Project 5

Nicholas Davidson

5/3/2023

(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 (http://www.stat.ucla.edu/~nchristo/statistics_c183_c283/statc183c283_tangent.pdf) . The approach here is based on Z = Σ−1R.

```
a <- read.csv("stockData.csv", sep=",", header=TRUE)

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

#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)]
#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 #se
t.

#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 <- 0.001


#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]
    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])
}


#Compute the Ci (col5):
for(i in (1:nrow(A))) {
col5[i] <- var(r[,1])*col2[i]/(1+var(r[,1])*col4[i])
}

#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)

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

```
##       stock         alpha       beta          Ribar         mse        Ratio
##  [1,]    19  0.0128568768 0.09599006  0.0147787917 0.002830250  0.143543942
##  [2,]    22  0.0226860588 0.18454304  0.0263809830 0.010159741  0.137534223
##  [3,]    13  0.0072257172 0.06077196  0.0084424946 0.001284923  0.122465925
##  [4,]    26  0.0084721243 0.07960230  0.0100659230 0.002403734  0.113890219
##  [5,]    27  0.0086981865 0.08347342  0.0103694930 0.002601563  0.112245224
##  [6,]    17  0.0118998199 0.14818837  0.0148668503 0.005325364  0.093575833
##  [7,]    15  0.0281211613 0.41177943  0.0363658169 0.012500750  0.085885342
##  [8,]    14  0.0127562873 0.18214885  0.0164032750 0.001927513  0.084564220
##  [9,]    18  0.0241431532 0.37177830  0.0315869055 0.006022497  0.082271895
## [10,]     8  0.0151869219 0.24021645  0.0199965403 0.003119281  0.079080930
## [11,]    23  0.0121520026 0.19456093  0.0160475052 0.003166575  0.077340839
## [12,]    16  0.0077663648 0.11894350  0.0101478539 0.001494667  0.076909237
## [13,]     1  0.0190084306 0.39265863  0.0268702493 0.002617901  0.065884835
## [14,]     7  0.0140563654 0.29930404  0.0200490365 0.001472136  0.063644436
## [15,]    12  0.0107431558 0.24844569  0.0157175403 0.002284329  0.059238460
## [16,]    29  0.0072426892 0.17508675  0.0107482796 0.003599384  0.055676854
## [17,]     2  0.0333136619 0.91530123  0.0516398407 0.011141031  0.055325874
## [18,]    24  0.0110070363 0.31258927  0.0172657046 0.003030013  0.052035391
## [19,]    25  0.0118782444 0.37457870  0.0193780662 0.004303695  0.049063298
## [20,]     6  0.0053835627 0.15477605  0.0084824917 0.001451472  0.048343990
## [21,]    28  0.0074172992 0.24121647  0.0122469399 0.003187635  0.046625921
## [22,]     4  0.0067417375 0.38568112  0.0144638523 0.003202944  0.034909285
## [23,]     3  0.0067090522 0.40667227  0.0148514521 0.003091125  0.034060479
## [24,]    11  0.0066637834 0.41088872  0.0148906053 0.003998211  0.033806246
## [25,]    10  0.0038248103 0.34604510  0.0107533320 0.003554502  0.028185147
## [26,]     5  0.0027538205 0.75913460  0.0179532280 0.011914407  0.022332308
## [27,]     9  0.0008876338 0.39325187  0.0087613303 0.004547327  0.019736283
## [28,]    21  0.0002518150 0.38974154  0.0080552275 0.004814547  0.018102324
## [29,]    20 -0.0062142341 0.27585055 -0.0006911491 0.003809642 -0.006130671
##             col1       col2      col3       col4        col5    z_short
##  [1,]  0.4673181  0.4673181  3.255575   3.255575 0.002668395  3.5170821
##  [2,]  0.4610239  0.9283420  3.352067   6.607642 0.005201296  1.7744706
##  [3,]  0.3520018  1.2803437  2.874283   9.481925 0.007059794  3.9077325
##  [4,]  0.3002280  1.5805718  2.636118  12.118043 0.008590381  2.4521423
##  [5,]  0.3006284  1.8812002  2.678318  14.796361 0.010077595  2.3230764
##  [6,]  0.3858715  2.2670716  4.123623  18.919984 0.011882229  1.4952074
##  [7,]  1.1649634  3.4320350 13.564170  32.484154 0.016794120  1.5166393
##  [8,]  1.4556002  4.8876352 17.212956  49.697110 0.022058875  4.2260915
##  [9,]  1.8881781  6.7758133 22.950463  72.647573 0.027710360  2.6191788
## [10,]  1.4629272  8.2387405 18.499115  91.146688 0.031323406  3.0216906
## [11,]  0.9245499  9.1632904 11.954227 103.100915 0.033323954  2.3039211
## [12,]  0.7279731  9.8912636  9.465354 112.566269 0.034774341  2.9496472
## [13,]  3.8802746 13.7715382 58.894807 171.461075 0.040110936  3.9059524
## [14,]  3.8729113 17.6444495 60.852316 232.313392 0.043654005  4.8390553
## [15,]  1.6006928 19.2451424 27.021176 259.334567 0.044630584  2.1094253
## [16,]  0.4741907 19.7193331  8.516837 267.851405 0.044844533  0.7701956
## [17,]  4.1603608 23.8796939 75.197381 343.048786 0.046375185  1.2719772
## [18,]  1.6780403 25.5577343 32.248059 375.296844 0.046708775  1.2577795
## [19,]  1.5995633 27.1572976 32.602034 407.898879 0.046841175  0.8024690
## [20,]  0.7978867 27.9551842 16.504361 424.403239 0.046882771  0.9064515
## [21,]  0.8510847 28.8062689 18.253467 442.656706 0.046875142  0.5132515
## [22,]  1.6212440 30.4275129 46.441627 489.098333 0.046034391 -0.5941380
## [23,]  1.8223143 32.2498272 53.502309 542.600643 0.045137745 -0.7608069
## [24,]  1.4275116 33.6773388 42.226266 584.826909 0.044505413 -0.6204259
## [25,]  0.9495263 34.6268651 33.688890 618.515799 0.043809793 -1.1349771
## [26,]  1.0801865 35.7070516 48.368779 666.884578 0.042571251 -1.1157305
## [27,]  0.6711982 36.3782498 34.008337 700.892915 0.041681462 -1.7388582
## [28,]  0.5711265 36.9493762 31.549899 732.442814 0.040858832 -1.7599572
## [29,] -0.1224536 36.8269226 19.973933 752.416747 0.039843389 -3.3289141
##           x_short
##  [1,]  0.09396519
##  [2,]  0.04740818
##  [3,]  0.10440212
##  [4,]  0.06551340
##  [5,]  0.06206517
##  [6,]  0.03994716
##  [7,]  0.04051975
##  [8,]  0.11290765
##  [9,]  0.06997608
## [10,]  0.08072991
## [11,]  0.06155340
## [12,]  0.07880514
## [13,]  0.10435456
## [14,]  0.12928408
## [15,]  0.05635710
## [16,]  0.02057716
## [17,]  0.03398316
## [18,]  0.03360385
## [19,]  0.02143940
## [20,]  0.02421749
## [21,]  0.01371244
## [22,] -0.01587347
## [23,] -0.02032633
## [24,] -0.01657580
## [25,] -0.03032296
## [26,] -0.02980875
## [27,] -0.04645673
## [28,] -0.04702043
## [29,] -0.08893794
```

```
cat("The final weights for the optimal portfolio with a Rf of 0.001 with short sales allowed is:", x_short)
```

```
## The final weights for the optimal portfolio with a Rf of 0.001 with short sales allowed is: 0.09396519 0.04740
818 0.1044021 0.0655134 0.06206517 0.03994716 0.04051975 0.1129077 0.06997608 0.08072991 0.0615534 0.07880514 0.1
043546 0.1292841 0.0563571 0.02057716 0.03398316 0.03360385 0.0214394 0.02421749 0.01371244 -0.01587347 -0.020326
33 -0.0165758 -0.03032296 -0.02980875 -0.04645673 -0.04702043 -0.08893794
```

```
cat("Check that the sum of the weights is 1:", sum(x_short))
```

```
## Check that the sum of the weights is 1: 1
```

(b)

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

```
a1 <- read.csv("stockData2015.csv", sep=",", header=TRUE)

a2 <- read.csv("stockData2023.csv", sep=",", header=TRUE)

spy <- read.csv("SPY2.csv", sep=",", header=TRUE)

#Convert adjusted close prices into returns:
r1 <- (a1[-1,3:ncol(a1)]-a1[-nrow(a1),3:ncol(a1)])/a1[-nrow(a1),3:ncol(a1)]

r2 <- (a2[-1,3:ncol(a2)]-a2[-nrow(a2),3:ncol(a2)])/a2[-nrow(a2),3:ncol(a2)]

rm <- (spy[-1,3:ncol(spy)]-spy[-nrow(spy),3:ncol(spy)])/spy[-nrow(spy),3:ncol(spy)]


#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]

#Adjust betas using the Blume's technique:
q1 <- lm(beta2 ~ beta1)
beta3adj_blume <- q1$coef[1] + q1$coef[2]*beta2


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

alpha2 <- rep(0,30)

sigma_e2 <- rep(0,30)

var_beta2 <- rep(0,30)

for(i in 1:30){
    q <- lm(r1[,i] ~ rm)
    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:
beta2adj_vasicek <- var_beta2*mean(beta2)/(var(beta2)+var_beta2) +
var(beta2)*beta2/(var(beta2)+var_beta2)

cat("The adjusted beta's using Blume's method:", beta3adj_blume, "\n")
```

```
## The adjusted beta's using Blume's method: 0.6513148 0.892809 0.5711414 0.5688403 0.61895 0.5640408 0.6213842
0.5097284 0.6305051 0.6490074 0.7178222 0.6165097 0.4907285 0.6308642 0.6193694 0.4756154 0.6932448 0.8260563 0.5
747907 0.7773759 0.695182 0.786382 0.6409142 0.6555884 0.6968731 0.401755 0.4501105 0.5787541 0.7510426
```

```
cat("The adjusted beta's using Vasicek's method:", beta2adj_vasicek)
```

```
## The adjusted beta's using Vasicek's method: 1.187177 1.133133 1.501654 1.127137 0.8474486 1.461058 0.8106808
0.9248067 1.121134 1.250505 1.466978 1.353996 0.848751 0.4517128 0.5293814 0.9976449 0.3722865 0.6375478 1.467466
0.6108178 1.001254 1.323176 0.8431415 0.8120955 1.04748 1.06334 0.5153818 0.5742489 1.02091 0.9936535
```

(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.)

```
PRESS <- sum((beta2adj_vasicek-beta2)^2) / 30
PRESS
```

```
## [1] 0.01498543
```