Google	Intel	Microsoft	Netflix	Oracle	Tesla	Walmart
Symbol	AAPL	AMZN	FB	GOOG	INTC	MSFT	NFLX	ORCL	TSLA	WMT

Table 1: Companies (assets) and their stock ticker symbols (in alphabetic order).

A dataset was collected for each of the aforementioned company which has the features `Date`, `Open`, `High`, `Low`, `Close`, `Adj Close` and `Volume`. These feature represent the trading date and time, opening price, highest price, lowest price, closing price, adjacent closing price and the volume of the stocks respectively. The datasets contain information about the stocks over the period of an year which is essentially 250 trading days. For the purpose of MC simulation, only the feature `Close` is considered sufficient which indicates to the price of the stock at the end of that business day. Thus, the 10 datasets were eventually reduced to a single dataset containing the closing price for these stocks for each trading days throughout the year. Thus the final dataset has 11 columns (including the indices) and 251 rows (including the stock ticker symbols).

3. Materials and Methods

3.1 Modern Portfolio Theory (MPT):

The MPT quantifies the benefits of diversification or in other words spreading out the investments in multiple stocks that have different risk levels associated with them. The principal task for implementing MPT concerns with calculating the Sharpe Ratio which works as the evaluation metric or the criterion for this method. It can theoretically obtain any value from positive to negative infinity. Mathematically, Sharpe Ratio, S_a is given by the following formula.

$$S_a = \frac{E[R_a - R_b]}{\sigma_a}$$

where,

$$\begin{aligned} E &= \text{Expectation} \\ R_a &= \text{Asset return} \\ R_b &= \text{Risk-free return} \\ \sigma_a &= \text{Standard deviation of the asset's expected return} \end{aligned}$$

The standard deviation of the asset's expected return is also termed as volatility or the portfolio risk. Essentially, the Sharpe ratio can be reduced to the annual portfolio return, R_p over the annual portfolio volatility, σ_p which are expressed as below.

$$R_p = \sum_{i=1}^{N} w_i r_i$$

$$\sigma_p^2 = \sum_{i=1}^{N} w_i^2 \sigma_i^2 + \sum_{i=1}^{N} \sum_{j=1}^{N} w_i w_j \sigma_{ij}^2$$

where, N is the total number of assets in the portfolio and r_i is the mean daily return. Also,

$$\sum_{i=1}^{N} w_i = 1 \qquad \text{and} \qquad 0 \leq w_i \leq 1$$

As observed from the equations, the expected return of the portfolio is calculated as a weighted sum of the returns of the individual assets. For example, if a portfolio contained four equally weighted assets with expected returns of 4%, 6%, 10%, and 14%, the portfolio's expected return would be:

$$(4\% \times 25\%) + (6\% \times 25\%) + (10\% \times 25\%) + (14\% \times 25\%) = 8.5\%$$

The portfolio's risk is a function of the variances of each asset and the correlations of each pair of assets. To calculate the risk of a four-asset portfolio, an investor needs each of the four assets' variances and six correlation values, since there are six possible two-asset combinations with four assets. Because of the asset correlations, the total portfolio risk, or standard deviation, is lower than what would be calculated by a weighted sum.

3.2 Monte Carlo (MC) Simulation:

Monte Carlo uses the Law of Large Numbers (LLN) to come up with an estimate for a population parameter using simulated values. LLN states that if $X_1, X_2, X_3, .., X_N$ are all independent random variables with the same underlying distribution (i.i.d), where all X's have the same mean μ and standard deviation σ, as the sample size grows, the probability that the average of all X's is equal to the mean μ is equal to 1. The LLN can be summarized as follows:

$$\text{If,} \qquad E(X_i) = \mu \qquad Var(X_i) = \sigma^2 \qquad \text{for } i = 1, 2, 3..., N$$

$$\text{and,} \qquad \bar{X_n} = \sum_{i=1}^{N} X_i / N$$

$$\text{then,} \qquad Pr(\bar{X_n} = \mu) = 1 \qquad \text{if} \qquad N \to \infty$$

Consequently, the idea behind Monte Carlo estimation is that when we obtain an estimate for a parameter a sufficiently large number of times, then the mean of these estimates will form a Monte Carlo unbiased estimate for that parameter.

$$\hat{\theta} = \frac{1}{M} \sum_{i=1}^{M} \theta_i$$

However, for this work the task is really to deal with an optimization problem where the aim is to find the 10 corresponsing weights that maximizes the objective function which is the Sharpe ratio, S_a. This can be mathematically expressed as below.

$$\hat{w_i} = \arg\max_{w_i} S_a$$

3.3 Coupling MPT with MC:

The core objective for the portfolio optimization task is finding the optimum set of weights associated with each asset in the portfolio so that the annual Sharpe ratio, S_a is maximized. For MC simulation, we initialize the weights to be equal for each asset. Since there are 10 assets, each one of them were assigned the weight of $w_i = 1/10 = 0.1$. We run the simulation sufficiently large number of times ($M = 30000$) and find the estimates for the weights that would maximize S_a.

4. Results and Discussions

The MC simulation has been run $M = 30000$ times. This means that M random portfolios have been generated from standard uniform distribution, $U(0, 1)$ by giving random weights to stocks in the portfolio. Based on the available stock data, it's possible to calculate the return and volatility from the formulae mentioned before. Since this is a stock portfolio optimization problem, the investor's goal is either to maximize the return or to minimize the risk. The first is observed when the Sharpe ratio reaches the maximum and the latter is observed when the volatility is minimum. These two scenarios are the two extremes that can be beneficial for the investor. However, on the other hand, the two opposite extremes would be the minimum Sharpe ratio and the maximum volatility which are the worst case scenarios. Those two events would the least desirable portfolio asset allocation.

Index	**Extremity**	**Sharpe Ratio, S_a**	**Annual Return, R_p**	**Annual Volatility, σ_p**
I	Max S_a	248.93	35.75	14.36
II	Min S_a	70.01	11.80	16.85
III	Max σ_p	118.28	29.87	25.25
IV	Min σ_p	162.43	21.70	13.36

Table 2: Sharpe ratio and its corresponding annual return and volatility (all in percentage) for four extreme cases.

The Sharpe ratio and the corresponding annual return and volatility (all in percentage) for the four extremes - the maximum Sharpe ratio (I), minimum Sharpe ratio (II), maximum volatility (III) and minimum volatility (IV) - are provided in Table 2. It is obvious to mention that for highest annual return, the best choice of combination for weights would be I and the worst would be II. On the other hand, for achieving the lowest risk on the investment, IV would be the best choice and III would be the worst. The corresponding asset weight percentages for these extremes are provided in Table 3. It's evident that for maximizing the return, the investor should put the most significant portion of his allocations on MSFT, GOOG and ORCL. For minimizing risk, the major four assets to consider would be WMT, ORCL, MSFT, AMZN. It seems that investing in MSFT and ORCL would be beneficial in both sense.

Asset	AAPL	AMZN	FB	GOOG	INTC	MSFT	NFLX	ORCL	TSLA	WMT
I_{w_i}	0.94	1.72	12.52	3.45	2.34	26.55	0.83	26.95	8.52	18.27
II_{w_i}	2.33	13.23	6.62	8.65	28.77	1.55	5.68	0.48	4.54	28.15
III_{w_i}	10.13	0.16	14.22	0.29	29.00	5.18	9.24	1.11	28.44	2.23
IV_{w_i}	5.15	9.77	2.39	2.67	7.48	3.03	9.26	24.89	4.82	30.53

Table 3: The optimized weight percentages of the portfolio assets for the four extremities.

However, rather than looking at these MC simulation and the relevant parameters that maximizes the outcome, a scatter plot of the return against the risk would serve a better purpose. One such plot is provided in Figure 1 where the four portfolios I, II, III and IV are represented by the blue, purple, green and orange triangles respectively. This plot gives an overall idea about the relation between the annual risk and return and a somewhat linear relationship is observed. This indicates that for achieving a extremely high return, it might require the investor to partake a significantly high risk. That is often not a reasonable idea and thus relying on the Sharpe ratio rather than the annual return is mostly preferred since it also takes the risk into account. The plot also provides an idea about the range of values the return and risk can obtain.

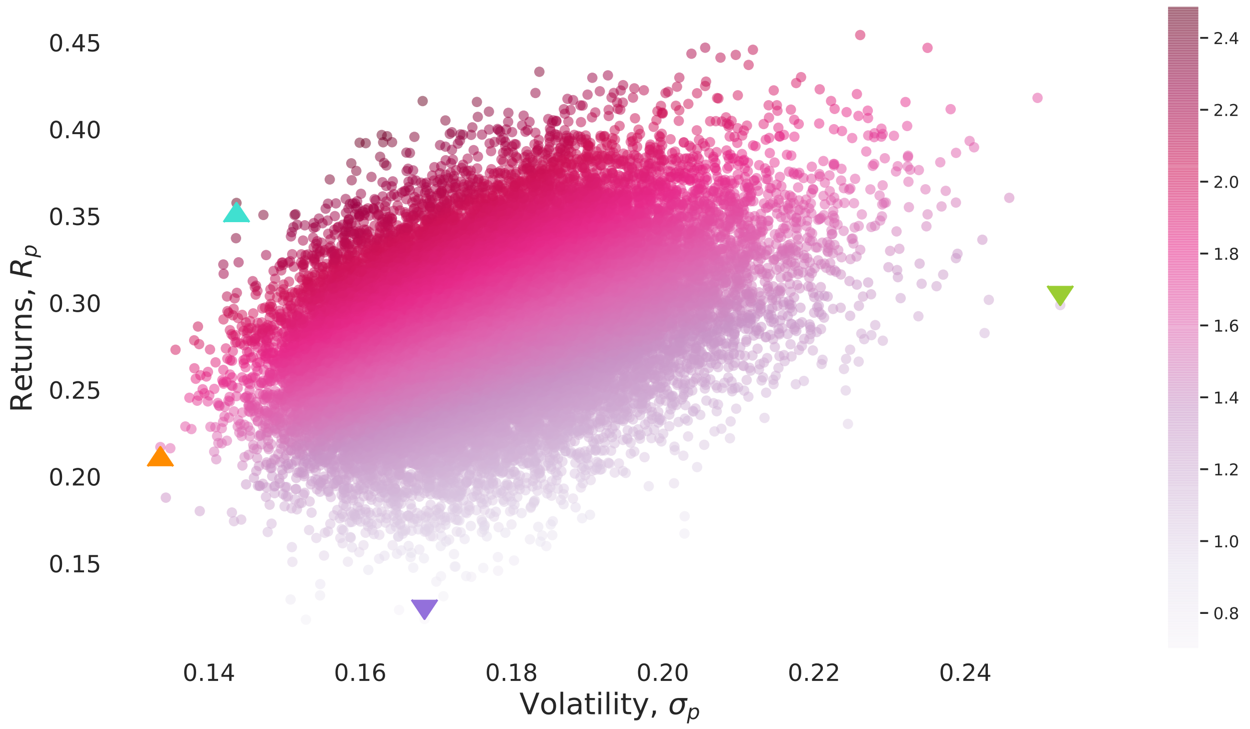

Fig 1. Anual return vs risk for 30000 MC simulations. The colorbar represents the Sharpe ratio.

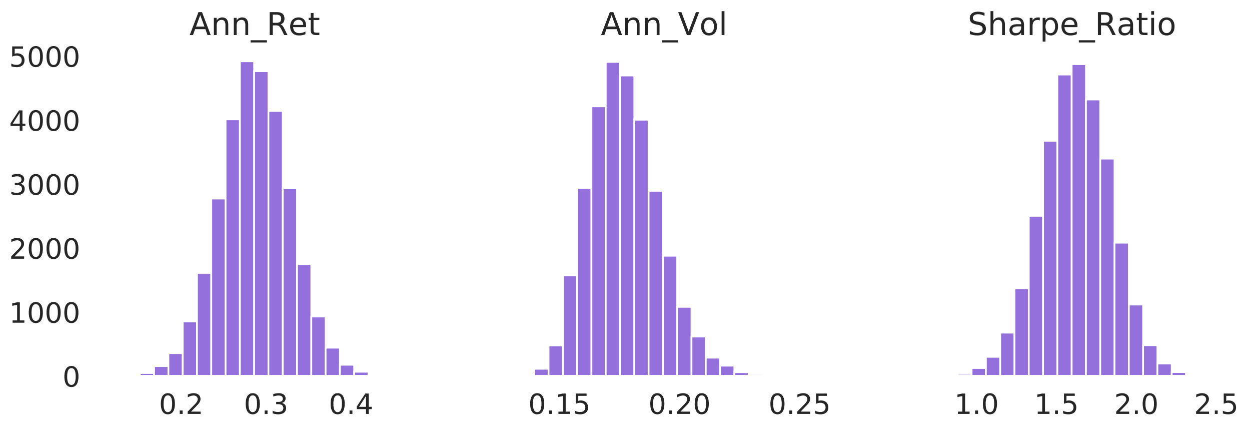

Fig 2: Distribution of the key metrics for portfolio optimization.

In Fig 2, the distributions for the annual return, annual volatility and the Sharpe ratio from the 30000 portfolios generated by MC simulations are provided. It appears that for abut 5000 of these portfolios, the annual return tended to be approximately 28% of the investment. On the other hand, the most frequent standard deviations for these assets were roughly 17% which accounts for the volatility of the assets in these portfolios over the year. The Sharpe Ratio, consequently, hovers around 1.5 times (150%) for roughly the same number of portfolios. This is interesting to see because from the 30000 simulated portfolios, almost 5000 of them (16.57%) have a a very high Sharpe Ratio. This is an outstanding scenario which is being observed supposedly because of the selection of the stocks. The 10 stocks selected for this analysis are well known Fortune 500 companies and thus, despite their high volatility throughout the year, the return from the invesestments were commensurate. This also reestablish the notion of “risking high volatility for high return” that was observed from Fig. 1.

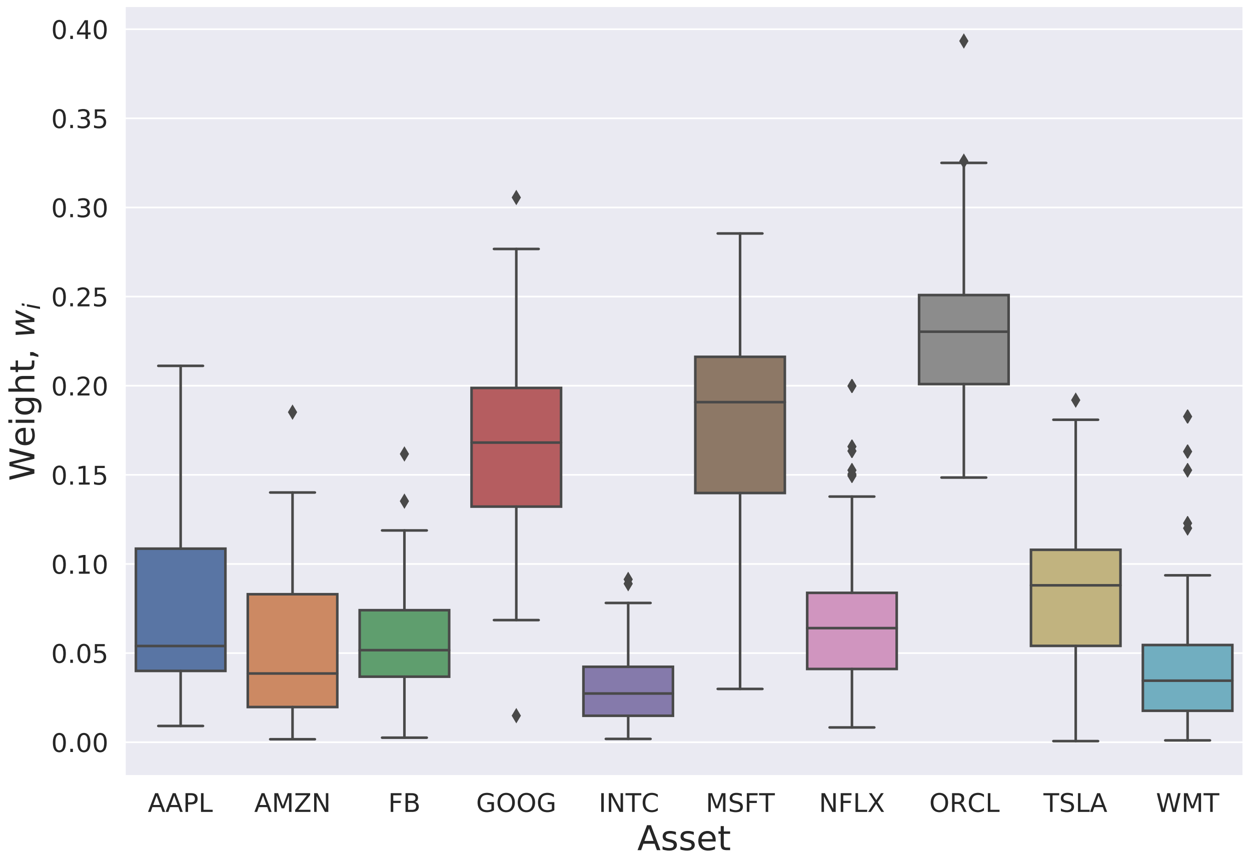

Fig 3. Weight Distribution of different assets for top 100 portfoilos according to highest Sharpe Ratio.

It would be more desirable to see how the assets were diversified for some of the top portfolios. Aimed at that, a boxplot has been provided in Fig. 3 to acquire and idea of how the weights for each assets were distributed for the top 100 portfolios based on the Sharpe Ratio. Boxplots are provided for all 10 assets and we can see a wide variety of summary statistics for each one of these. These boxplots, generated from 30000 MC simulations based on the last one year of stock data, indicates that investing in Google, Microsoft and Oracle has been the most beneficial in terms of Sharpe Ratio or essentially, the return on investments with minimum amount of risk. It is because they have the highest median value as well as the highest weight percentages among all.

Similarly, the box plots for each asset based on the top 100 portfolios that have the minimum volatility is provided in Fig. 4. It’s interesting to see that even though return from Walmart investments was among the minimums (Fig. 3), it shows the lowest volatility of all. Moreover, Oracle and Microsoft proves to be the next best choices for investments to minimize risks. It’s conveninent to make conclusions from these two figures because it indicates very clearly that Oracle and Microsoft should be the top two priorities for diversifying investments not only because they provide very high return but also they minimize the investor's risk to a significant amount. The rest of the assets and their weight percentage can also be chosen by taking informed decisions based on the results provided by the above two figures.

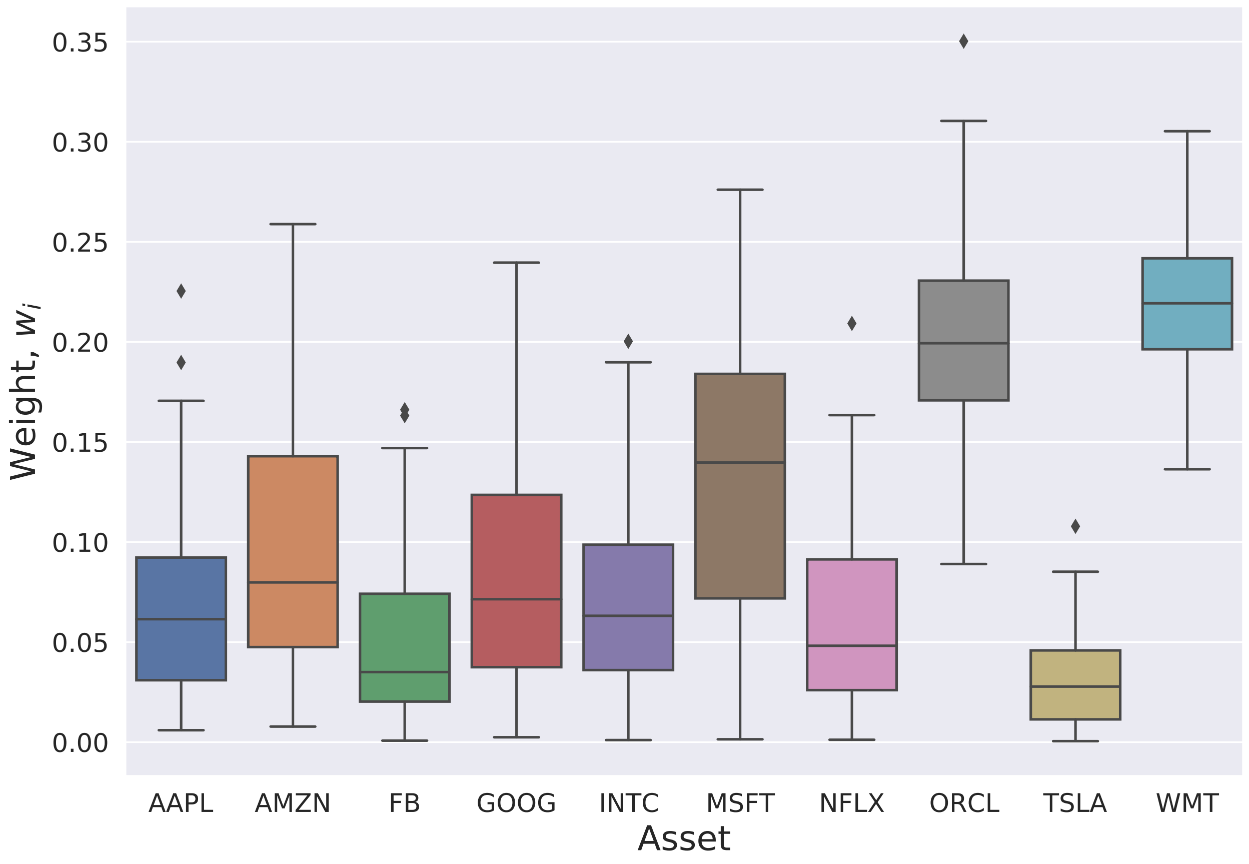

Fig 4. Weight Distribution of different assets for top 100 portfoilos according to lowest Volatility.

5. Conclusion

Stock portoflio optimization is more dificult to implement in practice than it sounds in theory. In this work, an oversimplified approach for this task has been presented. This approach can be extended to account for thousands of different assets, running a much higher number of MC simulations, on historical stock data that span over a time much longer than a year. These suggested changes can supposedly prvide the investors with a more concrete idea about the diversification of their assets. While this method provides some insgights into the potential stocks for investements while informing about the associated risk, has its criticism. The method is entirely founded upon the numerical simulations performed on previous and thus, making a prediction for future investments solely based on these results may breed unpleasant surprises. Rather than completely replying on a statistical method for this purpose, the investors must reply on their own discretion while aware of the circumstances of the economic world. For all is known, a pandemic may appear out of nowhere and can change everything. Thus, while Monte Carlo approach has its numerous beneficial application in multiple scientific fields, for effective portfolio optimization, it should be treated only as a tool to narrow down the options for a potential investor, nothing more than that.

Reference

[1] Harry Markowitz. “Portfolio Selection.” The Journal of Finance, Volume 7, No. 1, 1952, Pages 77-91.
[2] https://finance.yahoo.com

Appendix

The Python code used for this work is provided below.

```
###############################  Data Preprocessing  ################################

#importing necessary libraries
import random
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

np.random.seed(123)

#reading the datasets
raw_data = {}
raw_data['AAPL'] = pd.read_csv('AAPL.csv') #Apple
raw_data['AMZN'] = pd.read_csv('AMZN.csv') #Amazon
raw_data['FB'] = pd.read_csv('FB.csv')     #Facebook
raw_data['GOOG'] = pd.read_csv('GOOG.csv') #Google
raw_data['INTC'] = pd.read_csv('INTC.csv') #Intel
raw_data['MSFT'] = pd.read_csv('MSFT.csv') #Microsoft
raw_data['NFLX'] = pd.read_csv('NFLX.csv') #Netflix
raw_data['ORCL'] = pd.read_csv('ORCL.csv') #Oracle
raw_data['TSLA'] = pd.read_csv('TSLA.csv') #Tesla
raw_data['WMT'] = pd.read_csv('WMT.csv')   #Walmart

#store the datasets in a dictionary, keep only the closing price
datas = {}
for name,data in raw_data.items():
    datas[name] = raw_data[name]['Close'].reset_index()

#store the closing price columns of each dataset in a dictionary
CP_dict = {}
for name,data in datas.items():
    CP_dict[name] = datas[name]['Close']

#convert the dictionary to a dataframe
CP_df = pd.DataFrame(CP_dict)



###############################  MC Simulation  #####################################

#first, calculate the annual return and volatility with equal weights
```

```
#daily log return for each asset and store within the dataframe
daily_return = np.log(CP_df.pct_change() + 1).dropna()
#mean daily return for each asset
daily_return_mean = np.array(daily_return.mean())
#assigning equal weight of 0.1 (1/10) for each asset
weights = np.array([0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1])
#portfolio return can now be calculated as
Port_return = np.sum(weights * daily_return_mean)

#covariance for each asset
cov = daily_return.cov()
#portfolio volatility
Port_Vol = np.sqrt(np.dot(weights.T,np.dot(cov,weights)))



#the same calculations will now be done M times with different weight combos
#Declare the number of Portfolio to be generated
num_portfolio = 30000

#an empty list for storing annual returns, volatility, sharpe ratio
#and weightage of each stock in the generated portfolios
results = np.zeros((3 + len(daily_return.columns),num_portfolio))

#Monte Carlo simulation loop
for i in range(num_portfolio):

    #sample random weights from standard uniform
    weight = np.random.rand(len(daily_return.columns))
    #ensure that sum of all weight equal to 1
    weight = weight/np.sum(weight)

    #Annual return
    p_annual_return = np.sum(weight*daily_return_mean)*252
    #Annual volatility
    p_annual_volatility = np.sqrt(np.dot(weight.T,np.dot(cov,weight)))*np.sqrt(252)

    #Storing the values in results list
    results[0,i] = p_annual_return
    results[1,i] = p_annual_volatility
    results[2,i] = results[0,i]/results[1,i]

    for j in range(len(weight)):
        results[j+3,i] =  weight[j]


#store all the results in a single dataframe
cols = ['Ann_Ret','Ann_Vol','Sharpe_Ratio']
for num in range(len(list(daily_return.columns))):
    cols.append(list(daily_return.columns)[num])

#resulting dataframe
result_df = pd.DataFrame(results.T,columns=cols)
```

```
###############################  Visualizations  ####################################

#Fig 1 : return vs risk

fig=plt.figure(figsize=(15,8))
plt.scatter(result_df['Ann_Vol'],result_df['Ann_Ret'], \
            c=result_df['Sharpe_Ratio'],cmap='PuRd',alpha=0.5)
plt.colorbar()

plt.scatter(max_sharpe_ratio[1],max_sharpe_ratio[0], \
            marker = 6,color='turquoise',s=300) #portfolio I
plt.scatter(min_sharpe_ratio[1],min_sharpe_ratio[0], \
            marker = 7,color='mediumpurple',s=300) #portfolio II
plt.scatter(volatility_highest[1],volatility_highest[0], \
            marker = 7,color='yellowgreen',s=300) #portfolio III
plt.scatter(volatility_lowest[1],volatility_lowest[0], \
            marker = 6,color='darkorange',s=300)#portfolio IV

plt.xlabel('Volatility, $\sigma_p$',fontsize = 20)
plt.ylabel('Returns, $R_p$',fontsize = 20)
plt.xticks(fontsize = 16)
plt.yticks(fontsize = 16)
plt.show()
fig.savefig('fig.pdf', dpi=200)

#Fig 2 : dist of annual return, volatility and Sharpe ratio

result_df[['Ann_Ret', 'Ann_Vol', 'Sharpe_Ratio']].hist(bins=20, \
                figsize=(20, 8),layout=(2,5),grid=False, \
                xlabelsize=16,ylabelsize=16,sharex=False,sharey=True, \
                density=False, color='mediumpurple',alpha=1.0)
plt.savefig('histAnn.pdf')

#Fig 3 : weight distritbuion for top 100 portfolios with highest Sharpe ratio

top_df = result_df.sort_values(by='Sharpe_Ratio',ascending=False)
top_df = top_df.drop(['Ann_Ret', 'Ann_Vol', 'Sharpe_Ratio'], axis=1)

sns.set(rc={'figure.figsize':(11.7,8.27)})
b=sns.boxplot(data=top_df.iloc[0:100:,:])
b.set_xlabel("Asset",fontsize=20)
b.set_ylabel("Weight, $w_i$",fontsize=20)
b.tick_params(labelsize=15)
fig = b.get_figure()
fig.savefig('boxTop.pdf')

#Fig 4 : weight distritbuion for top 100 portfolios with lowest annual volatility

top_df2 = result_df.sort_values(by='Ann_Vol',ascending=True)
top_df2 = top_df2.drop(['Ann_Ret', 'Ann_Vol', 'Sharpe_Ratio'], axis=1)

sns.set(rc={'figure.figsize':(11.7,8.27)})
b=sns.boxplot(data=top_df2.iloc[0:100:,:])
```

```
b.set_xlabel("Asset",fontsize=20)
b.set_ylabel("Weight, $w_i$",fontsize=20)
b.tick_params(labelsize=15)
fig = b.get_figure()
fig.savefig('boxTop2.pdf')
```