Final Project

Peter Sullivan

2021-05-02

Introduction

Covid has hit the world in many strange and unexpected ways. Recently the stock market has been very volatile. Gamestop stock is rising up to 400 a share from previous shares below 10 dollars. A failing video game store, that sells hard copy video games saw prices increase exponently. There are stories about college students making millions of dollars from investing using apps like Robinhood.

Due to the ease of entering the market through free online platforms, there's a whole new market of investors that are adding a new and unexpected force to the market.

For this paper, I am going to look into the best way to compare stocks! As there are many many different ways to compare stocks, I'm going to focus on a few key points that I find useful.

Data

Instead of using pre gathered data, I've decided to gather the data my self. To get up to date stock data, I'm using the package tidyquant to gather data from Yahoo Finance.

I've set up a vector named stocks. This vector contains multiple stocks with with very different listed prices. For example, Google currently has a price of $2300 and MTNB (a pharmaceutical company) currently has a price of 90 cents.

##Disclaimer## Due to the fact that the data is constantly being pulled every time this report is run. There is a chance that the descriptions below the figures may have different amounts then what are currently in the figures. I plan to write inline code in the future that will automatically update every time the report is run.

The Following stocks I will be looking at are : MSFT, GOOG, AAPL, AMZN, FB, MTNB, LUV, CSCO, GLMD, RIOT, MRNA, CO.

Updating Dates

When running this report, I use multiple different date filters that will depend on the current date of running the report. For example I am pulling 4 years of data starting in 2017. When looking at the data, for my daily and monthly returns, I may only care about 1 month of data or even 6 months of data. To do this, I will need to filter the date column based on today's date. To make this process easier and capable of being run at any date and pull fresh information. I've created date objects are are dependent on the current date. The current date is acquired using sys.date().

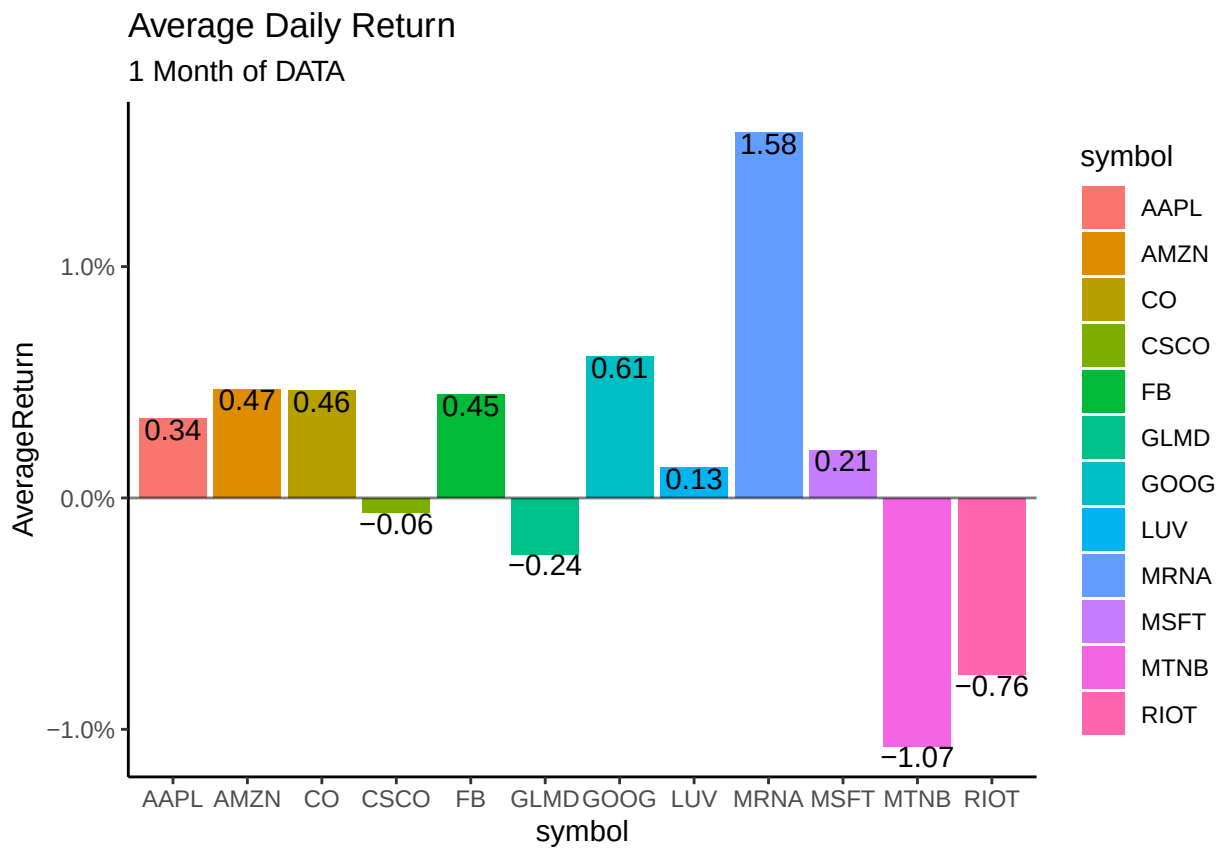
```
Today = Sys.Date()
month1_date = Today - months(1)
month6_date = Today - months(6)
Year_4_date = Today - years(4)
```

Mutating DATA

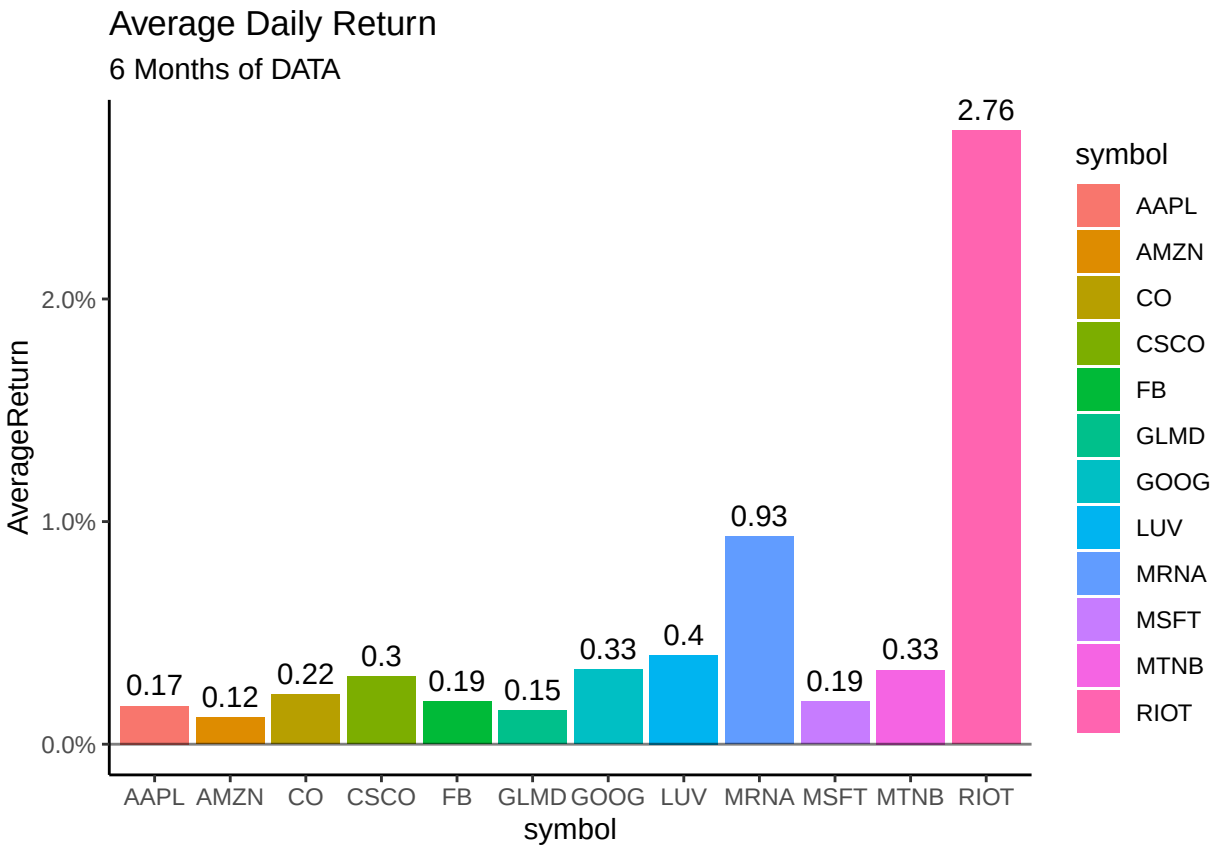
Right now I have the following variables: symbol, date, open, high, low, close, volume, adjusted. Using Tq_transmute, I will use the adjusted price to calculate the daily, weekly and monthly returns. This will allow me to look at the % returns and be useful when comparing stocks that have such different prices. I also wanted a good way to compare stocks that have very different prices. As mentioned above, Google has a price of 2300 dollars per share, while MTNB has a price of 90 cents per share. If we were to compare the stocks side by side, we could do a free y scale for the y axis, but I would question whether that is actually a good representation. I don't believe looking at the stocks side by side with a free scale is a fair representation of comparing stocks. We will look further into that later on!

Daily Returns

```
# Calculating the Daily Returns
dailyReturns <- data %>%
  group_by(symbol) %>%
  tq_transmute(select = adjusted, mutate_fun = periodReturn, period = "daily", type = "arithmetic")

dailyReturns %>%
  filter(date >= month1_date)%>%
  summarise(AverageReturn = mean(daily.returns))%>%
ggplot()+
  geom_col(aes(x = symbol, y = AverageReturn, fill = symbol))+
   geom_hline(yintercept = 0, alpha =.5)+
  geom_text(aes(x = symbol, y = AverageReturn, label = ((round(AverageReturn,4))*100)),vjust = 1)+
  scale_y_continuous(labels = scales::percent)+
  theme_classic()+
  labs(title = "Average Daily Return",
       subtitle = "1 Month of DATA")
```




```
dailyReturns %>%
  filter(date >= month6_date)%>%
  summarise(AverageReturn = mean(daily.returns))%>%
ggplot()+
  geom_col(aes(x = symbol, y = AverageReturn, fill = symbol))+
   geom_hline(yintercept = 0, alpha =.5)+
  geom_text(aes(x = symbol, y = AverageReturn, label = (round(AverageReturn,4))*100),vjust = -.5)+
  scale_y_continuous(labels = scales::percent)+
  theme_classic()+
  labs(title = "Average Daily Return",
       subtitle = "6 Months of DATA")
```

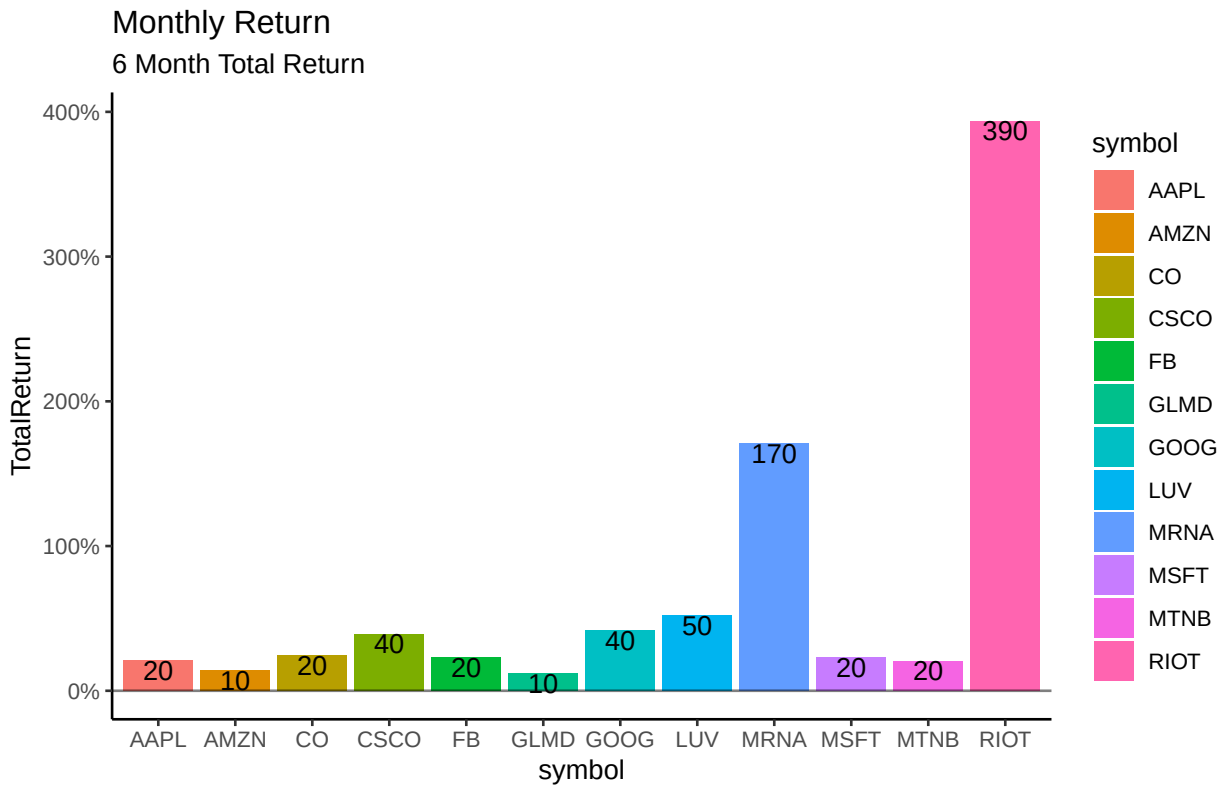


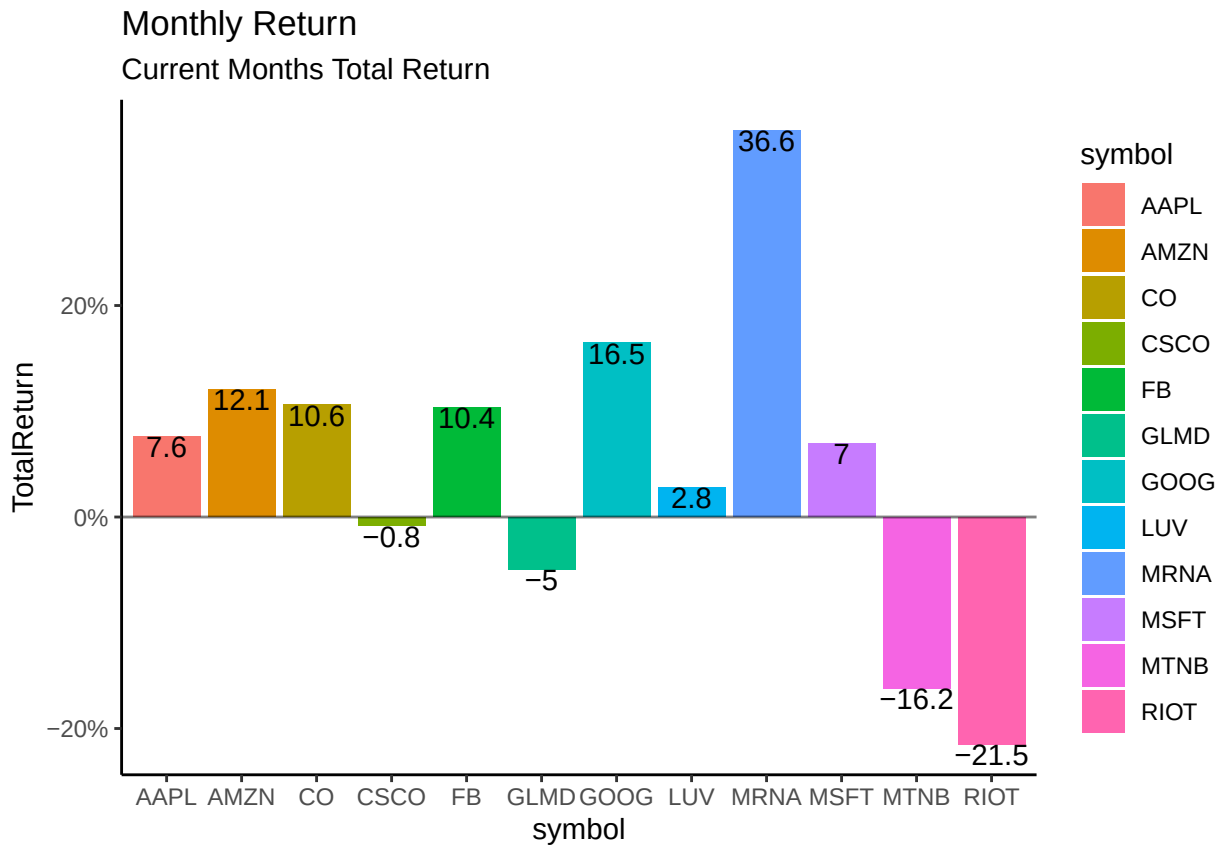
The two figures above show the average daily return for two date filters, one month and 6 months. When looking at the average daily return for one month of data, MRNA and GOOG look like good investments. MTNB And RIOT are the lowest average return of -1.07% and -.076%.

When looking at the average from 6 months of data RIOT and MRNA have the highest average return of .93% and 2.76% respectively.

Monthly Returns

```
# Calculating the Monthly Returns

MonthlyReturns <- data %>%
  group_by(symbol) %>%
  tq_transmute(select = adjusted, mutate_fun = periodReturn, period = "monthly", type = "arithmetic")%>%
  mutate(month_1 = month(date),
         Month = case_when(
    month_1 == 1 ~ "Jan",
    month_1 == 2 ~ "Feb",
    month_1 == 3 ~ "Mar",
    month_1 == 4 ~ "Apr",
    month_1 == 5 ~ "May",
    month_1 == 6 ~ "Jun",
    month_1 == 7 ~ "Jul",
    month_1 == 8 ~ "Aug",
    month_1 == 9 ~ "Sep",
    month_1 == 10 ~ "Oct",
    month_1 == 11 ~ "Nov",
    month_1 == 12 ~ "Dec"
  ))

#Creating Factors
MonthlyReturns$Month <- factor(MonthlyReturns$Month, levels = c("Jun","Jul","Aug","Sep","Oct",
                                                                "Nov","Dec","Jan", "Feb",
                                                                "Mar","Apr","May"))
MonthlyReturns %>%
  filter(date >= month6_date)%>%
  group_by(symbol)%>%
  summarise(TotalReturn = sum(monthly.returns))%>%
ggplot()+
  geom_col(aes(x = symbol, y = TotalReturn, fill = symbol))+
   geom_hline(yintercept = 0, alpha =.5)+
  geom_text(aes(x = symbol, y = TotalReturn, label = ((round(TotalReturn,1))*100)),vjust = 1)+
  scale_y_continuous(labels = scales::percent)+
  theme_classic()+
  labs(title = "Monthly Return",
       subtitle = "6 Month Total Return")
```

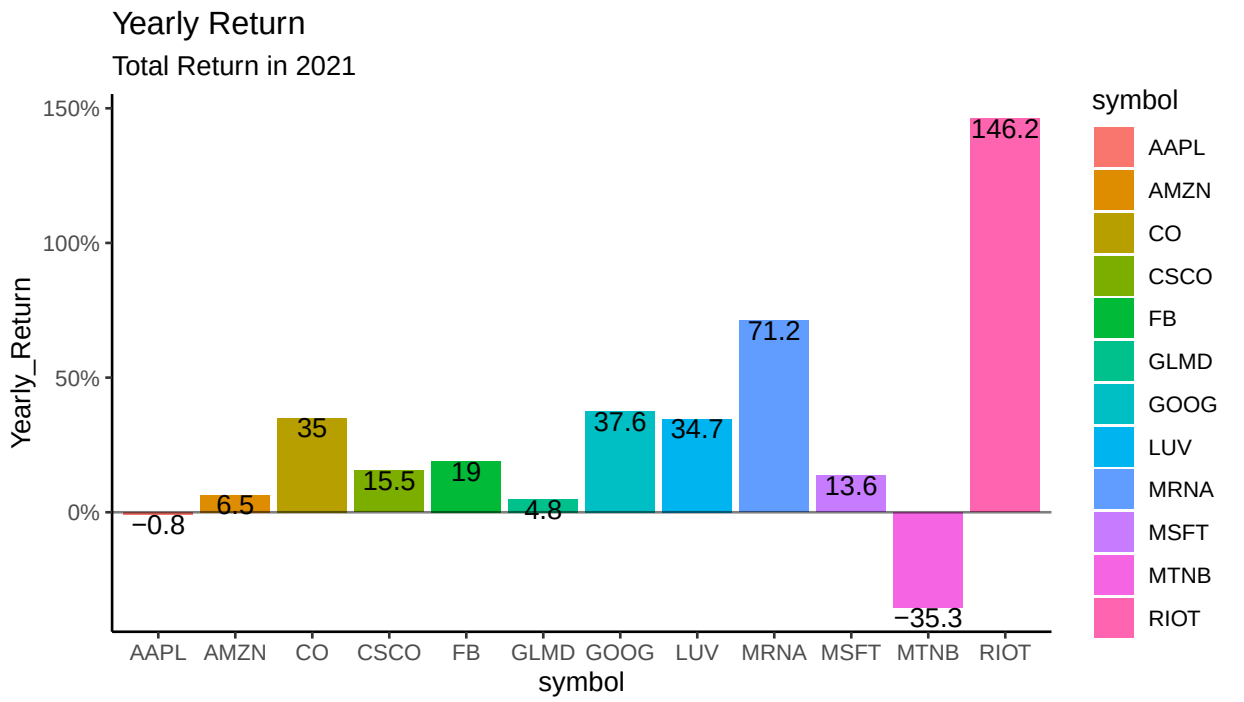

Monthly Return
6 Month Total Return
TotalReturn
400%
300%
200%
100%
0%
20
10
20
40
20
10
40
50
170
20
20
390
AAPL
AMZN
CO
CSCO
FB
GLMD
GOOG
LUV
MRNA
MSFT
MTNB
RIOT
symbol
symbol
AAPL
AMZN
CO
CSCO
FB
GLMD
GOOG
LUV
MRNA
MSFT
MTNB
RIOT


Monthly Return
Current Months Total Return
TotalReturn
20%
0%
−20%
7.6
12.1
10.6
−0.8
10.4
−5
16.5
2.8
36.6
7
−16.2
−21.5
AAPL
AMZN
CO
CSCO
FB
GLMD
GOOG
LUV
MRNA
MSFT
MTNB
RIOT
symbol
symbol
AAPL
AMZN
CO
CSCO
FB
GLMD
GOOG
LUV
MRNA
MSFT
MTNB
RIOT

I performed similar date filters for the two figures above, 1 month and 6 month. Instead of doing averages, I decided to summarize the monthly return. Above we can see that looking from the six month perspective, MRNA and RIOT have had large gains gains of 170% and 390% respectively. When looking at this current months return, MRNA and Google have the highest return of 36.6% and 16.5%. As expected MTNB and RIOT have the lowest return of -16.2% and -21.5%.

Yearly Return

```
yearlyReturns<- data %>%
  group_by(symbol) %>%
  tq_transmute(select = adjusted, mutate_fun = periodReturn, period = "yearly",
               type = "arithmetic")%>%
  mutate(Year = year(date))


yearlyReturns %>%
  filter(date >= "2021-01-01")%>%
  group_by(symbol)%>%
  rename(Yearly_Return = yearly.returns)%>%
ggplot()+
  geom_col(aes(x = symbol, y = Yearly_Return, fill = symbol))+
  geom_hline(yintercept = 0, alpha =.5)+
  geom_text(aes(x = symbol, y = Yearly_Return, label = (round(Yearly_Return,3))*100),vjust = 1)+
  scale_y_continuous(labels = scales::percent)+
  theme_classic()+
  labs(title = "Yearly Return",
       subtitle = "Total Return in 2021")
```



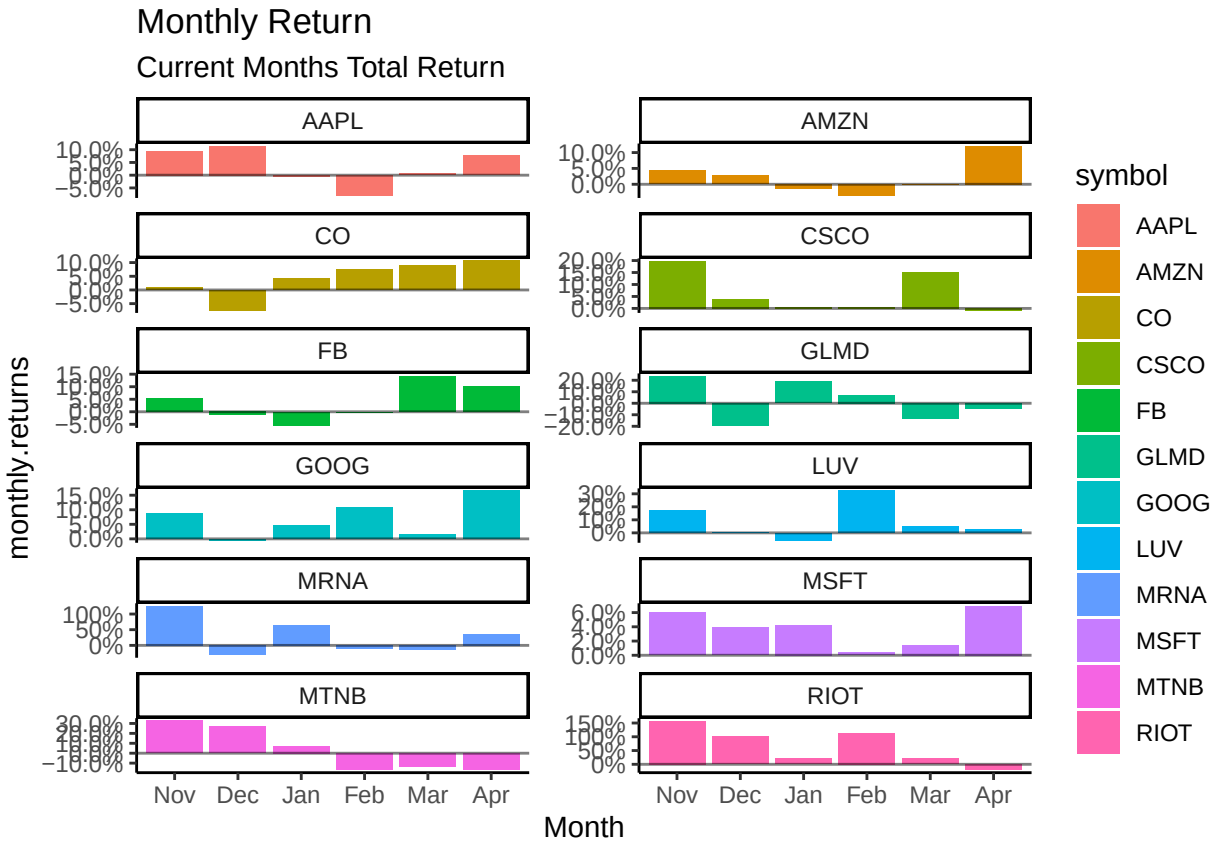
The figure above shows the total return for one year of data. It seems the best investments this year were RIOT, MRNA, and GOOG. The worst investments were MTNB and AAPL.

Looping through my Stocks

I've decided that I also want to view my stocks using facet wrap on company symbol. Right now I have 12 stocks that I'm analyzing. When I attempt to facet by company I end up getting something that is unreadable. In order to get this process done, in a timely manner I first need to break my stock vector into three parts, 4 groups of 3. After that I used the list function to group the stock groups into a list. I know have a list with three items and each item contains four stocks.

Now I can create a for loop that can go pull out the 4 stocks in each stock group and plot them in my ggplot that I created. I then used the facet command and the corresponding plots are much easier to understand!

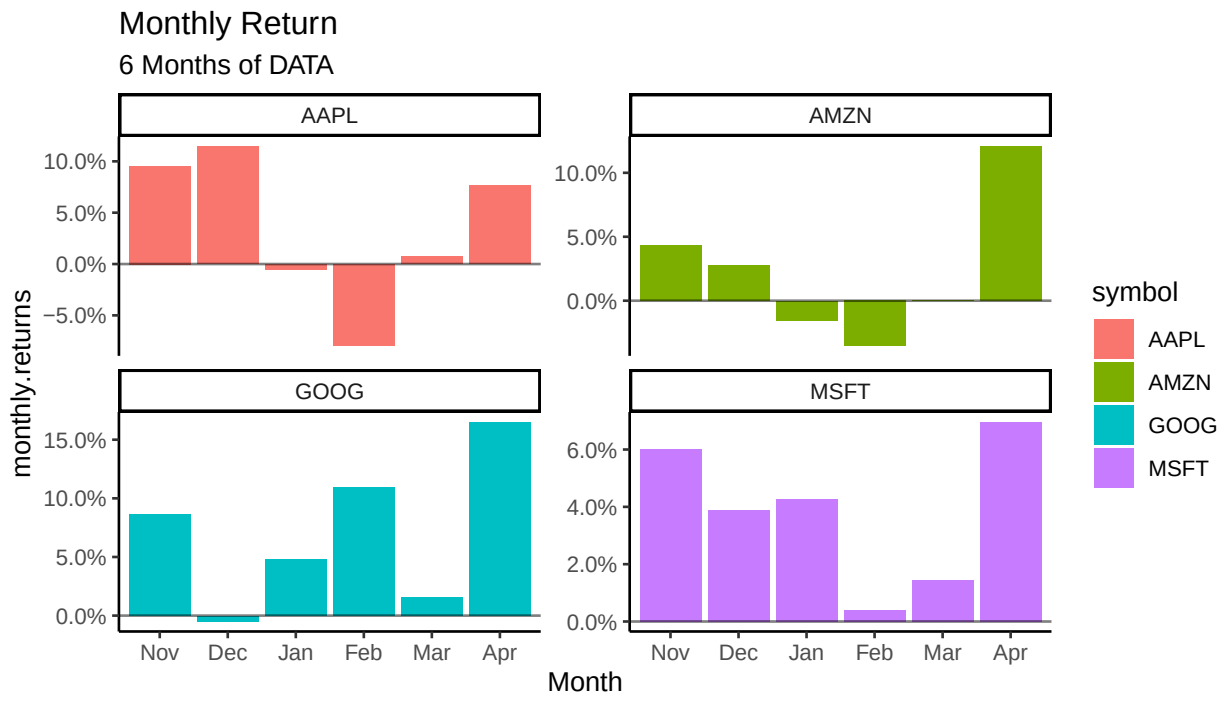
In the future I will figure out a way where I don't need to break up my list before the for loop. Ideally it would make the most sense to have a loop that would pull every four items. At this point in time, I do not know how to do this.

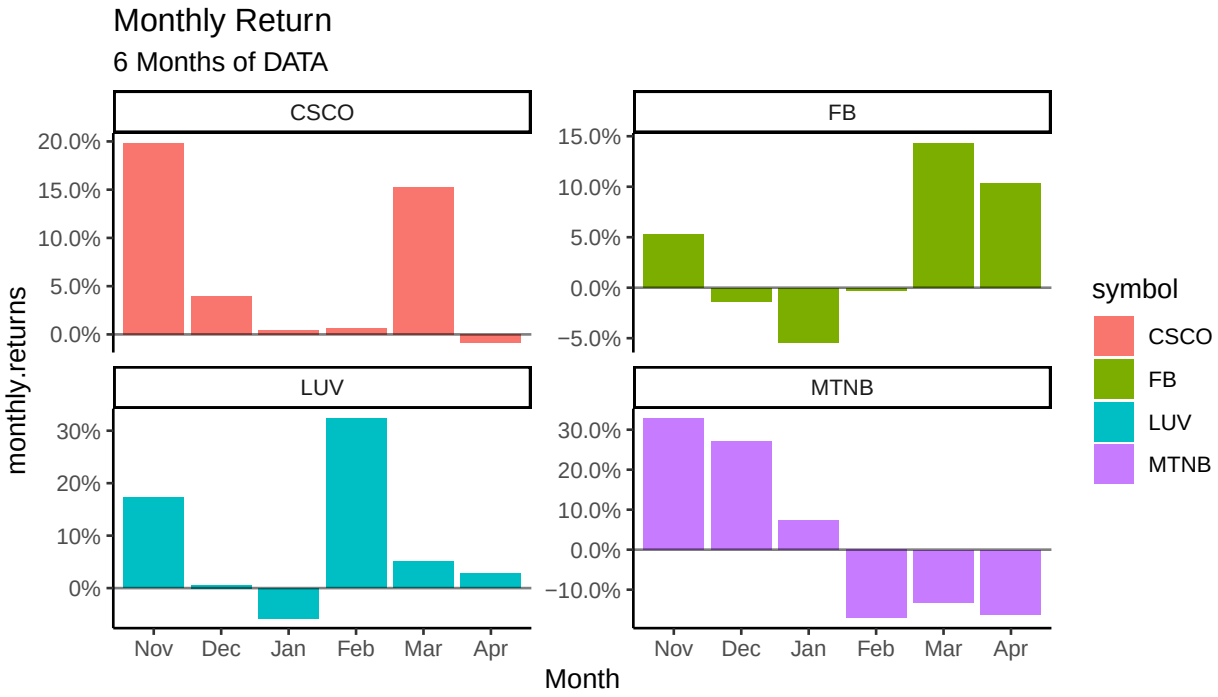


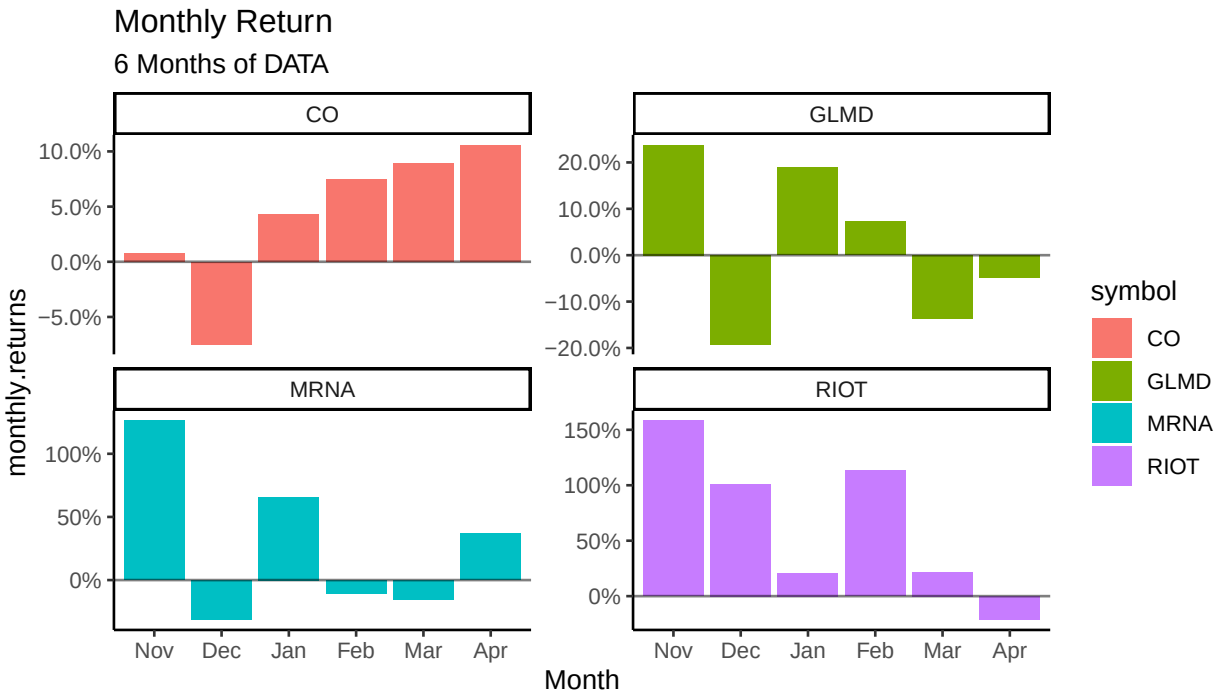
The figure above shows a facet using all 12 companys. It's quite hard to understand and not useful. Below I will break the stocks in into 3 groups of four and plot each group using a for loop.

,

```
stocks1 <- stocks[1:4]
stocks2 <- stocks[5:8]
stocks3 <- stocks[9:12]



Stock_block <- list(stocks1, stocks2, stocks3)



for (stock_group in Stock_block){
x <- MonthlyReturns %>%
  filter(symbol %in% stock_group)%>%
  filter(date >= month6_date)%>%
ggplot()+
  geom_col(aes(x = Month, y = monthly.returns, fill = symbol),
           position = position_dodge2(width = 1, preserve = "single"))+
   geom_hline(yintercept = 0, alpha =.5)+
  scale_y_continuous(labels = scales::percent)+
  theme_classic()+
  labs(title = "Monthly Return",
       subtitle = "6 Months of DATA")+
    facet_wrap(~symbol, ncol = 2, scales = "free_y")
print(x)
}
```

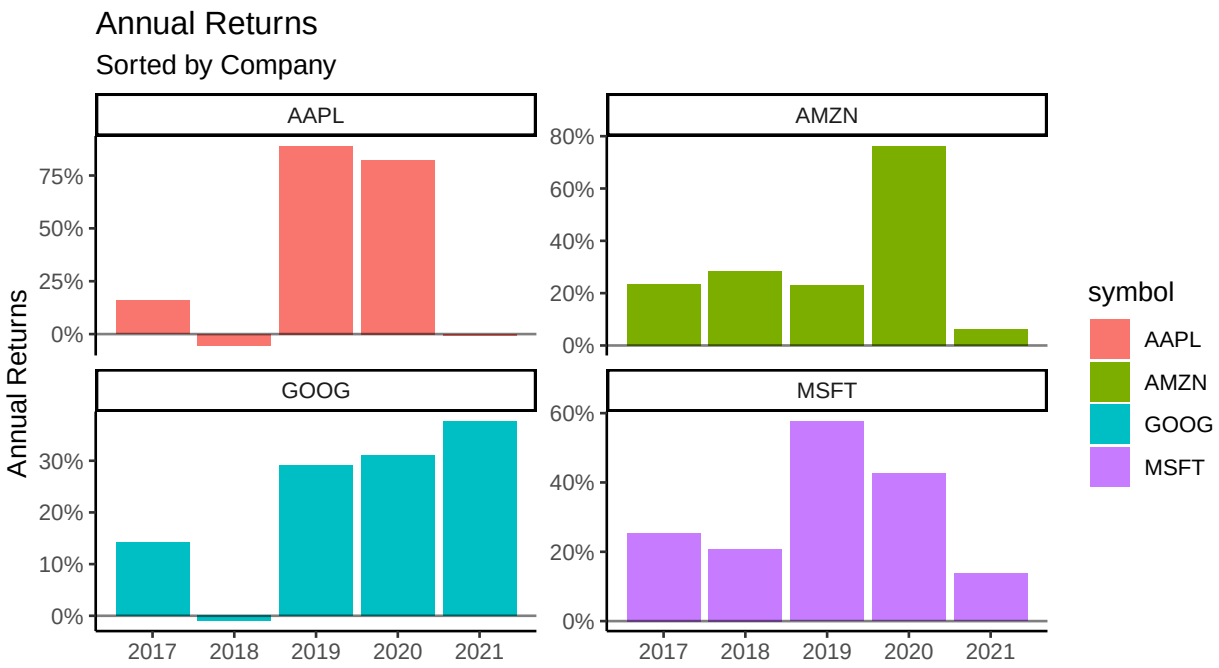


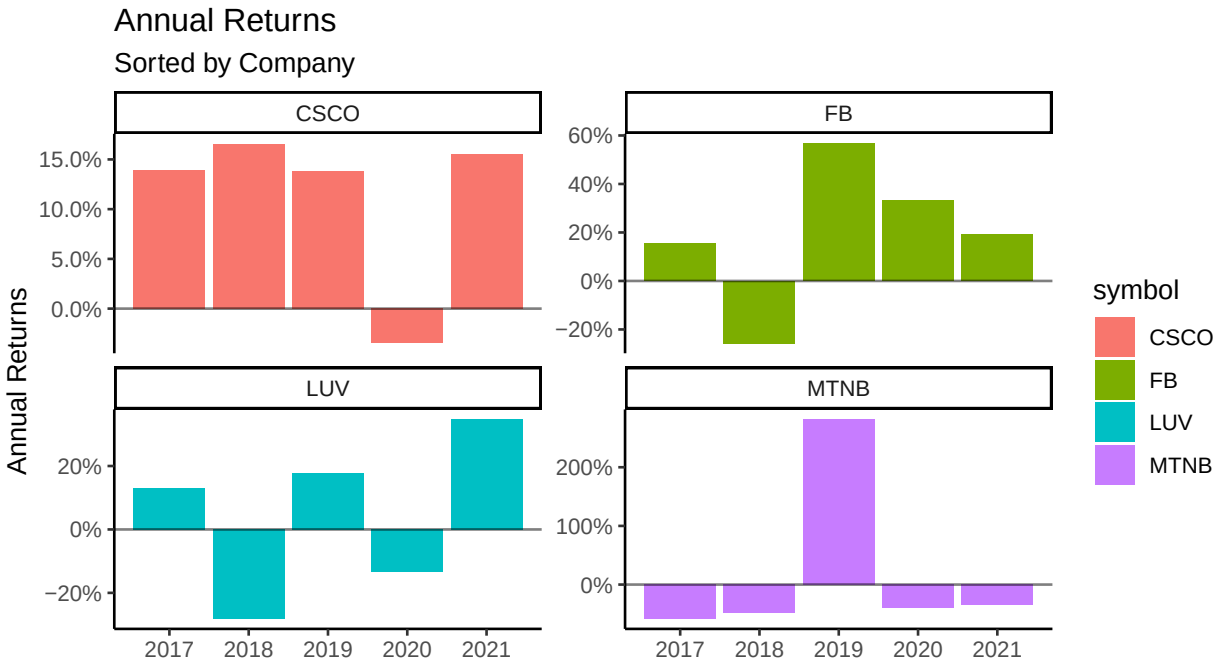


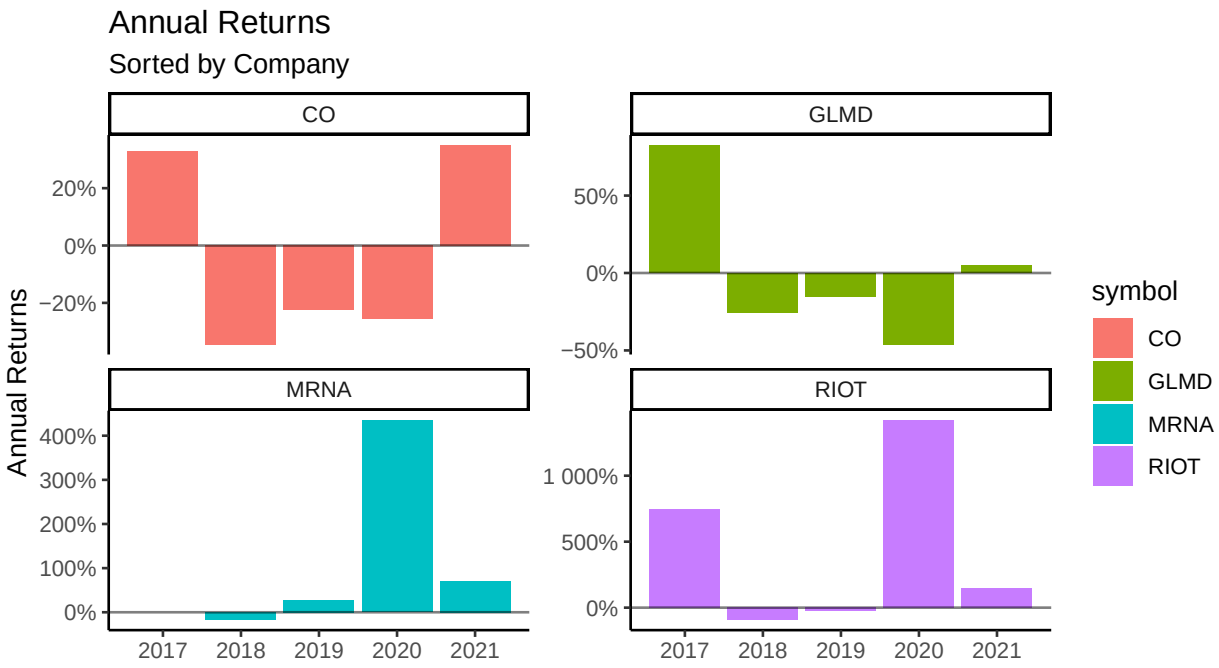



```
for (stock_group in Stock_block){
x <- yearlyReturns%>%
    filter(symbol %in% stock_group)%>%
  ggplot()+
  geom_col(aes(x = Year, y = yearly.returns, fill = symbol))+
  geom_hline(yintercept = 0, alpha = .5)+
  scale_y_continuous(labels = scales::percent)+
  labs(title = "Annual Returns",
```

```
       subtitle = "Sorted by Company",
       y = "Annual Returns", x = "")+
  facet_wrap(~symbol, ncol = 2, scales = "free_y")+
  theme_classic()
print(x)
}
```







Results

The first 3 figures are the monthly Returns faceted by company for 6 months of data. These charts are much easier to understand. The last three figures are the annual returns for faceted by company for the last 4 years.

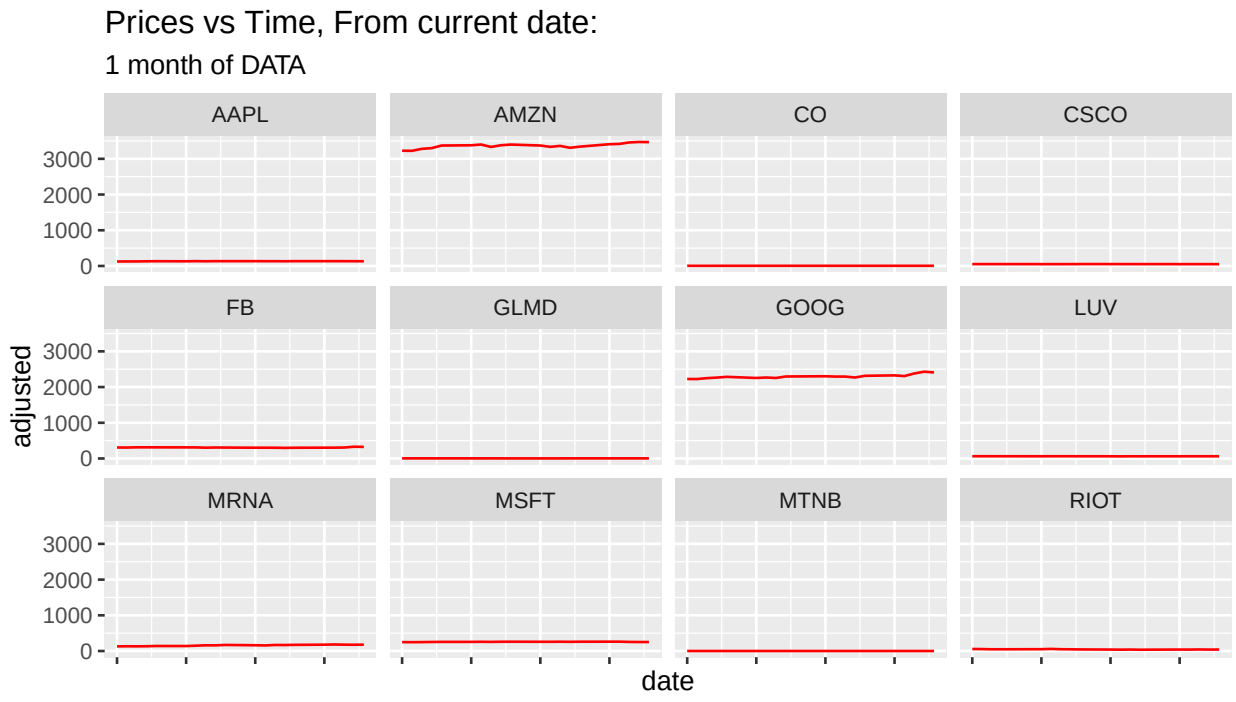
CO and MSFT seem to be on a trend for growing monthly returns. Google seems to be the only company with increasing returns every year since 2019.

Final look

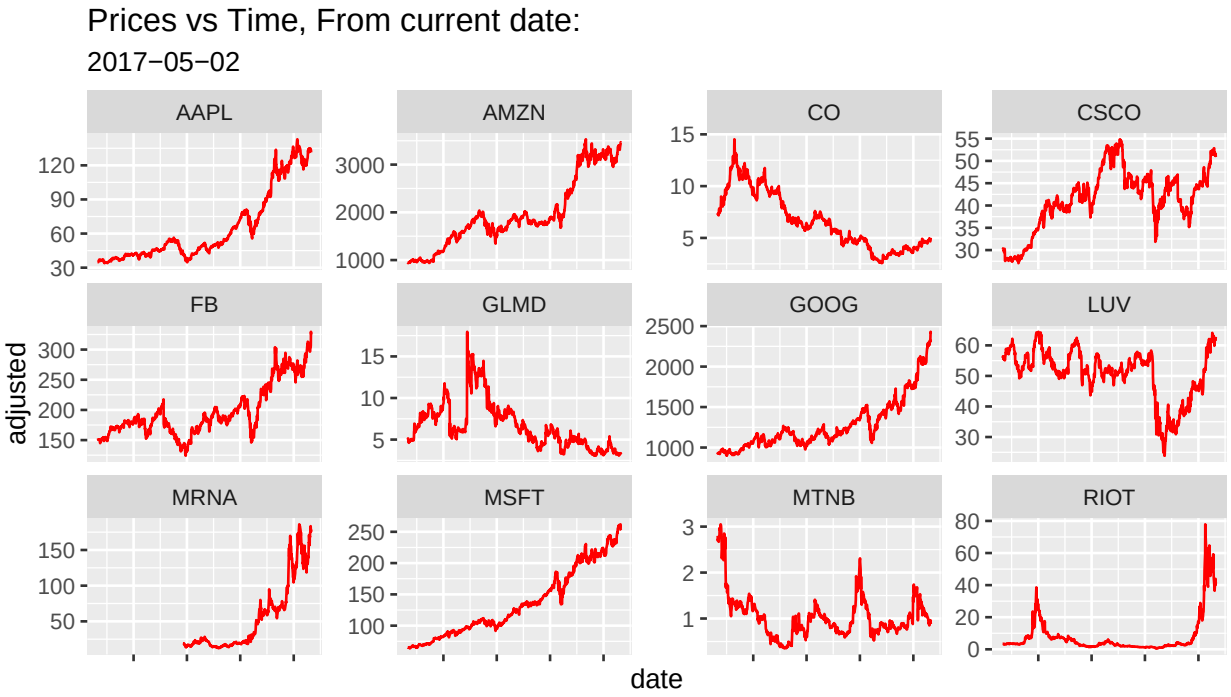
The last attempt I will use to observe the stocks is based on the price trends. Using the for loop and stock groups I will facet through the stocks 4 at a time and compare the prices. It can be seen from below that even when using free scale, or not using free scale, there's no easy way to compare stocks with such different price differences.

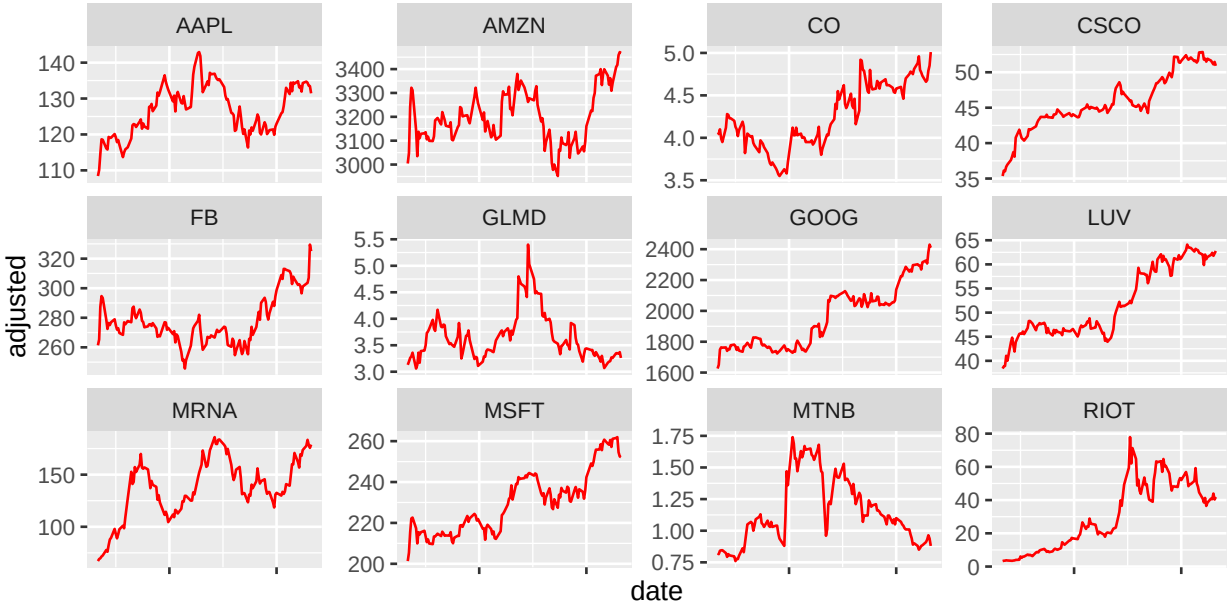
To come up with a way to fix this, I decided to normalize the prices themselves.

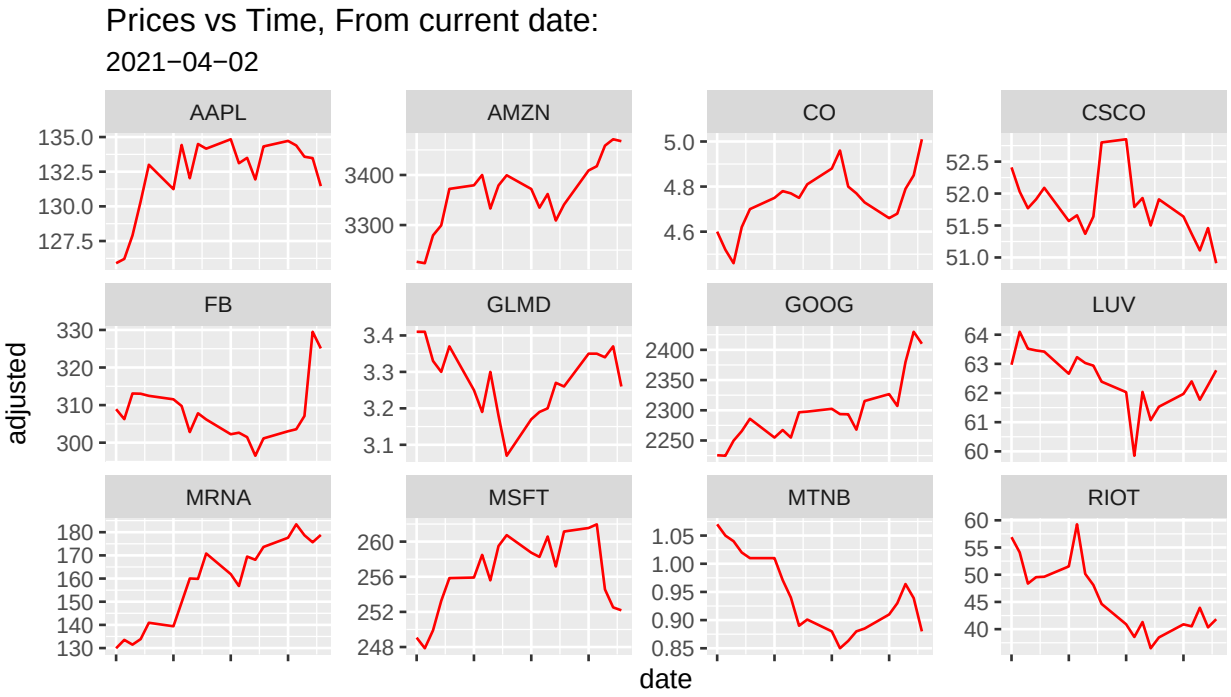
```
multiple_dates <- rev(list(month1_date, month6_date, Year_4_date))

data %>%
   filter(date >= month1_date)%>%
  ggplot()+
  geom_line(aes(x = date, y = adjusted),color = "red")+
  facet_wrap(~symbol)+
    theme(axis.text.x = element_blank())+
    labs(title = "Prices vs Time, From current date: ",
         subtitle = "1 month of DATA")
```




```
for (dates in multiple_dates){
print(data %>%
   filter(date >= dates)%>%
  ggplot()+
  geom_line(aes(x = date, y = adjusted),color = "red")+
  facet_wrap(~symbol, scales = "free_y")+
    theme(axis.text.x = element_blank())+
    labs(title = "Prices vs Time, From current date: ",
         subtitle = dates))
}
```


Prices vs Time, From current date:
2017−05−02
AAPL
AMZN
CO
CSCO
FB
GLMD
GOOG
LUV
MRNA
MSFT
MTNB
RIOT
adjusted
date


Prices vs Time, From current date:
2020−11−02
AAPL
AMZN
CO
CSCO
FB
GLMD
GOOG
LUV
MRNA
MSFT
MTNB
RIOT
adjusted
date



The very first figure shows the price vs date for 1 month of data faceted by the company. As you can tell, this is impossible to compare the stocks side by side due to google and amazon having such high stock prices.

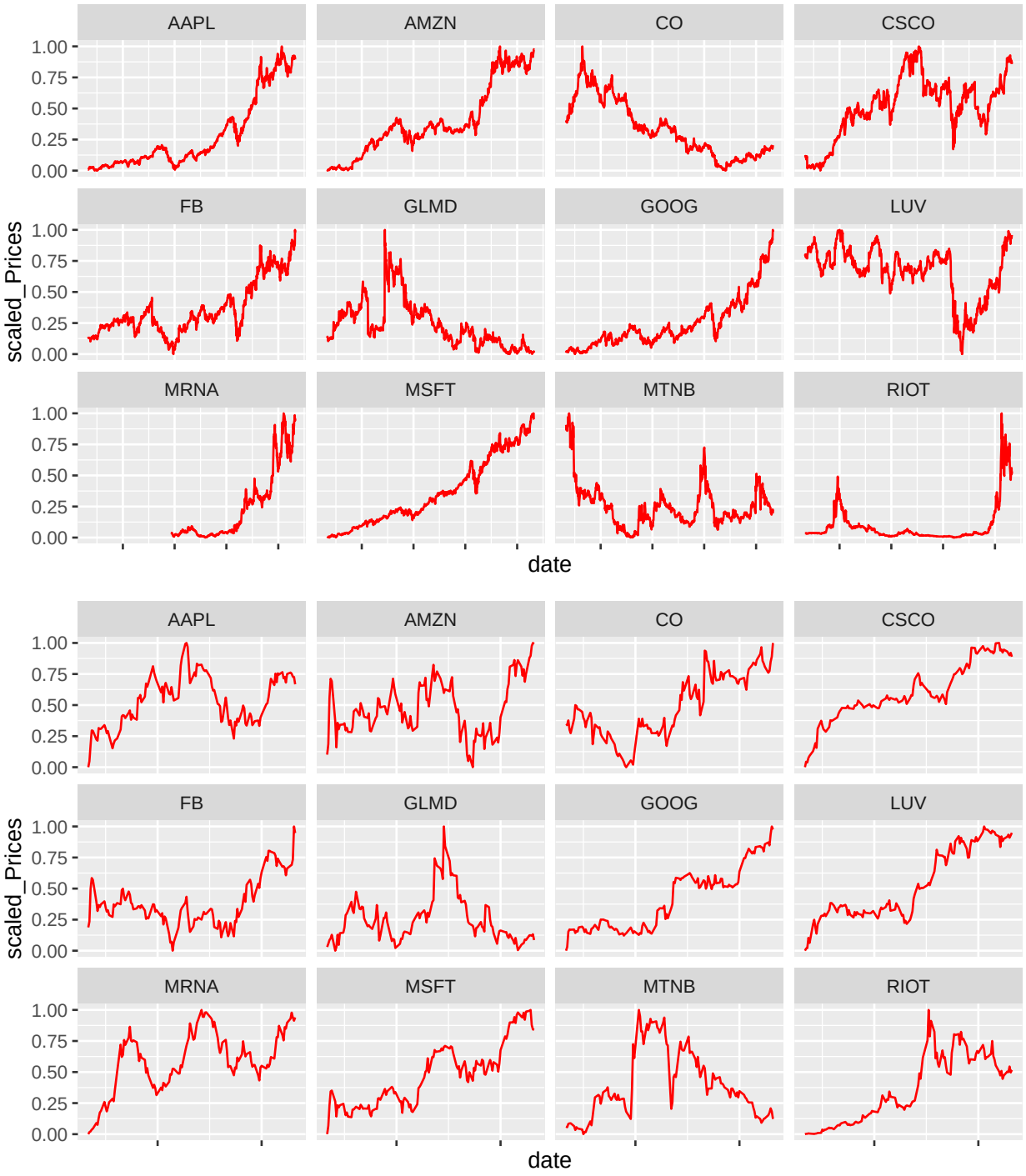
The last three figures show the price vs time for three different time periods, 1 month, 6 months and 4 years. I used the free scale for Y. It does look like we are able to compare the stocks side by side when we use a free scale. I do question the validity of the free scale. In order to get a true representation, I will normalize the prices.

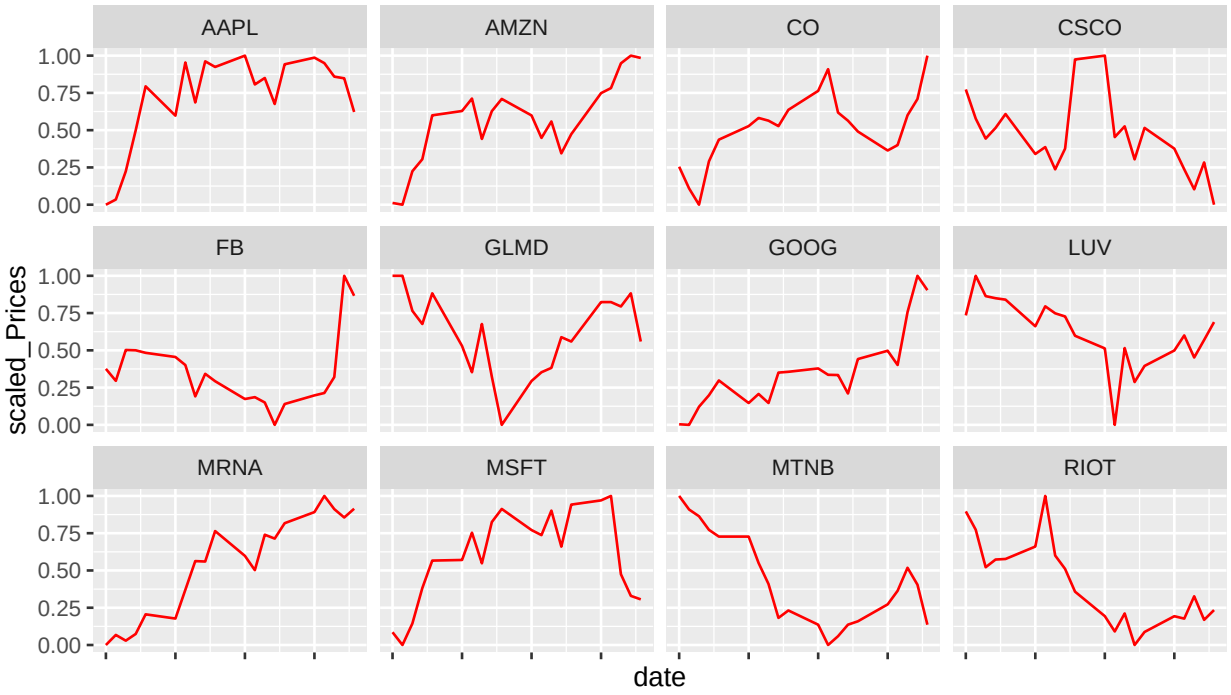
To Normalize the prices, I created a fucntion called "normalize_function", and I applied it to the adjusted price. What's fantastic about R is that I can run a for loop, normaize the price data, and plot the graphs all in one line of code!

Here is the normalize function:

Normalize = (x - min(x)/(max(x)- min(x)))

```
normalize_funciton <- function(x){
  return((x - min(x))/(max(x)- min(x)))
}

for (dates in multiple_dates){
print(data %>%
        filter(date >= dates)%>%
 group_by(symbol)%>%
  mutate(scaled_Prices = normalize_funciton(adjusted))%>%
   ggplot()+
  geom_line(aes(x = date, y = scaled_Prices),color = "red")+
  facet_wrap(~symbol)+
    theme(axis.text.x = element_blank()))

}
```


AAPL
AMZN
CO
CSCO
FB
GLMD
GOOG
LUV
MRNA
MSFT
MTNB
RIOT
scaled_Prices
1.00
0.75
0.50
0.25
0.00
date



Results

Amazingly, the scaled prices look exactly the same when compared to the free scale y figures from above. It looks like it was not needed to normalize the prices. Though it does look better on the y axis when when the y axis labels are the same. If I was using the free scaly y figures, I would blank out the y axis labels.

Conclusion

Sadly, I may have taken on too much to handle on this project. There are so many ways to look at each stock. If you are a short term investor, than the short term plots filtered by 1 month may be more important then the figures with 6 months or 4 years of data. If one is planning on holding the stocks for longer than 1 year, perhaps the 6 months and 4 years of data will be of more use. Other date filters may be more useful as well. Returns and Price trends seem like they can be useful, but I am unable to currently say which stocks are the best to invest in. I did find this project very useful though! I was able to properly create and apply a function to my data frame. I was also able to create and use a for loop that helped me develop multiple plots with out haivng to write out more code.

This project was a lot of fun, and there is much room for improvement. In the future, I plan on using tidyquant to create more indicators and plot those side by side to help determine the best stock to invest in.