# STATS183 Project 5

Takao Oba

2023-05-01

## Please answer the following questions assuming the single index model holds:

**a. Assume the single index model holds. Use only the stocks with positive betas in your data. Choose a value of Rf and find the optimal portfolio (point of tangency) using the optimization procedure as discussed in handout #12: http://www.stat.ucla.edu/~nchristo/statistics_c183_c283/statc183c283_tangent.pdf . The approach here is based on Z = Σ−1R.**

```r
#Read your csv file:
a <- read.csv("/Users/takaooba/STATS 183/stockData.csv", sep=",", header=TRUE)
train <- a[1:60,]
test <- a[61:dim(a)[1],]

#Convert adjusted close prices into returns:
r <- (train[-1,3:ncol(train)]-train[-nrow(train),3:ncol(train)])/train[-nrow(train),3:ncol(train)]
```

```r
rr <- r[,-1]
#Compute the variance-covariance matrix:
var_covar <- cov(rr)
# var_covar

#Compute the inverse of the variance-covariance matrix:
var_covar_inv <- solve(var_covar)
# var_covar_inv

#Create the vector R:
Rf <- 0.002
R_ibar <- apply(rr, 2, mean)
R <- R_ibar-Rf
# R

#Compute the vector Z:
z <- var_covar_inv %*% R
# z

#Compute the vector X:
x <- z/sum(z)
# x

#Compute the expected return of portfolio G:
R_Gbar <- t(x) %*% R_ibar
R_Gbar
```

```
##              [,1]
## [1,] 0.04754626
```

```
#Compute the variance and standard deviation of portfolio G:
var_G <- t(x) %*% var_covar %*% x
# var_G

sd_G <- var_G^0.5
sd_G
```

```
##              [,1]
## [1,] 0.03708783
```

```
#Compute the slope:
slope <- (R_Gbar-Rf)/(sd_G)
# slope
```

Side note: using C*

```
#Compute the betas:
covmat <- var(r)
beta <- covmat[1,-1] / covmat[1,1]

#Keep only the stocks with positive betas:
rrr <- r[,-c(1,which(beta<0)+1)]

#all of the stocks have positive betas
length(rrr)
```

```
## [1] 30
```

```
#Note: which(beta<0) gives the element in the beta vector with negative beta and add 1 because
#the first column in the iitial data set is the index.
# We also remove column 1 (index) from the initial data #set.

#Initialize
beta <- rep(0,ncol(rrr))
alpha <- rep(0,ncol(rrr))
mse <- rep(0,ncol(rrr))
Ribar <- rep(0,ncol(rrr))
Ratio <- rep(0,ncol(rrr))
stock <- rep(0,ncol(rrr))

#Risk free asset:
rf <- 0.005


#This for loop computes the required inputs:
for(i in 1:ncol(rrr)){
    q <- lm(data=rrr, formula=rrr[,i] ~ r[,1])
    beta[i] <- q$coefficients[2]
    alpha[i] <- q$coefficients[1]
```

```
     mse[i] <- summary(q)$sigma^2
    Ribar[i] <- q$coefficients[1]+q$coefficients[2]*mean(r[,1])
    Ratio[i] <- (Ribar[i]-rf)/beta[i] # (R_i - R_f)/B_i
    stock[i] <- i
}


#So far we have this table:
xx <- (cbind(stock,alpha, beta, Ribar, mse, Ratio))




#Order the table based on the excess return to beta ratio:
A <- xx[order(-xx[,6]),]

col1 <- rep(0,nrow(A))
col2 <- rep(0,nrow(A))
col3 <- rep(0,nrow(A))
col4 <- rep(0,nrow(A))
col5 <- rep(0,nrow(A))


#Create the last 5 columns of the table:
col1 <- (A[,4]-rf)*A[,3]/A[,5]
col3 <- A[,3]^2/A[,5]
for(i in(1:nrow(A))) {
col2[i] <- sum(col1[1:i])
col4[i] <- sum(col3[1:i])
}

#So far we have:
# cbind(A, col1, col2, col3, col4)
```

Recall that we can find c* by the following

$$C^* = \frac{\sigma_m^2 \, \Sigma \frac{\bar{R}_i - R_f}{\sigma_{\epsilon_i}^2} b_i}{1 + \sigma_m^2 \Sigma \frac{b_i^2}{\sigma_{\epsilon_i}^2}}$$

Where

$$\Sigma \frac{\bar{R}_i - R_f}{\sigma_{\epsilon_i}^2} b_i$$

is represented in column2

and

$$\Sigma \frac{b_i^2}{\sigma_{\epsilon_i}^2}$$

is represented in column 4

```
#Compute the Ci (col5):
for(i in (1:nrow(A))) {
```

```r
col5[i] <- var(r[,1])*col2[i]/(1+var(r[,1])*col4[i])
}
```

Since we have that short selling is allowed, we have

```r
#SHORT SALES ALLOWED:
#Compute the Zi:
z_short <- (A[,3]/A[,5])*(A[,6]-col5[nrow(A)])
#Compute the xi:
x_short <- z_short/sum(z_short)

weights <- data.frame("Stock Name" = colnames(covmat)[-1],
                      "Weights" = x_short)
weights
```

```
##    Stock.Name     Weights
## 1         MCD  0.10261518
## 2         NKE  0.32332147
## 3        SBUX  0.17283392
## 4           F  0.24104655
## 5         CMG  0.43834244
## 6        LULU  0.27261993
## 7           V  0.38841955
## 8         JPM  0.54359894
## 9          MA  0.12409951
## 10        BAC  0.42317996
## 11         MS  0.20140341
## 12        WFC  0.07147675
## 13        JNJ  0.12114633
## 14        UNH  0.07373381
## 15        PFE  0.05954028
## 16        CVS  0.03162145
## 17         CI  0.01003395
## 18        ZTS  0.01376412
## 19        RTX -0.01142462
## 20         BA -0.06581910
## 21        LMT -0.09273014
## 22         DE -0.17012972
## 23        CAT -0.18748504
## 24         GE -0.14903690
## 25       AAPL -0.04273267
## 26       MSFT -0.53692910
## 27       ADBE -0.39901147
## 28       INTU -0.42389646
## 29       NVDA -0.25639104
## 30        CRM -0.27721129
```

```r
#The final table when short sales allowed:
Weights_with_short <- cbind(A, col1, col2, col3, col4, col5, z_short, x_short)
```

**b. Adjusting the betas: Adjust the betas using Blume's and Vasicek's techniques. For the Blume technique use the two periods: 01- Jan-2015 to 01-Jan-2020 and 01-Jan-2020 to 31-Mar-2023. For the Vasicek technique use only the period 01-Jan-2014 to 01-Jan-2019. Note: For the Blume technique our goal is to adjust the betas in 01-Jan-2020 to 31-Mar-2023 to be better forecasts for the betas in period 01-Apr-2023 to 01-Apr-2027. For the Vasicek technique our goal is to adjust the betas in 01-Jan-2015 to 01-Jan-2020 to be better forecasts for the betas in period 01-Jan-2020 to 31-Mar-2023.**

Blume's Technique We will be using two periods from 01-Jan-2015 to 01-Jan-2020 and 01-Jan-2020 to 31-Mar-2023.

```r
#Convert adjusted close prices into returns:
r1 <- (train[-1,3:ncol(train)]-train[-nrow(train),3:ncol(train)])/train[-nrow(train),3:ncol(train)]

r2 <- (test[-1,3:ncol(test)]-test[-nrow(test),3:ncol(test)])/test[-nrow(test),3:ncol(test)]

#Compute the variance covariance matrix of the returns for each period:
covmat1 <- var(r1)
covmat2 <- var(r2)

#Compute the betas in each period:
beta1 <- covmat1[1,-1] / covmat1[1,1]
beta2 <- covmat2[1,-1] / covmat2[1,1]

#Correlation between the betas in the two periods:
cor(beta1, beta2)
```

```
## [1] 0.5507411
```

```r
#Adjust betas using the Blume's technique:
q1 <- lm(beta2 ~ beta1)


beta3adj_blume <- q1$coef[1] + q1$coef[2]*beta2
beta3adj_blume
```

```
##       MCD       NKE      SBUX         F       CMG      LULU         V       JPM
## 0.9112101 1.1037914 1.0600440 1.3981200 1.2589491 1.2309169 1.0453327 1.0763102
##        MA       BAC        MS       WFC       JNJ       UNH       PFE       CVS
## 1.1205174 1.2191256 1.2392187 1.1174170 0.7381099 0.8754591 0.8667130 0.7774759
##        CI       ZTS       RTX        BA       LMT        DE       CAT        GE
## 0.7961129 0.9362179 0.9865224 1.3401971 0.7846516 1.0255135 1.0210260 1.1818548
##      AAPL      MSFT      ADBE      INTU      NVDA       CRM
## 1.1904003 0.9908486 1.2151092 1.1956526 1.3688962 1.2159469
```

Vasicek's Technique Use period 01-Jan-2015 to 01-Jan-2020

```r
#Vasicek's method:
beta2 <- rep(0,30)

alpha2 <- rep(0,30)
```

```r
sigma_e2 <- rep(0,30)

var_beta2 <- rep(0,30)

for(i in 1:30){
    q <- lm(data=r1, formula=r1[,i+1] ~ r1[,1])
    beta2[i] <- q$coefficients[2]
    alpha2[i] <- q$coefficients[1]
    sigma_e2[i] <- summary(q)$sigma^2
    var_beta2[i] <- vcov(q)[2,2]
}


#Adjusting the betas using the Vasicek's technique:
beta3adj_vasicek <- var_beta2*mean(beta2)/(var(beta2)+var_beta2) +
var(beta2)*beta2/(var(beta2)+var_beta2)
beta3adj_vasicek
```

```
##  [1] 0.5423767 0.8841492 0.6650870 1.0459454 0.9044051 0.9313976 0.9327375
##  [8] 1.1400380 1.0151394 1.4280532 1.2496926 1.0821347 0.7371672 0.7664968
## [15] 0.7308134 0.9505549 0.8795790 0.8542972 1.2048472 1.1852432 0.9518934
## [22] 1.1135857 1.3307594 1.1204411 1.2051578 1.1343650 1.0818721 1.0323267
## [29] 1.4342176 1.1620504
```

**c. Compute PRESS only for the Vasicek technique. (You can compute the PRESS only for the Vasicek technique because you have the actual betas in the period 01-Jan-2020 to 31-Mar-2023.)**

Recall there are three components to the MSE:

$$MSE = (\bar{A} - \bar{P})^2 + (1 - \hat{b_1})^2 S_p^2 + (1 - R^2) S_A^2$$

Where the Bias term is

$$(\bar{A} - \bar{P})^2$$

and the inefficiency term is

$$(1 - \hat{b_1})^2 S_p^2$$

and the random error component is

$$(1 - R^2) S_A^2$$

```r
# Actual betas from testing is as follows:
r2 <- (test[-1,3:ncol(test)]-test[-nrow(test),3:ncol(test)])/test[-nrow(test),3:ncol(test)]
beta2 <- covmat2[1,-1] / covmat2[1,1]

#Using Vasicek's technique:
#1.  Bias component:
V1 <- ( mean(beta2) - mean(beta3adj_vasicek) )^2

#2.  Inefficiency component:
q3 <- lm(beta2 ~ beta3adj_vasicek)
Sp32 <- (29/30)*var(beta3adj_vasicek)
V2 <- (1-q3$coef[2])^2*Sp32
```

```
#3.  Random error component:
Sa2 <- (29/30)*var(beta2)
rap32 <- ( cor(beta3adj_vasicek,beta2) )^2
V3 <- (1-rap32)*Sa2

V1+V2+V3
```

```
## beta3adj_vasicek
##       0.07426014
```

Recall we can also achieve

```
PRESS <- sum((beta2 - beta3adj_vasicek)^2)/30
PRESS
```

```
## [1] 0.07426014
```