2.6 Principle component regression (PCR)

Task

Run PCR with the same features and response as in Section 2.5. Use the first 70% data as training data and the last 30% data as validation data. Use the validation MSE to seek the optimal number of principal components to include in PCR and generate the corresponding prediction for the whole year. Report the in-sample and out-of-sample correlation between your prediction and true response.

Load required libraries

```
library(pls)
```

```
## Warning: package 'pls' was built under R version 4.0.3
```

Calculate backward returns

```
data_pcr <- read_csv("data/final_project-1.csv")
colnames(data_pcr) <- c("X", "Asset_1", "Asset_2", "Asset_3")

time.horizons = c(3,10,30,60,120,180,240,360,480,600,720,960,1200,1440)

for (time in time.horizons) {
  data_pcr = mutate(data_pcr,
                    "Asset_1_BRet_{time}" :=
                      (data_pcr$Asset_1 - ifelse(X > (time-1),
                                                 lag(data_pcr$Asset_1, n=time),
                                                 data_pcr$Asset_1[1])) /
                      ifelse(X > (time-1), lag(data_pcr$Asset_1, n=time),
                             data_pcr$Asset_1[1]))
  data_pcr = mutate(data_pcr,
                    "Asset_2_BRet_{time}" :=
                      (data_pcr$Asset_2 - ifelse(X > (time-1),
                                                 lag(data_pcr$Asset_2, n=time),
                                                 data_pcr$Asset_2[1])) /
                      ifelse(X > (time-1), lag(data_pcr$Asset_2, n=time),
                             data_pcr$Asset_2[1]))
  data_pcr = mutate(data_pcr,
                    "Asset_3_BRet_{time}" :=
                      (data_pcr$Asset_3 - ifelse(X > (time-1),
                                                 lag(data_pcr$Asset_3, n=time),
                                                 data_pcr$Asset_3[1])) /
                      ifelse(X > (time-1), lag(data_pcr$Asset_3, n=time),
                             data_pcr$Asset_3[1]))
}
```

Calculate response

```
Asset_1 <- data_pcr %>% select(Asset_1)
Asset_1_lead <- lead(Asset_1, n=10, default=tail(Asset_1, 1))
Asset_1_HRet_10 <- (Asset_1_lead - Asset_1) / Asset_1
colnames(Asset_1_HRet_10) <- c("Asset_1_HRet_10")
data_pcr <- cbind(data_pcr, Asset_1_HRet_10)
```

Prepare data

```
# remove index column and underlying asset prices
data_pcr <- data_pcr[,-c(1:4)]

# using first 70% as train data
train_id <- 1:floor(nrow(data_pcr) * 0.7)
```

Train Principal Component Regression (PCR) Model

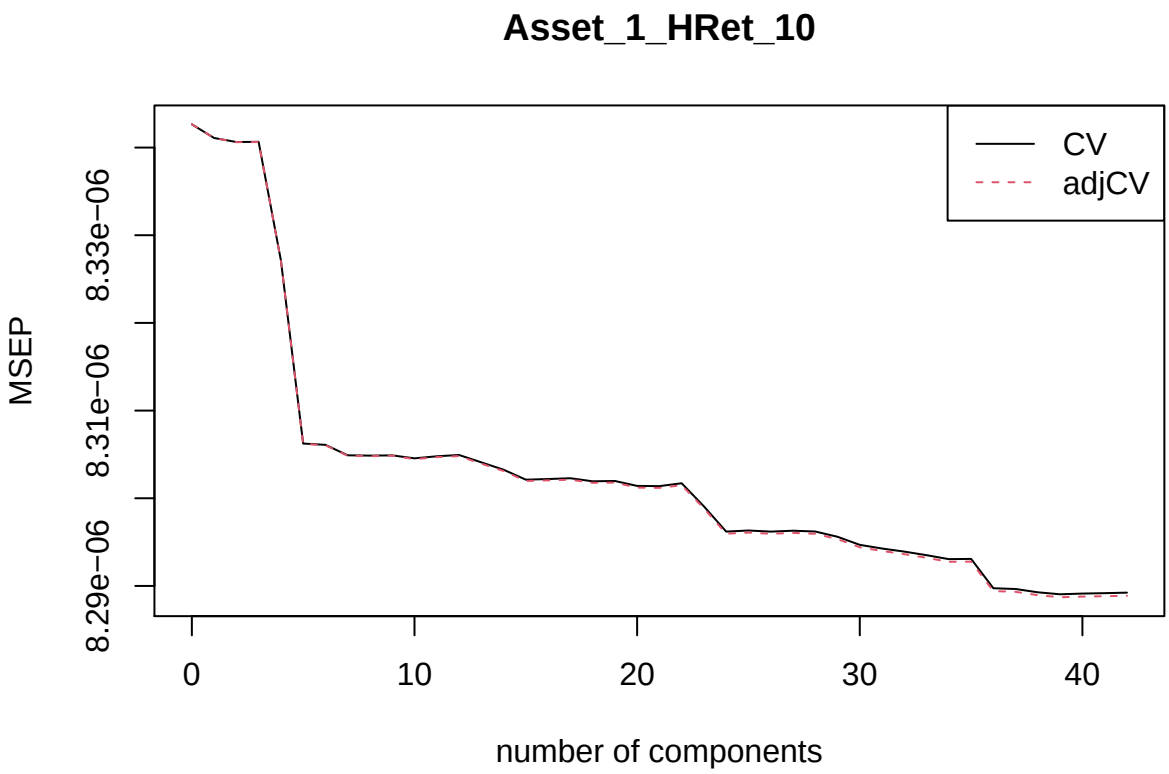
```
# train model without restrictions
returnPCR <- pcr(Asset_1_HRet_10 ~ ., data = data_pcr, subset = train_id,
                 scale = T, validation = "CV")

# evaluate model
summary(returnPCR)
```

```
## Data:    X dimension: 366912 42
##  Y dimension: 366912 1
## Fit method: svdpc
## Number of components considered: 42
##
## VALIDATION: RMSEP
## Cross-validated using 10 random segments.
##        (Intercept)   1 comps   2 comps   3 comps   4 comps   5 comps   6 comps
## CV        0.002888  0.002888  0.002888  0.002888  0.002886  0.002882  0.002882
## adjCV     0.002888  0.002888  0.002888  0.002888  0.002886  0.002882  0.002882
##         7 comps   8 comps   9 comps  10 comps  11 comps  12 comps  13 comps
## CV     0.002882  0.002882  0.002882  0.002882  0.002882  0.002882  0.002882
## adjCV  0.002882  0.002882  0.002882  0.002882  0.002882  0.002882  0.002882
##        14 comps  15 comps  16 comps  17 comps  18 comps  19 comps  20 comps
## CV     0.002882  0.002881  0.002881  0.002881  0.002881  0.002881  0.002881
## adjCV  0.002882  0.002881  0.002881  0.002881  0.002881  0.002881  0.002881
##        21 comps  22 comps  23 comps  24 comps  25 comps  26 comps  27 comps
## CV     0.002881  0.002881  0.002881   0.00288   0.00288   0.00288   0.00288
## adjCV  0.002881  0.002881  0.002881   0.00288   0.00288   0.00288   0.00288
##        28 comps  29 comps  30 comps  31 comps  32 comps  33 comps  34 comps
## CV      0.00288   0.00288   0.00288   0.00288   0.00288   0.00288   0.00288
## adjCV   0.00288   0.00288   0.00288   0.00288   0.00288   0.00288   0.00288
##        35 comps  36 comps  37 comps  38 comps  39 comps  40 comps  41 comps
```

```
## CV       0.00288  0.002879  0.002879  0.002879  0.002879  0.002879  0.002879
## adjCV    0.00288  0.002879  0.002879  0.002879  0.002879  0.002879  0.002879
##        42 comps
## CV      0.002879
## adjCV   0.002879
##
## TRAINING: % variance explained
##                    1 comps   2 comps   3 comps  4 comps  5 comps  6 comps
## X                 28.93112  40.73804  51.85405  61.6213  66.9560  70.8909
## Asset_1_HRet_10    0.01979   0.02712   0.02781   0.1925   0.4446   0.4487
##                   7 comps  8 comps  9 comps  10 comps  11 comps  12 comps
## X                 74.7715  77.8994   80.137   82.3562   84.5361   86.0183
## Asset_1_HRet_10    0.4645   0.4667    0.467    0.4741    0.4741    0.4797
##                   13 comps  14 comps  15 comps  16 comps  17 comps  18 comps
## X                  87.3321   88.6215   89.7789   90.6870   91.5769    92.433
## Asset_1_HRet_10     0.4937    0.5043    0.5193    0.5194    0.5211     0.527
##                   19 comps  20 comps  21 comps  22 comps  23 comps  24 comps
## X                   93.164   93.8542   94.4634   95.0470    95.522   95.9672
## Asset_1_HRet_10      0.529    0.5378    0.5399    0.5401     0.573    0.6093
##                   25 comps  26 comps  27 comps  28 comps  29 comps  30 comps
## X                  96.4036   96.8043   97.1620   97.4937   97.8005   98.0774
## Asset_1_HRet_10     0.6093    0.6119    0.6119    0.6148    0.6242    0.6365
##                   31 comps  32 comps  33 comps  34 comps  35 comps  36 comps
## X                  98.3418   98.6005   98.8124   99.0146   99.1859   99.3388
## Asset_1_HRet_10     0.6432    0.6499    0.6568    0.6637    0.6647    0.7051
##                   37 comps  38 comps  39 comps  40 comps  41 comps  42 comps
## X                   99.486   99.6192   99.7297   99.8256   99.9192   100.000
## Asset_1_HRet_10      0.709    0.7138    0.7188    0.7188    0.7195     0.721
```

```
validationplot(returnPCR, val.type = "MSEP", legendpos = "topright")
```



We find that 11 components collectively explain 95% of variance in response variable and that we get the lowest mean squared error of prediction (from 10-fold cross validation, adjusted) when using most or all of the 42 components. We can also test different values for ncomps with the training/test split we defined earlier:

```
ncompss = 1:42
cors <- vector(length = length(ncompss))
for (i in 1:length(ncompss)) {
  returnPCR.pred.test <- predict(returnPCR,
                                 data_pcr[-train_id, names(data_pcr)
                                          != 'Asset_1_HRet_10'],
                                 ncomp = ncompss[i])
  cors[i] <- cor(returnPCR.pred.test,
                 data_pcr[-train_id, names(data_pcr) == 'Asset_1_HRet_10'])
}
paste("Highest out-of-sample correlation for PCR:",
      round(max(cors),4), "(for", which.max(cors), "components)")
```

```
## [1] "Highest out-of-sample correlation for PCR: 0.0416 (for 6 components)"
```

Based on the CV and the test error - and mindful of the computational cost of our prediction - we decided to go with a relatively simple principal component regression model that used six components.

Evaluate Model

```
# predict train data using PCR with 22 principal components
returnPCR.pred.train <- predict(returnPCR, data_pcr[train_id, names(data_pcr)
                                                    != 'Asset_1_HRet_10'],
                                ncomp = 6)

# Calculate in-sample correlation
PCR.cor.train <- cor(returnPCR.pred.train, data_pcr[train_id, names(data_pcr)
                                                    == 'Asset_1_HRet_10'])
paste("In-sample correlation for PCR:", round(PCR.cor.train,4))
```

```
## [1] "In-sample correlation for PCR: 0.067"
```

```
# predict test data using PCR with 22 principal components
returnPCR.pred.test <- predict(returnPCR, data_pcr[-train_id, names(data_pcr)
                                                    != 'Asset_1_HRet_10'],
                               ncomp = 6)

# Calculate out-of-sample correlation
PCR.cor.test <- cor(returnPCR.pred.test, data_pcr[-train_id, names(data_pcr)
                                                   == 'Asset_1_HRet_10'])
paste("Out-of-sample correlation for PCR:", round(PCR.cor.test,4))
```

```
## [1] "Out-of-sample correlation for PCR: 0.0416"
```