Rmd _HW2

Apoorva_Reddy_Adavalli

28 February 2019

R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see http://rmarkdown.rstudio.com (http://rmarkdown.rstudio.com).

Loading Libraries

```
# The package scales is only used to present things nicely. You can ignore it.
library('scales')
library('lpSolve')
```

```
## Warning: package 'lpSolve' was built under R version 3.5.2
```

Basic Cleaning of Data

```
#setwd('D:/UT_Austin/Spring/OPTIMISATION/Project2')
# read in the data
data = read.csv("N100StkPrices.csv", header = TRUE)
# clean up data
data = na.omit(data)
ticker = data$TICKER
# spun off MDLZ
delete = seq(1, dim(data)[1])[ticker == "MDLZ"]
data = data[-delete, ]
date = apply(as.matrix(data$date), MARGIN = 1, FUN = "toString")
date = as.Date(date, "%Y%m%d")
ticker = data$TICKER
price = data$PRC
shares = data$SHROUT
# Accounting for changes in ticker names
# KFT changed to KRFT in Oct 2012.
ticker[ticker == "KFT"] = "KRFT"
# SXCI changed to CTRX in Jul 2012.
ticker[ticker == "SXCI"] = "CTRX"
# HANS changed to MNST in Jan 2012.
ticker[ticker == "HANS"] = "MNST"
# convert prices to a matrix, arranged by rows of dates and columns of tickers
unique_dates = sort(unique((date)))
unique_tickers = sort(unique(ticker))
```

Daily and Monthly Price matrices

```
priceMat = matrix(NA, length(unique_dates), length(unique_tickers))
sharesMat = matrix(0, length(unique_dates), length(unique_tickers))

for (i in 1:length(unique_tickers)) {
  tic = unique_tickers[i]
  #print (tic)
  idx = is.element(unique_dates, date[ticker == tic])
  priceMat[idx, i] = price[ticker == tic]
  sharesMat[idx, i] = shares[ticker == tic]
}

rownames(priceMat) = as.character(unique_dates)
rownames(sharesMat) = as.character(unique_dates)

rm(list = c("data", "delete", "i", "idx", "price", "shares", "tic", "ticker", "date"))

# Read Monthly Data -------------------------------------------------------

# read in the data
mdata = read.csv("N100Monthly.csv", header = TRUE, stringsAsFactors = FALSE)

# clean up data
mdate = apply(as.matrix(mdata$date), MARGIN = 1, FUN = "toString")
mdate = as.Date(mdate, "%Y%m%d")

mticker = mdata$TICKER
mprice = mdata$PRC
mshares = mdata$SHROUT
mticker[mticker == "FOXA"] = "NWSA"


unique_mdates = sort(unique((mdate)))
unique_mtickers = sort(unique(mticker))

idx = is.element(unique_mtickers, unique_tickers)

# if (!all(idx)) {
#   print("Warning: Some tickers seem to be missing")
# }

monthlyPriceMat = matrix(NA, length(unique_mdates), length(unique_tickers))

for (i in 1:length(unique_tickers)) {
  tic = unique_tickers[i]
  idx = is.element(unique_mdates, mdate[mticker == tic])
  monthlyPriceMat[idx, i] = mprice[mticker == tic]
}

rm("mdata", "i", "idx", "mprice", "mshares", "mticker", "tic", "mdate")
```

The Specifics

Question 1

Calculate the daily returns for each stock using the 2012 price data

```
priceToday = priceMat[2:nrow(priceMat),]
priceYesterday = priceMat[1:(nrow(priceMat)-1),]
dailyReturn = priceToday/priceYesterday - 1
# To keep the environment clean, delete variable priceToday and priceYesterday
rm('priceToday','priceYesterday')
# Print out the top 5 periods of the first 5 stocks' daily return
# The function percent() is from package 'scales' which gives the
# percentage presentation of a number.
display = apply(dailyReturn[1:5,1:5],2,percent)
rownames(display) = rownames(priceMat)[1:5]
print(display)
```

```
##            [,1]     [,2]     [,3]     [,4]     [,5]
## 2012-01-03 "0.54%"  "-1.02%" "-0.14%" "-0.12%" "-1.95%"
## 2012-01-04 "1.11%"  "0.71%"  "0.44%"  "0.74%"  "2.02%"
## 2012-01-05 "1.05%"  "0.84%"  "-0.66%" "0.15%"  "0.45%"
## 2012-01-06 "-0.16%" "-0.66%" "1.95%"  "-0.38%" "0.39%"
## 2012-01-09 "0.36%"  "2.35%"  "0.44%"  "-0.02%" "4.67%"
```

Question 2

As our initial candidate for the similarity matrix, find the correlation matrix for the returns of the 100 stocks. Note that there will be missing data in the price matrix (NA which stands for Not Available). You need to specify ‘use’ argument in the ‘cor’ function to handle NAs.

```
rhoCor = cor(dailyReturn,use = 'complete')
#display the correlation matrix of the first 5 stocks
print(rhoCor[1:5,1:5],digit = 2)
```

```
##      [,1] [,2] [,3] [,4] [,5]
## [1,] 1.00 0.24 0.40 0.36 0.31
## [2,] 0.24 1.00 0.58 0.54 0.64
## [3,] 0.40 0.58 1.00 0.64 0.57
## [4,] 0.36 0.54 0.64 1.00 0.51
## [5,] 0.31 0.64 0.57 0.51 1.00
```

Question 3

Code the integer program above as another function that returns the weights for each of the stock that needs to be in your portfolio.

Calling ConstructFund function

```
source('constructFund.R')
#Weights for each stock
allocation = constructFund(rhoCor, q = 25, priceMat, sharesMat, unique_tickers, unique_dates)
y_correlation=allocation$solution[1:100]
#allocation$weight_fraction
y_correlation
```

```
##   [1] 0 0 1 1 0 0 0 0 0 0 1 0 0 0 1 0 0 0 1 0 0 1 0 0 0 0 1 0 1 0 0 0 0 0 0
##  [36] 1 0 0 1 1 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 1 1 0 1 0 0 1 0 0 0 0 0 0 0
##  [71] 1 0 0 0 0 1 1 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 1 0 0 1 1 0 0 0
```

This will amount to simply formulating the integer program, solving it and then using the market capitalization of each company on the last date to compute weights. The output weights will be a vector of size n with only q non-zero elements denoting the weights.

```
#Basic checks
# Check1 - Number of stocks picked out of 100 in NASDAQ is 25
sum(y_correlation)
```

```
## [1] 25
```

```
# Check2 - The sum of weights is 1 after normalisation
sum(allocation$weight_fraction)
```

```
## [1] 1
```

Question 4

Use your weights to construct an index portfolio at the end of 2012.

Portfolio of stocks

```
Portfolio_weights_correlation <- data.frame("Stock_name" = as.character(unique_tickers),
                                   "weights" = as.vector(allocation$weight_fraction))
Portfolio_weights_correlation=Portfolio_weights_correlation[Portfolio_weights_correlation$weight
s>0,]
Portfolio_weights_correlation
```

```
##    Stock_name      weights
## 3         ADI 0.2343106663
## 4         ADP 0.0999696115
## 11       AMZN 0.1390916618
## 15       BIDU 0.0093990660
## 19       CELG 0.0488368463
## 22       CHRW 0.0080233965
## 27       CTRX 0.0033017861
## 29       CTXS 0.0348163151
## 36       ESRX 0.0150775179
## 39       FAST 0.0047217151
## 40         FB 0.0151813120
## 44       GMCR 0.0020976887
## 53       KRFT 0.0257411154
## 57       LMCA 0.0122518221
## 58        MAR 0.2603313344
## 60       MNST 0.0029896074
## 63        MWW 0.0002246126
## 71       ORLY 0.0065926647
## 76       REGN 0.0055596826
## 77       ROST 0.0041338535
## 78       SBAC 0.0030749654
## 87       TRIP 0.0018607787
## 93        VOD 0.0084671149
## 96        WDC 0.0117989434
## 97        WFM 0.0421459214
```

. Compare how this index portfolio performs monthly in 2013 as compared to the NASDAQ 100 index using the 2013 stock data provided.Here you may assume that you can directly invest in the Index as if it is stock.

Part 1: First, calculate the number of shares for every stock in your fund at the end of 2012 using 1 million dollars in cash.

```
Investment_money=1000000
price_dec_end=priceMat["2012-12-31",]
price_dec_beg=priceMat["2012-12-03",]
Invested_money=as.vector(allocation$weight_fraction)*Investment_money
Index_portfolio=data.frame("Stock" = as.character(unique_tickers),"Investment"=Invested_money,"P
rice_dec_end"=price_dec_end,
                           "Price_dec_beg"=price_dec_beg,"Shares_bought" = Invested_money/price_
dec_end)
Index_portfolio_25=(Index_portfolio[Index_portfolio$Investment>0,])
Index_portfolio_25
```

```
##    Stock  Investment Price_dec_end Price_dec_beg Shares_bought
## 3    ADI 234310.6663       42.0600       40.4400    5570.86701
## 4    ADP  99969.6115       56.9300       56.4100    1756.00934
## 11  AMZN 139091.6618      250.8700      250.3291     554.43721
## 15  BIDU   9399.0660      100.2900       95.9029      93.71888
## 19  CELG  48836.8463       78.4700       78.6600     622.36328
## 22  CHRW   8023.3965       63.2200       61.0000     126.91231
## 27  CTRX   3301.7861       47.1000       48.0600      70.10162
## 29  CTXS  34816.3151       65.6200       60.2500     530.57475
## 36  ESRX  15077.5179       54.0000       53.5100     279.21329
## 39  FAST   4721.7151       46.6500       42.0900     101.21576
## 40    FB  15181.3120       26.6197       27.0400     570.30365
## 44  GMCR   2097.6887       41.3400       37.8100      50.74235
## 53  KRFT  25741.1154       42.5200       41.8350     605.38841
## 57  LMCA  12251.8221      116.0100      111.0500     105.61005
## 58   MAR 260331.3344       37.2700       35.8900    6985.01031
## 60  MNST   2989.6074       52.8400       51.8800      56.57849
## 63   MWW    224.6126        5.6200        5.4400      39.96665
## 71  ORLY   6592.6647       89.4200       93.3200      73.72696
## 76  REGN   5559.6826      171.0700      180.7000      32.49946
## 77  ROST   4133.8535       54.0900       56.3700      76.42547
## 78  SBAC   3074.9654       70.9800       68.4200      43.32158
## 87  TRIP   1860.7787       41.9200       37.7000      44.38880
## 93   VOD   8467.1149       25.1900       25.6500     336.13001
## 96   WDC  11798.9434       42.4900       33.4100     277.68754
## 97   WFM  42145.9214       91.1600       92.4146     462.32911
```

Part 2: Then calculate the value of your fund starting December 2012.

```
sum(Index_portfolio$Shares_bought*Index_portfolio$Price_dec_beg)
```

```
## [1] 973918.3
```

Part3: Next using the value of the index in December and 1 million cash, calculate the units of index you will buy. Then calculate the value of the index.

```
Index_value_NASDAQ_dec=2660.93 # Given
units_NASDAQ_dec=Investment_money/Index_value_NASDAQ_dec
units_NASDAQ_dec
```

```
## [1] 375.8085
```

We can buy the above number of units of NASDAQ 100 index (Assuming we can directly invest in the index as if it is a stock)

Final Part: Portfolio vs NASDAQ Returns (Dec 2012)

Returns on our portfolio at the end of 2012

```
price_dec_beg_portfolio=price_dec_beg[which(y_correlation > 0)]
price_dec_end_portfolio=price_dec_end[which(y_correlation > 0)]
pct_returns_portfolio_dec <-(price_dec_end_portfolio - price_dec_beg_portfolio) / price_dec_beg_
portfolio
Return_on_portfolio <- sum(allocation$weight_fraction * pct_returns_portfolio_dec)
cat("Return on our protfolio for Dec 2012 is ", round(Return_on_portfolio*100,2),'%')
```

```
## Return on our protfolio for Dec 2012 is  3.25 %
```

Returns on NASDAQ at the end of 2012

```
share_dec_end_NASDAQ <- sharesMat["2012-12-31", 1:ncol(sharesMat)]
mcap_NASDAQ_dec_end=price_dec_end*share_dec_end_NASDAQ
NASDAQ_weight_correlation <- mcap_NASDAQ_dec_end/sum(mcap_NASDAQ_dec_end)
pct_returns_NASDAQ_dec <-(price_dec_end - price_dec_beg) / price_dec_beg
Return_on_NASDAQ <- sum(NASDAQ_weight_correlation * pct_returns_NASDAQ_dec)
cat("Return on our protfolio for Dec 2012 is ", round(Return_on_NASDAQ*100,2),'%')
```

```
## Return on our protfolio for Dec 2012 is  -0.25 %
```

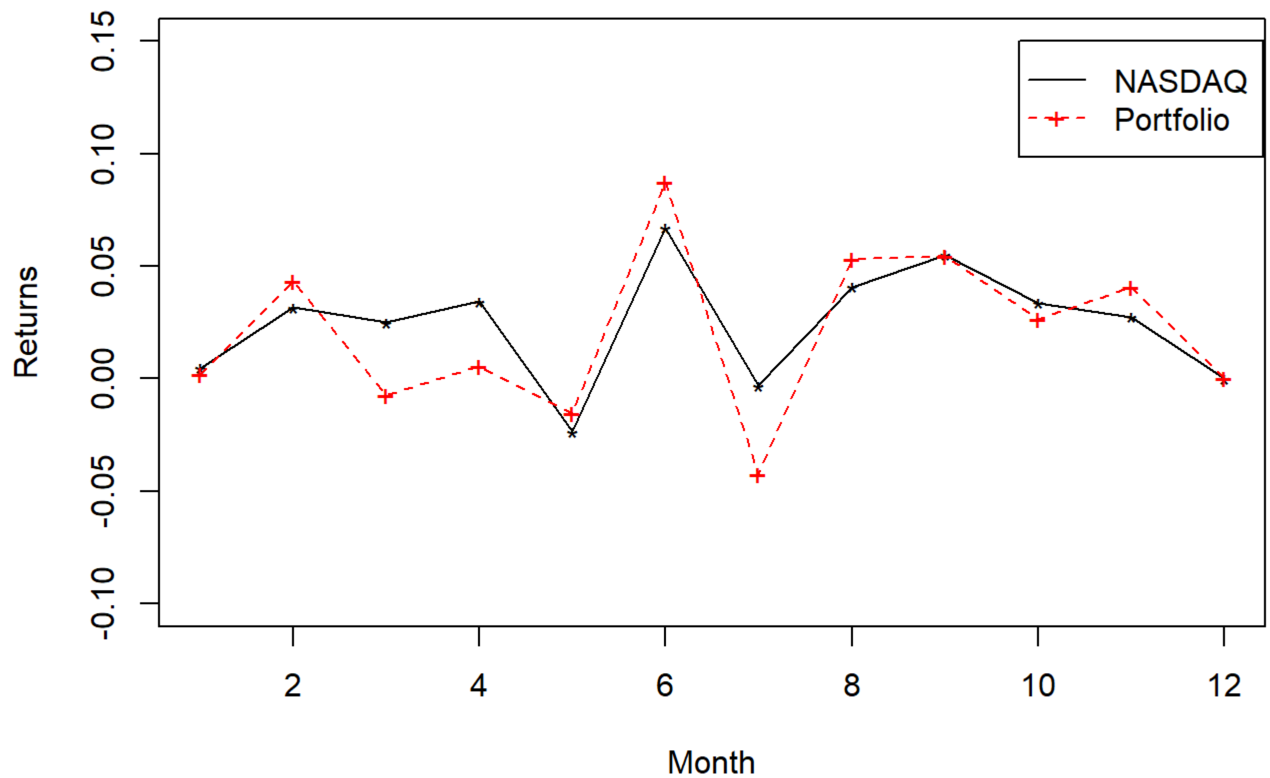
As we can see, our Portfolio performs much better than NASDAQ index in December. Now moving on to monthly performance comparision of our portfolio with NASDAQ 100 in 2013

2013 Monthly Returns

```
#Index_Portfolio Monthly returns

#Step 1:  Calculate stock wise percentage returns for each consecutive month in 2013.

monthly_return=rbind(diff(monthlyPriceMat),rep(0,100))
pct_returns <- monthly_return/monthlyPriceMat
pct_returns_index_portfolio <- pct_returns[1:nrow(pct_returns), which(y_correlation > 0)]

#Step 2: Because our investment proportion remains constant for our index, replicating these wei
ghts for each month in 2013
# and multiplying with returns to calucate overall monthly returns
overall_monthly_return_index=NULL
for (i in 1:nrow(pct_returns)){
  overall_monthly_return_index[i]=sum(pct_returns_index_portfolio[i,]*allocation$weight_fraction
[which(y_correlation > 0)])
          }

# NASDAQ Monthly returns

pct_returns_NASDAQ=(pct_returns)

overall_monthly_return_NASDAQ=NULL
for (i in 1:nrow(pct_returns_NASDAQ)){
  overall_monthly_return_NASDAQ[i]=sum(pct_returns_NASDAQ[i,]*NASDAQ_weight_correlation)
}
```

Visualisations: Present your findings using any visualizations or tabulations. You can leave the shares as non-integers because the effect that the non-integer parts of shares have should be marginal.

```
plot(overall_monthly_return_NASDAQ, col="black", pch="*", lty=1, ylim=c(-0.1,0.15),ylab='Return
s',xlab='Month' )
lines(overall_monthly_return_NASDAQ, col="black",lty=1)
points(overall_monthly_return_index, col="red", pch="+")
lines(overall_monthly_return_index, col="red",lty=2)
legend(9.8,0.15,legend=c("NASDAQ","Portfolio"),col=c("black","red"),pch=c("","+"),lty=c(1,2), nc
ol=1)
```



Insights & Conclusion: 1) On an average, it appears that our portfolio closely follows trends in NASDAQ 100 across months which means that our portfolio can be considered as a good representation of the entire Index. 2) Also our portfolio is doing better than NASDAQ in extreme highs. We can infer that our portfolio is slightly a riskier fund as the stocks picked are not diverse.

Question 5

Earlier you used correlation as the similarity measure. Now instead create your similarity measure and put it in a function similarityMat that has the same inputs and outputs.This criterion is linked to a Learning Outcome explanation of similarity matrix

Use this rho in your function call to constructFund and as in Step 4, evaluate the performance of this fund as well. Please compare the new fund to the previous fund. Explain why the performance is better (or worse). Repeating the above steps and comparing the performance in case of a new similarity measure.

We have selected Cosine similarity as our new measure

Reasons are stated below: Cosine similarity is a measure of similarity between two non-zero vectors of an inner product space that measures the cosine of the angle between them. Further, cosine similarity remains same for a subset of data, however correlation changes for each subset. Since we are dealing with a subset of stocks from NASDAQ 100, a constant measure of similarity across different subsets would result in more stable results.

```
#Step 1: Calculating cosine similarity matrix
source('similarityMat.R')
library(lsa)
```

```
## Warning: package 'lsa' was built under R version 3.5.2
```

```
## Loading required package: SnowballC
```

```
## Warning: package 'SnowballC' was built under R version 3.5.2
```

```
rhoCor_cosine=similarityMat(priceMat, sharesMat, unique_tickers, unique_dates)

#Step2 : Finding the weights for each stock based on new similarity matrix
allocation2 = constructFund(rhoCor_cosine, q = 25, priceMat, sharesMat, unique_tickers, unique_d
ates)
y_cosine <-allocation2$solution[1:100]
```

```
#Basic checks
# Check1 - Number of stocks picked out of 100 in NASDAQ is 25
sum(y_cosine)
```

```
## [1] 25
```

```
# Check2 - The sum of weights is 1 after normalisation
sum(allocation2$weight_fraction)
```

```
## [1] 1
```

```
# Allocation of weights in new portfolio
Invested_money_P2=as.vector(allocation2$weight_fraction)*Investment_money
Index_portfolio_new=data.frame("Stock" = as.character(unique_tickers),"Investment"=Invested_mone
y_P2,
                               "weights"= as.vector(allocation2$weight_fraction),
                               "Price_dec_end"=price_dec_end,"Price_dec_beg"=price_dec_beg,
                           "Shares_bought" = Invested_money_P2/price_dec_end)
Index_portfolio_new_25=(Index_portfolio_new[Index_portfolio_new$Investment>0,])
Index_portfolio_new_25
```

```
##    Stock  Investment      weights Price_dec_end Price_dec_beg
## 3    ADI 244371.5848 0.2443715848       42.0600       40.4400
## 4    ADP  92568.3263 0.0925683263       56.9300       56.4100
## 11  AMZN 139091.6618 0.1390916618      250.8700      250.3291
## 15  BIDU   9399.0660 0.0093990660      100.2900       95.9029
## 19  CELG  55065.5909 0.0550655909       78.4700       78.6600
## 22  CHRW   8023.3965 0.0080233965       63.2200       61.0000
## 27  CTRX   3301.7861 0.0033017861       47.1000       48.0600
## 29  CTXS  33920.4246 0.0339204246       65.6200       60.2500
## 32  DLTR   6592.6647 0.0065926647       40.5600       41.3400
## 36  ESRX  15077.5179 0.0150775179       54.0000       53.5100
## 39  FAST   4721.7151 0.0047217151       46.6500       42.0900
## 40    FB  15181.3120 0.0151813120       26.6197       27.0400
## 44  GMCR   2097.6887 0.0020976887       41.3400       37.8100
## 53  KRFT  25741.1154 0.0257411154       42.5200       41.8350
## 57  LMCA   9619.1905 0.0096191905      116.0100      111.0500
## 58   MAR 260331.3344 0.2603313344       37.2700       35.8900
## 60  MNST   2989.6074 0.0029896074       52.8400       51.8800
## 63   MWW    224.6126 0.0002246126        5.6200        5.4400
## 76  REGN   5559.6826 0.0055596826      171.0700      180.7000
## 77  ROST   4133.8535 0.0041338535       54.0900       56.3700
## 78  SBAC   3074.9654 0.0030749654       70.9800       68.4200
## 87  TRIP   1860.7787 0.0018607787       41.9200       37.7000
## 95  VRTX   3107.2591 0.0031072591       41.9000       39.0800
## 96   WDC  11798.9434 0.0117989434       42.4900       33.4100
## 97   WFM  42145.9214 0.0421459214       91.1600       92.4146
##    Shares_bought
## 3     5810.07097
## 4     1626.00257
## 11     554.43721
## 15      93.71888
## 19     701.74068
## 22     126.91231
## 27      70.10162
## 29     516.92205
## 32     162.54104
## 36     279.21329
## 39     101.21576
## 40     570.30365
## 44      50.74235
## 53     605.38841
## 57      82.91691
## 58    6985.01031
## 60      56.57849
## 63      39.96665
## 76      32.49946
## 77      76.42547
## 78      43.32158
## 87      44.38880
## 95      74.15893
## 96     277.68754
## 97     462.32911
```

Observations

1. New stocks that have come up with minor changes in the weights in the rest of the stocks

```
setdiff((Index_portfolio_new_25$Stock),(Index_portfolio_25$Stock))
```

```
## [1] "DLTR" "VRTX"
```

2. Old stocks that have been replaced with minor changes in the weights in the rest of the stocks

```
setdiff((Index_portfolio_25$Stock),(Index_portfolio_new_25$Stock))
```

```
## [1] "ORLY" "VOD"
```

3. Comparision of returns at the end of 2012

```
# Constructing new Index portfolion at the end of 2012.
price_dec_end=priceMat["2012-12-31",] #End of 2012 price matrix
Investment_money=1000000
Invested_money=as.vector(allocation2$weight_fraction)*Investment_money
Index_portfolio_cosine=data.frame("Stock" = as.character(unique_tickers),"Investment"=Invested_m
oney,"Price"=price_dec_end,
                          "Shares_bought" = Invested_money/price_dec_end)
#Index_portfolio_cosine[Index_portfolio_cosine$Shares_bought>0,]
#Returns on our portfolio at the end of 2012

price_dec_beg_new_portfolio=priceMat["2012-12-03",][which(y_cosine > 0)]
price_dec_end_new_portfolio=priceMat["2012-12-31",][which(y_cosine > 0)]

pct_returns_new_portfolio_dec <-(price_dec_end_new_portfolio - price_dec_beg_new_portfolio) / pr
ice_dec_beg_new_portfolio
Return_on_new_portfolio <- sum(allocation2$weight_fraction * pct_returns_new_portfolio_dec) *Inv
estment_money
ROI_new_portfolio <- round((Return_on_new_portfolio/Investment_money)*100,2)
ROI_new_portfolio
```

```
## [1] 5.03
```

Our new portfolio with ROI of 5.03% performs better than our NASDAQ 100. But earlier portfolio with correlation earned us return (3.25%)

4. Comparision of overall_monthly returns in 2013

```
#New Index Portfolio monthly returns

#Step 1:  Calculate stock wise percentage returns for each consecutive month in 2013.

monthly_return=rbind(diff(monthlyPriceMat),rep(0,100))
pct_returns <- monthly_return/monthlyPriceMat
pct_returns_index_new_portfolio <- pct_returns[1:nrow(pct_returns), which(y_cosine > 0)]

#Step 2: Because our investment proportion remains constant for our index, replicating these wei
ghts for each month in 2013
# and multiplying with returns to calucate overall monthly returns
new_overall_monthly_return_index=NULL
for (i in 1:nrow(pct_returns)){
  new_overall_monthly_return_index[i]=sum(pct_returns_index_new_portfolio[i,]*allocation2$weight
_fraction[which(y_cosine > 0)])
}
print ('old_portfolio_returns')
```

```
## [1] "old_portfolio_returns"
```

```
print (overall_monthly_return_index)
```

```
##  [1]  0.001930217  0.043438323 -0.007163045  0.005720330 -0.015416951
##  [6]  0.087584034 -0.042555125  0.053162856  0.054651912  0.026506382
## [11]  0.040487009  0.000000000
```

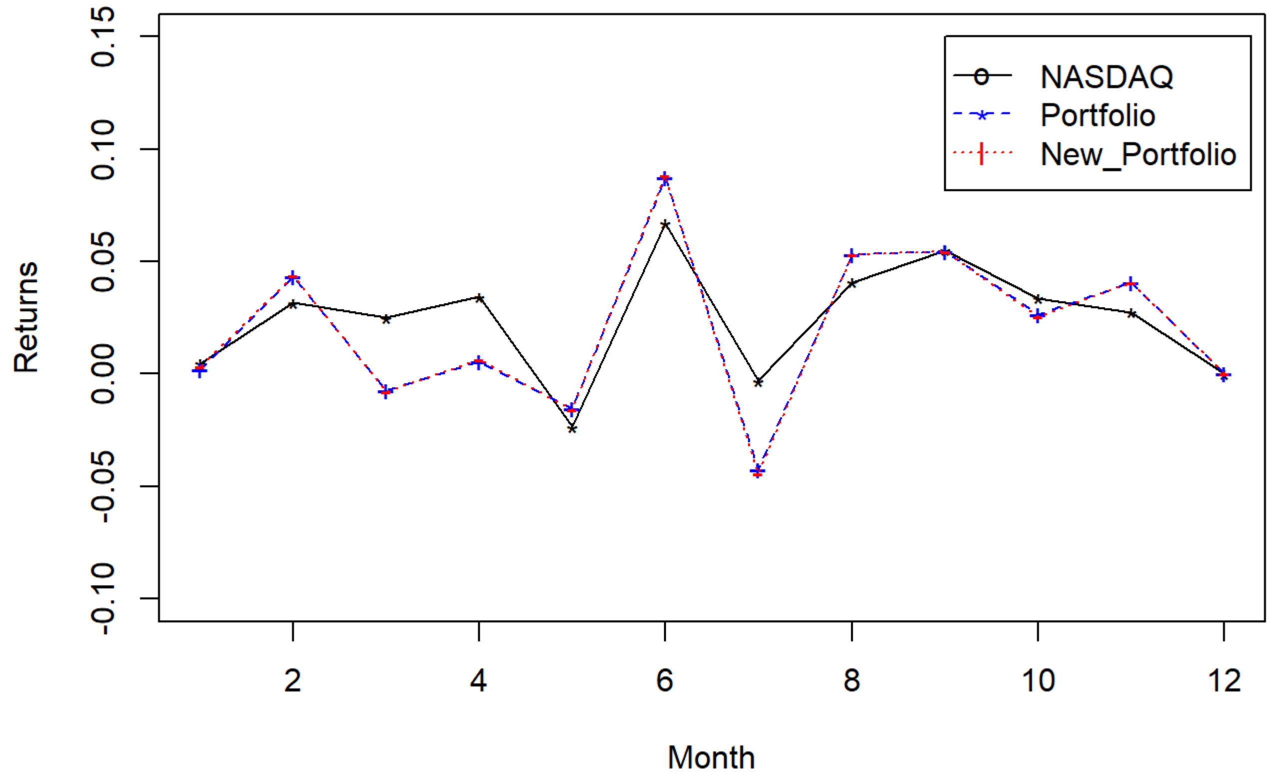
```
print ('new_portfolio_returns')
```

```
## [1] "new_portfolio_returns"
```

```
print (new_overall_monthly_return_index)
```

```
##  [1]  0.003441148  0.043810652 -0.007653883  0.006611718 -0.015734384
##  [6]  0.088525251 -0.044076062  0.053212263  0.054458803  0.025791015
## [11]  0.040926164  0.000000000
```

```
#Compaision of all 2 portfolios wrt NASDAQ
plot(overall_monthly_return_NASDAQ, col="black", pch="*", lty=1, ylim=c(-0.1,0.15),ylab='Return
s',xlab='Month' )
lines(overall_monthly_return_NASDAQ, col="black",lty=1)
points(overall_monthly_return_index, col="blue", pch="+")
lines(overall_monthly_return_index, col="blue",lty=2)
points(new_overall_monthly_return_index, col="red", pch="-")
lines(new_overall_monthly_return_index, col="red",lty=3)
legend(9,0.15,legend=c("NASDAQ","Portfolio","New_Portfolio"),col=c("black","blue","red"),
      pch=c("o","*","l"),lty=c(1,2,3), ncol=1)
```



Considering that cosine similarity is a very similar similarity metric as correlation, with only a small variation. Cosine similarity is a dot product of unit vectors, while correlation is a cosine similarity between centered vectors. Thus effectively ther ae very similar metrics. This is also reflected in the fact that they have very similar performance across all the months in 2013.