

## 6.7 Lab: PCR and PLS Regression

Zongyi Liu

2023-05-31

### 6.7 Lab: PCR and PLS Regression

#### 6.7.1 Principal Components Regression

Principal Components Regression (PCR) can be performed using the `pcr()` function.

```
library(ISLR)
Hitters=na.omit(Hitters)
x=model.matrix(Salary~.,Hitters)[,-1]
y=Hitters$Salary
```

```
set.seed(1)
train=sample(1:nrow(x), nrow(x)/2)
test=(-train)
y.test=y[test]
```

```
library(pls)
```

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

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

```
set.seed(2)
pcr.fit=pcr(Salary~., data=Hitters,scale=TRUE,validation="CV")
summary(pcr.fit)
```

```
## Data:      X dimension: 263 19
## Y dimension: 263 1
## Fit method: svdpc
## Number of components considered: 19
##
## VALIDATION: RMSEP
## Cross-validated using 10 random segments.
##      (Intercept)  1 comps  2 comps  3 comps  4 comps  5 comps  6 comps
## CV              452    351.9    353.2    355.0    352.8    348.4    343.6
```

```
## adjCV      452    351.6    352.7    354.4    352.1    347.6    342.7
##          7 comps  8 comps  9 comps 10 comps 11 comps 12 comps 13 comps
## CV        345.5    347.7    349.6    351.4    352.1    353.5    358.2
## adjCV      344.7    346.7    348.5    350.1    350.7    352.0    356.5
##          14 comps 15 comps 16 comps 17 comps 18 comps 19 comps
## CV        349.7    349.4    339.9    341.6    339.2    339.6
## adjCV      348.0    347.7    338.2    339.7    337.2    337.6
##
## TRAINING: % variance explained
##          1 comps  2 comps  3 comps  4 comps  5 comps  6 comps  7 comps  8 comps
## X          38.31    60.16    70.84    79.03    84.29    88.63    92.26    94.96
## Salary     40.63    41.58    42.17    43.22    44.90    46.48    46.69    46.75
##          9 comps 10 comps 11 comps 12 comps 13 comps 14 comps 15 comps
## X          96.28    97.26    97.98    98.65    99.15    99.47    99.75
## Salary     46.86    47.76    47.82    47.85    48.10    50.40    50.55
##          16 comps 17 comps 18 comps 19 comps
## X          99.89    99.97    99.99    100.00
## Salary     53.01    53.85    54.61    54.61
```

We can examine its summeries:

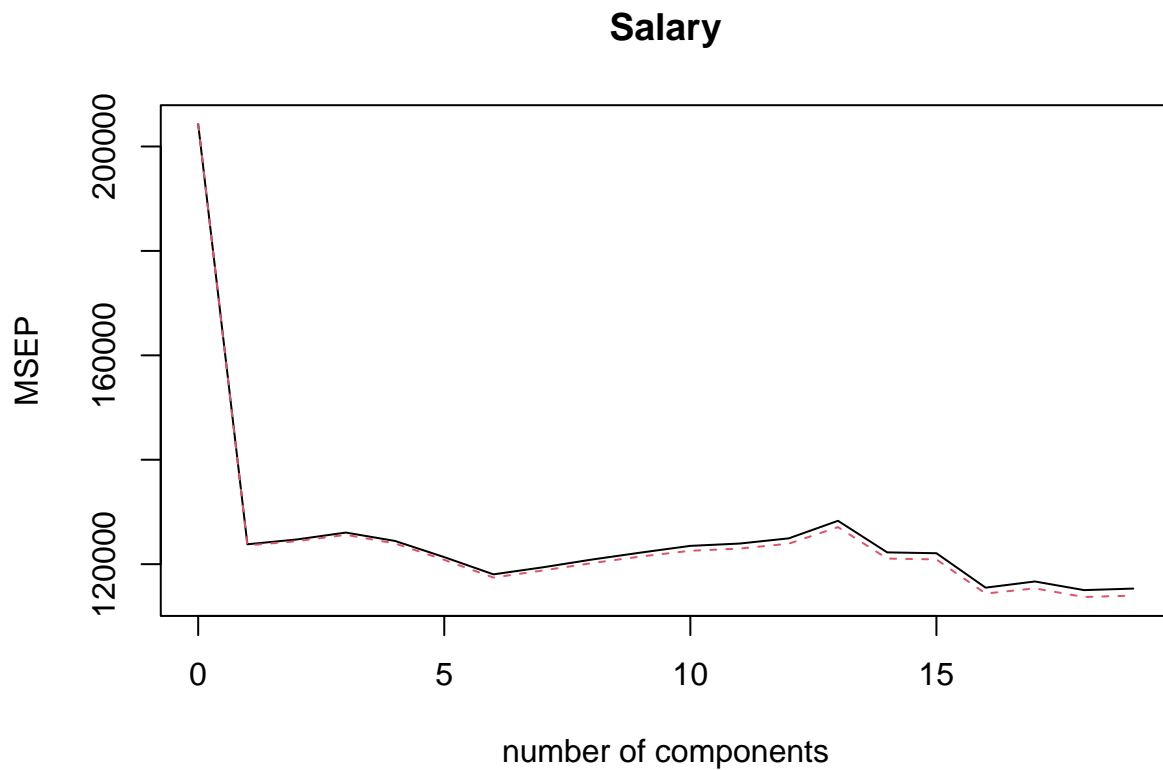
```
summary(pcr.fit)
```

```
## Data:      X dimension: 263 19
## Y dimension: 263 1
## Fit method: svdpc
## Number of components considered: 19
##
## VALIDATION: RMSEP
## Cross-validated using 10 random segments.
##      (Intercept)  1 comps  2 comps  3 comps  4 comps  5 comps  6 comps
## CV              452    351.9    353.2    355.0    352.8    348.4    343.6
## adjCV           452    351.6    352.7    354.4    352.1    347.6    342.7
##          7 comps  8 comps  9 comps 10 comps 11 comps 12 comps 13 comps
## CV        345.5    347.7    349.6    351.4    352.1    353.5    358.2
## adjCV      344.7    346.7    348.5    350.1    350.7    352.0    356.5
##          14 comps 15 comps 16 comps 17 comps 18 comps 19 comps
## CV        349.7    349.4    339.9    341.6    339.2    339.6
## adjCV      348.0    347.7    338.2    339.7    337.2    337.6
##
## TRAINING: % variance explained
##          1 comps  2 comps  3 comps  4 comps  5 comps  6 comps  7 comps  8 comps
## X          38.31    60.16    70.84    79.03    84.29    88.63    92.26    94.96
## Salary     40.63    41.58    42.17    43.22    44.90    46.48    46.69    46.75
##          9 comps 10 comps 11 comps 12 comps 13 comps 14 comps 15 comps
## X          96.28    97.26    97.98    98.65    99.15    99.47    99.75
## Salary     46.86    47.76    47.82    47.85    48.10    50.40    50.55
##          16 comps 17 comps 18 comps 19 comps
## X          99.89    99.97    99.99    100.00
## Salary     53.01    53.85    54.61    54.61
```

The CV score is provided for each possible number of components, ranging from  $M = 0$  onwards.

To plot the model and cv scores, we can use the `validationplot()` function. Using `val.type="MSEP"` will cause the cross-validation MSE to be plotted.

```
validationplot(pcr.fit, val.type="MSEP")
```

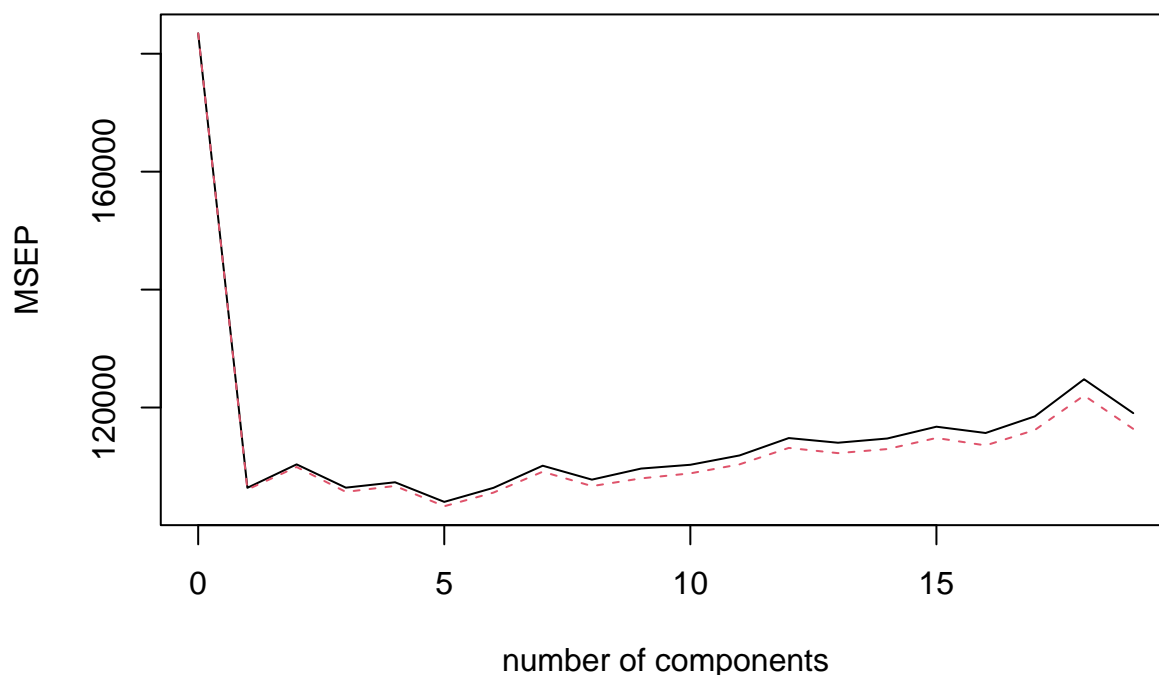


Here we can see that the smallest cv-error occurs when  $M=16$ .

We now perform PCR on the training data and evaluate its test set performance.

```
set.seed(1)
pcr.fit=pcr(Salary~., data=Hitters, subset=train, scale=TRUE, validation="CV")
validationplot(pcr.fit, val.type="MSEP")
```

## Salary



Here the lowest cv-error occurs when  $M=7$ , thus, we will have:

```
pcr.pred=predict(pcr.fit,x[test,],ncomp=7)
mean((pcr.pred-y.test)^2)
```

```
## [1] 140751.3
```

Finally, we fit PCR on the full data set, using  $M = 7$ , the number of components identified by cross-validation.

```
pcr.fit=pcr(y~x,scale=TRUE,ncomp=7)
summary(pcr.fit)
```

```
## Data:      X dimension: 263 19
## Y dimension: 263 1
## Fit method: svdpc
## Number of components considered: 7
## TRAINING: % variance explained
##   1 comps 2 comps 3 comps 4 comps 5 comps 6 comps 7 comps
## X   38.31  60.16  70.84  79.03  84.29  88.63  92.26
## y   40.63  41.58  42.17  43.22  44.90  46.48  46.69
```

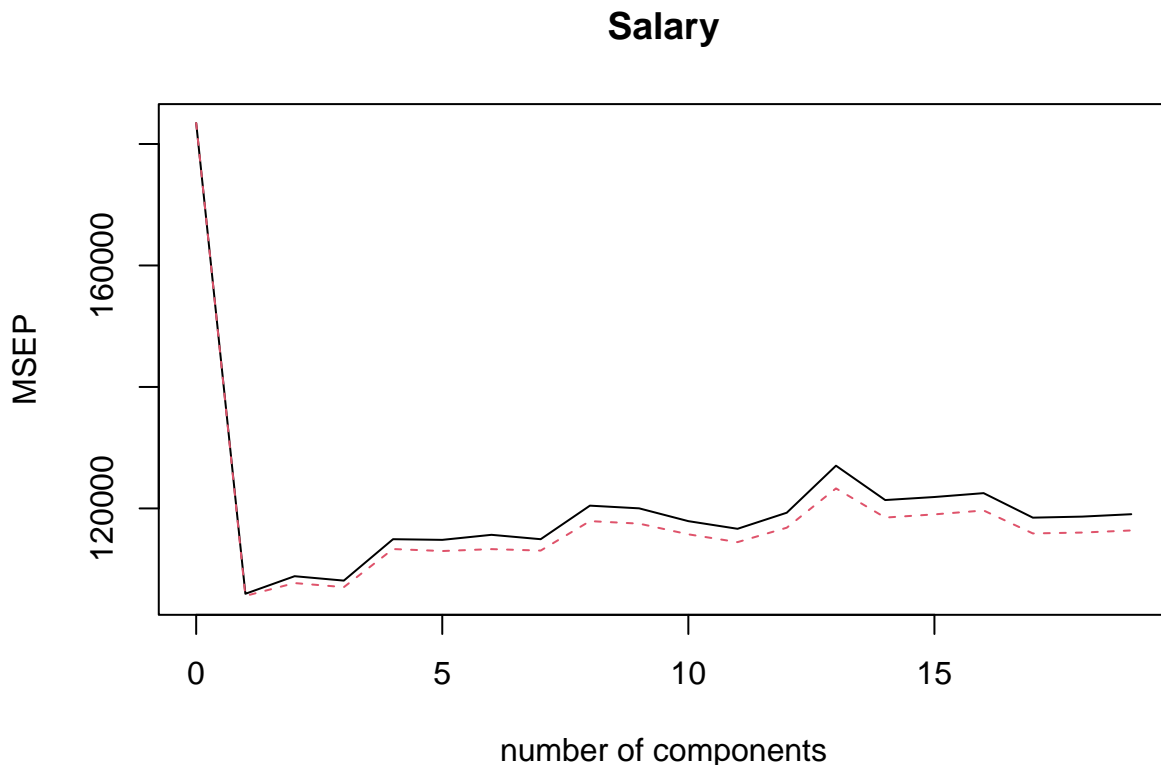
### 6.7.2 Partial Least Squares

Here we will use `pls()` to do the Partial Least Squares (PLS).

```
set.seed(1)
pls.fit=pls(Salary~., data=Hitters,subset=train,scale=TRUE, validation="CV")
summary(pls.fit)
```

```
## Data:      X dimension: 131 19
## Y dimension: 131 1
## Fit method: kernelppls
## Number of components considered: 19
##
## VALIDATION: RMSEP
## Cross-validated using 10 random segments.
##      (Intercept)  1 comps  2 comps  3 comps  4 comps  5 comps  6 comps
## CV           428.3   325.5   329.9   328.8   339.0   338.9   340.1
## adjCV         428.3   325.0   328.2   327.2   336.6   336.1   336.6
##      7 comps  8 comps  9 comps 10 comps 11 comps 12 comps 13 comps
## CV           339.0   347.1   346.4   343.4   341.5   345.4   356.4
## adjCV         336.2   343.4   342.8   340.2   338.3   341.8   351.1
##      14 comps 15 comps 16 comps 17 comps 18 comps 19 comps
## CV           348.4   349.1   350.0   344.2   344.5   345.0
## adjCV         344.2   345.0   345.9   340.4   340.6   341.1
##
## TRAINING: % variance explained
##      1 comps  2 comps  3 comps  4 comps  5 comps  6 comps  7 comps  8 comps
## X           39.13   48.80   60.09   75.07   78.58   81.12   88.21   90.71
## Salary       46.36   50.72   52.23   53.03   54.07   54.77   55.05   55.66
##      9 comps 10 comps 11 comps 12 comps 13 comps 14 comps 15 comps
## X           93.17   96.05   97.08   97.61   97.97   98.70   99.12
## Salary       55.95   56.12   56.47   56.68   57.37   57.76   58.08
##      16 comps 17 comps 18 comps 19 comps
## X           99.61   99.70   99.95  100.00
## Salary       58.17   58.49   58.56   58.62
```

```
validationplot(pls.fit, val.type="MSEP")
```



The lowest cv-error occurs when  $M=2$ .

Finally, we perform PLS using the full data set, using  $M = 2$ , the number of components identified by cross-validation.

```
pls.pred=predict(pls.fit,x[test,],ncomp=2)
mean((pls.pred-y.test)^2)
```

```
## [1] 145367.7
```

```
pls.fit=plsr(Salary~., data=Hitters,scale=TRUE,ncomp=2)
summary(pls.fit)
```

```
## Data:      X dimension: 263 19
## Y dimension: 263 1
## Fit method: kernelpls
## Number of components considered: 2
## TRAINING: % variance explained
##           1 comps  2 comps
## X           38.08   51.03
## Salary      43.05   46.40
```