Homework 1, by Somesh Srivastava, Jan 21, 2017

Executive Summary

S&P 500 average arithmetic and geometric historical mean rate of returns and excess return over the prevailing treasury rates for the same period shows self-financed portfolio would have earned around 10% annualized return or around 5% annualized excess return during 1/1/1973 through 1/1/2015. Overlapping longer-term series shows higher return on arithmetic return but it doesn't lead to different inference on geometric returns.

Details

S&P 500 Average arithmetic and geometric historical mean rate of return for daily returns, for monthly returns, for annual returns, and for 5-year returns from 1/1/1973 through 1/1/2015 have been calculated using S&P500 with dividends data from CRSP. Daily and monthly returns file used are given in the "Table and Figures" section.

The historical arithmatic and geometric excess S&P500 return over prevailing treasury rates viz. short rate, the 30-day Treasury rate, the 1-year Treasury rate, and the 5-year Treasury rates, respectively have been calculated using data from CRSP. All the returns have been annualized for comparison.

Based on the results shown in below, it shows that self-financed portfolio would have earned around 10% annualized return or around 5% annualized excess return. Overlapping longer-term series shows higher return on arithmetic return but it doesn't lead to different inference on geometric returns.

Tables and Figures

Tables Used from CRSP:

Rate/ Return	Data File	Variable Weight	Details
s&p 500 daily	crspq.dsp500	vwretd	value weighted return
s&p 500 monthly	crspq.msp500	vwretd	value weighted return
short/overnight rate	crspq.bxcalind	ffefrt	federal fund effective rate
30 days treasury rate	crspq.tfz_mth_rf	tmytm	monthly annualized yield (where kytreasnox = 2000001)
1 year treasury rate	crspq.bxdlyind	ytm	termtype=112
5 year treasury rate	crspq.bxdlyind	ytm	termtype=512

Results:

	Arithmetic	Geometric	Arithmatic Excess	Geometric Excess
Daily	0.1133	0.10337	0.06663	0.05319
Monthly	0.1117	0.10439	0.06315	0.05226
Yearly	0.1201	0.10439	0.06695	0.05065
5yearly	0.1402	0.09861	0.08474	0.04965

Computer Code

```
## Loading required libraries

if (!require("RJDBC")) install.packages("RJDBC")
if (!require("rJava")) install.packages("rJava")
if (!require("data.table")) install.packages("data.table")
if (!require("xts")) install.packages("xts")


###### wrds connection ################
# ---- INPUTS ---- #
username <- "somesh"
password <- '{SAS002}940343080D2FB6A03BBDBBBD587730081C6A411D'
# local path to the sas files
sasPath <- "D:/wrds_drivers/SAS-JDBC-Drivers"
# ---- CODE ---- #
library(rJava)
options(java.parameters = '-Xmx4g')
library(RJDBC)
sasCore <- paste0(sasPath, "/sas.core.jar")
sasDriver <- paste0(sasPath, "/sas.intrnet.javatools.jar")
.jaddClassPath(c(sasCore, sasDriver))
driver <- RJDBC::JDBC(
  "com.sas.net.sharenet.ShareNetDriver",
  sasDriver, identifier.quote = "`")
wrds <- RJDBC::dbConnect(driver,
                         "jdbc:sharenet://wrds-cloud.wharton.upenn.edu:8551/",
                         username, password)


##Load Daily data
sql <- "SELECT caldt, vwretd from CRSPQ.DSP500"
res <- dbSendQuery(wrds, sql)
dbHasCompleted(res) #check that this is true
dsp500 <- as.data.table(fetch(res, n = -1))
dbClearResult(res) # free up memory
```

```
##Load Monthly data
sql <- "SELECT caldt, vwretd FROM CRSPQ.MSP500"
res <- dbSendQuery(wrds, sql)
dbHasCompleted(res) #check that this is true
msp500 <- as.data.table(fetch(res, n = -1))
dbClearResult(res) # free up memory

##Formatting text dates to date format
dsp500$caldt <- as.Date(dsp500$caldt)
dsp500 <- dsp500[dsp500$caldt %between% c("1973-01-01", "2015-01-01")]
msp500$caldt <- as.Date(msp500$caldt)
msp500 <- msp500[msp500$caldt %between% c("1973-01-01", "2015-01-01")]


###############Part (a) return calculation ######################

##Daily return
AMReturnDaily <- mean(dsp500$vwretd)*252
GMReturnDaily <-  (prod(1+dsp500$vwretd)^(1/length(dsp500$vwretd)))^252-1


##Monthly return
AMReturnMonthly <- mean(msp500$vwretd)*12
GMReturnMonthly <- (prod(1+msp500$vwretd)^(1/length(msp500$vwretd)))^12-1

##Yearly return
ep <- endpoints(msp500$caldt, on="years")
AnnualReturn <- period.apply(msp500, ep, FUN = function(x) prod(1+x))-1
AnnualReturn <- as.data.table(AnnualReturn)
colnames(AnnualReturn) <- c("dates", "Annual_vwretd")
AMReturnYearly <- mean(AnnualReturn$Annual_vwretd)
GMReturnYearly <- prod(1+AnnualReturn$Annual_vwretd)^(1/length(AnnualReturn$Annual_vwretd))-1

##Five yearly return
fiveYrEndPt <- ep*5
fiveYrEndPt <- fiveYrEndPt[fiveYrEndPt<=max(ep)]
FiveYearlyReturns <- period.apply(msp500, fiveYrEndPt, FUN = function(x) prod(1+x))-1
FiveYearlyReturns <- as.data.table(FiveYearlyReturns)
colnames(FiveYearlyReturns) <- c("dates", "FiveY_vwretd")
#leave last year value as after 2012 till 2015, less than 5 year period
FiveYearlyReturns <- head(FiveYearlyReturns, -1)

AMReturnFiveY <- mean(FiveYearlyReturns$FiveY_vwretd)/5
GMReturnFiveY <- (prod(1+FiveYearlyReturns$FiveY_vwretd)^(1/length(FiveYearlyReturns$FiveY_vwretd)))^(1/5)-1

## collecting answers
Ans1 <- matrix(c(AMReturnDaily, AMReturnMonthly, AMReturnYearly, AMReturnFiveY,GMReturnDaily, GMReturnMonthly, GMReturnYearly, GMReturnFiveY), ncol=2)
rownames(Ans1) <- c("Daily","Monthly","Yearly","5yearly")
colnames(Ans1) <- c("Arithmetic","Geometric")



############## Part (b) excess over treasury rate ###########


###Loading short rate (overnight rate)
sql <- "SELECT qdate, ffefrt FROM CRSPQ.BXCALIND"
res <- dbSendQuery(wrds, sql)
dbHasCompleted(res) #check that this is true
TshortRate <- as.data.table(fetch(res, n = -1))
dbClearResult(res) # free up memory
TshortRate$ffefrt <- (TshortRate$ffefrt/360)/100 ##converting annualized return to actual daily return


###Loading 1 month rate
sql <- "SELECT MCALDT, TMYTM FROM CRSPQ.TFZ_MTH_RF where KYTREASNOX = 2000001 "
res <- dbSendQuery(wrds, sql)
dbHasCompleted(res) #check that this is true
TmonthlyRate <- as.data.table(fetch(res, n = -1))
dbClearResult(res) # free up memory
TmonthlyRate$TMYTM <- (TmonthlyRate$TMYTM/12)/100 ##converting annualized return to actual monthly return

###Loading 1 year and 5 year rate
sql <- "SELECT termtype, qdate, ytm FROM CRSPQ.BXDLYIND"
res <- dbSendQuery(wrds, sql)
dbHasCompleted(res) #check that this is true
AnnualOneYFiveY <- as.data.table(fetch(res, n = -1))
dbClearResult(res) # free up memory
AnnualOneYFiveY$qdate <- as.Date(AnnualOneYFiveY$qdate)

TYearlyRate <- as.data.table(AnnualOneYFiveY[trimws(termtype)=="112"])
TFiveYRate <- as.data.table(AnnualOneYFiveY[trimws(termtype)=="512"])
TYearlyRate$ytm <- TYearlyRate$ytm/100
TFiveYRate$ytm <- ((1+TFiveYRate$ytm/100)^5-1) #converting annual to actual 5 year

## date format and merging with S&P table

## daily excess ret table
TshortRate$qdate <- as.Date(TshortRate$qdate)
colnames(dsp500) <- c("dates", "snpdaily")
colnames(TshortRate) <- c("dates", "TRateDaily")
ExcessDaily <- merge(dsp500, TshortRate)
ExcessDaily <- na.omit(ExcessDaily)


## monthly excess ret table
TmonthlyRate$MCALDT <- as.Date(TmonthlyRate$MCALDT)
```

```
colnames(msp500) <- c("dates", "snpmonthly")
colnames(TmonthlyRate) <- c("dates", "TRatesMonthly")
ExcessMonthly <- merge(msp500, TmonthlyRate)
ExcessMonthly <- na.omit(ExcessMonthly)

## yearly excess return table
colnames(TYearlyRate) <- c("termtype", "dates", "ytm")
ExcessYearly <- merge(AnnualReturn, TYearlyRate)


## Five Yearly excess return table
colnames(TFiveYRate) <- c("termtype", "dates", "ytm")
Excess5Yearly <- merge(FiveYearlyReturns, TFiveYRate)

### daily excess return
ExcessDaily$excessRet <- ExcessDaily$snpdaily-ExcessDaily$TRateDaily
AMExcessDaily <- mean(ExcessDaily$excessRet)*252
GMExcessDaily <-  (prod(1+ExcessDaily$excessRet)^(1/length(ExcessDaily$excessRet)))^252-1

## monthly excess return
ExcessMonthly$ExcessRet <- ExcessMonthly$snpmonthly-ExcessMonthly$TRatesMonthly
AMExcessMonthly <- mean(ExcessMonthly$ExcessRet)*12
GMExcessMonthly <- (prod(1+ExcessMonthly$ExcessRet)^(1/length(ExcessMonthly$ExcessRet)))^12-1

## Annual Excess return
ExcessYearly$ExcessRet <- ExcessYearly$Annual_vwretd - ExcessYearly$ytm
AMExcessYearly <- mean(ExcessYearly$ExcessRet)
GMExcessYealy <- prod(1+ExcessYearly$ExcessRet)^(1/length(ExcessYearly$ExcessRet))-1

## Five Yearly excess return
Excess5Yearly$ExcessRet <- Excess5Yearly$FiveY_vwretd-Excess5Yearly$ytm
AMExcess5Yearly <- mean(Excess5Yearly$ExcessRet)/5
GMExcess5Yealy <- (prod(1+Excess5Yearly$ExcessRet)^(1/length(Excess5Yearly$ExcessRet)))^(1/5)-1

## collecting answers
Ans2 <- matrix(c(AMExcessDaily, AMExcessMonthly, AMExcessYearly, AMExcess5Yearly,GMExcessDaily, GMExcessMonthly, GMExcessYealy, GMExcess5Yealy),ncol=2)
rownames(Ans2) <- c("Daily","Monthly","Yearly","5yearly")
colnames(Ans2) <- c("Arithmetic_Excess","Geometric_Excess")


Ans <- cbind(Ans1, Ans2)
```

References

- Wharton Research Data Services (WRDS) CRSP data taken on Jan 21, 2018.