GLD_trade1_smac1.R

peter

2021-05-14

```
#install.packages("tidyquant")
library(tidyverse)
```

```
## Warning: package 'tidyverse' was built under R version 3.6.3
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.0 --
```

```
## v ggplot2 3.3.2     v purrr   0.3.3
## v tibble  3.0.3     v dplyr   1.0.2
## v tidyr   1.1.1     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.4.0
```

```
## Warning: package 'ggplot2' was built under R version 3.6.3
```

```
## Warning: package 'tibble' was built under R version 3.6.3
```

```
## Warning: package 'tidyr' was built under R version 3.6.3
```

```
## Warning: package 'dplyr' was built under R version 3.6.3
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(tidyquant)
```

```
## Warning: package 'tidyquant' was built under R version 3.6.3
```

```
## Loading required package: lubridate
```

```
##
## Attaching package: 'lubridate'
```

```
## The following object is masked from 'package:base':
##
##     date
```

```
## Loading required package: PerformanceAnalytics
```

```
## Warning: package 'PerformanceAnalytics' was built under R version 3.6.3
```

```
## Loading required package: xts
```

```
## Loading required package: zoo
```

```
##
## Attaching package: 'zoo'
```

```
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```
## Registered S3 method overwritten by 'xts':
##   method     from
##   as.zoo.xts zoo
```

```
##
## Attaching package: 'xts'

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

##
## Attaching package: 'PerformanceAnalytics'

## The following object is masked from 'package:graphics':
##
##     legend

## Loading required package: quantmod

## Warning: package 'quantmod' was built under R version 3.6.3

## Loading required package: TTR

## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo

## Version 0.4-0 included new data defaults. See ?getSymbols.

## == Need to Learn tidyquant? ==========================================================
## Business Science offers a 1-hour course - Learning Lab #9: Performance Analysis & Portfolio Optimiza
## </> Learn more at: https://university.business-science.io/p/learning-labs-pro </>
```

```
library(farver)
```

```
## Warning: package 'farver' was built under R version 3.6.3
```

```
load(file="symbolsData.Rdata")

symbol0<-list_symbols[2]

symbolsData_symbol0<-symbolsData %>% filter(symbol==symbol0)

# Define logReturn as forward return
symbolsData_symbol0$logReturn<- c(diff(log(symbolsData_symbol0$close)),0)

head(data.frame(symbolsData_symbol0))
```

```
##   symbol       date   open   high    low  close   volume adjusted    logReturn
## 1    GLD 2011-01-03 138.67 139.00 137.88 138.00 11510200   138.00 -0.023832475
## 2    GLD 2011-01-04 136.24 136.28 134.16 134.75 26154300   134.75 -0.002824058
## 3    GLD 2011-01-05 133.50 134.68 133.10 134.37 16700900   134.37 -0.004026799
## 4    GLD 2011-01-06 134.05 134.38 133.14 133.83 15965300   133.83 -0.001869788
## 5    GLD 2011-01-07 133.38 134.61 133.18 133.58 16761400   133.58  0.004034320
## 6    GLD 2011-01-10 133.85 134.20 133.24 134.12  8429900   134.12  0.005873035
```

```
symbolsData_symbol0$SMA50<- runMean(symbolsData_symbol0$close,n=50)

fcn.dot0<-function(x){
  x0<-ifelse(is.na(x)==TRUE,0,x)
  return(x0)
}
```

```
symbolsData_symbol0$SMA50<- fcn.dot0(runMean(symbolsData_symbol0$close,n=50))
symbolsData_symbol0$SMA100<-fcn.dot0(runMean(symbolsData_symbol0$close,n=100))


date0<-symbolsData_symbol0$date

y<-symbolsData_symbol0$logReturn
xclose<-symbolsData_symbol0$close
xSMA50<- runMean(xclose, n=20)
xSMA100<- runMean(xclose, n=40)

head(y)
```

```
## [1] -0.023832475 -0.002824058 -0.004026799 -0.001869788  0.004034320
## [6]  0.005873035
```

```
tail(y)
```

```
## [1] -0.007566032 -0.006082064  0.007061489 -0.004353783  0.016033537
## [6]  0.000000000
```

```
length(xSMA50)
```

```
## [1] 2588
```

```
state0<-100*(xclose>xSMA50) + 10*(xSMA50>xSMA100)
state0.tb<-tibble(behind=lag(state0), state0, ahead=lead(state0))

df00<-rbind(data.frame(
  date=date0,
  value=xclose,
  type="Asset"),
  data.frame(
    date=date0,
    value=xSMA50,
    type="SMA50"),
  data.frame(
    date=date0,
    value=xSMA100,
    type="SMA100")
)

df00 %>% filter(date >=as.Date("2020-01-01")) %>%
ggplot(., aes(x=date, y=value, col=type)) +
  geom_line()
```


```
#
# States to define:
# Indicator of close larger than lag 1 close
# and lags of this indicator


ind.smac1<-ifelse(xclose>lag(xclose,n=1), 1,0)

smac0.tb<-tibble(
  smac1=ind.smac1,
  smac1.lag1=lag(ind.smac1),
  smac1.lag2=lag(lag(ind.smac1)))

tmp3<-split(y, f=smac0.tb)
df3<-data.frame(
  sum=sapply(tmp3,sum),
  n=sapply(tmp3,length)
)
df3
```

```
##                sum   n
## 0.0.0 -0.128173639 260
## 1.0.0 -0.079026851 313
## 0.1.0 -0.017263717 340
## 1.1.0  0.004456068 331
## 0.0.1  0.027622588 312
## 1.0.1  0.093663219 359
```

```
## 0.1.1  0.214372949 331
## 1.1.1  0.095843507 339
```

```
df3$return=df3$sum/df3$n
df3
```

```
##                sum   n        return
## 0.0.0 -0.128173639 260 -4.929755e-04
## 1.0.0 -0.079026851 313 -2.524820e-04
## 0.1.0 -0.017263717 340 -5.077564e-05
## 1.1.0  0.004456068 331  1.346244e-05
## 0.0.1  0.027622588 312  8.853393e-05
## 1.0.1  0.093663219 359  2.609003e-04
## 0.1.1  0.214372949 331  6.476524e-04
## 1.1.1  0.095843507 339  2.827242e-04
```

```
code_smac1<-100*(ind.smac1>0) + 10*(lag(ind.smac1,1)>0) +
  1*(lag(lag(ind.smac1,1),1)>0)

table(code_smac1)
```

```
## code_smac1
##   0   1  10  11 100 101 110 111
## 260 312 340 331 313 359 331 339
```

```
# One step transitions -- Markov Chain

# Table of one-step transitions:
smac_onestep<-table(lag(code_smac1,1), code_smac1)
smac_onestep
```

```
##      code_smac1
##         0   1  10  11 100 101 110 111
##   0   121   0   0   0 139   0   0   0
##   1   138   0   0   0 174   0   0   0
##   10    0 163   0   0   0 177   0   0
##   11    0 149   0   0   0 182   0   0
##   100   0   0 163   0   0   0 150   0
##   101   0   0 177   0   0   0 181   0
##   110   0   0   0 163   0   0   0 168
##   111   0   0   0 168   0   0   0 171
```

```
round(prop.table(smac_onestep,margin =1), digits=4)
```

```
##      code_smac1
##            0      1     10     11    100    101    110    111
##   0   0.4654 0.0000 0.0000 0.0000 0.5346 0.0000 0.0000 0.0000
##   1   0.4423 0.0000 0.0000 0.0000 0.5577 0.0000 0.0000 0.0000
##   10  0.0000 0.4794 0.0000 0.0000 0.0000 0.5206 0.0000 0.0000
##   11  0.0000 0.4502 0.0000 0.0000 0.0000 0.5498 0.0000 0.0000
##   100 0.0000 0.0000 0.5208 0.0000 0.0000 0.0000 0.4792 0.0000
##   101 0.0000 0.0000 0.4944 0.0000 0.0000 0.0000 0.5056 0.0000
##   110 0.0000 0.0000 0.0000 0.4924 0.0000 0.0000 0.0000 0.5076
##   111 0.0000 0.0000 0.0000 0.4956 0.0000 0.0000 0.0000 0.5044
```

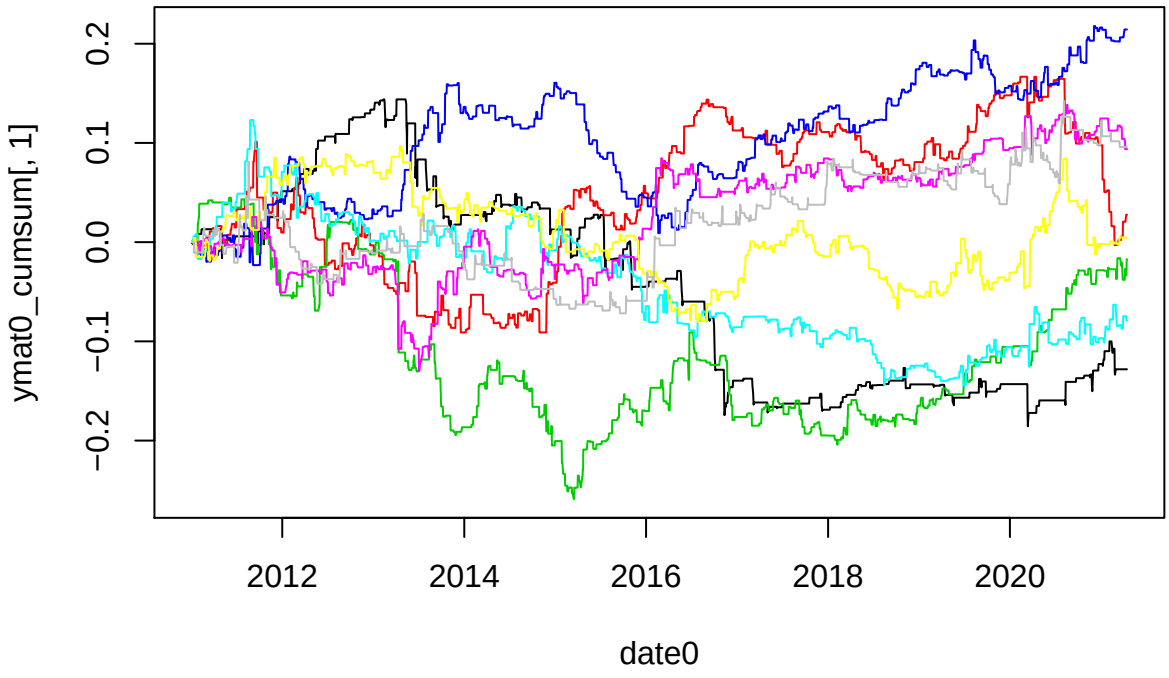
```
# Create ymat with columns corresonding to state-dependent returns
#
ymat<-cbind(
```

```
  y*(code_smac1==0),
  y*(code_smac1==1),
  y*(code_smac1==10),
  y*(code_smac1==11),
  y*(code_smac1==100),
  y*(code_smac1==101),
  y*(code_smac1==110),
  y*(code_smac1==111)
  )

ymat0<-apply(ymat,2,fcn.dot0)
ymat0_cumsum<-apply(ymat0,2,cumsum)

plot(x=date0,y=ymat0_cumsum[,1],ylim=c(min(as.matrix(ymat0_cumsum)),
                                       max(as.matrix(ymat0_cumsum))),type="l")

for (j in c(2:ncol(ymat0_cumsum))){
  lines(date0, ymat0_cumsum[,j], col=j)
}
```

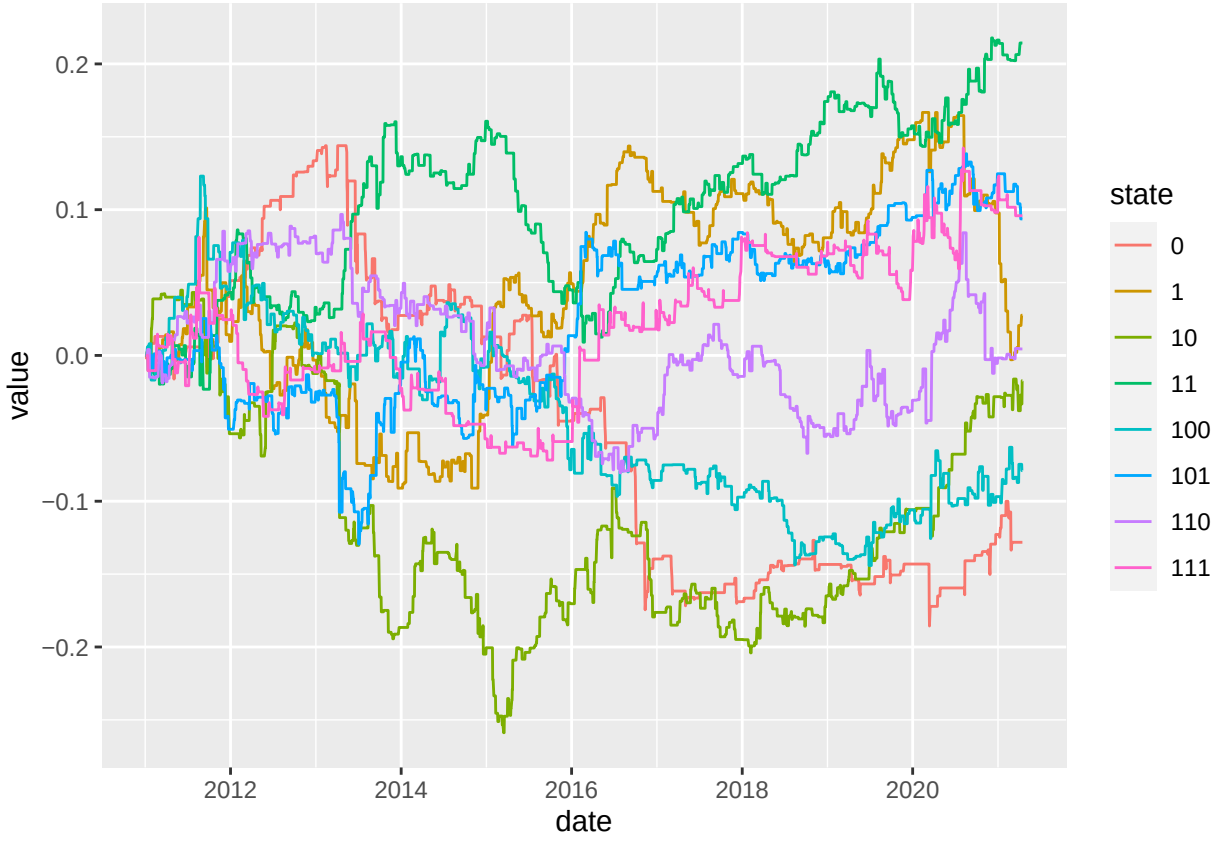



```
# Do same plot with ggplot

df0<-rbind(
  data.frame(date=date0, value=ymat0_cumsum[,1], state=0),
  data.frame(date=date0, value=ymat0_cumsum[,2], state=1),
  data.frame(date=date0, value=ymat0_cumsum[,3], state=10),
```

```
  data.frame(date=date0, value=ymat0_cumsum[,4], state=11),
  data.frame(date=date0, value=ymat0_cumsum[,5], state=100),
  data.frame(date=date0, value=ymat0_cumsum[,6], state=101),
  data.frame(date=date0, value=ymat0_cumsum[,7], state=110),
  data.frame(date=date0, value=ymat0_cumsum[,8], state=111)
)

df0$state<-as.factor(df0$state)
ggplot(df0, aes(x=date,y=value, col=state)) + geom_line()
```




```
# Each col of ymat0 corresponds to a trading  long
# only on days when the state condition holds
#

vec_means=apply(ymat0,2,mean)*252
vec_vol = sqrt(apply(ymat0,2,var))*sqrt(252)
vec_sharpe=vec_means/vec_vol

tab_perf<-cbind(
  mean=vec_means,
  vol=vec_vol,
  sharpe=vec_sharpe)

dimnames(tab_perf)[[1]]<- sort(unique(code_smac1))
tab_perf
```

```
##              mean        vol       sharpe
## 0   -0.0124805862 0.05145065 -0.242573913
## 1    0.0026896801 0.06197650  0.043398385
## 10  -0.0016810111 0.06308356 -0.026647371
## 11   0.0208740275 0.05060242  0.412510481
## 100 -0.0076950412 0.05778567 -0.133165205
## 101  0.0091202207 0.05460362  0.167025932
## 110  0.0004338984 0.05582398  0.007772617
## 111  0.0093325208 0.05214225  0.178981951
```

```
# This gives annualized means/volatilities/share ratios of
# trades

round(cor(ymat0), digits=5)
```

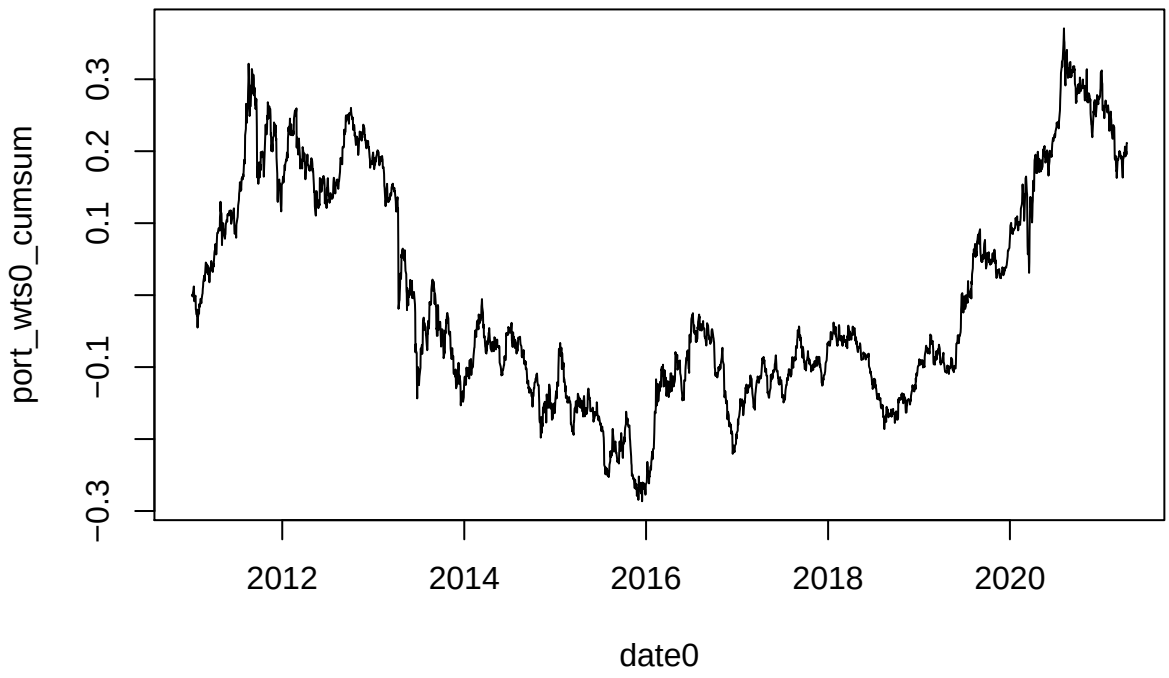
```
##          [,1]   [,2]   [,3]     [,4]     [,5]     [,6]   [,7]     [,8]
## [1,]  1.00000  4e-05 -3e-05  0.00040 -0.00013  0.00016  1e-05  0.00017
## [2,]  0.00004  1e+00  0e+00 -0.00007  0.00002 -0.00003  0e+00 -0.00003
## [3,] -0.00003  0e+00  1e+00  0.00004 -0.00001  0.00002  0e+00  0.00002
## [4,]  0.00040 -7e-05  4e-05  1.00000  0.00022 -0.00027 -1e-05 -0.00029
## [5,] -0.00013  2e-05 -1e-05  0.00022  1.00000  0.00009  0e+00  0.00009
## [6,]  0.00016 -3e-05  2e-05 -0.00027  0.00009  1.00000 -1e-05 -0.00012
## [7,]  0.00001  0e+00  0e+00 -0.00001  0.00000 -0.00001  1e+00 -0.00001
## [8,]  0.00017 -3e-05  2e-05 -0.00029  0.00009 -0.00012 -1e-05  1.00000
```

```
# Define wts vectors corresponding to trading strategies
# that are long on days with the respective states

wts0<-as.matrix(c(1:ncol(ymat0))*0 +1)
wts0
```

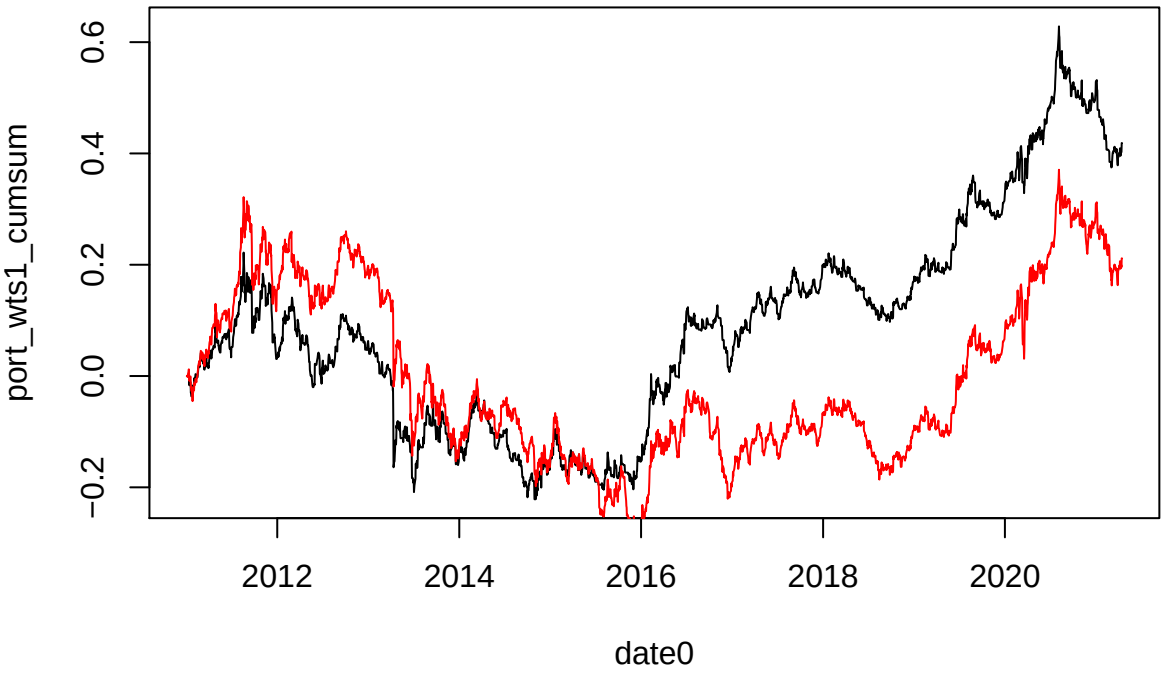
```
##      [,1]
## [1,]    1
## [2,]    1
## [3,]    1
## [4,]    1
## [5,]    1
## [6,]    1
## [7,]    1
## [8,]    1
```

```
port_wts0_cumsum<-cumsum(ymat0 %*% wts0)
plot(date0, port_wts0_cumsum,type="l")
```




```
# wts1 which zeros cases 1 and 5

wts1<-wts0
wts1[1]<-0
wts1[5]<-0
port_wts1_cumsum<-cumsum(ymat0 %*% wts1)
plot(date0, port_wts1_cumsum,type="l")
lines(date0,port_wts0_cumsum,col='red')
```


port_wts1_cumsum
−0.2
0.0
0.2
0.4
0.6
2012
2014
2016
2018
2020
date0