Homework 2 Solutions

Pratik Dahal

December 13, 2021

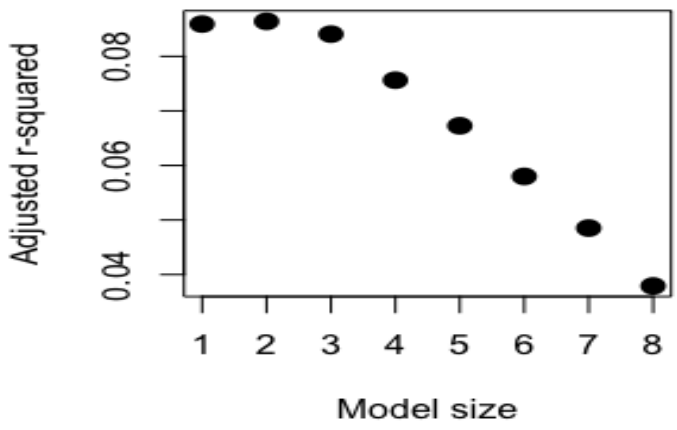
In this assignment we use the leaps package to perform a "best-subset" linear regression analysis. Let us consider the compact structure of the prostate cancer data.

```
> str(prostate)
'data.frame':   97 obs. of  10 variables:
 $ lcavol : num  -0.58 -0.994 -0.511 -1.204 0.751 ...
 $ lweight: num  2.77 3.32 2.69 3.28 3.43 ...
 $ age    : int  50 58 74 58 62 50 64 58 47 63 ...
 $ lbph   : num  -1.39 -1.39 -1.39 -1.39 -1.39 ...
 $ svi    : int  0 0 0 0 0 0 0 0 0 0 ...
 $ lcp    : num  -1.39 -1.39 -1.39 -1.39 -1.39 ...
 $ gleason: int  6 6 7 6 6 6 6 6 6 6 ...
 $ pgg45  : int  0 0 20 0 0 0 0 0 0 0 ...
 $ lpsa   : num  -0.431 -0.163 -0.163 -0.163 0.372 ...
 $ train  : logi  TRUE TRUE TRUE TRUE TRUE TRUE ...
```

The main objective is to come up with the "best" model that predicts `lpsa` by using some subset of the predictors used in the full model. As suggested, we discard Column 10 i.e., `train`. Then, the remaining features are numeric in nature.

After applying the regsubset function, we get best n-variable model where $n = 1, 2, \cdots, 8$. Now, let us use Adjusted r-squared and BIC criterion to select the best model among the models of different sizes.

Firstly, using Adjusted r-squared we observe the following plot:



Generally, the higher adjusted r-squared is better since it penalizes the model with the

increase in model size. If we squint our eyes sufficiently enough, then we should see that the adjusted r-squared value is highest when the size of the model is 2.

We can also use the following R command to find the model with highest adjusted r-squared value:

```
> match(max(out_summary$adjr2), out_summary$adjr2)
[1] 2
```

So, the features are:

```
> names(coef(out_subsets, id = 2))
[1] "lcavol"  "lweight"
```

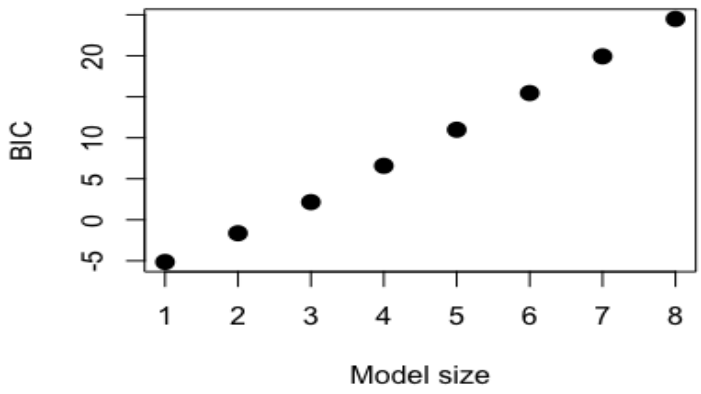
Finally, we fit the model to the entire data.

```
fit_1 <- lm(lpsa ~ lcavol + lweight, data = data.frame(prostate))
summary(fit_1)
```

```
> summary(fit_1)

Call:
lm(formula = lpsa ~ lcavol + lweight, data = data.frame(prostate))

Residuals:
     Min       1Q   Median       3Q      Max
-1.61051 -0.44135 -0.04666  0.53542  1.90424

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) -0.81344    0.65309  -1.246 0.216033
lcavol       0.65154    0.06693   9.734 6.75e-16 ***
lweight      0.66472    0.18414   3.610 0.000494 ***
---

Residual standard error: 0.7419 on 94 degrees of freedom
Multiple R-squared:  0.5955,    Adjusted R-squared:  0.5869
F-statistic: 69.19 on 2 and 94 DF,  p-value: < 2.2e-16
```

Notice that the Adjusted R-squared is 0.5869. This means about 58.7% of the variance in the log PSA score can be predicted by log cancer volume and log prostate weight.

Secondly, using BIC we observe the following plot:



Generally, models with lower values of BIC are preferred since BIC accounts for the penalty term for the number of parameters in the model. Clearly, the model with size 1 has the lowest BIC score.

We can also use the following R command to find the model with lowest BIC score:

```
> match(min(out_summary$bic), out_summary$bic)
[1] 1
```

So, the required feature is:

```
> names(coef(out_subsets, id = 1))
[1] "lcavol"
```

Finally, we fit the model to the entire data.

```
> fit_2 <- lm(lpsa ~ lcavol, data = data.frame(prostate))
> summary(fit_2)

Call:
lm(formula = lpsa ~ lcavol, data = data.frame(prostate))

Residuals:
     Min       1Q   Median       3Q      Max
-1.67624 -0.41648  0.09859  0.50709  1.89672

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  1.50730    0.12194   12.36   <2e-16 ***
lcavol       0.71932    0.06819   10.55   <2e-16 ***
---

Residual standard error: 0.7875 on 95 degrees of freedom
Multiple R-squared:  0.5394,    Adjusted R-squared:  0.5346
F-statistic: 111.3 on 1 and 95 DF,  p-value: < 2.2e-16
```

Notice that the Adjusted R-squared is 0.5346. This means about 53.5% of the variance in the log PSA score can be predicted by log cancer volume.

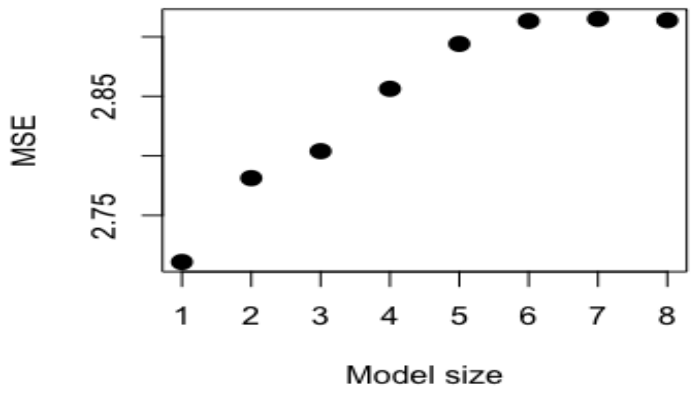
Now, let us use cross validation to select the best model among the models of different sizes.

We will perform 5-fold cross validation and obtain a 8 x 5 error matrix.

```
> Err_cv
         [,1]     [,2]     [,3]     [,4]     [,5]
[1,] 7.656069 6.851481 7.859552 6.014917 8.488855
[2,] 8.526302 7.360037 7.967339 6.247267 8.710914
[3,] 8.724878 6.710855 8.231084 6.883389 8.896071
[4,] 9.187808 6.937613 8.497029 7.038687 9.292919
[5,] 9.119710 7.516247 8.454446 7.113138 9.822584
[6,] 9.134129 7.823327 8.697019 7.199819 9.701507
[7,] 8.988637 7.879415 8.853369 7.174991 9.709126
[8,] 8.989541 7.870250 8.728512 7.209269 9.773160
```

The (i,j) entry in the matrix corresponds to mean squared error (MSE) of the i^{th} best model in j^{th} cross validation fold. Now, taking the average i^{th} row sum yields the average MSE across the best i^{th} variable model.

So we get the followiong plot:



Notice that the lowest MSE corresponds to the model of size 1. So, the required feature is:

```
> names(coef(out_subsets, id = 1))
[1] "lcavol"
```

Finally, we fit the model to the entire data. Notice, this is the same model we got by model selection using BIC criteria.

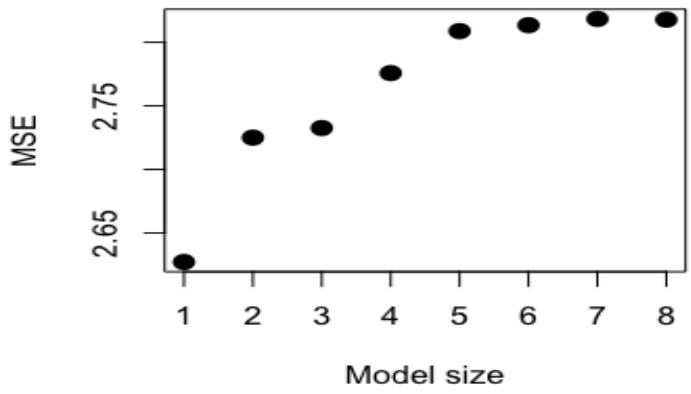
```
> fit_3 <- lm(lpsa ~ lcavol, data = data.frame(prostate))
```

Similarly, we will now perform 10-fold cross validation and obtain a 8 x 10 error matrix.

```
> Err_cv
          [,1]     [,2]     [,3]     [,4]     [,5]     [,6]     [,7]
[1,] 15.20831 8.257881 6.268091 5.601155 7.000087 6.705187 6.406396
[2,] 16.52461 8.564773 7.499931 6.242660 7.709104 6.735304 6.320479
[3,] 13.92561 8.953352 7.605389 6.165394 8.605558 6.612505 6.362860
[4,] 14.51495 9.056560 7.794447 6.352649 8.715904 8.730599 6.389220
[5,] 14.21202 8.948557 7.889061 6.638002 8.892146 8.468977 6.669134
[6,] 13.93635 9.026114 7.624464 6.662281 9.161989 8.435338 6.628346
[7,] 14.77455 8.912359 7.385614 6.585075 9.150452 8.592128 6.430361
[8,] 14.50016 8.912361 7.407263 6.585301 9.294990 8.784332 6.518926
          [,8]     [,9]     [,10]
[1,] 7.724420 6.198271 6.745112
[2,] 8.102450 6.652350 6.859978
[3,] 7.906653 6.817515 6.576358
[4,] 8.009162 7.864619 6.892514
[5,] 8.002453 7.586130 6.975812
[6,] 8.076704 7.565964 6.999495
[7,] 8.056962 7.592232 6.954891
[8,] 8.066565 7.563650 6.919200
```

Like the case with 5 fold CV, the (i,j) entry in the matrix corresponds to mean squared error (MSE) of the i^{th} best model in j^{th} cross validation fold. Now, taking the average i^{th} row sum yields the average MSE across the best i^{th} variable model.

So we get the followiong plot:



Note that the lowest MSE corresponds to the model of size 1. So, the required feature is:

```
> names(coef(out_subsets, id = 1))
[1] "lcavol"
```

Finally, we fit the model to the entire data. Notice, this is the same model we got by model selection using BIC criteria.

```
> fit_4 <- lm(lpsa ~ lcavol, data = data.frame(prostate)).
```

All together we can summarize best model as follows:

i. Adjusted r-squared
   lpsa $\sim$ lcavol + lweight

ii. BIC
   lpsa $\sim$ lcavol

iii. 5-fold cross validation
   lpsa $\sim$ lcavol

iv. 10-fold cross validation
   lpsa $\sim$ lcavol

□

Appendix

```
# Name: Pratik Dahal
# Computational Statistics

set.seed(12132021)
prostate <- read.table("~/Desktop/comp_stat/homework_2/prostrate_cancer.txt")

# removing the last column.
prostate$train <- NULL

# standardizing the predictor variables
X <- scale(as.matrix(prostate[, 1:8]))
Y <- prostate$lpsa

require(leaps)

ncol(X)
## since the number of predictors is 8, we set nvmax = 8.
## This will return best jth variable model where j = {1,2,...,8}

out_subsets <- regsubsets(X, Y, nvmax = 8, intercept = FALSE)
summary(out_subsets)
(out_summary <- summary(out_subsets))

#best 1-variable model
coef(out_subsets, id = 1)
#best 2-variable model
coef(out_subsets, id = 2)
#best 3-variable model
coef(out_subsets, id = 3)
#best 4-variable model
coef(out_subsets, id = 4)
#best 5-variable model
coef(out_subsets, id = 5)
#best 6-variable model
coef(out_subsets, id = 6)
#best 7-variable model
coef(out_subsets, id = 7)
#best 8-variable model
coef(out_subsets, id = 8)

# Adjusted R-squared error.
out_summary$adjr2

plot(1:8, out_summary$adjr2, main = "",
     pch = 16, cex = 1.5, col = "black",
     xlab = "Model␣size", ylab = "Adjusted␣r-squared")

match(max(out_summary$adjr2), out_summary$adjr2)

### final model using all the data
fit_1 <- lm(lpsa ~ lcavol + lweight, data = data.frame(prostate))
summary(fit_1)
```

```r
# BIC
out_summary$bic

plot(1:8, out_summary$bic, main = "",
     pch = 16, cex = 1.5, col = "black",
     xlab = "Model size", ylab = "BIC")

match(min(out_summary$bic), out_summary$bic)
names(coef(out_subsets, id = 1))

### final model using all the data
fit_2 <- lm(lpsa ~ lcavol, data = data.frame(prostate))
summary(fit_2)

### cross-validation 5-fold
K <- 5    # 5-fold
M <- 8    # Total number of ith best models for i=1:8
ind <- rep_len(1:K, length = nrow(X))
ind <- sample(ind)
Err_cv <- matrix(data = NA, nrow = M, ncol = K)

# filling the error matrix
for (k in 1:K) {
  out_subsets <- regsubsets(X[ind != k,], Y[ind!=k], nvmax = 8, intercept = FAL
  for (m in 1:M) {
    predicted_value <- as.matrix(X[ind == k,][,names(coef(out_subsets, id = m))
    %*% as.matrix(coef(out_subsets, id = m))
    Err_cv[m,k] <- mean((Y[ind == k] - predicted_value)^2)
  }
}

# Get the mean of each row of Err_cv matrix
mean_vector = rowMeans(sqrt(Err_cv))

plot(1:8, mean_vector, main = "",
     pch = 16, cex = 1.5, col = "black",
     xlab = "Model size", ylab = "MSE")

# full model for prediction
fit_3 <- lm(lpsa ~ lcavol, data = data.frame(prostate))


### cross-validation 10-fold
K <- 10   # 10-fold
M <- 8    # Total number of ith best models for i=1:8
ind <- rep_len(1:K, length = nrow(X))
ind <- sample(ind)
Err_cv <- matrix(data = NA, nrow = M, ncol = K)

# filling the error matrix
for (k in 1:K) {
  out_subsets <- regsubsets(X[ind != k,], Y[ind!=k],
                            nvmax = 8, intercept = FALSE)
  for (m in 1:M) {
```

```
    predicted_value <- as.matrix(X[ind == k,][,names(coef(out_subsets, id = m))
    %*% as.matrix(coef(out_subsets, id = m))
    Err_cv[m,k] <- mean((Y[ind == k] - predicted_value)^2)
  }
}

# Get the mean of each row of Err_cv matrix
mean_vector = rowMeans(sqrt(Err_cv))

plot(1:8, mean_vector, main = "",
     pch = 16, cex = 1.5, col = "black",
     xlab = "Model size", ylab = "MSE")

# final model for prediction
fit_4 <- lm(lpsa ~ lcavol, data = data.frame(prostate))
```