MA615_Final_Project

Kunyu Liu

12/10/2020

---

1 Introduction

Since the whole economic system is inextricably linked to nearly everything we do as a society, the global pandemic- COVID-19 has significantly impacted it, especially the stock market. The stock market began to recover after the first lock-down in early March 2020.

In this project, I choose stock from Nike, Procter & Gamble and The Walt Disney Company from July.1st,2020 to Dec.1st,2020. As the top appeal company, consumer goods company and entertainment and media company respectively, these three company all experience stock market crash influenced by COVID-19. I will perform stock analysis on a portfolio consisting of Nike, Procter & Gamble and The Walt Disney Company.

---

2 Data

2.1 Data Preparation

I use stock from NIKE, Procter & Gamble and The Walt Disney Company and their stock symbols are NKE, PG and DIS respectively. By using `tq_get()`, I get stock data from July.1st,2020 to Dec.1st,2020.

```
# Create a vector of stock symbols
dt_symbols <- c("NKE","PG","DIS")

# Pass symbols to tq_get to get daily prices
daily_p <- dt_symbols %>%
    tq_get(
        get = "stock.prices",
        from = "2020-07-01", to = "2020-12-02")
```

2.2 Data Exploration

2.2.1 Closing Price

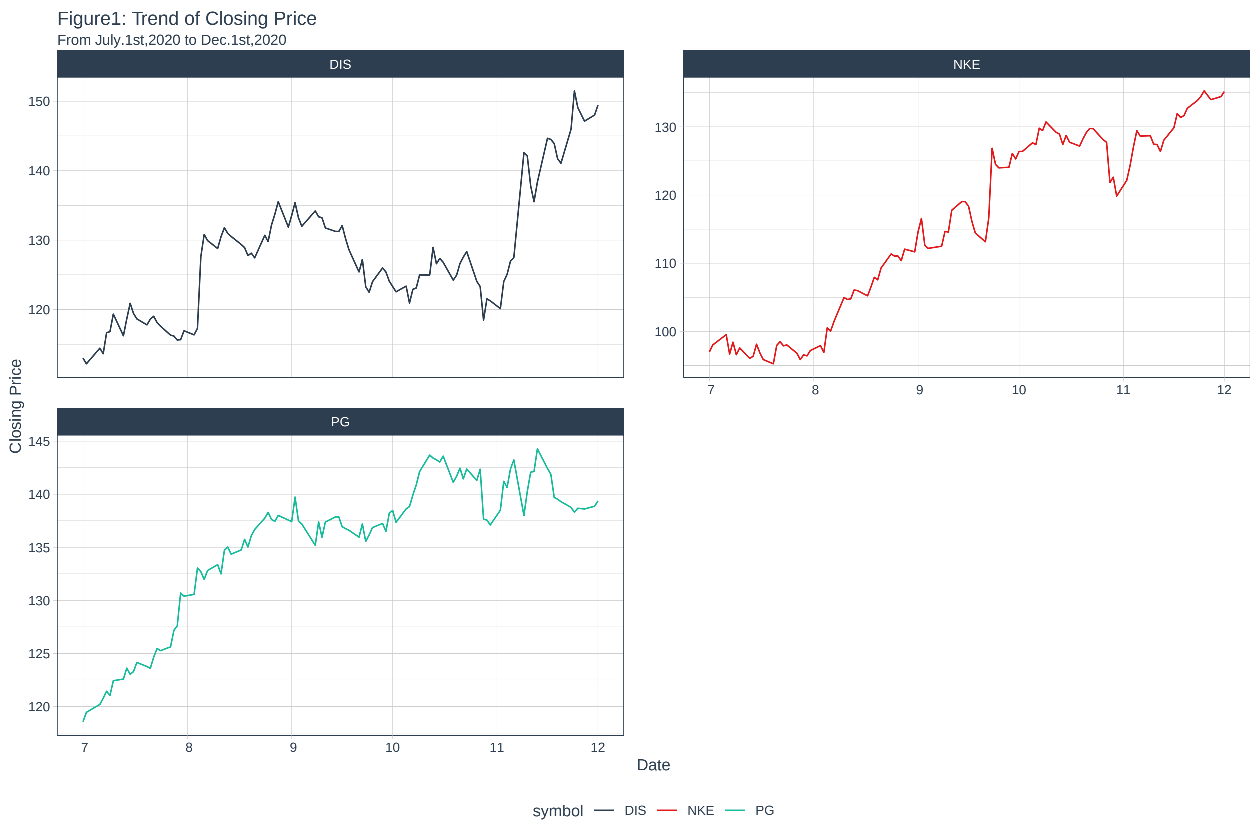
First, I do a summary to find the minimum, maximum and mean closing price for these three stocks.

```r
summary_p <- daily_p %>%
  select(symbol,date,close) %>%
  group_by(symbol) %>%
  summarise(`Min Closing Price` = min(close),
            `Max Closing Price` = max(close),
            `Mean Closing Price` = mean(close))
summary_p2 = as.data.frame(summary_p)
head(summary_p2)
```

```
##   symbol Min Closing Price Max Closing Price Mean Closing Price
## 1    DIS            112.18            151.49           127.6914
## 2    NKE             95.65            135.54           116.0765
## 3     PG            119.98            144.49           136.0289
```

Then, I visualize the trend of closing price from July.1st,2020 to Dec.1st,2020.

```r
daily_p %>%
    ggplot(aes(x = date, y = adjusted, color = symbol)) +
    geom_line() +
    facet_wrap(~ symbol, ncol = 2, scales = "free_y") +
    theme_tq() +
    scale_color_tq() +
    labs(title = "Figure1: Trend of Closing Price",
         subtitle = "From July.1st,2020 to Dec.1st,2020",
         x = "Date",y="Closing Price")
```



Based on the results above, we can find that all of these three stocks' closing price have upper trend and the closing price of Disney shows big fluctuation.

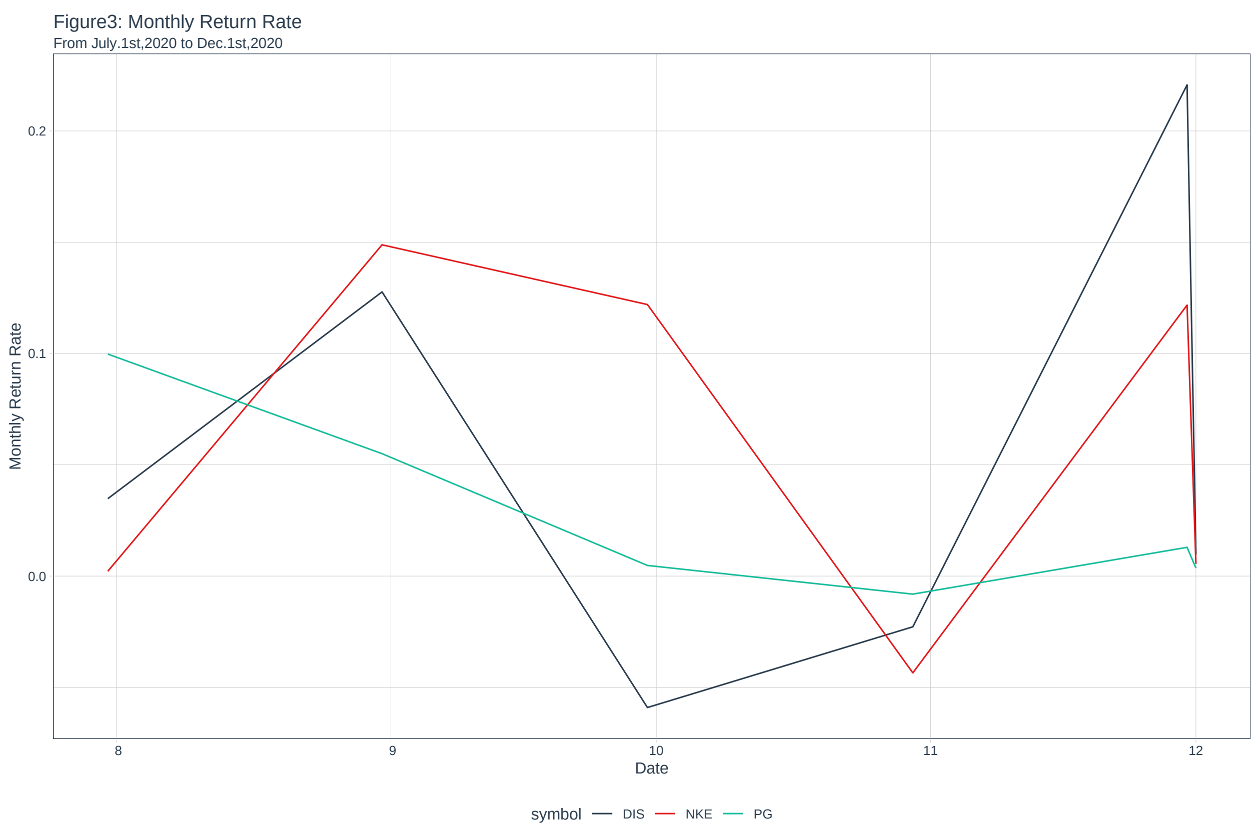
2.2.2 Monthly Return

I change the daily data of these three stocks into monthly data by using `tq_transmute` function, to calculate their monthly return.

```
# Compute monthly return rate
month_r <- daily_p %>%
    group_by(symbol) %>%
    tq_transmute(select     = adjusted,
                 mutate_fun = periodReturn,
                 period     = "monthly",
                 col_rename = "monthly_return_rate")
```

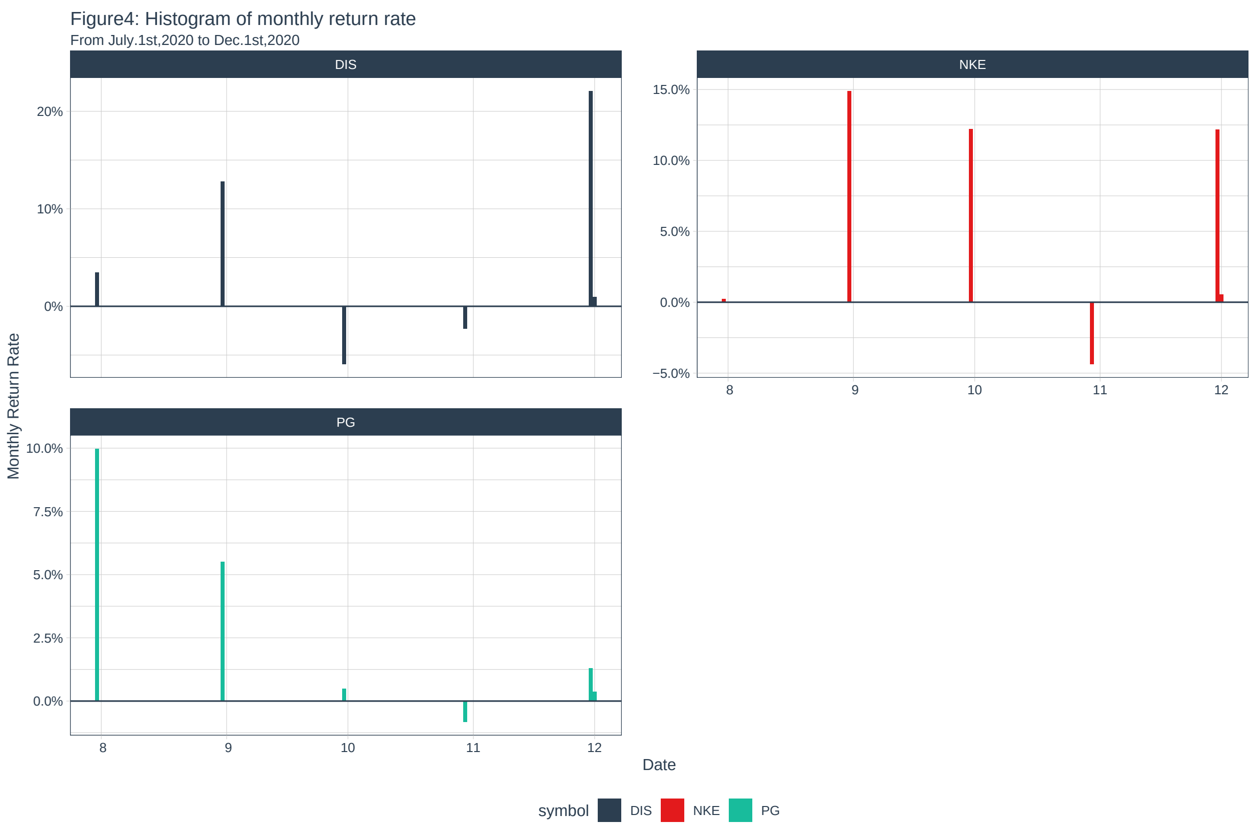
Then, we visualize the trend of these three stocks' monthly return rate.

```
# Visualize: Monthly Return Rate
month_r %>%
  ggplot(aes(x = date, y = monthly_return_rate, color = symbol)) +
    geom_line() +
    theme_tq() +
    scale_color_tq() +
    labs(title = "Figure3: Monthly Return Rate",
         subtitle = "From July.1st,2020 to Dec.1st,2020",
         x = "Date",y="Monthly Return Rate")
```




```
# Histogram of monthly return rate
month_r %>%
    ggplot(aes(x = date, y = monthly_return_rate, fill = symbol)) +
    geom_col() +
```

```
    geom_hline(yintercept = 0, color = palette_light()[[1]]) +
    scale_y_continuous(labels = scales::percent) +
    labs(title = "Figure4: Histogram of monthly return rate",
         subtitle = "From July.1st,2020 to Dec.1st,2020",
         x = "Date",y="Monthly Return Rate") +
    facet_wrap(~ symbol, ncol = 2, scales = "free_y") +
    theme_tq() +
    scale_fill_tq()
```



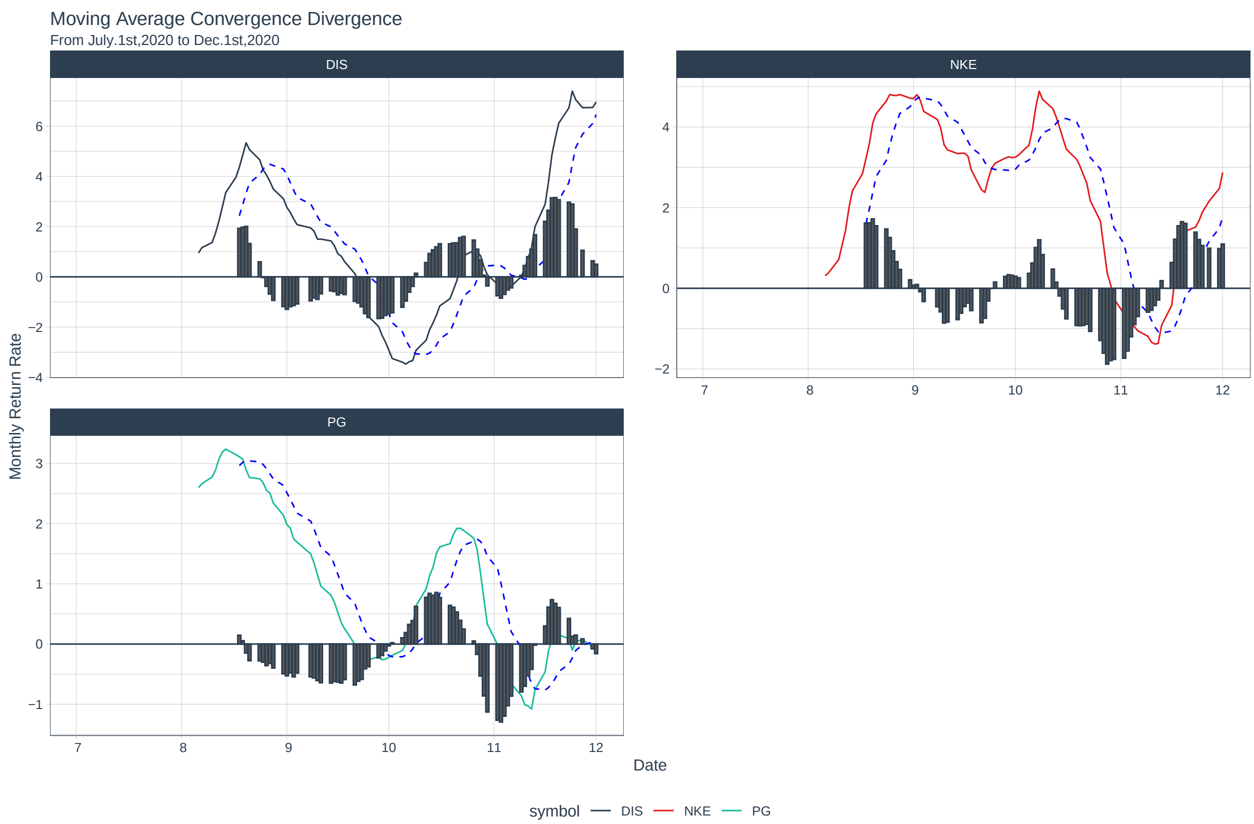
From the two plot above, we can conclude that Disney has the highest monthly return rate in late November, and in the start of December, there is not much difference in the return rate among these three stocks.

2.2.3 Moving Average Convergence Divergence

I also use Moving Average Convergence Divergence, which can show the relationship between two moving averages of a security's price. I use the function `tq_mutate()`, MACD requires a price, so I select close.

```
macd <- daily_p %>%
    group_by(symbol) %>%
    tq_mutate(select     = close,
              mutate_fun = MACD,
              nFast      = 12,
              nSlow      = 26,
              nSig       = 9,
              maType     = SMA) %>%
    mutate(diff = macd - signal) %>%
    select(-(open:volume))
```

```r
macd %>%
    ggplot(aes(x = date)) +
    geom_hline(yintercept = 0, color = palette_light()[[1]]) +
    geom_line(aes(y = macd, col = symbol)) +
    geom_line(aes(y = signal), color = "blue", linetype = 2) +
    geom_bar(aes(y = diff), stat = "identity", color = palette_light()[[1]]) +
    facet_wrap(~ symbol, ncol = 2, scale = "free_y") +
    labs(title = "Moving Average Convergence Divergence",
        subtitle = "From July.1st,2020 to Dec.1st,2020",
         x = "Date",y="Monthly Return Rate") +
    theme_tq() +
    scale_color_tq()
```



Based on the MACD, Disney and Nike are crossing above zero in November, which are considered bullish, while PG is crossing below zero, which is considered bearish. Thus, I choose to buy more proportion of Disney and Nike and less proportion of PG. MACD has the guiding significance to the stock exchange transaction, but there is a lag when the stock market is not stable.

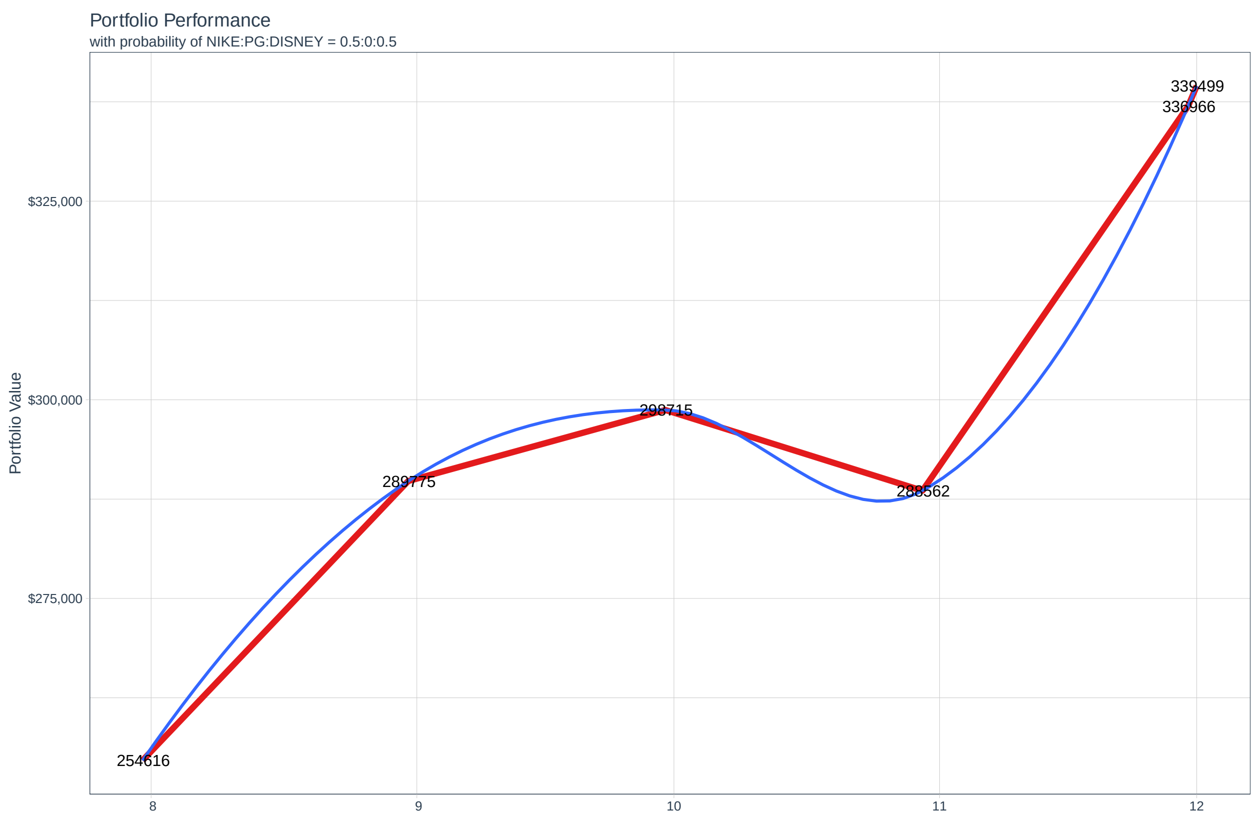
3 Results

3.1 Evaluating Portfolio Performance

```r
start <- 250000
prob1 <- c(0.5,0,0.5) # Compute investment result with probability of NIKE:PG:DISNEY = 0.5:0:0.5
month_g1 <- month_r %>%
```

```
    tq_portfolio(assets_col   = symbol,
                 returns_col  = monthly_return_rate,
                 weights      = prob1,
                 col_rename   = "investment.growth",
                 wealth.index = TRUE) %>%
    mutate(investment.growth = investment.growth * start)
```

```
# Visualize the performance
month_g1 %>%
    ggplot(aes(x = date, y = investment.growth)) +
    geom_line(size = 2, color = palette_light()[[2]]) +
    labs(title = "Portfolio Performance",
         subtitle = "with probability of NIKE:PG:DISNEY = 0.5:0:0.5",
         x = "", y = "Portfolio Value") +
    geom_smooth(method = "loess") +
    geom_text(aes(label = round(investment.growth)),nudge_x=0.1,nudge_y=0.1)+
    theme_tq() +
    scale_color_tq() +
    scale_y_continuous(labels = scales::dollar)
```



Aggregate Portfolio Returns for Multiple Portfolios

```
multi_r <- month_r %>% tq_repeat_df(n = 3)
```

Create a portfolio weight table as the weight I set in 3.1

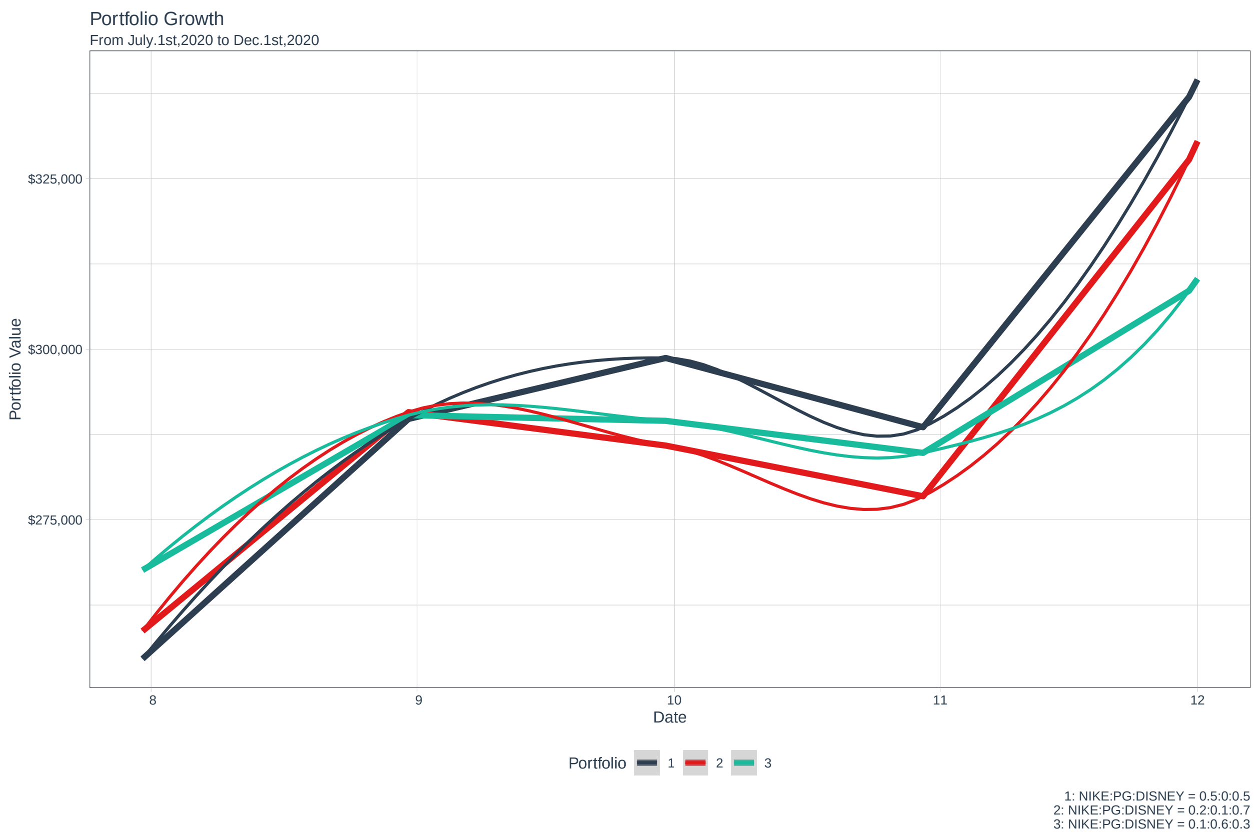
```
weights <- c(
    0.50, 0, 0.5,
    0.2, 0.1, 0.7,
```

```
    0.1, 0.6, 0.3
)

weights_table <-  tibble(dt_symbols) %>%
    tq_repeat_df(n = 3) %>%
    bind_cols(tibble(weights)) %>%
    group_by(portfolio)
```

```
multi_growth <- multi_r %>%
    tq_portfolio(assets_col   = symbol,
                 returns_col  = monthly_return_rate,
                 weights      = weights_table,
                 col_rename   = "investment_growth",
                 wealth.index = TRUE) %>%
    mutate(investment_growth = investment_growth * start)
```

```
# # Visualize the portfolio performance
multi_growth %>%
    ggplot(aes(x = date, y = investment_growth, color = factor(portfolio))) +
    geom_line(size = 2) +
    labs(title = "Portfolio Growth",
         subtitle = "From July.1st,2020 to Dec.1st,2020",
         caption = "1: NIKE:PG:DISNEY = 0.5:0:0.5\n2: NIKE:PG:DISNEY = 0.2:0.1:0.7\n3: NIKE:PG:DISNEY =
         x = "Date", y = "Portfolio Value",
         color = "Portfolio") +
    geom_smooth(method = "loess") +
    theme_tq() +
    scale_color_tq() +
    scale_y_continuous(labels = scales::dollar)
```



The above plots suggests that a larger percentage of investment in Disney and Nike will bring a higher portfolio value than put PG as the main investment.

3.2 Evaluating Portfolio Performance by using baseline

A baseline is helpful in doing portfolio analysis, in this project, I choose Netflix stock (NFLX) as the baseline.

First, use `tq_get` function to get the baseline prices

```
nflx <- c("NFLX") %>%
    tq_get(
        get = "stock.prices",
        from = "2020-07-01", to = "2020-12-02")
```

Next, mutate to returns

```
nflx_r <- nflx %>%
    group_by(symbol) %>%
    tq_transmute(select     = adjusted,
                 mutate_fun = periodReturn,
                 period     = "monthly",
                 col_rename = "nflx_return")
```

Now pass the the expanded `multi_r` and the `weights_table` to `tq_portfolio` for portfolio aggregation.

```
portfolio_aggregation <- multi_r %>%
    tq_portfolio(assets_col  = symbol,
                 returns_col = monthly_return_rate,
```

```r
                  weights      = weights_table,
                  col_rename   = "multi_return")
```

Merge with the baseline using "date" as the key.

```r
combine_dt = left_join(portfolio_aggregation, nflx_r, by = "date")
```

I use `tq_performance` to convert investment returns into performance metrics and set performance_fun equal to SharpeRatio, measuring the return of an investment compared to its risk.

```r
combine_dt %>%
    tq_performance(Ra = multi_return, Rb = nflx_return, performance_fun = SharpeRatio)
```

```
## # A tibble: 3 x 4
## # Groups:   portfolio [3]
##   portfolio `ESSharpe(Rf=0.8%,p=9~ `StdDevSharpe(Rf=0.8%,~ `VaRSharpe(Rf=0.8%,p~
##       <int>                  <dbl>                   <dbl>                 <dbl>
## 1         1                  0.697                   0.593                 0.843
## 2         2                  0.563                   0.519                 0.700
## 3         3                  0.816                   0.641                 0.863
```

Based on the Sharpe Ratio result, portfolio 3 has the highest Sharpe Ratio, which means the better a fund's returns have been relative to the risk it has taken on.

---

4 Reference

1. Wickham et al., (2019). Welcome to the tidyverse. Journal of Open Source Software, 4(43), 1686, https://doi.org/10.21105/joss.01686
2. Matt Dancho and Davis Vaughan (2020). tidyquant: Tidy Quantitative Financial Analysis. R package version 1.0.2. https://CRAN.R-project.org/package=tidyquant
3. Jeroen Ooms (2019). curl: A Modern and Flexible Web Client for R. R package version 4.3. https://CRAN.R-project.org/package=curl