RM 294 - Optimization II

Dr. Daniel Mitchell

Project 2 – Integer Programming

Group 6

Apurva **Audi (aa85254)**

Soumya **Agrawal (sa55638)**

Troy **Austin (ta23234)**

Vivek **Dhulipalla (vd6543)**

Introduction

An investment that tracks a market index is called an index fund. We decided to use the NASDAQ-100 index for this project. We could typically locate at least one index fund that follows the index. When faced with a variety of options, we chose the index fund that most closely tracks the performance of the index. We decide how many shares of each chosen stock to purchase after choosing the index fund.

Method 1: Stock Selection and Weights Calculation - Integer Programming

For stock selection, we used two methods. To maximize the resemblance between the index stock and its representative in the fund, we first calculate the correlation between the fund and each stock in the index. Constraints and the objective function are:

$$\max_{x,y} \sum_{i=1}^{n} \sum_{i=1}^{n} \rho_{ij} x_{ij}$$

$$s.t. \sum_{j=1}^{n} y_j = m$$

$$\sum_{j=1}^{n} x_{ij} = 1 \ for \ i = 1, 2, \ldots, n$$

$$x_{ij} - y_j \leq 0 \ for \ i, j = 1, 2, \ldots, n$$

$$x_{ij}, y_j \in \{0, 1\}$$

An example of the correlation matrix is shown in Table 1, which indicates that a stock has the strongest correlation (equal to 1) with itself.

We then calculated the weighted return of each chosen stock to match the returns of the index. The objective function and constraints are:

$$\min_{w} \sum_{t=1}^{T} d_t$$

$$s.t. \sum_{i=1}^{m} w_i = 1$$

$$d_t + \sum_{i=1}^{m} w_i r_{it} \geq q_t \ for \ t = 1, 2, \ldots, T$$

$$d_t - \sum_{i=1}^{m} w_i r_{it} \geq -q_t \ for \ t = 1, 2, \ldots, T$$

$$w_i \geq 0$$

The return of each stock, r_{it}, is calculated using the pandas pct_changes function, which computes the percentage change from the immediately previous day. We calculated the return for both 2019 and 2020, as shown in Tables 1 and 2.

Date	NDX	ATVI	ADBE	AMD	ALXN	ALGN	GOOGL	GOOG	AMZN	AMGN	...
2019-01-03	-0.033602	-0.035509	-0.039498	-0.094530	0.022030	-0.085791	-0.027696	-0.028484	-0.025242	-0.015216	...
2019-01-04	0.044824	0.039903	0.048632	0.114370	0.057779	0.010445	0.051294	0.053786	0.050064	0.034184	...
2019-01-07	0.010211	0.028196	0.013573	0.082632	0.018302	0.017192	-0.001994	-0.002167	0.034353	0.013457	...
2019-01-08	0.009802	0.030309	0.014918	0.008751	0.006207	0.015954	0.008783	0.007385	0.016612	0.012824	...
2019-01-09	0.007454	0.017210	0.011819	-0.026988	0.012430	0.038196	-0.003427	-0.001505	0.001714	-0.001196	...
...	...	...	...	...	...	...	...	...	...	...	...
2019-12-23	0.002019	-0.005572	0.004090	0.029672	0.006469	0.019239	-0.000437	-0.000556	0.003638	-0.000123	...
2019-12-24	0.000402	-0.001358	0.002098	0.023757	-0.001630	-0.000899	-0.004590	-0.003914	-0.002114	-0.002880	...
2019-12-26	0.009058	0.001360	0.004732	0.001934	-0.012242	0.001331	0.013418	0.012534	0.044467	-0.001774	...
2019-12-27	-0.000835	0.005094	-0.001238	-0.009650	-0.003488	-0.002228	-0.005747	-0.006256	0.000551	-0.001530	...
2019-12-30	-0.006983	-0.005237	-0.007407	-0.014292	-0.011147	-0.007240	-0.011021	-0.011650	-0.012253	-0.005217	...

Table 1: Sample of return of each stock in 2019

Date	NDX	ATVI	ADBE	AMD	ALXN	ALGN	GOOGL	GOOG	AMZN	AMGN	...
2020-01-03	-0.008827	0.000341	-0.007834	-0.010183	-0.013260	-0.011421	-0.005231	-0.004907	-0.012139	-0.006789	...
2020-01-06	0.006211	0.018238	0.005726	-0.004321	0.001598	0.019398	0.026654	0.024657	0.014886	0.007674	...
2020-01-07	-0.000234	0.010043	-0.000959	-0.002893	0.002533	-0.009864	-0.001932	-0.000624	0.002092	-0.009405	...
2020-01-08	0.007452	-0.007623	0.013438	-0.008705	0.016191	0.010386	0.007118	0.007880	-0.007809	0.000756	...
2020-01-09	0.008669	-0.009018	0.007636	0.023834	0.019893	0.036853	0.010498	0.011044	0.004799	0.002980	...
...	...	...	...	...	...	...	...	...	...	...	...
2020-09-24	0.005828	-0.005491	-0.005782	0.014586	-0.003647	0.011546	0.009557	0.009242	0.006644	-0.009357	...
2020-09-25	0.023371	0.016188	0.025894	0.029544	0.021514	0.019398	0.011386	0.011671	0.024949	0.014564	...
2020-09-28	0.019130	0.011978	0.018196	0.018191	-0.009875	0.015507	0.013620	0.013537	0.025498	0.013165	...
2020-09-29	-0.003652	-0.014277	0.001679	0.028812	-0.007414	-0.008332	0.005046	0.003284	-0.009190	0.005141	...
2020-09-30	0.008400	0.002104	0.002248	0.002690	0.017517	0.022457	-0.000287	0.000184	0.001224	0.023600	...

Table 2: Sample of return of each stock in 2020

Method 2: Only used Weight Calculation - Mixed Integer Programming

In this method, the stock selection integer programming issue is completely disregarded in favor of a MIP problem that required an integer to represent the number of non-zero weights. To do this, we also added some constraints that force $w_i = 0$ if $y_i = 0$ and a series of binary variables called y_i, y_2, ..., y_n. As a result, the objective function and constraints are:

$$\min_{w} \sum_{t=1}^{T} d_t$$

$$s.t. \sum_{i=1}^{n} w_i = 1$$

$$\sum_{i=1}^{n} y_i = m$$

$$w_i - M y_i \leq 0 \ for \ i = 1, 2, \ldots, n$$

$$d_t + \sum_{i=1}^{n} w_i r_{it} \geq q_t \ for \ t = 1, 2, \ldots, T$$

$$d_t - \sum_{i=1}^{n} w_i r_{it} \geq -q_t \ for \ t = 1, 2, \ldots, T$$

$$w_i \geq 0$$

The smallest value for M here would be 1 since the largest value that wi can have is 1 given that the sum of the weights must be 1. We limited this method to roughly 10 hours of runtime due to the fact that we had 2T+n+2 constraints and 2n+T decision variables (T = time period and n = total number of stocks)

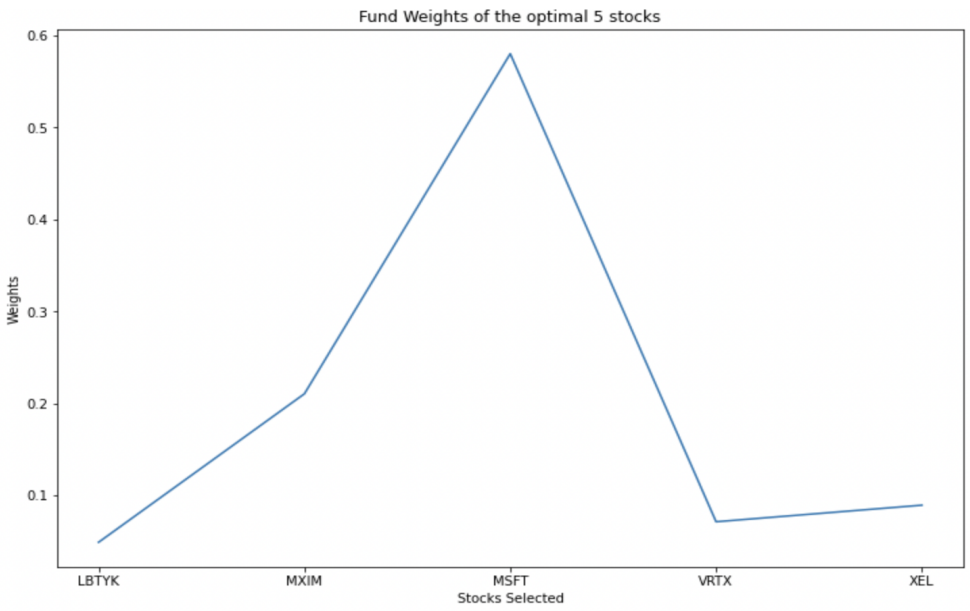
Outcomes

Method 1:

When m = 5, the following is the result along with a line graph representing the corresponding weights for each of the top five stocks. Using 2019 data, the five best stocks are MSFT, MXIM, XEL, VRTX, and LBTYK.

```
print("Top 5 stocks based on weights : ",fund_stocks19)
print("The corresponding weights of these stocks are : ",funds_weight19)
```

```
Top 5 stocks based on weights :  ['LBTYK', 'MXIM', 'MSFT', 'VRTX', 'XEL']
The corresponding weights of these stocks are :  [0.04886175 0.21038806 0.58035198 0.07119022 0.089208  ]
```

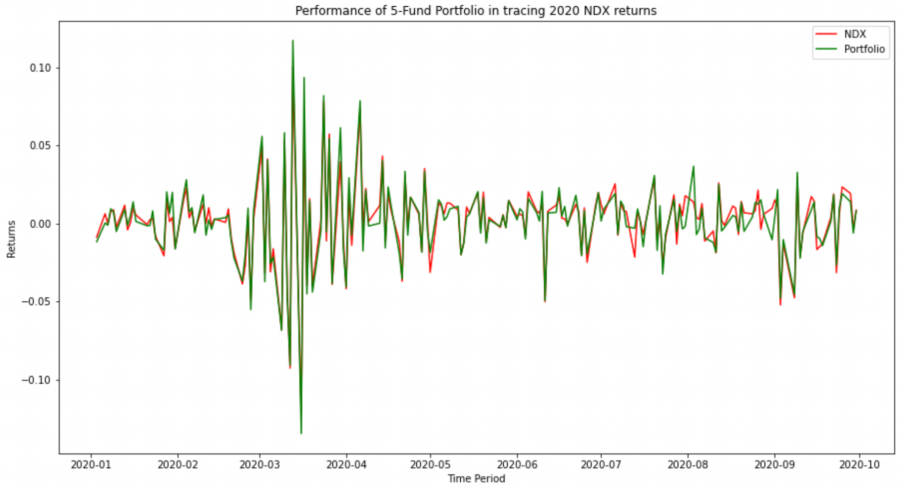


The total absolute deviation of the portfolio from NASDAQ in 2020 is approximately 0.8 units higher based on the portfolio developed using 2019 stock data. This means that portfolio returns in 2020 will be more evenly distributed relative to real NASDAQ returns. This problem is most likely caused by the fact that a portfolio of only five stocks is insufficient to accurately replicate the index fund. This belief is supported by the fact that, even in 2019, deviations from the NASDAQ are significant. As a result, increasing the number of stocks in the portfolio increases the likelihood that the portfolio will accurately reflect the NASDAQ-100 results.

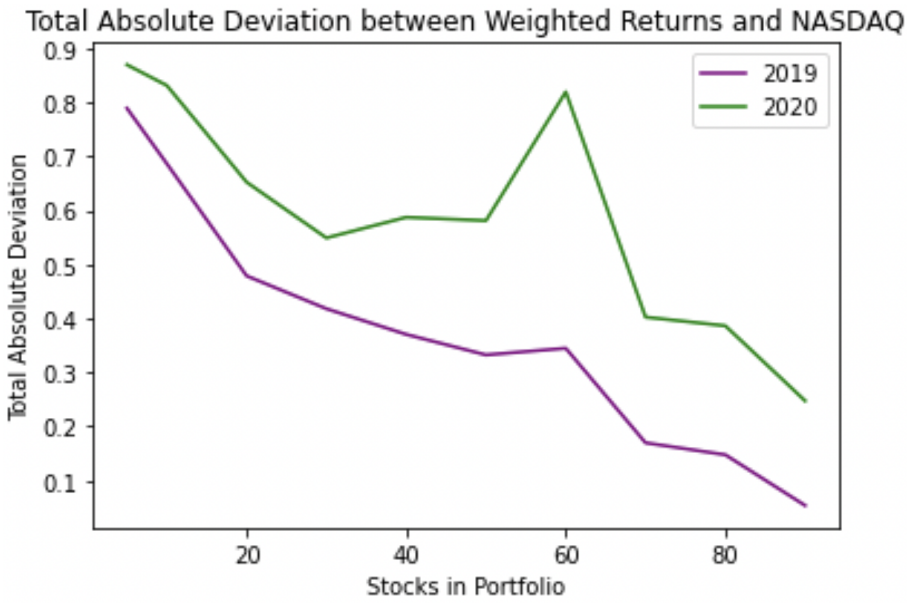
```
perform1 = perform_measure(return_stock19,fund_stocks19,funds_weight19)
perform2 = perform_measure(return_stock20,fund_stocks19,funds_weight20)

print('Total absolute deviation from NASDAQ in 2019: ',round(perform1,3))
print('Total absolute deviation from NASDAQ in 2020: ',round(perform2,3))
```

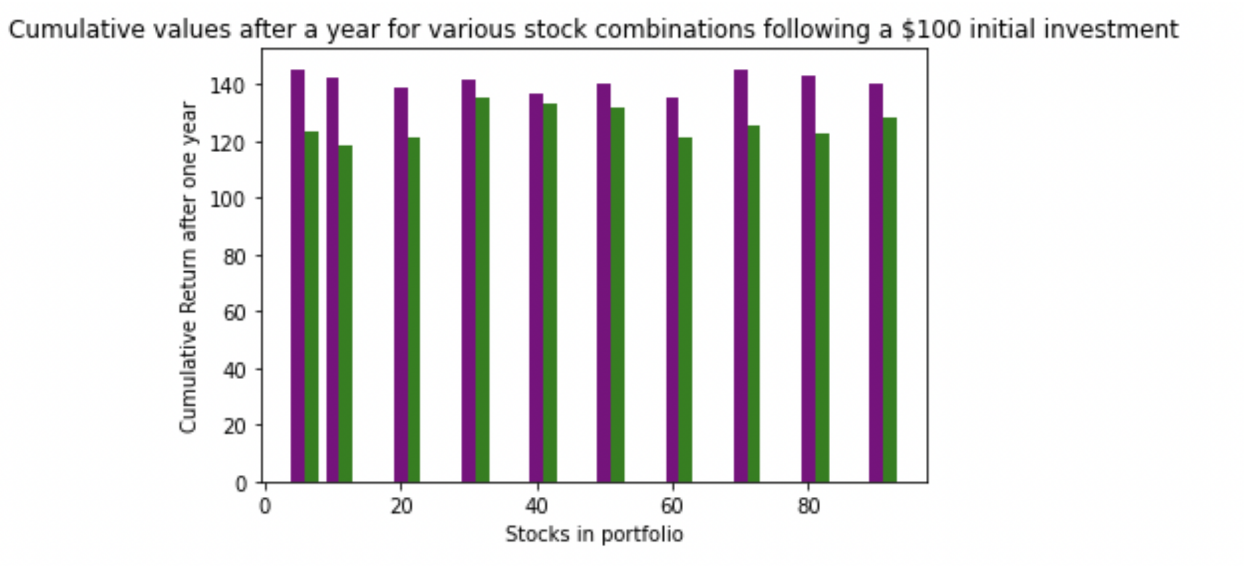
```
Total absolute deviation from NASDAQ in 2019:  0.789
Total absolute deviation from NASDAQ in 2020:  0.851
```

Following that, we created a graph to demonstrate how accurate the portfolio is at monitoring the index fund in 2020. The graph below shows that the portfolio is reasonably accurate in tracking the index fund in 2020. Particularly when the returns are subject to large swings.



According to the graph below, the overall absolute deviation is consistently decreasing for both 2019 and 2020 until the portfolio contains 50 stocks. Furthermore, with 60 stocks in the portfolio, we begin to see declining returns as a result of adding more stocks to the portfolio. The 2020 graph makes this shift much more visible. This is because the model is using weights from 2019. After 60 stocks, the divergence from the index fund begins to decline once more. It will eventually stabilize in 2020, when the number of stocks in the portfolio is between 90 and 100.

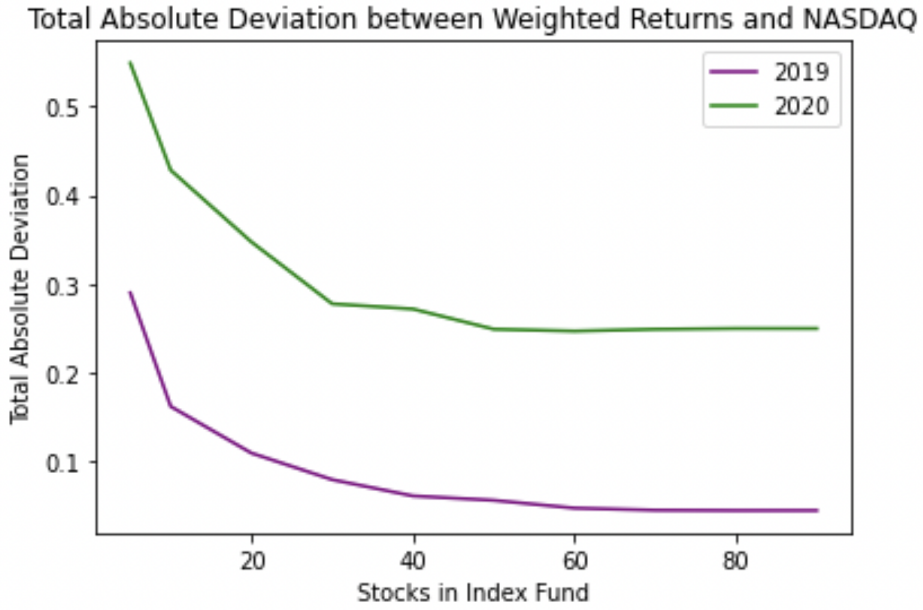




This above bar graph depicts the number of stocks required to achieve the maximum cumulative value after a year of investing $100 at time 0 when investing $100 at time 0. In 2019, 70 stocks must reach a total cumulative value of 145.2. For 2020, 30 stocks must reach a total cumulative value of 134.8.

Method 2

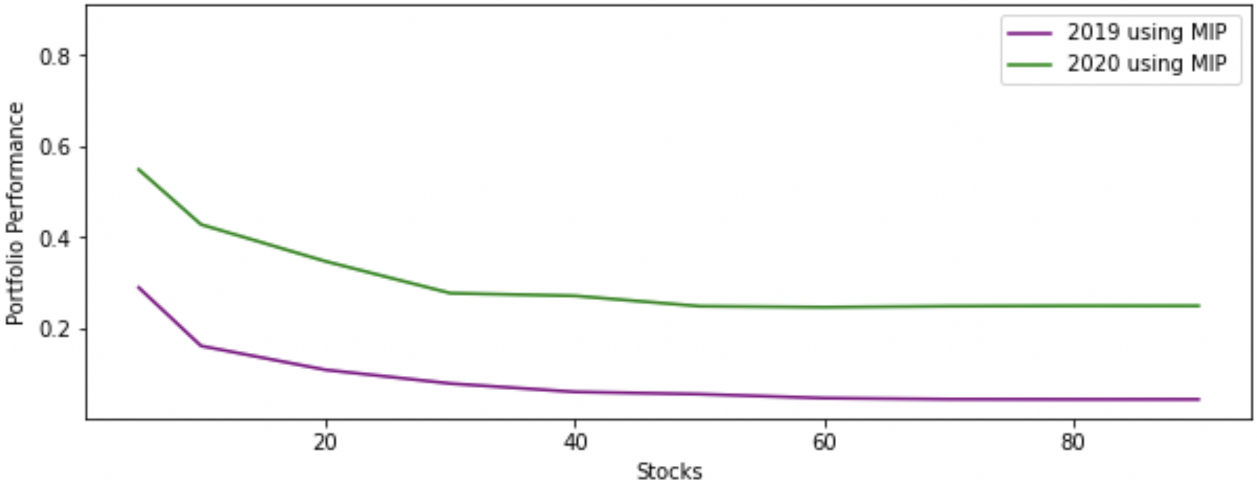
The graph below depicts how the total absolute deviation has changed as the index fund's stock holdings have grown. Unlike Method 1 which has a small increase in deviation as the number of stocks approaches 50, Method 2 shows a constant decrease in deviation as the number of stocks increases. Because the model is based on 2019 stock data, we can see that 2019 stock data has a much lower total absolute deviation than 2020 stock data. Given that the slopes of both graphs flatten out around 50, we can conclude that including 50 stocks in the index fund would be sufficient to precisely track the NASDAQ index when using Method 2.



Comparison Between Two Methods

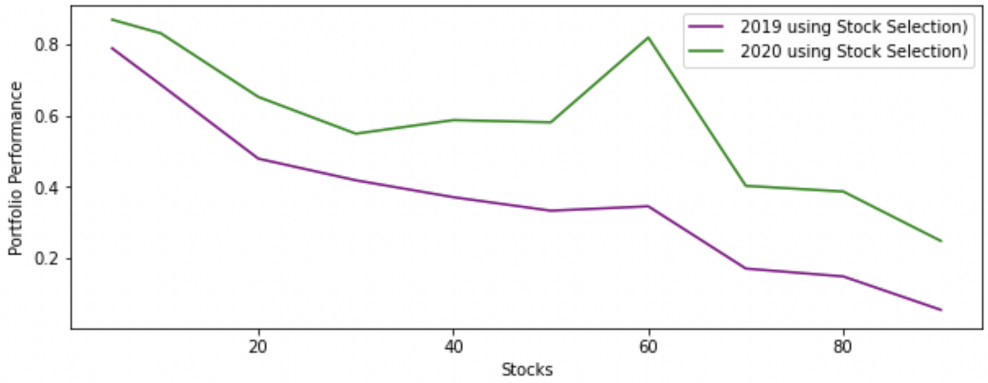
Method 1:

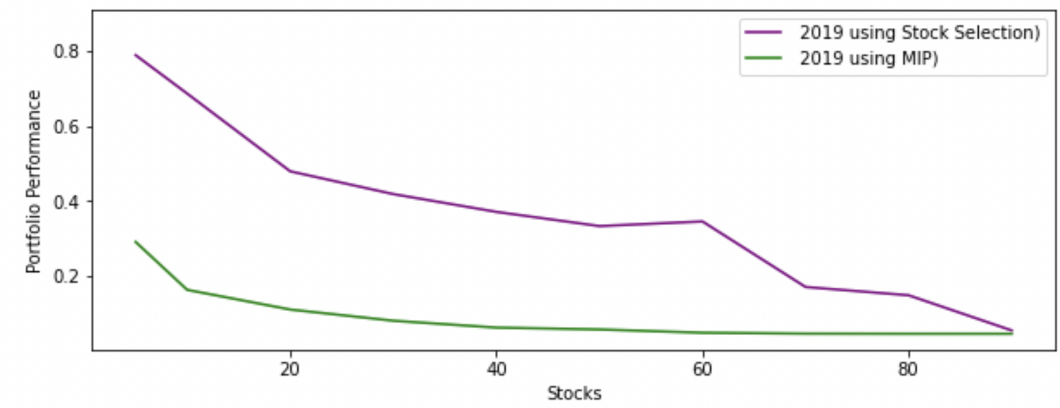
In 2019, selecting 60 stocks performed worse than selecting 50 stocks. After selecting 30 companies in 2020, there are declining returns because the loss remains constant and even rises between m = 50 and m = 70. Therefore, 30 equities would be the ideal amount of stocks to select for modeling the NASDAQ-100 index.

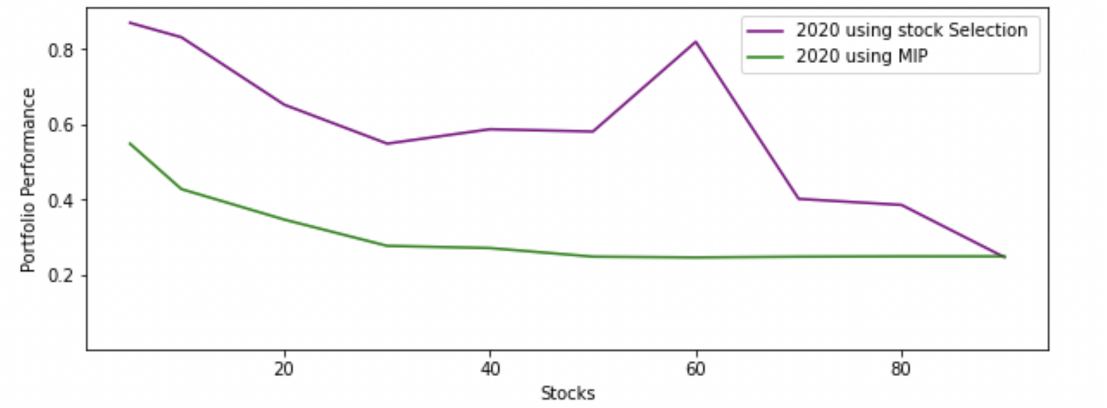


Method 2:

With more stocks used in this method, the performance keeps getting better. There is only one instance in 2019 where selecting 20 stocks performed worse than selecting just 10. After selecting 40 stocks in 2020, there are declining rewards because the loss then remains constant. As a result, 40 equities would be the ideal number to use for modeling the NASDAQ-100 index.







Conclusion

The graphics shown above makes it abundantly evident that Method 2 performs better on the 2020 data. Both approaches produce comparable results, however Method 1 exhibits far greater variance due to its larger loss fluctuations. Our reasonable conclusion is to utilize the Second Method moving forward, which involves selecting 40 stocks to model the NASDAQ-100 index and modeling the optimization to determine the weights without first selecting the stocks, as Method 2 is superior. By figuring out the price of adding more stocks to our portfolio and factoring that price into our optimization restrictions, we can enhance our model.