STAT 534: Homework 4 Solution

Fall 2021

Due: Friday, October 8

1. Using the *Boston* dataset from the *MASS* package, the goal is to predict the crime rate by the other variables.

   (a) Create bivariate plots to explore relations between variables. Comment on your observations. (Hint: you may use ggpair() and remember to factorize any categorical variables.)

```
library(MASS)
data(Boston)
Boston$chas <- as.factor(Boston$chas)
library(GGally)
ggpairs(data = Boston,
        upper = list(continuous = wrap("cor", size = 2.5))
)
```

   The plot is omitted. The distribution of the crime rate is clearly right-skewed, which violates the normality assumption of multiple linear regression. Some of the predictors are highly correlated. For example, dis is highly correlated with indus, nox, age; tax is hightly correlated with indus.

   (b) Log-transform the crime rate and repeat part (a).

```
Boston$crim <- log(Boston$crim)
ggpairs(data = Boston,
        upper = list(continuous = wrap("cor", size = 2.5))
)
```

   The plot is omitted. The crime rate is more correlated with some predictors after the log transformation.

   (c) Create a correlation matrix of all the continuous variables and make comments. (Hint: you may use ggcorr().)

```
ggcorr(data = Boston[,-4],
       method = c("pairwise.complete.obs", "pearson"),
       label = TRUE, label_size = 4)
```

   The plot is omitted. The multicollinearity among predictors is more obvious using the correlation matrix. In addition, the crime rate is highly correlated with indus, nox, age (negatively), dis, rad and tax.

   (d) Split the data into training and testing subsets.

```
n <- nrow(Boston)
set.seed(1)
train <- sample(1:n, n/2)
```

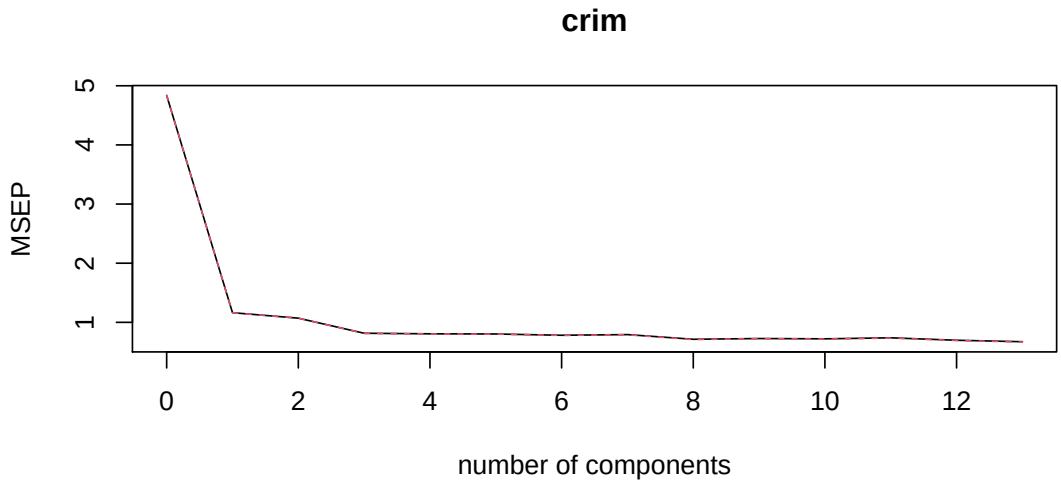
   (e) Build the following models using the training set:

   - multiple linear regression
   - principal components regression (indicate how many principal components are selected)
   - partial least squares (indicate how many directions are selected)
   - lasso

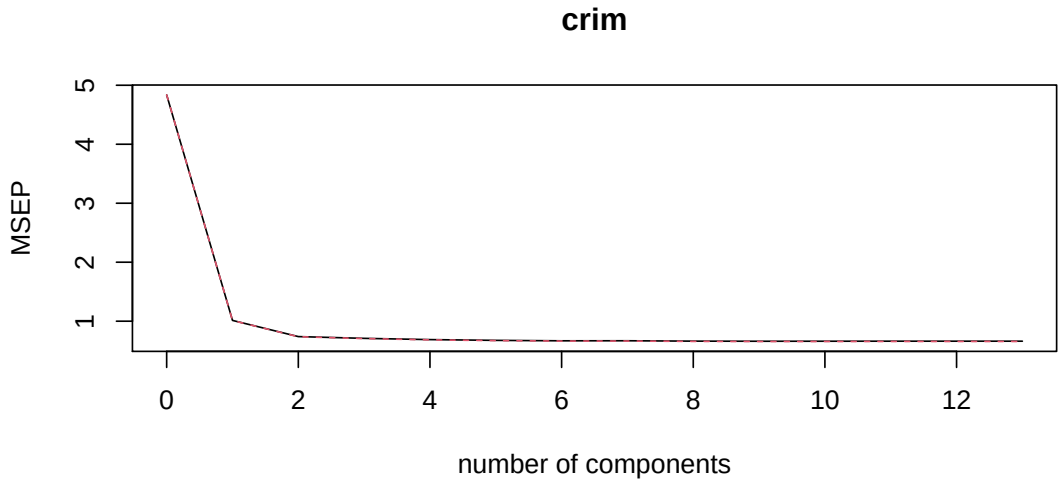
```r
#mlr
mlr_fit <- lm(crim~., data = Boston, subset = train)
summary(mlr_fit)
```

```
##
## Call:
## lm(formula = crim ~ ., data = Boston, subset = train)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.31007 -0.51732 -0.03626  0.48879  1.95731
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -5.1241547  1.2751335  -4.019 7.85e-05 ***
## zn          -0.0113677  0.0033005  -3.444 0.000676 ***
## indus        0.0264042  0.0136422   1.935 0.054112 .
## chas1       -0.0136987  0.2071427  -0.066 0.947328
## nox          3.8071425  0.9273673   4.105 5.54e-05 ***
## rm          -0.0579030  0.1079929  -0.536 0.592337
## age          0.0074932  0.0030783   2.434 0.015659 *
## dis          0.0574569  0.0501635   1.145 0.253192
## rad          0.1422165  0.0154881   9.182  < 2e-16 ***
## tax         -0.0001540  0.0008831  -0.174 0.861737
## ptratio      0.0055181  0.0333428   0.165 0.868695
## black       -0.0017982  0.0006242  -2.881 0.004326 **
## lstat        0.0410325  0.0134870   3.042 0.002609 **
## medv         0.0187922  0.0115422   1.628 0.104817
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.7826 on 239 degrees of freedom
## Multiple R-squared:  0.8794,Adjusted R-squared:  0.8729
## F-statistic: 134.1 on 13 and 239 DF,  p-value: < 2.2e-16
```

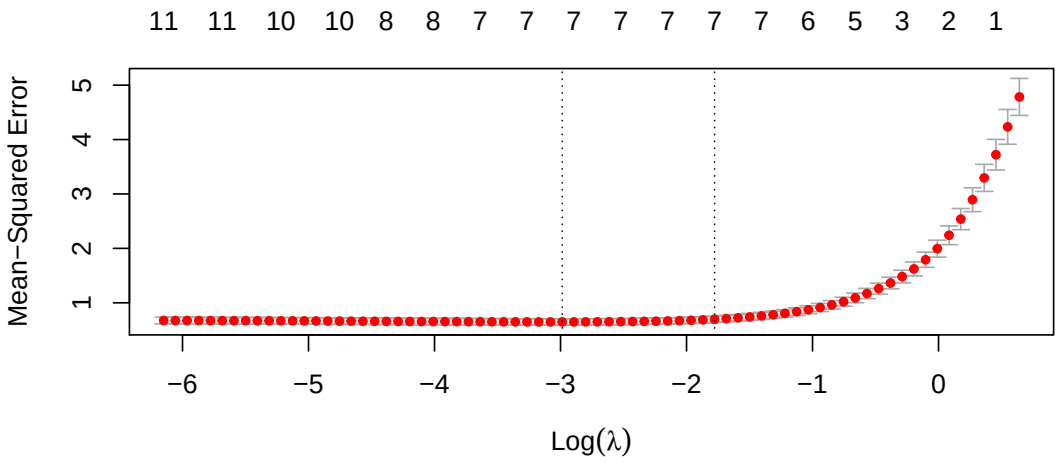
```r
#pcr
set.seed(1)
library(pls)
pcr_fit <- pcr(crim~., data = Boston, subset = train, scale = T, validation = "CV")
validationplot(pcr_fit, val.type = "MSEP")
```




```
#pls
set.seed(2)
pls_fit <- plsr(crim~., data = Boston, subset = train, scale = T, validation = "CV")
validationplot(pls_fit, val.type = "MSEP")
```




```
#lasso
set.seed(3)
x <- model.matrix(crim~., Boston)[,-1]
y <- Boston$crim
ytrain <- y[train]
ytest <- y[-train]
library(glmnet)
lasso_cv_out <- cv.glmnet(x[train,], ytrain,
                          alpha = 1)
plot(lasso_cv_out)
```



Notice for multiple linear regression, highly correlated variables are usually not significant simultaneously (e.g. rad and tax, indus and nox, etc.). Three PCs and two new directions seem appropriate for PCR and PLS, respectively.

(f) Compare the effectiveness of each model on training vs. testing data. Which model is the best?

```
#mlr
mlr_tr <- summary(mlr_fit)$sigma^2
mlr_pred <- predict(mlr_fit, Boston[-train,])
mlr_te <- mean((mlr_pred-ytest)^2)
#pcr
pcr_pred <- predict(pcr_fit, Boston[train,], ncomp = 3)
pcr_tr <- mean((pcr_pred-ytrain)^2)
pcr_pred <- predict(pcr_fit, Boston[-train,], ncomp = 3)
pcr_te <- mean((pcr_pred-ytest)^2)
#pls
pls_pred <- predict(pls_fit, Boston[train,], ncomp = 2)
pls_tr <- mean((pls_pred-ytrain)^2)
pls_pred <- predict(pls_fit, Boston[-train,], ncomp = 2)
pls_te <- mean((pls_pred-ytest)^2)
#lasso
lasso_pred <- predict(lasso_cv_out,
                      s = lasso_cv_out$lambda.min,
                      newx = x[train,])
lasso_tr <- mean((lasso_pred-ytrain)^2)
lasso_pred <- predict(lasso_cv_out,
                      s = lasso_cv_out$lambda.min,
                      newx = x[-train,])
lasso_te <- mean((lasso_pred-ytest)^2)
#summary
errsum <- tribble(
  ~model, ~train, ~test,
  "MLR", mlr_tr, mlr_te,
  "PCR", pcr_tr, pcr_te,
  "PLS", pls_tr, pls_te,
  "LASSO", lasso_tr, lasso_te
)
```

```
errsum
```

```
## # A tibble: 4 x 3
##   model train  test
##   <chr> <dbl> <dbl>
## 1 MLR   0.612 0.614
## 2 PCR   0.781 0.800
## 3 PLS   0.694 0.705
## 4 LASSO 0.593 0.606
```

The lasso model has the smallest training error as well as testing error.

(g) Refit the principal components regression model and the lasso model to the entire dataset. Comment on the differences between the two methods. (Hint: also pay attention to highly correlated variables that you found in part (c).)

```
#pcr
pcr_fit <- pcr(crim~., data = Boston, scale = T, ncomp = 3)
summary(pcr_fit)
```

```
## Data:  X dimension: 506 13
##  Y dimension: 506 1
## Fit method: svdpc
## Number of components considered: 3
## TRAINING: % variance explained
##       1 comps  2 comps  3 comps
## X       47.70    60.36    69.67
## crim    75.72    78.27    83.33
```

```
pcr_fit[["coefficients"]][,,1]
```

```
##           zn        indus        chas1          nox           rm          age
## -0.196714027  0.260614374 -0.001916924  0.254693971 -0.160478678  0.233969465
##          dis          rad          tax      ptratio        black        lstat
## -0.234141527  0.229300495  0.247698464  0.161681734 -0.149012878  0.242197034
##         medv
## -0.207339066
```

```
#lasso
lasso_out <- glmnet(x, y, alpha = 1)
coef(lasso_out, s = lasso_cv_out$lambda.min)
```

```
## 14 x 1 sparse Matrix of class "dgCMatrix"
##                         1
## (Intercept) -4.567533734
## zn          -0.008466202
## indus        0.013660255
## chas1        .
## nox          4.183109289
## rm           .
## age          0.005293734
## dis         -0.028822090
## rad          0.133523414
## tax          .
## ptratio      .
## black       -0.001099941
## lstat        0.021689430
## medv         .
```

The PCR uses all predictors and tends to give similar loadings (hence, coefficients) to correlated variables, whereas the lasso model gives quite different coefficient values to correlated variables and some coefficients may be reduced to 0. For example, rad and tax receive similar coefficients in PCR but only rad is selected by lasso. In general, PCR handles multicollinearity by creating uncorrelated PCs with all variables. Lasso on the other hand may eliminate predictors that are highly correlated with the others but have insignificant contribution to the response.
For the Boston dataset, the first PC can be regarded as a summary index of all variables except for chas1 and it explains about 76% of the variation in the crime rate. The lasso model identifies nitrogen oxides concentration (nox) as a prominent predictor for crime rate.

(h) Refit the partial least squares model to the entire dataset, and compare with the principal components regression model.

```
pls_fit <- plsr(crim~., data = Boston, scale = T, ncomp = 2)
summary(pls_fit)
```

```
## Data:  X dimension: 506 13
##  Y dimension: 506 1
## Fit method: kernelpls
## Number of components considered: 2
## TRAINING: % variance explained
##       1 comps  2 comps
## X       47.52    58.15
## crim    79.09    85.03
```

```
pls_fit[["coefficients"]][,,1]
```

```
##          zn       indus       chas1         nox          rm         age
## -0.18377669  0.25973729  0.01012778  0.28027768 -0.10908890  0.23395702
##         dis         rad         tax     ptratio       black       lstat
## -0.24235154  0.30330476  0.29435805  0.13844917 -0.17015180  0.22270187
##        medv
## -0.16146101
```

Since the directions in PLS are obtained by integrating the crim rate, they explain more variation in the crime rate than the principal components do. PLS also gives similar loadings to correlated variables, but variables that are correlated with the crim rate receive more weights compared with PCR. The first PLS direction is also a summary index of all variables except for chas1 and it explains about 79% variation in the crime rate.