STA 4320 CHAP 6.1.2, 6.1.3

Prof. He Jiang

```
require(ISLR2) # Hitters dataset
```

```
## Loading required package: ISLR2
```

```
require(leaps) # subset selection
```

```
## Loading required package: leaps
```

Hitters dataset and NA terms

The Hitters dataset consists of Major League Baseball data from the 1986 and 1987 seasons.

We remove the NA terms.

```
dat = na.omit(Hitters)
any(is.na(dat))
```

```
## [1] FALSE
```

6.1.2 Stepwise selection

Forward stepwise selection.

```
# regsubsets is from the leaps package
regfit_fwd = regsubsets(Salary ~ ., data = dat, nvmax = 19, method = "forward")
summary(regfit_fwd)
```

```
## Subset selection object
## Call: regsubsets.formula(Salary ~ ., data = dat, nvmax = 19, method = "forward")
## 19 Variables  (and intercept)
##            Forced in Forced out
## AtBat          FALSE      FALSE
## Hits           FALSE      FALSE
## HmRun          FALSE      FALSE
## Runs           FALSE      FALSE
## RBI            FALSE      FALSE
## Walks          FALSE      FALSE
## Years          FALSE      FALSE
## CAtBat         FALSE      FALSE
## CHits          FALSE      FALSE
## CHmRun         FALSE      FALSE
## CRuns          FALSE      FALSE
## CRBI           FALSE      FALSE
## CWalks         FALSE      FALSE
## LeagueN        FALSE      FALSE
## DivisionW      FALSE      FALSE
```

```
## PutOuts         FALSE      FALSE
## Assists         FALSE      FALSE
## Errors          FALSE      FALSE
## NewLeagueN      FALSE      FALSE
## 1 subsets of each size up to 19
## Selection Algorithm: forward
##           AtBat Hits HmRun Runs RBI Walks Years CAtBat CHits CHmRun CRuns CRBI
## 1  ( 1 )  " "   " "  " "   " "  " " " "   " "   " "    " "   " "    " "   "*"
## 2  ( 1 )  " "   "*"  " "   " "  " " " "   " "   " "    " "   " "    " "   "*"
## 3  ( 1 )  " "   "*"  " "   " "  " " " "   " "   " "    " "   " "    " "   "*"
## 4  ( 1 )  " "   "*"  " "   " "  " " " "   " "   " "    " "   " "    " "   "*"
## 5  ( 1 )  "*"   "*"  " "   " "  " " " "   " "   " "    " "   " "    " "   "*"
## 6  ( 1 )  "*"   "*"  " "   " "  " " "*"   " "   " "    " "   " "    " "   "*"
## 7  ( 1 )  "*"   "*"  " "   " "  " " "*"   " "   " "    " "   " "    " "   "*"
## 8  ( 1 )  "*"   "*"  " "   " "  " " "*"   " "   " "    " "   " "    "*"   "*"
## 9  ( 1 )  "*"   "*"  " "   " "  " " "*"   " "   "*"    " "   " "    "*"   "*"
## 10  ( 1 ) "*"   "*"  " "   " "  " " "*"   " "   "*"    " "   " "    "*"   "*"
## 11  ( 1 ) "*"   "*"  " "   " "  " " "*"   " "   "*"    " "   " "    "*"   "*"
## 12  ( 1 ) "*"   "*"  " "   "*"  " " "*"   " "   "*"    " "   " "    "*"   "*"
## 13  ( 1 ) "*"   "*"  " "   "*"  " " "*"   " "   "*"    " "   " "    "*"   "*"
## 14  ( 1 ) "*"   "*"  "*"   "*"  " " "*"   " "   "*"    " "   " "    "*"   "*"
## 15  ( 1 ) "*"   "*"  "*"   "*"  " " "*"   " "   "*"    "*"   " "    "*"   "*"
## 16  ( 1 ) "*"   "*"  "*"   "*"  "*" "*"   " "   "*"    "*"   " "    "*"   "*"
## 17  ( 1 ) "*"   "*"  "*"   "*"  "*" "*"   " "   "*"    "*"   " "    "*"   "*"
## 18  ( 1 ) "*"   "*"  "*"   "*"  "*" "*"   "*"   "*"    "*"   " "    "*"   "*"
## 19  ( 1 ) "*"   "*"  "*"   "*"  "*" "*"   "*"   "*"    "*"   "*"    "*"   "*"
##           CWalks LeagueN DivisionW PutOuts Assists Errors NewLeagueN
## 1  ( 1 )  " "    " "     " "       " "     " "     " "    " "
## 2  ( 1 )  " "    " "     " "       " "     " "     " "    " "
## 3  ( 1 )  " "    " "     " "       "*"     " "     " "    " "
## 4  ( 1 )  " "    " "     "*"       "*"     " "     " "    " "
## 5  ( 1 )  " "    " "     "*"       "*"     " "     " "    " "
## 6  ( 1 )  " "    " "     "*"       "*"     " "     " "    " "
## 7  ( 1 )  "*"    " "     "*"       "*"     " "     " "    " "
## 8  ( 1 )  "*"    " "     "*"       "*"     " "     " "    " "
## 9  ( 1 )  "*"    " "     "*"       "*"     " "     " "    " "
## 10  ( 1 ) "*"    " "     "*"       "*"     "*"     " "    " "
## 11  ( 1 ) "*"    "*"     "*"       "*"     "*"     " "    " "
## 12  ( 1 ) "*"    "*"     "*"       "*"     "*"     " "    " "
## 13  ( 1 ) "*"    "*"     "*"       "*"     "*"     "*"    " "
## 14  ( 1 ) "*"    "*"     "*"       "*"     "*"     "*"    " "
## 15  ( 1 ) "*"    "*"     "*"       "*"     "*"     "*"    " "
## 16  ( 1 ) "*"    "*"     "*"       "*"     "*"     "*"    " "
## 17  ( 1 ) "*"    "*"     "*"       "*"     "*"     "*"    "*"
## 18  ( 1 ) "*"    "*"     "*"       "*"     "*"     "*"    "*"
## 19  ( 1 ) "*"    "*"     "*"       "*"     "*"     "*"    "*"
```

Backward stepwise selection.

```
regfit_bwd = regsubsets(Salary ~ ., data = dat, nvmax = 19, method = "backward")
summary(regfit_bwd)
```

```
## Subset selection object
## Call: regsubsets.formula(Salary ~ ., data = dat, nvmax = 19, method = "backward")
## 19 Variables  (and intercept)
```

```
##             Forced in Forced out
## AtBat           FALSE      FALSE
## Hits            FALSE      FALSE
## HmRun           FALSE      FALSE
## Runs            FALSE      FALSE
## RBI             FALSE      FALSE
## Walks           FALSE      FALSE
## Years           FALSE      FALSE
## CAtBat          FALSE      FALSE
## CHits           FALSE      FALSE
## CHmRun          FALSE      FALSE
## CRuns           FALSE      FALSE
## CRBI            FALSE      FALSE
## CWalks          FALSE      FALSE
## LeagueN         FALSE      FALSE
## DivisionW       FALSE      FALSE
## PutOuts         FALSE      FALSE
## Assists         FALSE      FALSE
## Errors          FALSE      FALSE
## NewLeagueN      FALSE      FALSE
## 1 subsets of each size up to 19
## Selection Algorithm: backward
##           AtBat Hits HmRun Runs RBI Walks Years CAtBat CHits CHmRun CRuns CRBI
## 1  ( 1 )  " "   " "  " "   " "  " " " " " "   " "   " "    " "   " "    "*"   " "
## 2  ( 1 )  " "   "*"  " "   " "  " " " " " "   " "   " "    " "   " "    "*"   " "
## 3  ( 1 )  " "   "*"  " "   " "  " " " " " "   " "   " "    " "   " "    "*"   " "
## 4  ( 1 )  "*"   "*"  " "   " "  " " " " " "   " "   " "    " "   " "    "*"   " "
## 5  ( 1 )  "*"   "*"  " "   " "  " " " " "*"   " "   " "    " "   " "    "*"   " "
## 6  ( 1 )  "*"   "*"  " "   " "  " " " " "*"   " "   " "    " "   " "    "*"   " "
## 7  ( 1 )  "*"   "*"  " "   " "  " " " " "*"   " "   " "    " "   " "    "*"   " "
## 8  ( 1 )  "*"   "*"  " "   " "  " " " " "*"   " "   " "    " "   " "    "*"   "*"
## 9  ( 1 )  "*"   "*"  " "   " "  " " " " "*"   " "   "*"    " "   " "    "*"   "*"
## 10  ( 1 ) "*"   "*"  " "   " "  " " " " "*"   " "   "*"    " "   " "    "*"   "*"
## 11  ( 1 ) "*"   "*"  " "   " "  " " " " "*"   " "   "*"    " "   " "    "*"   "*"
## 12  ( 1 ) "*"   "*"  " "   "*"  " " " " "*"   " "   "*"    " "   " "    "*"   "*"
## 13  ( 1 ) "*"   "*"  " "   "*"  " " " " "*"   " "   "*"    " "   " "    "*"   "*"
## 14  ( 1 ) "*"   "*"  "*"   "*"  " " " " "*"   " "   "*"    " "   " "    "*"   "*"
## 15  ( 1 ) "*"   "*"  "*"   "*"  " " " " "*"   " "   "*"    "*"   " "    "*"   "*"
## 16  ( 1 ) "*"   "*"  "*"   "*"  "*" "*" "*"   " "   "*"    "*"   " "    "*"   "*"
## 17  ( 1 ) "*"   "*"  "*"   "*"  "*" "*" "*"   " "   "*"    "*"   " "    "*"   "*"
## 18  ( 1 ) "*"   "*"  "*"   "*"  "*" "*" "*"   "*"   "*"    "*"   " "    "*"   "*"
## 19  ( 1 ) "*"   "*"  "*"   "*"  "*" "*" "*"   "*"   "*"    "*"   "*"    "*"   "*"
##           CWalks LeagueN DivisionW PutOuts Assists Errors NewLeagueN
## 1  ( 1 )  " "    " "     " "       " "     " "     " "    " "
## 2  ( 1 )  " "    " "     " "       " "     " "     " "    " "
## 3  ( 1 )  " "    " "     " "       "*"     " "     " "    " "
## 4  ( 1 )  " "    " "     " "       "*"     " "     " "    " "
## 5  ( 1 )  " "    " "     " "       "*"     " "     " "    " "
## 6  ( 1 )  " "    " "     "*"       "*"     " "     " "    " "
## 7  ( 1 )  "*"    " "     "*"       "*"     " "     " "    " "
## 8  ( 1 )  "*"    " "     "*"       "*"     " "     " "    " "
## 9  ( 1 )  "*"    " "     "*"       "*"     " "     " "    " "
## 10  ( 1 ) "*"    " "     "*"       "*"     "*"     " "    " "
## 11  ( 1 ) "*"    "*"     "*"       "*"     "*"     " "    " "
```

```
## 12  ( 1 ) "*"    "*"     "*"       "*"     "*"     " "    " "
## 13  ( 1 ) "*"    "*"     "*"       "*"     "*"     "*"    " "
## 14  ( 1 ) "*"    "*"     "*"       "*"     "*"     "*"    " "
## 15  ( 1 ) "*"    "*"     "*"       "*"     "*"     "*"    " "
## 16  ( 1 ) "*"    "*"     "*"       "*"     "*"     "*"    " "
## 17  ( 1 ) "*"    "*"     "*"       "*"     "*"     "*"    "*"
## 18  ( 1 ) "*"    "*"     "*"       "*"     "*"     "*"    "*"
## 19  ( 1 ) "*"    "*"     "*"       "*"     "*"     "*"    "*"
```

Best subset selection (as reference).

```
regfit_best = regsubsets(Salary ~ ., data = dat, nvmax = 19)
```

Compare a 7 component model for forward, backward stepwise selection, and best subset selection.

```
coef(regfit_best, 7)
```

```
##  (Intercept)         Hits        Walks       CAtBat        CHits       CHmRun
##   79.4509472    1.2833513    3.2274264   -0.3752350    1.4957073    1.4420538
##    DivisionW      PutOuts
## -129.9866432    0.2366813
```

```
coef(regfit_fwd, 7)
```

```
##  (Intercept)        AtBat         Hits        Walks         CRBI       CWalks
##  109.7873062   -1.9588851    7.4498772    4.9131401    0.8537622   -0.3053070
##    DivisionW      PutOuts
## -127.1223928    0.2533404
```

```
coef(regfit_bwd, 7)
```

```
##  (Intercept)        AtBat         Hits        Walks        CRuns       CWalks
##  105.6487488   -1.9762838    6.7574914    6.0558691    1.1293095   -0.7163346
##    DivisionW      PutOuts
## -116.1692169    0.3028847
```

Selecting the resulting model based on a criteria (here BIC).

```
num_bic_fwd = which.min( summary(regfit_fwd)$bic )
num_bic_bwd = which.min( summary(regfit_bwd)$bic )
num_bic_best = which.min( summary(regfit_best)$bic )
paste("Forward stepwise selection: ", num_bic_fwd)
```

```
## [1] "Forward stepwise selection:  6"
```

```
paste("Back stepwise selection: ", num_bic_bwd)
```

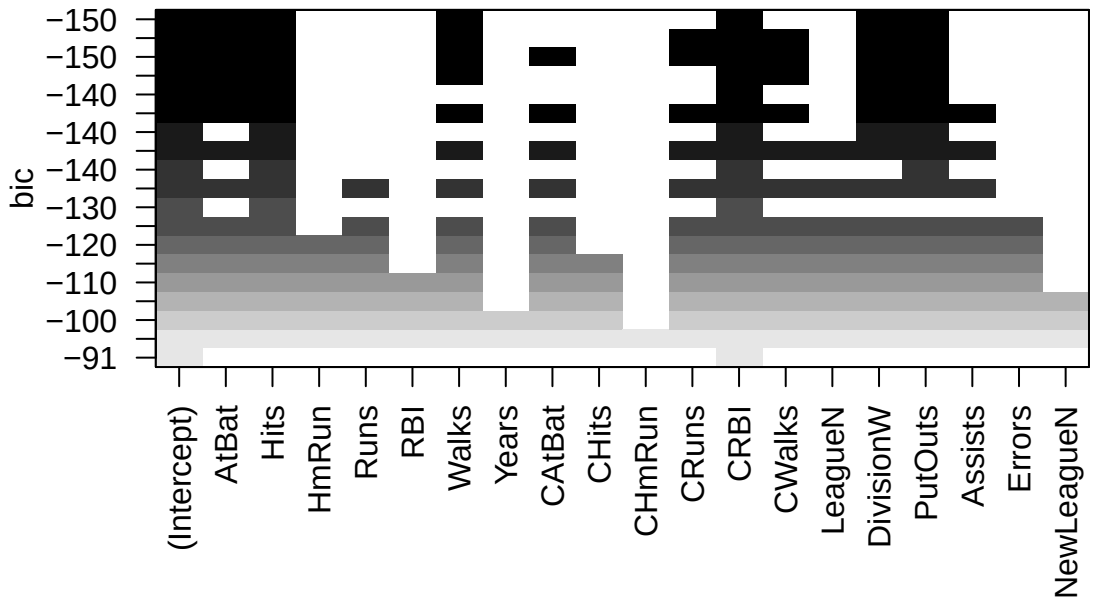
```
## [1] "Back stepwise selection:  8"
```

```
paste("Best subset selection: ", num_bic_best)
```

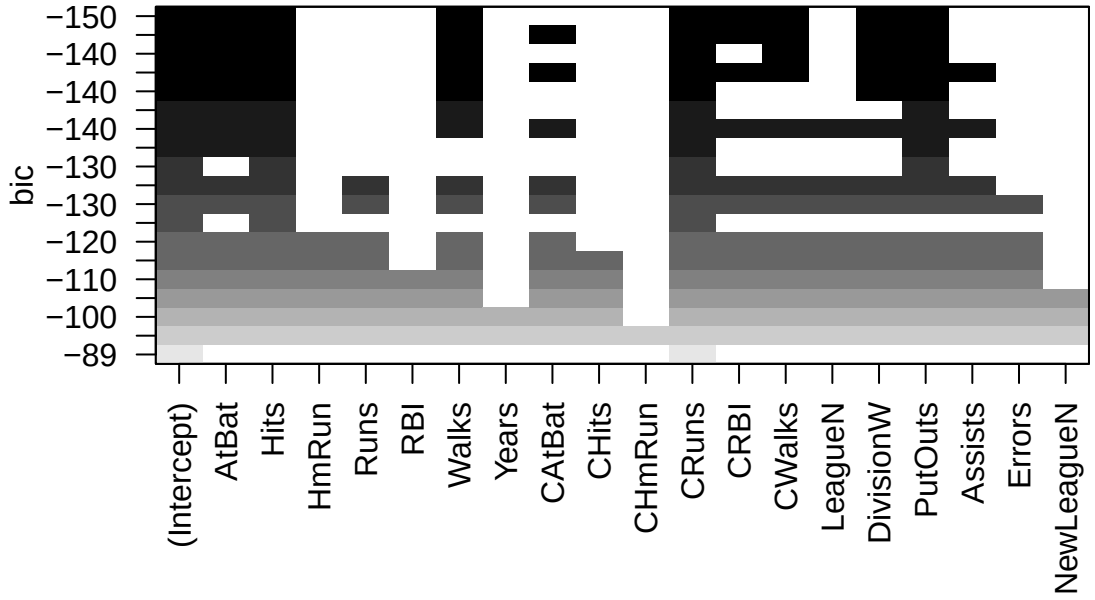
```
## [1] "Best subset selection:  6"
```

In addition, we have the model selection plots.

```
#par(mfrow = c(1, 2))
plot(regfit_fwd, scale = "bic")
```




```
plot(regfit_bwd, scale = "bic")
```



The coefficients should represent the same variables as the best row in the previous pictures.

```
coef(regfit_fwd, num_bic_fwd)
```

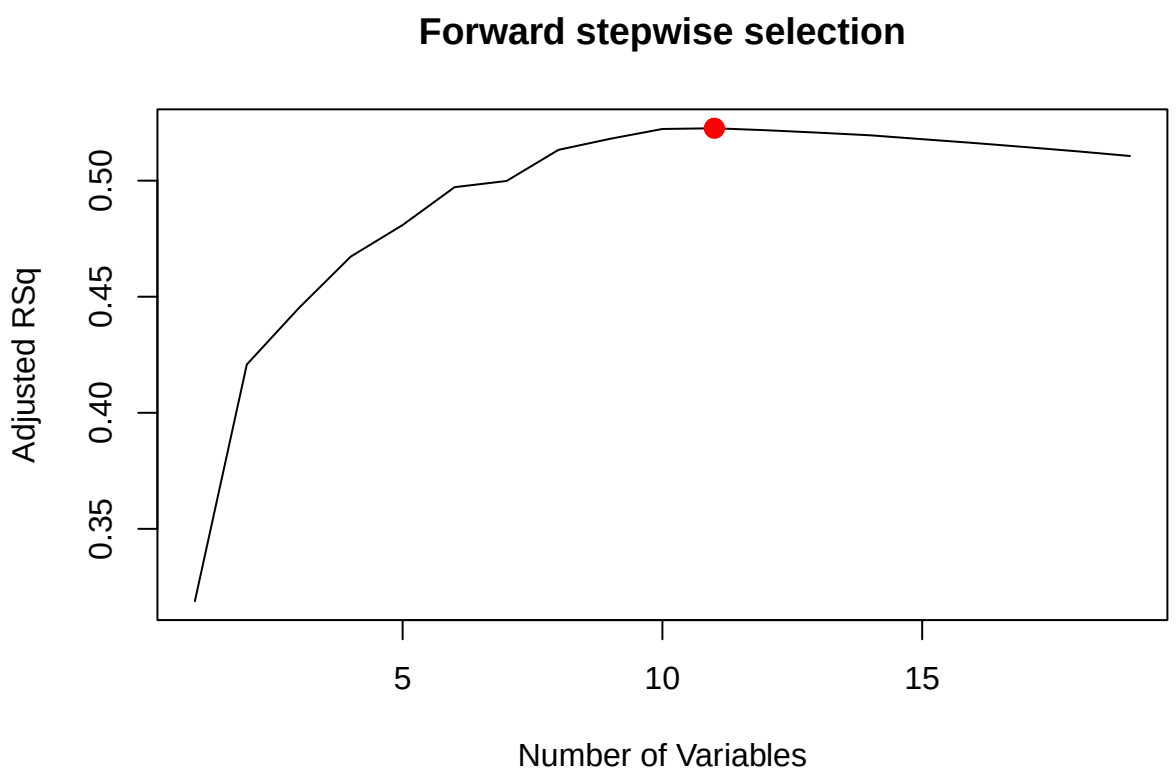
```
##  (Intercept)        AtBat         Hits        Walks         CRBI    DivisionW
##   91.5117981   -1.8685892    7.6043976    3.6976468    0.6430169 -122.9515338
##      PutOuts
##    0.2643076
```

```
coef(regfit_bwd, num_bic_bwd)
```

```
##  (Intercept)        AtBat         Hits        Walks        CRuns         CRBI
##  117.1520434   -2.0339209    6.8549136    6.4406642    0.7045391    0.5273238
##       CWalks    DivisionW      PutOuts
##   -0.8066062 -123.7798366    0.2753892
```

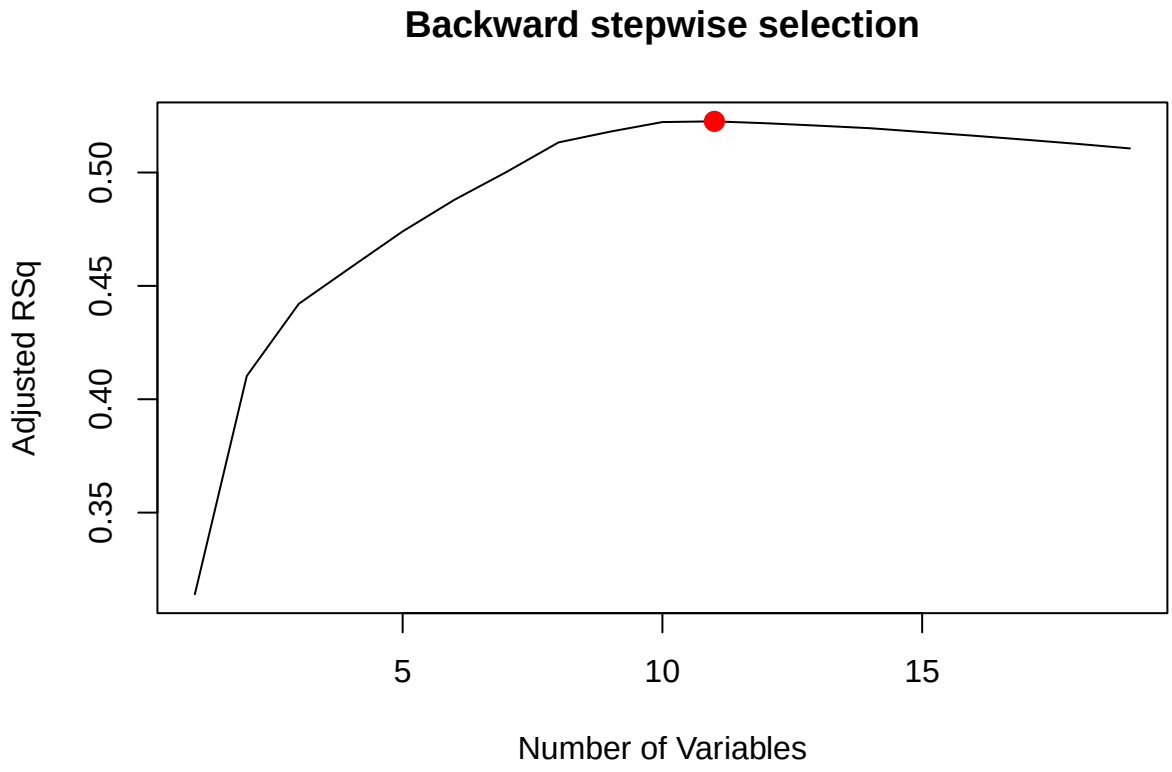
As before, we can also see the plots (using adjusted R2 as an example).

```
num_adjr_fwd = which.max(summary(regfit_fwd)$adjr2)
plot(summary(regfit_fwd)$adjr2,
     xlab = "Number of Variables", ylab = "Adjusted RSq", type = "l",
     main = "Forward stepwise selection")
points(num_adjr_fwd, summary(regfit_fwd)$adjr2[num_adjr_fwd], col = "red", cex = 2, pch = 20)
```



As before, we can also see the plots (using adjusted R2 as an example).

```
num_adjr_bwd = which.max(summary(regfit_bwd)$adjr2)
plot(summary(regfit_bwd)$adjr2,
     xlab = "Number of Variables", ylab = "Adjusted RSq", type = "l",
     main = "Backward stepwise selection")
points(num_adjr_bwd, summary(regfit_bwd)$adjr2[num_adjr_bwd], col = "red", cex = 2, pch = 20)
```



Note the above plot is slightly different than the forward stepwise one.

Validation set approach

Separation of the entire data into two parts

```
set.seed(1)
train = sample(c(TRUE, FALSE), nrow(dat), replace = TRUE)
test = (!train)
```

Apply the regsubsets command to the training set

```
regfit_best = regsubsets(Salary ~ ., data = dat[train, ], nvmax = 19)
```

There is no "predict" command to regsubsets, so we have to write the functions by hand.

```
test_mat = model.matrix(Salary ~ ., data = dat[test, ])
head(test_mat)
```

```
##                   (Intercept) AtBat Hits HmRun Runs RBI Walks Years CAtBat
## -Alvin Davis                1   479  130    18   66  72    76     3   1624
## -Alfredo Griffin            1   594  169     4   74  51    35    11   4408
## -Andre Thornton             1   401   92    17   49  66    65    13   5206
## -Alan Trammell              1   574  159    21  107  75    59    10   4631
## -Buddy Biancalana           1   190   46     2   24   8    15     5    479
## -Bruce Bochy                1   127   32     8   16  22    14     8    727
##                   CHits CHmRun CRuns CRBI CWalks LeagueN DivisionW PutOuts
## -Alvin Davis        457     63   224  266    263       0         1     880
## -Alfredo Griffin   1133     19   501  336    194       0         1     282
## -Andre Thornton    1332    253   784  890    866       0         0       0
## -Alan Trammell     1300     90   702  504    488       0         0     238
## -Buddy Biancalana   102      5    65   23     39       0         1     102
```

```
## -Bruce Bochy        180     24    67   82     56       1         1     202
##                   Assists Errors NewLeagueN
## -Alvin Davis           82     14          0
## -Alfredo Griffin      421     25          0
## -Andre Thornton         0      0          0
## -Alan Trammell        445     22          0
## -Buddy Biancalana     177     16          0
## -Bruce Bochy           22      2          1
```

Compute the validation errors for best subset selection.

```
p = ncol(test_mat) - 1
val_errors = numeric(p)

for (i in 1:p){
  coef_i = coef(regfit_best, id = i)
  pred = test_mat[, names(coef_i)] %*% coef_i
  val_errors[i] = mean( (dat$Salary[test] - pred)^2 )
}
```

To avoid having to retype all of the above, the textbook provided a function. This function returns the predicted values.

```
predict.regsubsets = function(object, newdata, id){
  form = as.formula(object$call[[2]])
  mat = model.matrix(form, newdata)
  coef_i = coef(object, id = id)
  xvars = names(coef_i)
  return(mat[, xvars] %*% coef_i)
}
```

The model with the lowest best subset selection validation error.

```
val_errors
```

```
##  [1] 164377.3 144405.5 152175.7 145198.4 137902.1 139175.7 126849.0 136191.4
##  [9] 132889.6 135434.9 136963.3 140694.9 140690.9 141951.2 141508.2 142164.4
## [17] 141767.4 142339.6 142238.2
```

```
which.min(val_errors)
```

```
## [1] 7
```

```
# We can see the best 7 component model on the training set.
coef(regfit_best, 7)
```

```
##  (Intercept)        AtBat         Hits        Walks        CRuns       CWalks
##   67.1085369   -2.1462987    7.0149547    8.0716640    1.2425113   -0.8337844
##    DivisionW      PutOuts
## -118.4364998    0.2526925
```

Now we determined that the "best" model from best subset selection is a 7 component model, we next select the 7 component model based on the full data set.

Note that the 7 variables from the entire dataset is different to the 7 variables from only the training set.

```
regfit_best = regsubsets(Salary ~ ., data = dat, nvmax = 19)
coef(regfit_best, 7)
```

```
##  (Intercept)         Hits        Walks       CAtBat        CHits       CHmRun
```

```
##   79.4509472    1.2833513    3.2274264   -0.3752350    1.4957073    1.4420538
##    DivisionW      PutOuts
## -129.9866432    0.2366813
```

With the same idea, we can compute the validation errors for forward stepwise selection.

```
regfit_fwd = regsubsets(Salary ~ ., data = dat[train, ], nvmax = 19, method = "forward")

p = ncol(test_mat) - 1
val_errors = numeric(p)

for (i in 1:p){
  coef_i = coef(regfit_fwd, id = i)
  pred = test_mat[, names(coef_i)] %*% coef_i
  val_errors[i] = mean( (dat$Salary[test] - pred)^2 )
}
```

The model with the lowest forward stepwise selection validation error.

```
val_errors
```

```
##  [1] 164377.3 144405.5 144024.6 139228.7 131529.4 130853.9 126849.0 136191.4
##  [9] 132889.6 134277.2 137267.1 138348.9 141538.4 141951.2 141508.2 142102.1
## [17] 141767.4 142339.6 142238.2
```

```
which.min(val_errors)
```

```
## [1] 7
```

```
# We can see the best 7 component model on the training set.
coef(regfit_best, 7)
```

```
##  (Intercept)         Hits        Walks       CAtBat        CHits       CHmRun
##   79.4509472    1.2833513    3.2274264   -0.3752350    1.4957073    1.4420538
##    DivisionW      PutOuts
## -129.9866432    0.2366813
```

Now we determined that the "best" model from forward stepwise selection is a 7 component model, we next select the 7 component model based on the full data set.

Note that the 7 variables from the entire dataset might be different to the 7 variables from only the training set.

Here, even though we selected the best model using forward stepwise selection, we can acquire the best 7 component model on the entire dataset using best subset selection (due to p=19).

If p is too large, we can use forward stepwise selection on the entire dataset.

```
regfit_fwd = regsubsets(Salary ~ ., data = dat, nvmax = 19)
coef(regfit_fwd, 7)
```

```
##  (Intercept)         Hits        Walks       CAtBat        CHits       CHmRun
##   79.4509472    1.2833513    3.2274264   -0.3752350    1.4957073    1.4420538
##    DivisionW      PutOuts
## -129.9866432    0.2366813
```

Cross validation

We must perform best subset selection within each of the k training sets.

We first separate the data into k folds.

```
k = 10
n = nrow(dat)
set.seed(1)
# the following repeats 1:k until n items are formed
# regardless of whether n is a multiple of k
folds = sample(rep(1:k, length = n), replace = FALSE)

# this matrix records the cross validation errors
cv_errors = matrix(0,
                   nrow = k,
                   ncol = p,
                   dimnames = list(NULL, paste(1:p)))
```

Each column represent a fold. Each row represent increasing the number of components from 1 to 19. The (j, i)-th element corresponds to the test MSE, using the j-th fold as the validation fold and using the remaining folds as training folds, while getting the best i-component model.

```
for (j in 1:k){
  best_fit = regsubsets(Salary ~ ., data = dat[folds != j, ], nvmax = 19)
  for (i in 1:19) {
    pred = predict(best_fit, dat[folds == j, ], id = i)
    cv_errors[j, i] = mean( (dat$Salary[folds == j] - pred)^2 )
  }
}
```

We then average over each column, to obtain the cross validation errors for each i-component model.

```
mean_cv_errors = apply(cv_errors, 2, mean)
# 2 indicates columns
# mean is the function to be taken
mean_cv_errors
```

```
##        1        2        3        4        5        6        7        8
## 143439.8 126817.0 134214.2 131782.9 130765.6 120382.9 121443.1 114363.7
##        9       10       11       12       13       14       15       16
## 115163.1 109366.0 112738.5 113616.5 115557.6 115853.3 115630.6 116050.0
##       17       18       19
## 116117.0 116419.3 116299.1
```

Plot of CV errors vs the number of components.

```
which.min( as.numeric(mean_cv_errors) )
```

```
## [1] 10
```

```
plot(mean_cv_errors, type = "l")
points(10, mean_cv_errors[10], col = "red", cex = 2, pch = 20)
```

Finally, we fit a 10 component model on the entire dataset.

```
reg_best = regsubsets(Salary ~ ., data = dat, nvmax = 19)
coef(reg_best, 10)
```

```
##  (Intercept)        AtBat         Hits        Walks       CAtBat        CRuns
##  162.5354420   -2.1686501    6.9180175    5.7732246   -0.1300798    1.4082490
##         CRBI       CWalks    DivisionW      PutOuts      Assists
##    0.7743122   -0.8308264 -112.3800575    0.2973726    0.2831680
```