Optimization Project 2

Chyavan Chandrashekar, Mihir Deshpande, Neel Sheth, Tanushree Balaji

Introduction

Building an investment portfolio requires a comprehensive understanding of the financial market & trading strategies coupled with careful fund allocation. While we could develop a tailored portfolio by “actively” seeking high-performing stocks to outperform the market, a less aggressive approach would be ‘Indexing’, to mimic established financial market indices (such as NASDAQ) and achieve similar results. Given that this is a more “passive” / low-risk strategy, one can expect lower but assured returns spread out over an extended timespan, making this a credible way to plan finances for the distant future. Hence, it is worthwhile to spend some time building a model that identifies the best stock options to represent & track the broad market index well and also allocates funds to mirror its returns.

Approach

Finding the right stocks to add to the index fund portfolio is crucial to represent a market index like NASDAQ. Factors to be considered in this process are as follows:

1. Decide on how many stocks go into the portfolio. This is a vital step in the process because it is not feasible to include all the stocks of an index, and continuously track and adjust based on the changes in the index.
2. Identify stocks that best represent industries and businesses included in NASDAQ through the use of a similarity metric such as correlation.
3. Once the stocks to invest in are identified, decide how the funds are going to be allocated amongst them
4. In the process of weighting the stocks, ensure that the returns from this group at any instant closely resemble the performance of the market index to ensure similar returns

Let us go through each step of the process and formulate them for further analysis. We are looking at the 2019 daily stock returns of NASDAQ and the 100 stocks that constitute it. To find the best stocks:

- Create a correlation matrix (call it cor) amongst all the stocks using 2019 data to understand how similar their price movement in the market is.

	ATVI	ADBE	AMD	ALXN	ALGN	GOOGL	GOOG	AMZN	AMGN	ADI	...
ATVI	1.000000	0.399939	0.365376	0.223162	0.216280	0.433097	0.426777	0.467076	0.203956	0.329355	...
ADBE	0.399939	1.000000	0.452848	0.368928	0.363370	0.552125	0.540404	0.598237	0.291978	0.473815	...
AMD	0.365376	0.452848	1.000000	0.301831	0.344252	0.418861	0.417254	0.549302	0.151452	0.503733	...
ALXN	0.223162	0.368928	0.301831	1.000000	0.332433	0.315993	0.307698	0.363170	0.342022	0.317040	...
ALGN	0.216280	0.363370	0.344252	0.332433	1.000000	0.248747	0.250316	0.399281	0.264599	0.328280	...
...	...	...	...	...	...	...	...	...	...	...	...

- Next, we consider a set of variables (call it x) in the form of a matrix with the same dimensions as that of the aforementioned correlation matrix that tells us which stocks present in the portfolio represent the other stocks that are not in the portfolio. This is to be achieved by picking the “m” stocks such that they correlate most to the other stocks that are not picked.
- Another set of variables (call it y) would also be required to tell us which of the stocks we will be buying - this would result in 100 decision variables, one for each stock with 1 representing buy, and 0 representing no-buy.

In total, we have **10100 decision variables** to wrap our head around for this specific data set. Given this variable setup, the following would be the objective and constraints.

Objective

The objective that we are trying to maximize here can be formulated as follows

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

For example, let us assume our stock funds have a high correlation if they are from the same sector. If we choose Google to be one of the stocks in the portfolio, then we expect that it best represents the other stocks not being picked, i.e. the correlation between Google and all other stocks that Google represents is maximum.

Constraints

1. Let’s say we are trying to find **m** stocks to buy. This implies that the sum of all y variables is exactly **m.**

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

2. Each stock has one and only one representative in the stocks picked for the portfolio

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

3. We need to make sure that a stock can represent another stock, only if it is itself present in the portfolio. This can be achieved with a logical constraint. For example, if Google is represented by Facebook, then it's important that Facebook is in the portfolio.

$$x_{ij} \leq y_j \qquad for\ i,j = 1,2,\dots,n$$

Considering all the aforementioned cases, we have a total of **10101 constraints**.

```
# Define the number of constraints, variables, matrices - A,b, sense, obj
n_constraints = 1 + self.n_rows + self.n_rows*self.n_cols
n_variables = self.n_cols + self.n_rows*self.n_cols
A = np.zeros((n_constraints, n_variables))
sense = np.array(['=']*n_constraints)
b = np.array([0]*n_constraints)
constraints_added = 0

# Make correlation matrix linear 1-D Array
correlation_linear = []
for i in self.no_idx_correlation_df.columns:
    correlation_linear += self.no_idx_correlation_df[i].to_list()
obj = np.array([0]*self.n_cols + correlation_linear)

# Constraint to select a max of "m" stocks
A[constraints_added, :self.n_cols] = 1
b[constraints_added] = m
constraints_added += 1

# Constraints to have only one representative stock for each stock in the portfolio
for i in range(self.n_rows):
    A[constraints_added, self.n_cols+i*self.n_rows+arr_cols] = 1
    b[constraints_added] = 1
    constraints_added += 1

# Logical constraint for each x to be non-zero when the representative stock is present in the portfolio
for i in range(self.n_rows):
    for j in range(self.n_cols):
        A[constraints_added, self.n_cols*(1+i) + j] = 1
        A[constraints_added, j] = -1
        b[constraints_added] = 0
        sense[constraints_added] = '<'
        constraints_added += 1
```

Weight Calculation

Once we decide on the stocks to buy, it's important to allocate the right amount of funds to each of them. This is a driving factor in realizing adequate returns that resemble the index.

- Create a matrix to find the return of each chosen stock at time t (call it r)
- Create a weight matrix that allocates weights to each of the chosen stocks (Call it w)
- The returns of NASDAQ over time can also be obtained from the data (call it q)

Our goal is to find the right weights and thus, we have a total of **'m'** decision variables to solve for

Given this variable set up, following would be objective and constraints

Objective

Minimize the sum of absolute errors over all time T

$$min \sum_{t=1}^{T} \left| q_t - \sum_{i=1}^{m} w_i r_{it} \right|$$

This is just a means to ensure that we are closely keeping up with the performance of NASDAQ. If the error is too high, it's an indication that the index isn't properly represented

Constraints

1. The total money allocated cannot exceed your budget (i.e.)

$$\sum_{i=1}^{m} w_i = 1$$

2. Also, none of these weights can be lower than 0 since they represent a proportion of money to be allocated and we are not going short on any positions. This need not be coded since Gurobi makes all variables non-negative by default.
3. Further, given that we are working with an absolute function in the objective, we need two constraints in each time period to convert it to linear form. Hence our new objective function becomes
    a. **Y** now represents the "absolute difference" between the index and portfolio performance at each time period.

$$min \sum_{t=1}^{T} y_t$$

b. Now we constrain y to be positive by forcing both sides of the absolute values to be positive

$$y_t \geq \left(q_t - \sum_{i=1}^{m} w_i r_{it} \right) \text{ for t=1,2,...T}$$

$$y_t \geq \left(\sum_{i=1}^{m} w_i r_{it} - q_t \right) \text{ for t=1,2,...T}$$

Findings

Choosing 5 stocks (m=5), the best stocks for the portfolio are

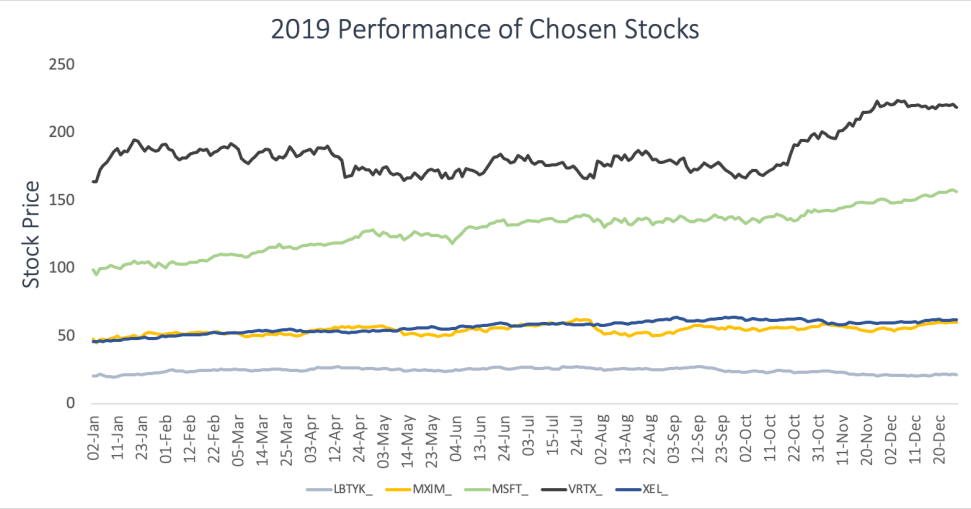
1. **LBTYK** - Liberty Global PLC Class C - *Telecom*
2. **MXIM** - Maxim Integrated - *Technology*
3. **MSFT** - Microsoft Corporation - *Technology*
4. **VRTX** - Vertex Pharmaceuticals Incorporated - *Healthcare*
5. **XEL** - Xcel Energy Inc - *Energy*

The weights (in percentages) are as follows.

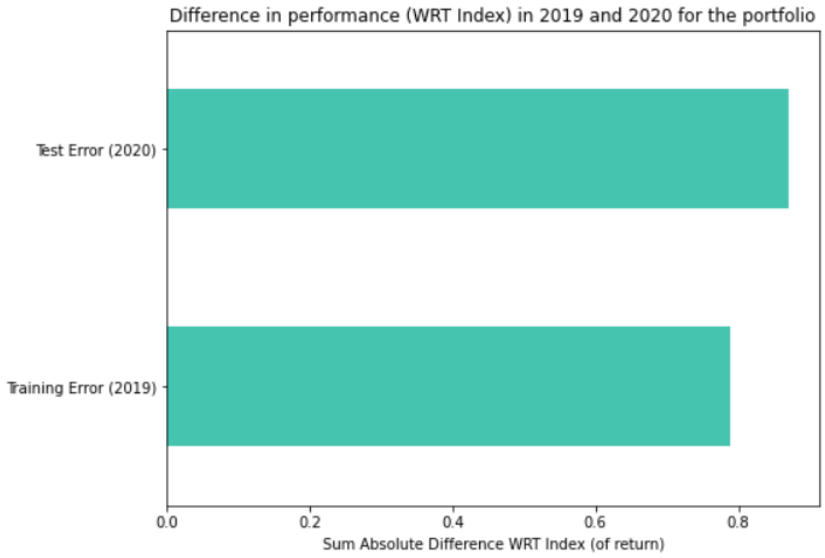
	weight
LBTYK	4.89
MXIM	21.04
MSFT	58.04
VRTX	7.12
XEL	8.92

We can see that technology stocks are garnering the most weight, as technology stocks were driving the index in 2019. The NASDAQ, which is a tech-heavy index, had great years in 2019 and even 2020, as tech stocks, such as MSFT, which we see is calculated to be 58.04% by weight, boomed. However, stocks from the Telecom, Healthcare and Energy sector are also captured as key index trackers. This could be due to the effect of the Tariff wars and advent of 5G which affected the entire market in a similar manner as the telecom industry. Pharmaceutical companies saw similar growth in 2019 which supported market growth.

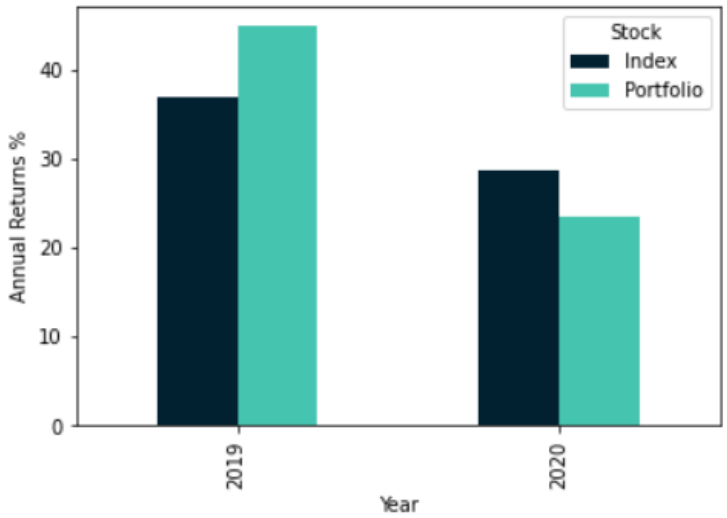
How the performance of these stocks look like in 2019



Let us also verify the performance of the portfolio using the 2020 data. The criteria for measurement of performance is the “Sum Absolute Difference” between the returns of the Index Fund and the returns of the portfolio.

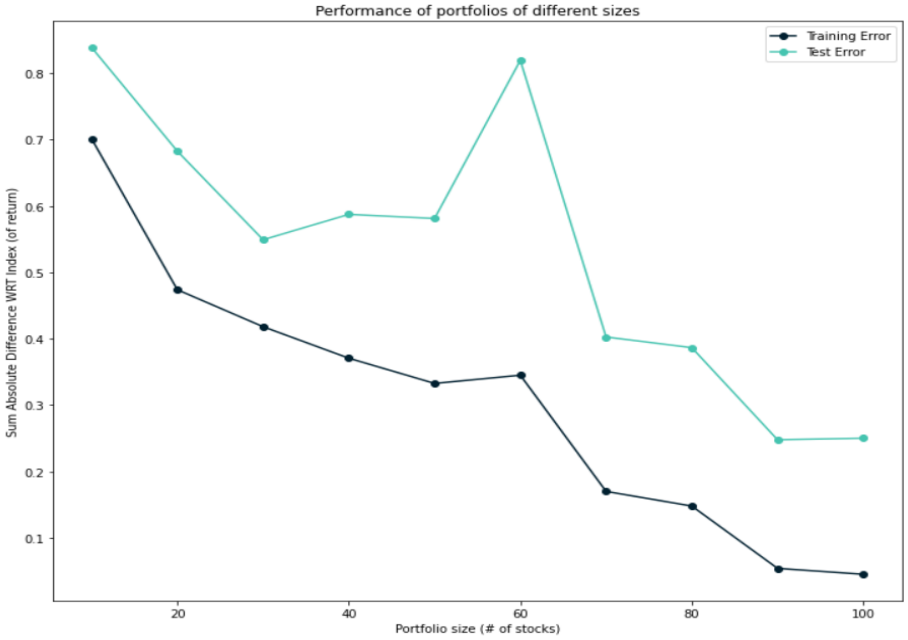


For a more intuitive understanding, let us see how well the Index and Portfolio have performed in the two years



As we can see in the graph above, the portfolio outperformed the index in 2019, but vice versa for 2020. Overall, returns across the board were higher in 2019, lower with the same distribution in 2020 but closely matching the returns of the index in both cases.

NASDAQ covers a plethora of industries such as Tech, Telecom, Energy, Healthcare, Real Estate, Consumer Staples etc. While some of these seem like they've been covered, maybe buying more stocks would give us better results. Let us see how using a different number of stocks relates to how well the portfolio matches the index.



From the previous graph, we can see that choosing more stocks in the portfolio decreases the error between the portfolio and index. However, there is a trade-off here - as a smaller size portfolio is easier to maintain, but a larger size portfolio has more accurate tracking. We would have to choose a portfolio size that is optimal for this trade-off.

Alternative Approach

An alternative approach is to consider a Mixed Integer Program, where we do not model the portfolio stocks to be representative of all the stocks, but instead, directly model the mixture of all the 100 stocks, where a particular combination of “m” stocks represent the Index as closely as possible, and the rest are forced to have a weight of 0. We can do this by considering the weight optimization from the previous approach with some extra constraints.

Objective

Minimize the sum of absolute errors over all time T for all “n” stocks.

$$min \sum_{t=1}^{T} \left| q_t - \sum_{i=1}^{n} w_i r_{it} \right|$$

Constraints

The constraints mostly remain the same, except, we are modeling for the weights of all the 100 stocks.

1. The total money allocated cannot exceed your budget (i.e.)

$$\sum_{i=1}^{n} w_i = 1$$

2. Also, none of these weights can be lower than 0 since they represent a proportion of money to be allocated. This need not be coded since Gurobi makes all variables non-negative by default.
3. Further, given that we are working with an absolute function in the objective, we need two constraints in each time period to convert it to linear form. Hence our new objective function becomes
    a. **Y** now represents the “absolute difference” between the index and portfolio performance at each time period.

$$min \sum_{t=1}^{T} y_t$$

b. Now we constrain y to be positive by forcing both sides of the absolute values to be positive

$$y_t \geq \left(q_t - \sum_{i=1}^{n} w_i r_{it} \right) \quad \text{for t=1,2,...T}$$

$$y_t \geq \left(\sum_{i=1}^{n} w_i r_{it} - q_t \right) \quad \text{for t=1,2,...T}$$

4. Additionally, since we only need “m” weights to be non-zero. We will add a big-M constraint for each weight, where the weight is non-zero only when the stock is selected. In our case, M=1 since x is binary and represents if the stock is picked or not.

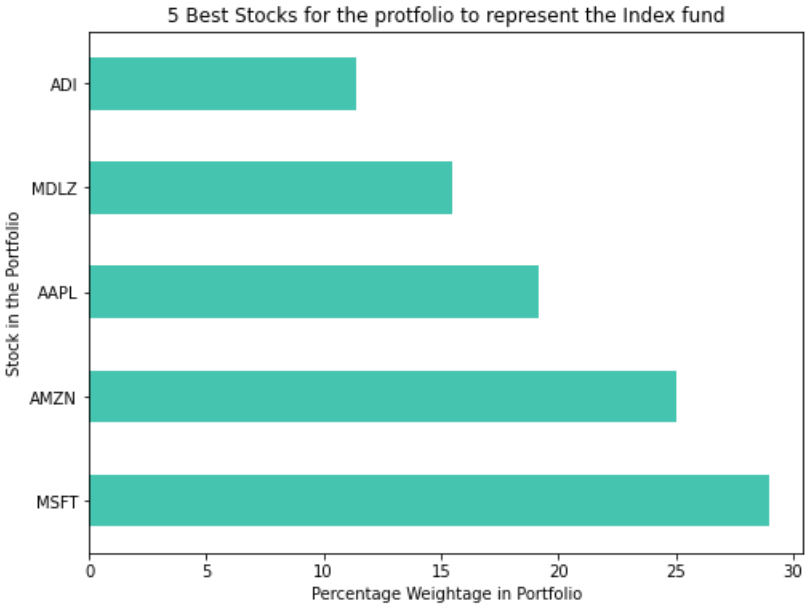
$$w_i < x_i \qquad \text{for i=1,2,...,n}$$

Findings for the Alternative Approach

Choosing 5 stocks (m=5), the best stocks for the portfolio are

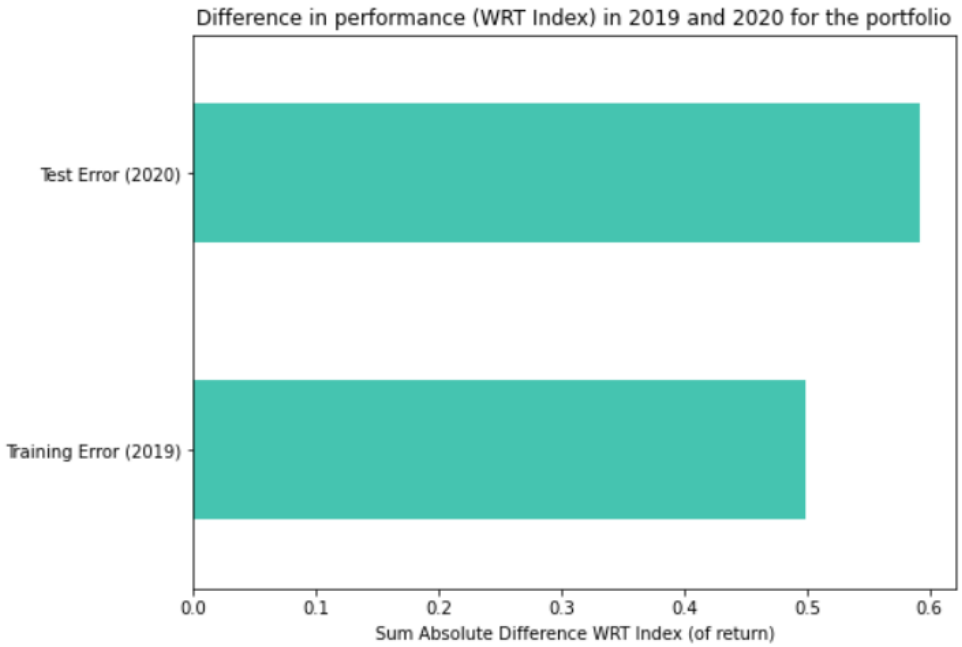
1. **AMZN** - Amazon.com, Inc - *Consumer Cyclical*
2. **ADI** - Analog Devices, Inc - *Technology*
3. **AAPL** - Apple Inc - *Technology*
4. **MSFT** - Microsoft Corporation - *Technology*
5. **MDLZ** - Mondelez International - *Consumer Defensive*

The weights (in percentages) are as follows



	weight
AMZN	25.01
ADI	11.38
AAPL	19.17
MSFT	28.99
MDLZ	15.46

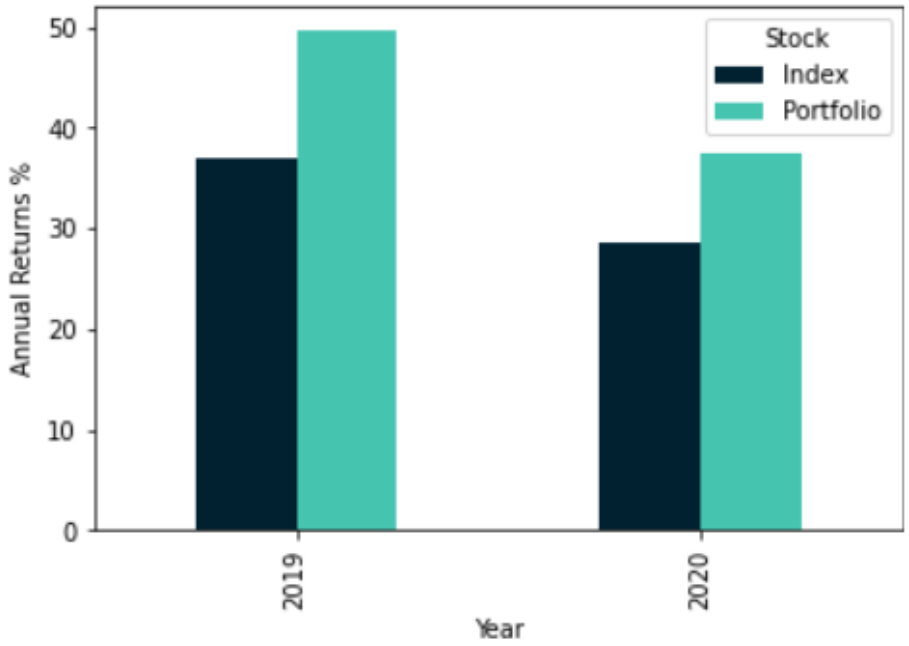
Notice how most of the stocks fall in the technology sector. This ties back to our previous hypothesis that technology stocks majorly drove the NASDAQ index in 2019. However, the new distribution is not as skewed towards technology and better distributed amongst multiple sectors.



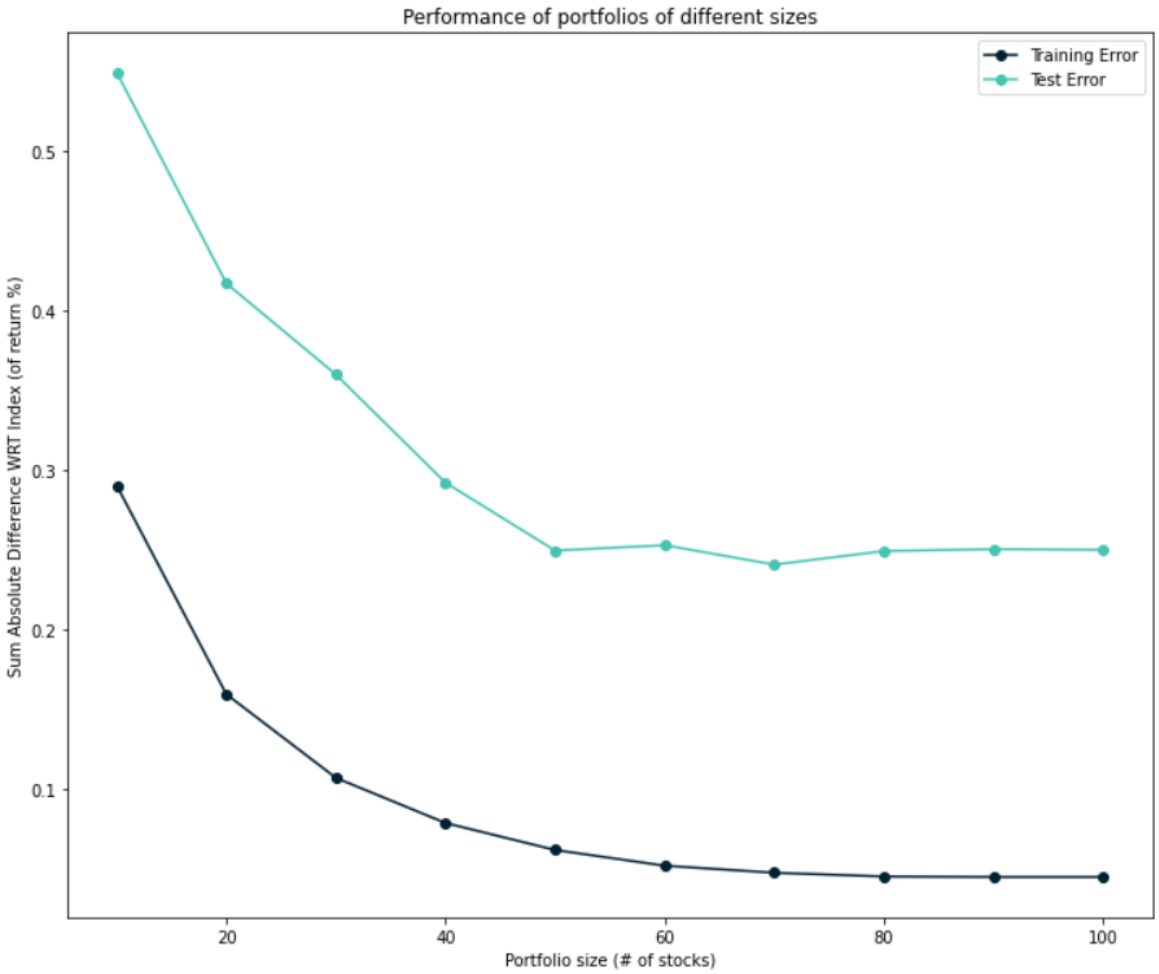
Looking at the graph above, both the training error (tracking error for 2019) and test error (tracking error for 2020) are lower than their counterparts calculated using the first method. A probable cause is we are "freeing" the stock selection criteria by not constraining them to be diversified among industries (or stocks that represent most

other stocks in the index). By “freeing” the stock selection, the technology stocks, which performed very well in 2019 and 2020, were the dominant stocks and thus, showed to be a better match to the index.

Annual returns are still close to ideal as the portfolio is aimed towards the index "drivers", i.e. the technology sector which was not as adversely affected by the pandemic as most other sectors. This means that even though overall returns were lesser, the portfolio was still able to track and slightly outperform the index in 2020 as we can see in the following graph -



Let’s study how the tracking error changes as we increase the portfolio size. Looking at the following graph which shows tracking error for both years 2019 and 2020 against the portfolio size ‘m’, the overall tracking does appear to decrease consistently with increasing number of stocks in the portfolio, which is to be expected (at least for the training sample of year 2019). However, for both the years the graph has an ‘elbow’ point following the principle of diminishing returns after which the decrease in tracking error is almost flat for increasing portfolio size. This ‘elbow’ point appears to occur around a portfolio size of 50, and persists until a portfolio size of 100.



We can choose the size of the portfolio as per the requirements and computational constraints of the business.
Moving on, we can improve upon this model selection problem by considering other approaches, modifying the question to be asked, adding/changing a few constraints and other such strategies.

Here are a few other strategies that we can consider:

1) Is the portfolio overfitting the 2019 returns? Can we add constraints that can try to generalize the model so it can fit the 2020 index better? (For eg. Add a few constraints limiting any one stock from receiving more than 35% weightage)

2) What happens if you don't select stocks at all but instead directly optimize on all the weights at once (i.e) optimize weights for the entire index? Any stocks with their weightage above a certain threshold could be added to the fund and any stocks whose weightage falls below could be removed, thus continuously updating the portfolio allocation.

Business Recommendation and Conclusion

We would first recommend a further analysis to be done, in which we would calculate the ideal portfolio size based on the flexibility we can provide. For example, is it feasible for us to switch the stocks in and out in terms of cost and frequency? If this is not the case, we would be better off choosing a smaller portfolio size, and trade-off the accuracy of following the index. However, based on what we have done, we would recommend picking a portfolio size of 50 based on a realistic time-accuracy tradeoff. If the portfolio needs to be updated very often, for eg. multiple times a day to better track the index, the first method of solving two different problems might be faster. When it comes to the size, if ease of rebalancing or minimizing transaction cost is important, a smaller size portfolio would be preferable. If the fund is focusing on passive long-term returns with minimal rebalancing, a higher number of funds using the second method of solving a mixed integer program would be preferable.

In conclusion, we formulated multiple ways to solve the stock selection problem, each with its own set of pros and cons. The business should select the method based on the specific use-case it is trying to tackle.