lab 06 PLS Solution

Partial Least Squares Regression (PLS)

Load data & remove NA

```
library(ISLR)
library(pls)
```

```
##
## Attaching package: 'pls'
```

```
## The following object is masked from 'package:stats':
##
##     loadings
```

```
str(Hitters, vec.len = 1)
```

```
## 'data.frame':    322 obs. of  20 variables:
##  $ AtBat    : int  293 315 ...
##  $ Hits     : int  66 81 ...
##  $ HmRun    : int  1 7 ...
##  $ Runs     : int  30 24 ...
##  $ RBI      : int  29 38 ...
##  $ Walks    : int  14 39 ...
##  $ Years    : int  1 14 ...
##  $ CAtBat   : int  293 3449 ...
##  $ CHits    : int  66 835 ...
##  $ CHmRun   : int  1 69 ...
##  $ CRuns    : int  30 321 ...
##  $ CRBI     : int  29 414 ...
##  $ CWalks   : int  14 375 ...
##  $ League   : Factor w/ 2 levels "A","N": 1 2 ...
##  $ Division : Factor w/ 2 levels "E","W": 1 2 ...
##  $ PutOuts  : int  446 632 ...
##  $ Assists  : int  33 43 ...
##  $ Errors   : int  20 10 ...
##  $ Salary   : num  NA 475 ...
##  $ NewLeague: Factor w/ 2 levels "A","N": 1 2 ...
```

```
Hitters <- na.omit(Hitters)
Hitters_mat <- model.matrix(Salary ~ ., Hitters)
X <- scale(Hitters_mat[, -1], center = T, scale = T)
Y <- scale(Hitters$Salary, center = T, scale = T)
rankX <- qr(X)$rank
# alternatice way
# require(Matrix); rankMatrix(X)[1]
```

partial least squares regression (without CV)

```
pls_fit <- plsr(Salary ~ ., data = Hitters, scale = TRUE, validation = "none")
names(pls_fit)
```

```
##  [1] "coefficients"    "scores"          "loadings"
##  [4] "loading.weights" "Yscores"         "Yloadings"
##  [7] "projection"      "Xmeans"          "Ymeans"
## [10] "fitted.values"   "residuals"       "Xvar"
## [13] "Xtotvar"         "fit.time"        "ncomp"
## [16] "method"          "scale"           "call"
## [19] "terms"           "model"
```

3.1) First iteration in PLSR

```
X0 <- X; Y0 <- Y
scalar1 <- function(x) {x / sqrt(sum(x^2))}  # normalize
(w1 <- scalar1(t(X0) %*% Y0))
```

```
##                    [,1]
## AtBat       0.225613698
## Hits        0.250704950
## HmRun       0.196042374
## Runs        0.239951404
## RBI         0.256867120
## Walks       0.253672504
## Years       0.228977607
## CAtBat      0.300689133
## CHits       0.313704738
## CHmRun      0.300000612
## CRuns       0.321573310
## CRBI        0.324023910
## CWalks      0.279935903
## LeagueN    -0.008162140
## DivisionW  -0.110023005
## PutOuts     0.171726124
## Assists     0.014536887
## Errors     -0.003086530
## NewLeagueN -0.001619909
```

```
# Obtain the first PLS component
z1 <- X0 %*% w1
head(z1)
```

```
##                         [,1]
## -Alan Ashby       -0.1090169
## -Alvin Davis       0.6670947
## -Andre Dawson      3.4717021
## -Andres Galarraga -2.1298594
## -Alfredo Griffin   0.9770842
## -Al Newman        -4.0036686
```

```
# Obtain a vector p1 of loadings
(p1 <- t(X0) %*% z1 / as.numeric(t(z1) %*% z1))
```

```
##                   [,1]
## AtBat       0.225618535
## Hits        0.223197232
## HmRun       0.217916095
## Runs        0.224969645
## RBI         0.256635914
## Walks       0.229200091
## Years       0.266002421
## CAtBat      0.319851627
## CHits       0.321135571
## CHmRun      0.311269101
## CRuns       0.329115965
## CRBI        0.331190018
## CWalks      0.306858535
## LeagueN    -0.046244531
## DivisionW  -0.039992026
## PutOuts     0.099952296
## Assists     0.009595614
## Errors      0.004863680
## NewLeagueN -0.032327743
```

```
# Comparison
all.equal(as.numeric(pls_fit$loading.weights[,1]), as.numeric(w1))
```

```
## [1] TRUE
```

```
all.equal(as.numeric(pls_fit$scores[, 1]), as.numeric(z1))
```

```
## [1] TRUE
```

```
all.equal(as.numeric(pls_fit$loadings[, 1]), as.numeric(p1))
```

```
## [1] TRUE
```

```
# Obtain regression coefficient d1
(b1 <- t(Y0) %*% z1 / as.numeric(t(z1) %*% z1))
```

```
##           [,1]
## [1,] 0.2460445
```

```
yhat <- z1 %*% b1
# Convert back to original scale
yhat_org <- mean(Hitters$Salary) + sd(Hitters$Salary) * yhat
all.equal(as.numeric(pls_fit$fitted.values[, , 1]), as.numeric(yhat_org))
```

```
## [1] TRUE
```

3.2) Implement the PLSR algorithm

```
weights <- loadings <- matrix(NA, nrow = NCOL(X), ncol = rankX)
components <-  matrix(NA, nrow = NROW(X), ncol = rankX)
coefficients <- rep(NA, rankX)
fitted <- fitted_org <- rep(0, NCOL(X))
colnames(weights) <- colnames(loadings) <- colnames(components) <- colnames(X)
X_temp <- X; Y_temp <- Y
```

```r
for (h in 1:rankX) {
  w <- scalar1(t(X_temp) %*% Y_temp)
  z <- X_temp %*% w
  p <- t(X_temp) %*% z / as.numeric(t(z) %*% z)
  b <- t(Y_temp) %*% z / as.numeric(t(z) %*% z)
  yhat <- z %*% b
  weights[, h] <- w
  components[, h] <- z
  loadings[, h] <- p
  coefficients[h] <- b
  fitted <- yhat + fitted

  # Iterative steps
  X_temp <- X_temp - z %*% t(p)
  Y_temp <- Y_temp - z %*% b
}
```

```
## Warning in yhat + fitted: longer object length is not a multiple of shorter
## object length
```

```r
fitted_org <- mean(Hitters$Salary) + sd(Hitters$Salary) * fitted
all.equal(as.numeric(pls_fit$fitted.values[, , 19]), as.numeric(fitted_org))
```

```
## [1] TRUE
```