Quandmod-Quantitative Finance R tutorial Youtube

Janis Corona

1/20/2020

This is a youtube tutorial on quant finance from 'Quant Finance with R Part 1 intro and Data": This tutorial link.

The github repository for these tutorials are at: https://github.com/fdupuis659/Quant-Finance-with-R

- $install.packages('quantmod')$
- $install.packages('PerformanceAnalytics)$

The above package, quantmod, is used for quantitative finance in R. The PerformanceAnalytics package is used to analyze the data using quantmod in R.

```
library(quantmod)
library(PerformanceAnalytics)
```

- $?quantmod$
- $?getSymbols$

This sets a minimum date to grab out of the AAPL finance data so that all dates will be after Feb 1, 2017.

```
dt <- '2017-2-1'

aapl <- getSymbols.yahoo('AAPL', from = dt, auto.assign=F)
```

The above object is an 'xts' object or extensible time series object used in financial markets.

```
head(aapl)

##            AAPL.Open AAPL.High AAPL.Low AAPL.Close AAPL.Volume
AAPL.Adjusted
## 2017-02-01    127.03    130.49   127.01     128.75   111985000
122.9902
## 2017-02-02    127.98    129.39   127.78     128.53    33710400
122.7800
## 2017-02-03    128.31    129.19   128.16     129.08    24507300
123.3054
## 2017-02-06    129.13    130.50   128.90     130.29    26845900
124.4613
## 2017-02-07    130.54    132.09   130.45     131.53    38183800
125.6458
```

```
## 2017-02-08    131.35    132.22   131.22     132.04    23004100
126.1330

dim(aapl)

## [1] 746   6

row.names(aapl)[1:20]

## NULL

colnames(aapl)

## [1] "AAPL.Open"     "AAPL.High"     "AAPL.Low"      "AAPL.Close"
## [5] "AAPL.Volume"   "AAPL.Adjusted"
```

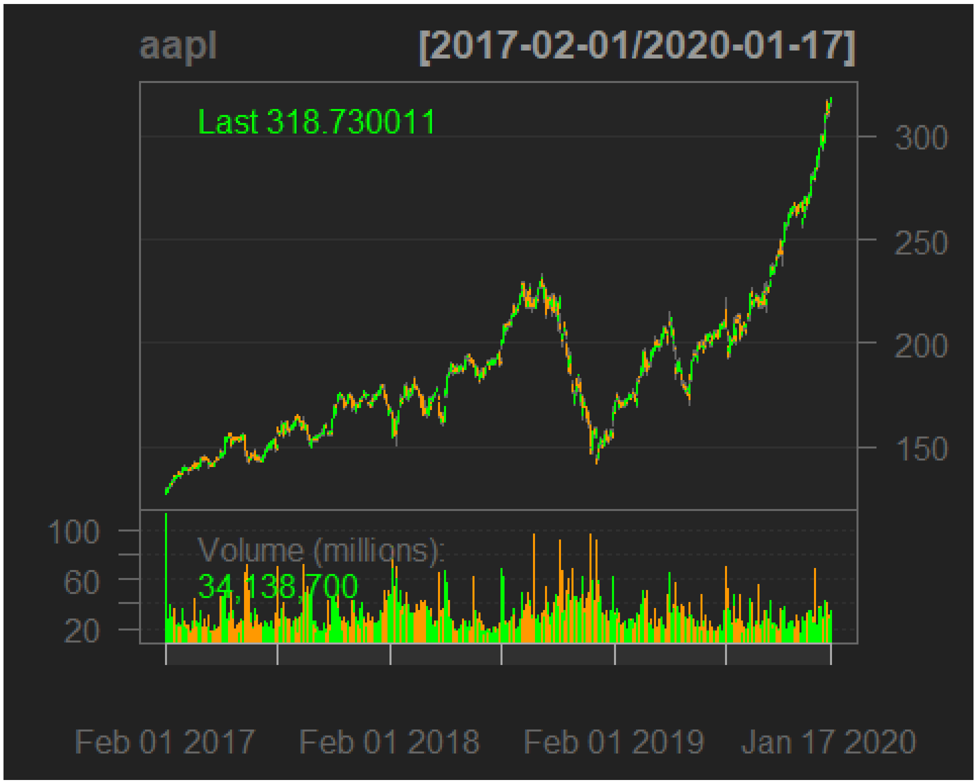
The following are also xts objects.

```
aaplClose <- aapl[,6]


# use of the PerformanceAnalytics library loaded earlier
appleReturns <- dailyReturn(aaplClose, type='log')

appleReturns1 <- na.omit(dailyReturn(aaplClose, type='log'))

head(appleReturns)

##            daily.returns
## 2017-02-01   0.000000000
## 2017-02-02  -0.001710085
## 2017-02-03   0.004269986
## 2017-02-06   0.009330424
## 2017-02-07   0.009471964
## 2017-02-08   0.003870171

head(appleReturns1)

##            daily.returns
## 2017-02-01   0.000000000
## 2017-02-02  -0.001710085
## 2017-02-03   0.004269986
## 2017-02-06   0.009330424
## 2017-02-07   0.009471964
## 2017-02-08   0.003870171
```

The NAs should have been removed but are being read in as zeros, from quantmod when getting the ‘AAPL’ xts object from the web.

The following will chart a graph of the xts object, aapl.

```
chartSeries(aapl)
```



Quant Finance with R Part 2: Portfolio Analysis: (https://www.youtube.com/watch?v=2Y4HX0UUcrA) of 4 parts for this Quantitative Finance tutorial on Youtube from December 2018.

```
library(quantmod)
library(PerformanceAnalytics)
```

Function that uses the closing price column to add the prices since a set date.

```
tickers <- c('FB', 'AAPL','NFLX')

weights <- c(0.25, 0.25, 0.25)

portfolioPrices <- NULL

for (ticker in tickers){
  portfolioPrices <- cbind(portfolioPrices, getSymbols.yahoo(ticker,
  from = '2016-01-03', periodicity='daily', auto.assign=FALSE)[,4])

}
```

Check NAs not in data.

```
colSums(is.na(portfolioPrices))
```

```
##   FB.Close AAPL.Close NFLX.Close
##          0          0          0
```

S&P benchmark

```
benchmarkPrices <- getSymbols.yahoo('^GSPC', from='2016-01-03',
periodicity='daily', auto.assign=FALSE)[,4]
```

Calculate daily change in each column.

```
benchmarkReturns <- na.omit(ROC(benchmarkPrices))
colSums(is.na(benchmarkReturns))

## GSPC.Close
##          0

portfolioReturns <- na.omit(ROC(portfolioPrices))
colSums(is.na(portfolioReturns))

##   FB.Close AAPL.Close NFLX.Close
##          0          0          0

portfolioReturn <- Return.portfolio(portfolioReturns)
```

To find out more on the Return.portfolio function, use: * ? $Return.portfolio$

Some side information about a few financial algorithms:

- **CAPM**: formula for expected return with calculated risk on an asset or stock.
- **ALPHA**: risk adjustment metric for performances compares to an index and shows how much better that index is beat by your benchmark.
- **BETA**: measure of volatility with <1 => less risky and >1 => more risky.
- **SHARPE RATIO**: risk metric for every standard deviation unit, how much return is achieved, gives risk & reward, and most widely used metric with finance managers.

The number of trading days is 252 days a year.

```
CAPM.beta(portfolioReturn, benchmarkReturns, 0.035/252)

## [1] 1.391641

CAPM.jensenAlpha(portfolioReturn, benchmarkReturns, 0.035/252)

## [1] 0.0524188

SharpeRatio(portfolioReturn, 0.035/252)

##                                portfolio.returns
## StdDev Sharpe (Rf=0%, p=95%):         0.05044925
## VaR Sharpe (Rf=0%, p=95%):          0.03085222
## ES Sharpe (Rf=0%, p=95%):           0.01879745
```

```
table.AnnualizedReturns(portfolioReturn)
```

```
##                           portfolio.returns
## Annualized Return                    0.2242
## Annualized Std Dev                   0.2473
## Annualized Sharpe (Rf=0%)            0.9067
```

```
table.CalendarReturns(portfolioReturn)
```

```
##       Jan  Feb  Mar  Apr  May  Jun  Jul  Aug Sep  Oct  Nov  Dec
## 2016  1.4  3.8  1.9 -0.2 -0.8  1.6  1.6  0.2 0.8 -1.4 -1.3 -1.0
## 2017  2.7  1.3 -0.2  1.8  0.1 -0.1  0.5  0.0 0.7  0.4 -0.7 -0.8
## 2018  0.3 -1.1  2.8  1.1  1.8 -0.8  1.7 -0.3 0.8  2.6 -0.2  1.7
## 2019 -0.1  0.3  1.8  2.3 -2.5  1.4 -1.4 -0.4 0.0  1.3 -0.3  0.4
## 2020  0.6   NA   NA   NA   NA   NA   NA   NA  NA   NA   NA   NA
##      portfolio.returns
## 2016               6.8
## 2017               5.8
## 2018              10.9
## 2019               2.8
## 2020               0.6
```

Quant Finance Part 3: Portfolio Optimization: https://www.youtube.com/watch?v=6Pi0fjARtUI

Same libraries and code above used, but add in more tickers.

```
tickers <- c('FB', 'AAPL','NFLX','AMZN','GOOGL','SQ','NVDA')

weights <- c(0.25, 0.25, 0.25)

portfolioPrices <- NULL

for (ticker in tickers){
  portfolioPrices <- cbind(portfolioPrices, getSymbols.yahoo(ticker,
  from = '2016-01-03', periodicity='daily', auto.assign=FALSE)[,4])

}
```

S&P benchmark

```
benchmarkPrices <- getSymbols.yahoo('^GSPC', from='2016-01-03',
periodicity='daily', auto.assign=FALSE)[,4]
```

Calculate daily change in each column.

```
benchmarkReturns <- na.omit(ROC(benchmarkPrices))
portfolioReturns <- na.omit(ROC(portfolioPrices))
portfolioReturn <- Return.portfolio(portfolioReturns)
```

$install.packages('imputeTS') install.packages('PortfolioAnalytics')$

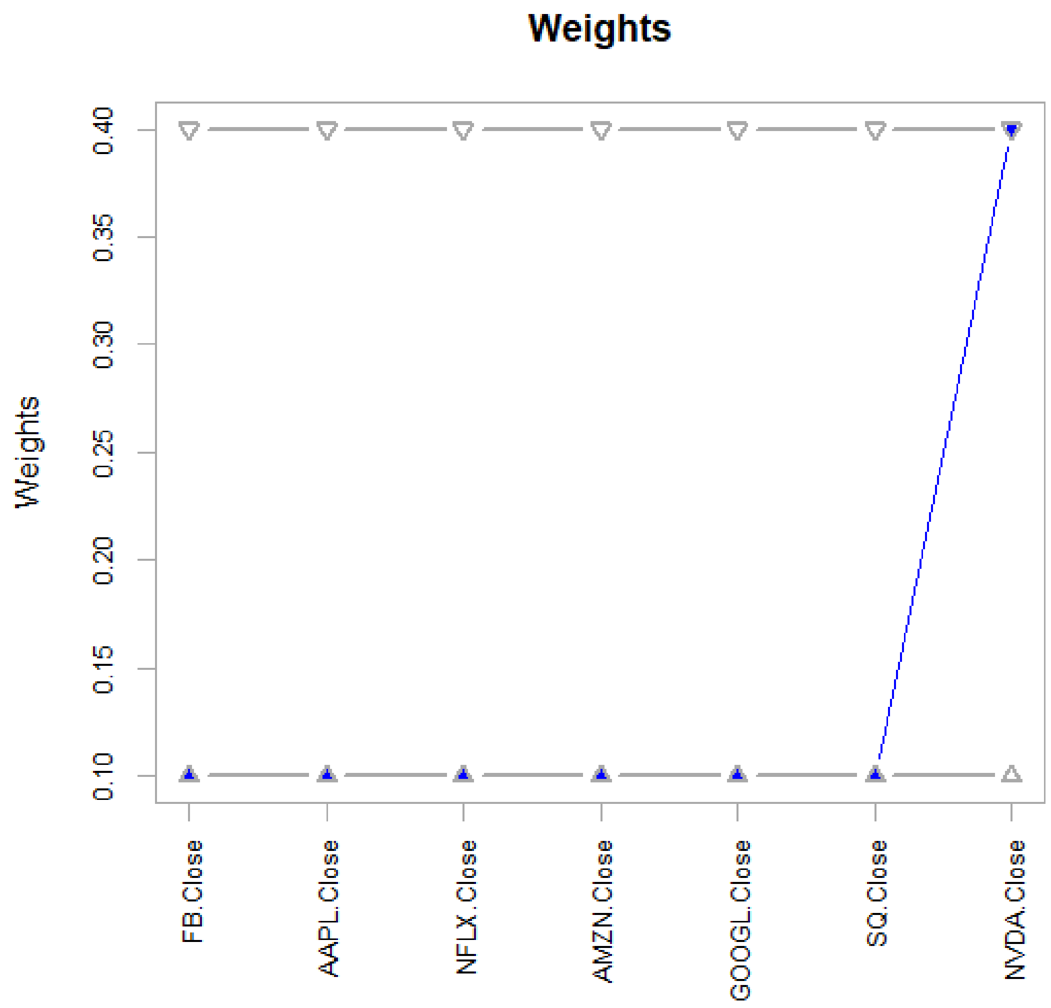
```
library(dplyr)

##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:xts':
##
##     first, last

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union

library(quantmod)
library(PerformanceAnalytics)
library(imputeTS)

## Registered S3 methods overwritten by 'forecast':
##   method             from
##   fitted.fracdiff    fracdiff
##   residuals.fracdiff fracdiff

##
## Attaching package: 'imputeTS'

## The following object is masked from 'package:zoo':
##
##     na.locf

library(PortfolioAnalytics)

## Loading required package: foreach

portf <- portfolio.spec(colnames(portfolioReturns))

portf <- add.constraint(portf, type="weight_sum", min_sum=1, max_sum=1)
portf <- add.constraint(portf, type="box", min=.10, max=.40)
portf <- add.objective(portf, type="return", name="mean")
portf <- add.objective(portf, type="risk", name="StdDev")
```

Need to install some more libraries to run the optimize.portfolio().

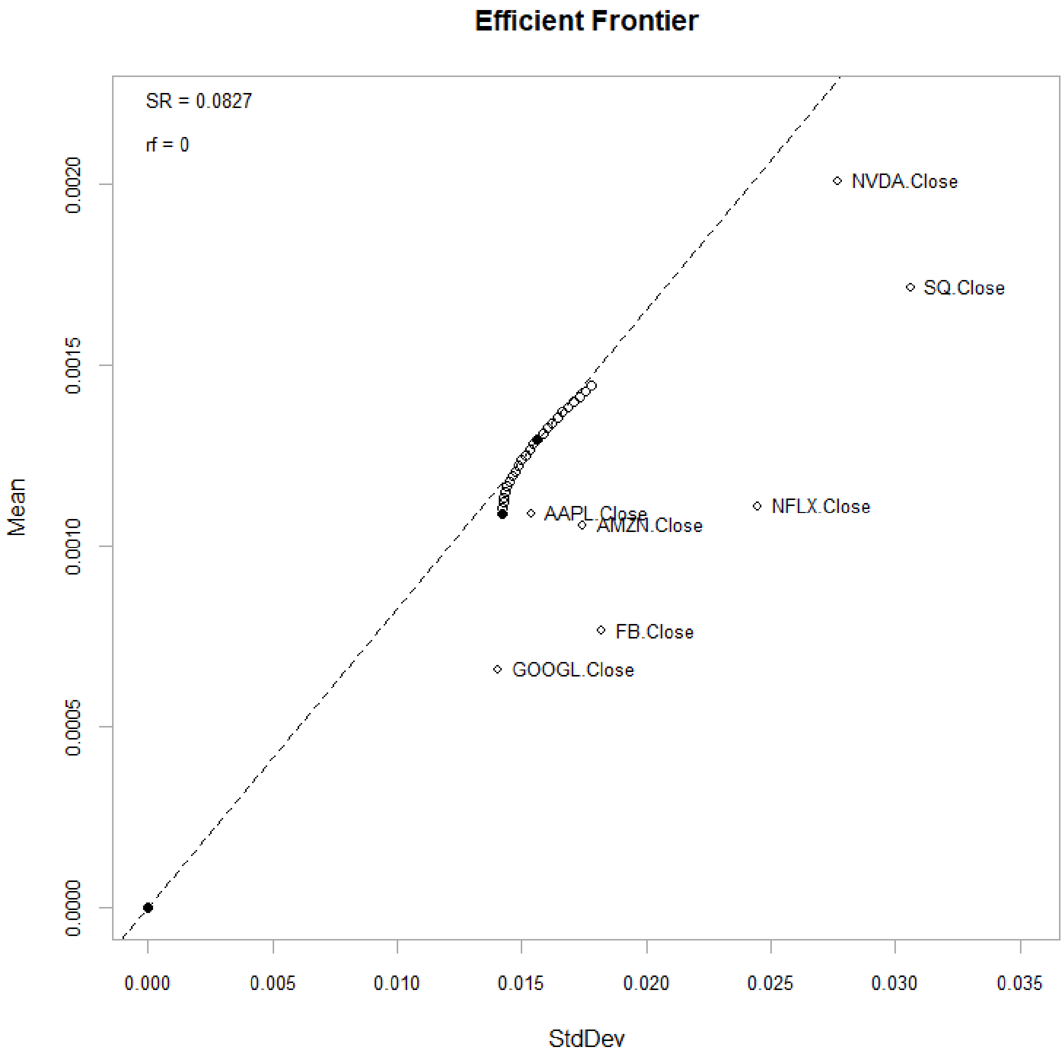
- $install.packages('ROI')$

- $install.packages('ROI.plugin.quadprog')$
- $install.packages('ROI.plugin.glpk')$

```
library(ROI)

## Registered S3 method overwritten by 'ROI':
##   method           from
##   print.constraint PortfolioAnalytics

## ROI: R Optimization Infrastructure

## Registered solver plugins: nlminb, glpk, quadprog.

## Default solver: auto.

##
## Attaching package: 'ROI'

## The following objects are masked from 'package:PortfolioAnalytics':
##
##     is.constraint, objective

library(ROI.plugin.quadprog)
library(ROI.plugin.glpk)

optPort <- optimize.portfolio(portfolioReturns, portf, optimize_method =
"ROI", trace=TRUE)

chart.Weights(optPort)
```

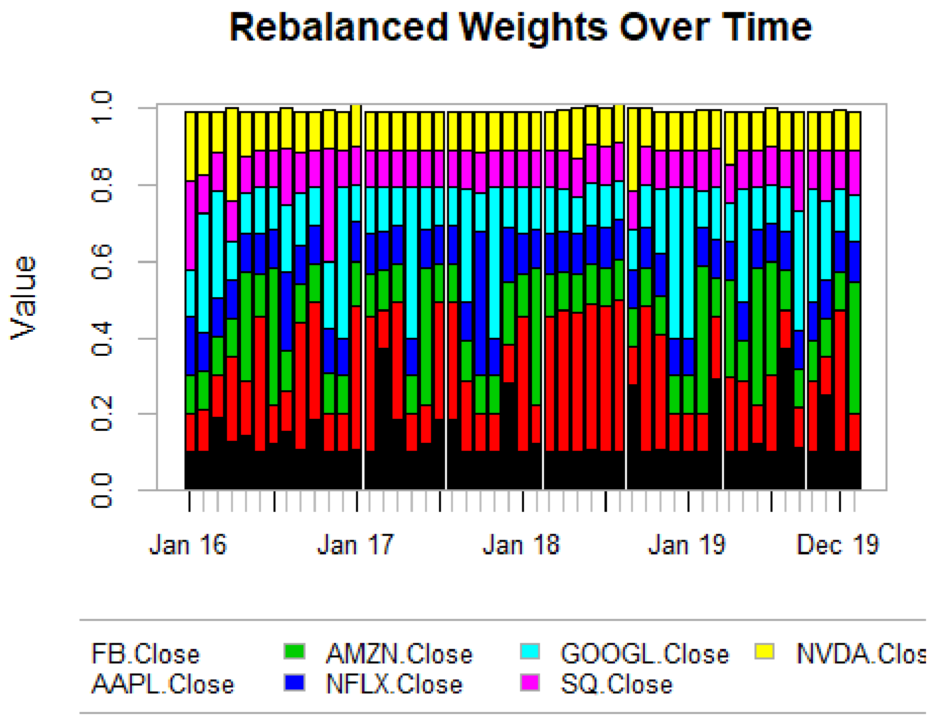


```
ef <- extractEfficientFrontier(optPort, match.col = "StdDev", n.portfolios =
25,
                         risk_aversion = NULL)

## Warning: executing %dopar% sequentially: no parallel backend registered

chart.EfficientFrontier(ef,
                        match.col = "StdDev", n.portfolios = 25, xlim = NULL,
ylim = NULL,
                        cex.axis = 0.8, element.color = "darkgray", main =
"Efficient Frontier",
                        RAR.text = "SR", rf = 0, tangent.line = TRUE,
cex.legend = 0.8,
                        chart.assets = TRUE, labels.assets = TRUE, pch.assets
= 21,
                        cex.assets = 0.8)
```



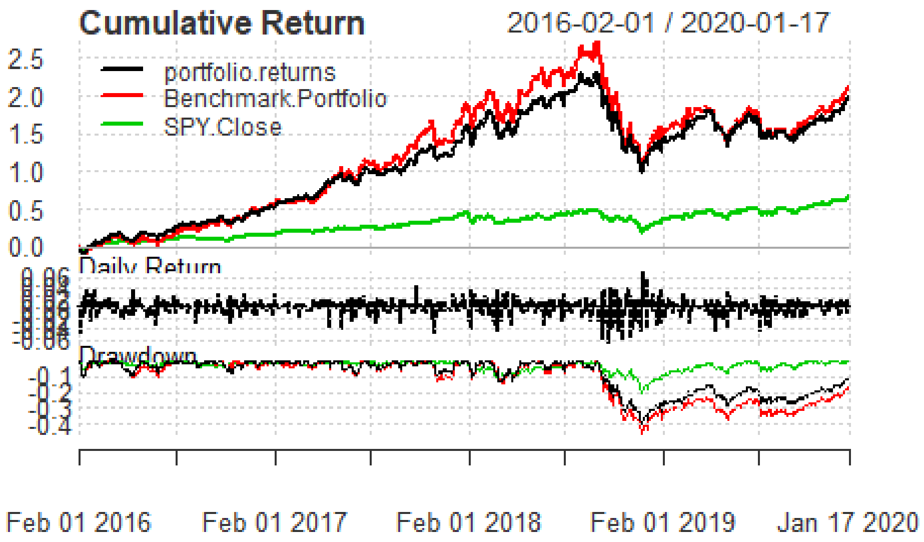
Quant Finance Part 4: Portfolio Optimization Backtest: https://www.youtube.com/watch?v=mBjdkAVdhgM

```r
library(quantmod)
library(PerformanceAnalytics)
library(PortfolioAnalytics)

tickers <- c("FB", "AAPL", "AMZN", "NFLX", "GOOGL", "SQ", "NVDA")

portfolioPrices <- NULL
for(ticker in tickers) {
  portfolioPrices <- cbind(portfolioPrices,
                           getSymbols.yahoo(ticker, from='2016-01-03',
periodicity = 'daily', auto.assign=FALSE)[,4])
}

portfolioReturns <- na.omit(ROC(portfolioPrices))

portf <- portfolio.spec(colnames(portfolioReturns))

portf <- add.constraint(portf, type="weight_sum", min_sum=0.99, max_sum=1.01)
portf <- add.constraint(portf, type="transaction_cost", ptc = 0.001)
portf <- add.constraint(portf, type="box", min=.10, max=.40)
portf <- add.objective(portf, type="return", name="mean")
portf <- add.objective(portf, type="risk", name="StdDev", target=0.005)

rp <- random_portfolios(portf, 10000, "sample")

opt_rebal <- optimize.portfolio.rebalancing(portfolioReturns,
                                           portf,
                                           optimize_method="random",
                                           rp=rp,
                                           rebalance_on="months",
                                           training_period=1,
                                           rolling_window=10)

equal_weight <- rep(1 / ncol(portfolioReturns), ncol(portfolioReturns))
benchmark <- Return.portfolio(portfolioReturns, weights = equal_weight)
colnames(benchmark) <- "Benchmark Portfolio"

sp500prices <- getSymbols.yahoo("SPY", from='2016-01-03', periodicity =
'daily', auto.assign=FALSE)[,4]
sp500Rets <- na.omit(ROC(sp500prices))
sp500Rets <- as.xts(sp500Rets)

chart.Weights(opt_rebal, main="Rebalanced Weights Over Time")
```



```
rebal_weights <-extractWeights(opt_rebal)
rebal_returns <- Return.portfolio(portfolioReturns, weights=rebal_weights)

## Warning in Return.portfolio.geometric(R = R, weights = weights,
wealth.index =
## wealth.index, : The weights for one or more periods do not sum up to 1:
assuming
## a return of 0 for the residual weights

rets_df <- cbind(rebal_returns, benchmark, sp500Rets)

charts.PerformanceSummary(rets_df, main="P/L Over Time")
```

P/L Over Time


Cumulative Return
2016-02-01 / 2020-01-17
2.5
2.0
1.5
1.0
0.5
0.0
portfolio.returns
Benchmark.Portfolio
SPY.Close
Daily Return
0.06
-0.06
Drawdown
-0.1
-0.2
-0.3
-0.4
Feb 01 2016
Feb 01 2017
Feb 01 2018
Feb 01 2019
Jan 17 2020