Mean-variance portfolio

Georg, Iliyana

28.01.2019

```
setwd("D:\\Portfolio Optimization/Possible Datasets/csv")

data=read.csv("Average_value_weighted_inklsv_US.csv",sep=";",header=TRUE)

# Function for calculating the means of every column/asset in the dataset:
get_means_vector=function(data){
  means_vector= NULL
  for (i in 1:length(data)){
    # Filling in the mean of each asset in every iteration:
    means_vector=c(means_vector,mean(data[[i]]))
  }
  return(means_vector)
}



# Function for calculating the weights of the assets in the dataset:
```

$$\max_{x_t} x_t^\top \mu_t - \frac{\gamma}{2} x_t^\top \Sigma_t x_t,$$

$$x_t = (1/\gamma)\Sigma_t^{-1}\mu.$$

```
get_weights_vector= function(means_vector,cov_matrix){
  # Create the inverse of the passed covariance matrix:
  inverse_cov=solve(cov_matrix)
  # Implement the formula for Xt (DeMiguel, Garlappi, Uppal; page 1922):
  x_t=inverse_cov%*%means_vector
  x_t_vector=c(x_t)
  return (x_t_vector)
}
```

```
# Function for calculating the relative weights of the assets in the dataset:
```

$$w_t = \frac{x_t}{|1_N^\top x_t|},$$

```
get_rel_weights_vector=function(weights_vector){
  # Calculate the absolute sum of the weights vector:
  abs_sum=abs(sum(weights_vector))
  # Divide each of the values in the weights vector by the absolute sum:
  rel_weights=weights_vector/abs_sum
  return (rel_weights)
}



#1.) Sharpe ratio out-of-sample:
```

$$\widehat{SR}_k = \frac{\hat{\mu}_k}{\hat{\sigma}_k}.$$

```
  #1.1.) Drop the column with the months:
  data=data[,-1]
  #1.2.) Set the rolling window:
  rolling_window=120
  #1.3.) Calculate length of the new vector with the portfolio returns:
  len_portfolio_returns=length(data[,1])-rolling_window


  #1.4.) Create the vector with the respective length:
  portfolio_returns=c(length=len_portfolio_returns)

  #1.5.) Calculate the (in this case 497 - 120) excess returns
  # and add each value in the portfolio_returns vector:
  for(i in 1:len_portfolio_returns){
    # Set the start index for each iteration:
    start_window=i
    # Set the start index for each iteration:
    end_window=rolling_window+i-1
    # Create a new "time"-matrix with the start index and the end index
      (rowise):
    time_matrix=data[start_window:end_window,]
    # Create the covariance matrix of the "time"-matrix:
    cov_time_matrix=cov(time_matrix)
```

```
    # Calculate the weights of the assets in row end_window + 1/
    # the weights for the new portfolio based on the last 120 rows:
    weights_vct=get_weights_vector(get_means_vector(time_matrix),cov_time_mat
rix)
    # Calculate the relative weights:
    rel_weights_vct=get_rel_weights_vector(weights_vct)
    # Calculate the portfolio return using the excess returns in row
    # end_window + 1 of the initial data and the computed weights:
    single_pf_return=unlist(data[end_window+1,])%*%rel_weights_vct
    # Add each value in the vector portfolio returns:
    portfolio_returns[start_window]=single_pf_return
}
  # Calculate the sharpe ratio out-of-sample
  sharpe_ratio_out_of_sample=mean(portfolio_returns)/sd(portfolio_returns)
  sharpe_ratio_out_of_sample
```

```
## [1] 0.5300793 (result in paper: 0.0679)
```

```
#2.) Sharpe ratio in-sample:
```

$$\widehat{SR}_k^{IS} = \frac{Mean_k}{Std_k} = \frac{\hat{\mu}_k^{IS\top}\hat{w}_k}{\sqrt{\hat{w}_k^{\top}\hat{\Sigma}_k^{IS}\hat{w}_k}},$$

```
  #2.1.) Alternative 1:
  # The means of the columns/ assets:
  means_vct_in_sample=get_means_vector(data)
  # The weights for each asset:
  weights=get_weights_vector(means_vct_in_sample,cov(data))
```

```
> cov(data)
          NoDur     Durbl     Manuf     Enrgy     HiTec     Telcm     Shops
NoDur  20.11980 17.57148 18.01022 11.418361 17.061059 12.128129 20.51272
Durbl  17.57148 32.55282 22.85251 13.301178 25.833099 14.634323 23.92160
Manuf  18.01022 22.85251 23.48934 15.066811 24.836671 13.153163 22.05955
Enrgy  11.41836 13.30118 15.06681 26.869282 14.866604  9.016275 12.27441
HiTec  17.06106 25.83310 24.83667 14.866604 45.784469 18.375535 25.43032
Telcm  12.12813 14.63432 13.15316  9.016275 18.375535 21.809255 15.32838
Shops  20.51272 23.92160 22.05955 12.274407 25.430317 15.328376 29.37509
Hlth   17.41806 15.58367 17.70245 11.494469 20.720553 12.037808 18.43435
Utils  11.23223 10.32490 10.23887 11.811149  7.475781  9.074115 10.37015
Other  19.75688 23.07145 22.63842 16.104075 24.672021 15.375996 23.67763
US     16.60370 20.42796 20.15495 15.165646 25.655797 14.810241 20.79053
          Hlth     Utils     Other       US
NoDur  17.41806 11.232230 19.75688 16.60370
Durbl  15.58367 10.324901 23.07145 20.42796
Manuf  17.70245 10.238868 22.63842 20.15495
Enrgy  11.49447 11.811149 16.10407 15.16565
HiTec  20.72055  7.475781 24.67202 25.65580
Telcm  12.03781  9.074115 15.37600 14.81024
Shops  18.43435 10.370145 23.67763 20.79053
Hlth   25.93442  9.594538 19.00786 17.25610
Utils   9.59454 16.727251 13.02878 10.51367
Other  19.00786 13.028784 27.22261 21.50227
US     17.25610 10.513666 21.50227 19.89224
```

```r
  # The relative weights for each asset:
  rel_wg=get_rel_weights_vector(weights)
  # The portfolio return:
  pf_return_in_sample=rel_wg%*%means_vct_in_sample
  # The variance of the portfolio:
  var_pf=c(rel_wg%*%cov(data))%*%rel_wg
  # The sd of the portfolio:
  sd=sqrt(var_pf)
  # The sharpe ratio in-sample:
  sharpe_ratio_in_sample=pf_return_in_sample/sd
  sharpe_ratio_in_sample
```

```
##           [,1]
## [1,] 0.9289298 (result in paper: 0.2124)
```

```r
  #2.2.) Alternative 2:
  # Create a new vector to be filled in a loop rowise:
  pf_rtr=c(length=length(data[,1]))
  for(i in 1:length(data[,1])){
    pf_rtr[i]=unlist(data[i,])%*%rel_wg
  }
  mean(pf_rtr)/sd(pf_rtr)
```

```
## [1] 0.9289298
```

```r
  # Leads to the same result
```