Trade Rule Backtesting

Niraj Sardar

Introduction

An important aspect of trading is knowing when to buy and sell in the market. There are many ways to do this. The first is by back testing. By looking at past data, analysts can predict how a company will perform in the future. This will dictate when they should buy and sell that stock. Another is through the Direction Volume Index (DVI). This is a predictor for S&P 500 over the past 30 years. DVI takes the magnitude of returns over different time windows smoothed and the relative number of up versus down days into account over different time windows. From this it can be used as a trend indicator. Finally, traders need to be aware of signaling. Signaling is the trigger for when to buy or sell stocks in the market. Trade signaling also helps modify portfolios by determining when it is a good time to buy more of a particular sector or offload more of another sector.

In the following code and analysis the team will develop 3 functions that only need inputs of ticker name, begin date and end date for the back testing, and threshold for DVI. The first function has a specific default constant DVI of 0.5. This function will be the baseline for the remaining function as it establishes an output summary that includes: total number of long trades, total number of short trades, percentage of time that the portfolio is long, the percentage of time that the portfolio is short, and the cumulative return from the strategy. We identify a long portfolio versus a short by applying the rule of DVI less than 0.5 as long, and a DVI greater than 0.5 as short. The second function applies the first function, but is able to simulate the back testing for multiple back test periods, given an input number of testing periods and date range. This second function will provide the average cumulative return across each iteration, while still using a constant DVI of 0.5. The third final function will allow us to simulate multiple DVI thresholds using the inputs of the low threshold, high threshold, and increment step size.

Load necessary packages

Performance Analytics Package - “Collection of econometric functions for performance and risk analysis. In addition to standard risk and performance metrics, this package aims to aid practitioners and researchers in utilizing the latest research in analysis of non-normal return streams”

https://cran.r-project.org/web/packages/PerformanceAnalytics/index.html (https://cran.r-project.org/web/packages/PerformanceAnalytics/index.html)

```
library(quantmod)             # for return calculations
library(TTR)                  # for signals
library(dplyr)                # for data management
library(PerformanceAnalytics) # for return metrics
library(lubridate)
library(ggplot2)
```

Base level Back Testing

Trading strategies are important to obtain profits.A simple understanding of the concept is that if a strategy has done well in the past it is most likely going to do well in the future and vice verse. When a back testing is conducted correctly it yeild the user or trader a strategy that is going to yeild a positive output or profit.

```
main <- function(ticker, fromDate, toDate, defaultDvi) {

fromDate <- as.Date(fromDate , format = '%Y%m%d');
toDate <- as.Date(toDate , format = '%Y%m%d');

Price <- getSymbols(ticker, auto.assign = F, from = fromDate, to = toDate ,periodicity = "daily")

Price = Price[,4]
Retn <- periodReturn(Price, period = 'daily', type = 'arithmetic')
Retn <- na.omit(Retn)

dvi <- DVI(Price)
dvi <- na.omit(dvi)

Position <- ifelse(dvi$dvi < defaultDvi, 1, -1)
lagPosition <- Lag(Position)
lagPosition <- na.omit(lagPosition)

dfxts <- merge(Price, Retn, dvi$dvi,Position, lagPosition)
dfxts <- na.omit(dfxts)
colnames(dfxts) <- c("Price", "Retn", "dvi", "Position", "lagPosition")
df1 <- data.frame(Date = index(dfxts), coredata(dfxts))
df1 <- na.omit(df1)
df3 <- df1 %>% mutate( LongOrShort  =  ifelse( df1$Position == -1 & df1$lagPosition ==  1 , "Long" , ifelse(
    df1$Position == 1 & df1$lagPosition ==  -1 ,"Short" ,0) ))


totalNoOfLongTrades <- length(which(df3$LongOrShort=='Long'))
totalNoOfShortTrades <- length(which(df3$LongOrShort=='Short'))
percntTimeLong <- (length(which(df3$Position=='1'))/ length(df3$Position))*100
percntTimeShort <- (length(which(df3$Position=='-1'))/ length(df3$Position))*100
return_strategy = Retn*lagPosition
df <- data.frame(return_strategy)
df <- na.omit(df)
df$retplus = df$daily.returns + 1
cumulativeReturn <- tail(cumprod(df$retplus),1) -1
finalDataFrame <- data.frame(totalNoOfLongTrades,totalNoOfShortTrades,percntTimeLong,percntTimeShort,cumulativeReturn)

return(finalDataFrame)

}
```

```
finalData <- main("JNJ", "20140101", "20171231", 0.5)
finalData
```

```
##   totalNoOfLongTrades totalNoOfShortTrades percntTimeLong percntTimeShort
## 1                  30                   30             48              52
##   cumulativeReturn
## 1       -0.1670116
```

Applying the rule of long when DVI < 0.5 and short otherwise. The dataframe obtained shows the total number of long trades and short trades at 30 each. While the cumulative return is at -0.167. The required output was for the company Jonson & Jonson, with a ticker of JNJ, for the time period of Jan 1st 2014 to December 31st 2017.

Multiple Back test Periods.

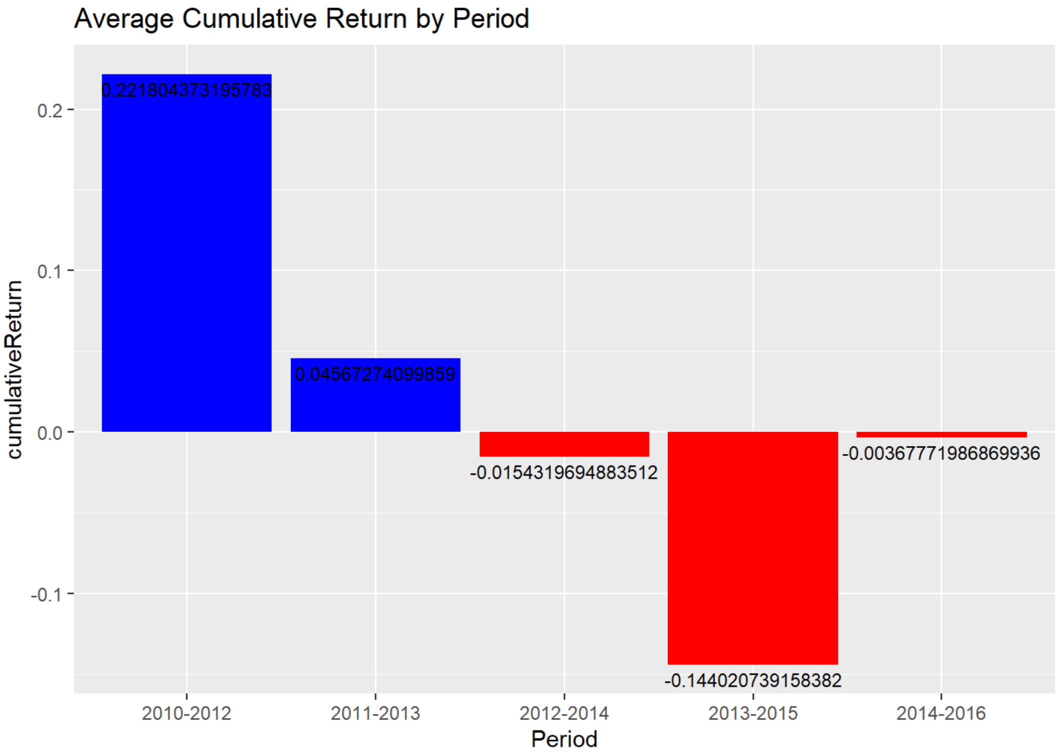
```
multipleBackTest <- function(ticker , testPeriods , years , dvi ){
counter <- 0
y <- years
x <- as.numeric(y[-1]) - as.numeric(y[1]) +1
 for ( i in 1: x ){
   from <- make_date((as.numeric(years[1] )+ counter ),01,01)
   to <- make_date((as.numeric(years[1] ) + counter + testPeriods -1))
   if(to > make_date((as.numeric(y[-1]) ),12,31)){
     break
   }
   df1 <- main(ticker , from , to , 0.5 )
   df1 <- df1 %>% mutate(fromDate = from  ) %>% mutate(toDate = to)
   if(counter == 0){
     df <- df1
   }
    else{
     df<- rbind(df,df1)
    }
   counter<- counter +1
 }
df$Period <- paste(format(as.Date(df$fromDate, format="%d/%m/%Y"),"%Y"),format(as.Date(df$toDate, format="%d/%m/%Y"),"%Y"),s
ep ="-")
return(df)
}
```

Multiple Back testing graph.

```
df4<-multipleBackTest("JNJ",3, c("2010","2016"),0.5)
df4
```

```
##   totalNoOfLongTrades totalNoOfShortTrades percntTimeLong percntTimeShort
## 1                  10                   10       53.06122        46.93878
## 2                   5                    5       44.13793        55.86207
## 3                   5                    5       60.00000        40.00000
## 4                   6                    7       55.10204        44.89796
## 5                   8                    7       44.89796        55.10204
##   cumulativeReturn   fromDate     toDate    Period
## 1       0.22180437 2010-01-01 2012-01-01 2010-2012
## 2       0.04567274 2011-01-01 2013-01-01 2011-2013
## 3      -0.01543197 2012-01-01 2014-01-01 2012-2014
## 4      -0.14402074 2013-01-01 2015-01-01 2013-2015
## 5      -0.00367772 2014-01-01 2016-01-01 2014-2016
```

```
df4 %>% mutate(Color = ifelse(cumulativeReturn <0, "red","blue")) %>%
  ggplot(aes(x = Period, y = cumulativeReturn, fill = Color))+
  geom_col()+
  geom_text(aes(label=cumulativeReturn), vjust=1.5, color="black", size=3)+
  ggtitle("Average Cumulative Return by Period")+
  scale_fill_identity(guide = FALSE)
```



Simulating multiple back testing periods, considering all the possible period intervals, with a default DVI of 0.5. We obtain the bar graph and the data frame shown above.In the data frame, we can see the total number of Short and Long trades along with their periods and their Cumulative Return. The same is visualized in the bargraph for a better understanding.

Simulating Multiple DVI thresholds.

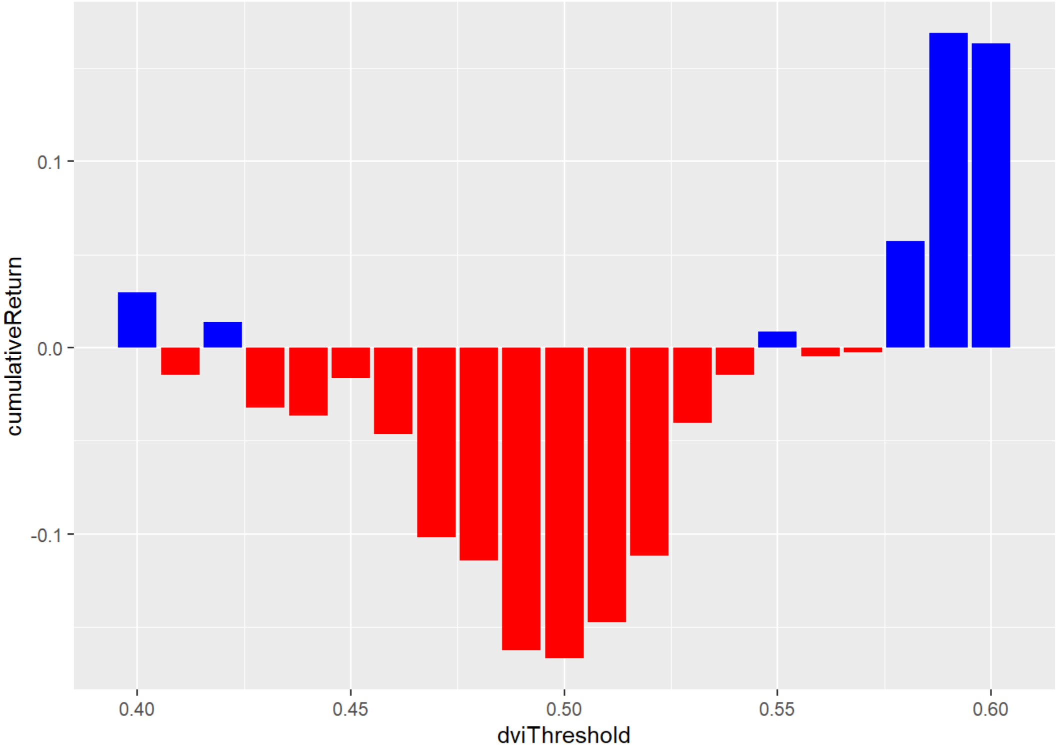
```
multipleBackTestWithDvi <- function(ticker,fromDate, toDate , intitailDvi ,finalDvi, steps){
  for(i in seq(intitailDvi, finalDvi, steps)){
    df1 <- main(ticker,fromDate,toDate,i)
    df1$dviThreshold <- i
    if(i == intitailDvi){df <- df1}
    else{df <- rbind(df,df1)}
  }
  return(df)
}
```

Obtaining the graph of the multiple DVI thresholds.

```
df <- multipleBackTestWithDvi("JNJ", "20140101", "20171231", 0.4, 0.6, 0.01)
df
```

```
##    totalNoOfLongTrades totalNoOfShortTrades percntTimeLong percntTimeShort
## 1                   30                   30       38.46154        61.53846
## 2                   29                   29       39.38462        60.61538
## 3                   29                   29       40.46154        59.53846
## 4                   28                   28       41.38462        58.61538
## 5                   28                   28       42.15385        57.84615
## 6                   29                   29       43.23077        56.76923
## 7                   29                   29       44.00000        56.00000
## 8                   28                   28       44.92308        55.07692
## 9                   29                   29       46.30769        53.69231
## 10                  29                   29       47.07692        52.92308
## 11                  30                   30       48.00000        52.00000
## 12                  30                   30       49.23077        50.76923
## 13                  31                   31       49.84615        50.15385
## 14                  32                   32       51.69231        48.30769
## 15                  33                   33       53.38462        46.61538
## 16                  33                   33       54.46154        45.53846
## 17                  31                   31       55.38462        44.61538
## 18                  30                   30       56.30769        43.69231
## 19                  31                   31       57.38462        42.61538
## 20                  34                   34       58.30769        41.69231
## 21                  32                   32       59.84615        40.15385
##    cumulativeReturn dviThreshold
## 1       0.029467250         0.40
## 2      -0.014840823         0.41
## 3       0.013626310         0.42
## 4      -0.032340622         0.43
## 5      -0.036423881         0.44
## 6      -0.016490685         0.45
## 7      -0.046668649         0.46
## 8      -0.101722105         0.47
## 9      -0.114357188         0.48
## 10     -0.162615995         0.49
## 11     -0.167011579         0.50
## 12     -0.147489727         0.51
## 13     -0.111696884         0.52
## 14     -0.040239974         0.53
## 15     -0.014640373         0.54
## 16      0.008646664         0.55
## 17     -0.004821547         0.56
## 18     -0.002753345         0.57
## 19      0.057067132         0.58
## 20      0.168874665         0.59
## 21      0.163403285         0.60
```

```
df %>% mutate(Color = ifelse(cumulativeReturn <0, "red","blue")) %>%
  ggplot(aes(x = dviThreshold, y = cumulativeReturn, fill = Color))+
  geom_col()+
  scale_fill_identity(guide = FALSE)
```



Considerations For Future Analysis

When developing future financial functions for trading rules, we will want to carefully consider and address the response time to execute a trade following a signal indicator. Once we receive the signal to perform a trade, the scaling of that trade can also be important to understand. If the ticker has a high liquidity of the market, a large trade will not have as much of a ripple effect and upset the market then if the company is very illiquid. A penalty to not trading a high enough quantity can be relative to how high the transaction costs are for each sale. It could be useful to incorporate multiple signals to help the algorithm see a larger perspective of data that could lead to a more accurate portfolio prediction. Similarly, an analyst can ensure lower risk across the board by implementing a large diversification classification of companies. It is beneficial to stay away from large amounts of trading on one specific portfolio and to attempt to hedge your funds by incorporating many different firms at a smaller scale. Finally, when it comes to automating the finalized trading rules, one should be careful to have some sort of human intervention if the dollar amount of the transaction reaches a certain level.