Model Selection and Model Validation

Debopriya Ghosh

2019-07-25

In this lecture we will discuss some ways in which the simple linear model can be improved, by replacing plain least squares fitting with some alternative fitting procedures. There are many alternatives, both classical and modern, to using least squares to fit. Here, we discuss two important classes of methods.

- Subset Selection
- Shrinkage Methods

Subset Selection

In this section we consider some methods for selecting subsets of predictors. These include best subset and stepwise model selection procedures.

Best Subset Selection

- To perform best subset selection, we fit a separate least squares regression best subset selection for each possible combination of the p predictors.
- We then look at all of the resulting models with the aim of identifying the best model.

Algorithm

- Let M_0 denote the null model, which contains no predictors. This model simply predicts the sample mean for each observation.
- For $k = 1, 2, \ldots p$:
  - Fit all $\binom{p}{k}$ kmodels that contain exactly k predictors.
  - Pick the best among these $\binom{p}{k}$ models, and call it M_k. Here, best is defined as having the smallest RSS, or equivalently largest R^2.

- Select a single best model from among $M_0, \ldots, M_p$ using crossvalidated prediction error, C_p, AIC, BIC, or adjusted R^2.

Example

- Here we apply the best subset selection approach to the Hitters data. We wish to predict a baseball player's Salary on the basis of various statistics associated with performance in the previous year.
- We note that the Salary variable is missing for some of the players.

```
library(ISLR)
data(Hitters)
help(Hitters)

sum(is.na(Hitters$Salary))
```

Hitters R Documentation

Baseball Data

Description

Major League Baseball Data from the 1986 and 1987 seasons.

Usage

```
Hitters
```

Format

A data frame with 322 observations of major league players on the following 20 variables.

AtBat

Number of times at bat in 1986

Hits

Number of hits in 1986

HmRun

Number of home runs in 1986

Runs

Number of runs in 1986

RBI

Number of runs batted in in 1986

Walks

Number of walks in 1986

Years

Number of years in the major leagues

CAtBat

Number of times at bat during his career

CHits

Number of hits during his career

CHmRun

Number of home runs during his career

CRuns

Number of runs during his career

CRBI

Number of runs batted in during his career

CWalks

Number of walks during his career

League

A factor with levels `A` and `N` indicating player's league at the end of 1986

Division

A factor with levels `E` and `W` indicating player's division at the end of 1986

PutOuts

Number of put outs in 1986

Assists

Number of assists in 1986

Errors

Number of errors in 1986

Salary

1987 annual salary on opening day in thousands of dollars

NewLeague

A factor with levels `A` and `N` indicating player's league at the beginning of 1987

Source

This dataset was taken from the StatLib library which is maintained at Carnegie Mellon University. This is part of the data that was used in the 1988 ASA Graphics Section Poster Session. The salary data were originally from Sports Illustrated, April 20, 1987. The 1986 and career statistics were obtained from The 1987 Baseball Encyclopedia Update published by Collier Books, Macmillan Publishing Company, New York.

References

James, G., Witten, D., Hastie, T., and Tibshirani, R. (2013) *An Introduction to Statistical Learning with applications in R*, www.StatLearning.com, Springer-Verlag, New York

Examples

```
summary(Hitters)
lm(Salary~AtBat+Hits,data=Hitters)
```

```
## [1] 59
```

- Hence we see that Salary is missing for 59 players. The na.omit() function removes all of the rows that have missing values in any variable.

```
Hitters = na.omit(Hitters)
```

- The regsubsets() function performs best subset selection by identifying the best model that contains a given number of predictors, where best is quantified using RSS.
- The summary() command outputs the best set of variables for each model size.

```
library(leaps)
regfit.full = regsubsets (Salary ~ .,Hitters)
summary(regfit.full)
```

```
## Subset selection object
## Call: regsubsets.formula(Salary ~ ., Hitters)
## 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 8
## Selection Algorithm: exhaustive
##          AtBat Hits HmRun Runs RBI Walks Years CAtBat CHits CHmRun CRuns
## 1  ( 1 ) " "   " "  " "   " "  " " " " " "   " "    " "    " "    " "
## 2  ( 1 ) " "   "*"  " "   " "  " " " " " "   " "    " "    " "    " "
## 3  ( 1 ) " "   "*"  " "   " "  " " " " " "   " "    " "    " "    " "
## 4  ( 1 ) " "   "*"  " "   " "  " " " " " "   " "    " "    " "    " "
## 5  ( 1 ) "*"   "*"  " "   " "  " " " " " "   " "    " "    " "    " "
## 6  ( 1 ) "*"   "*"  " "   " "  " " " "*"   " "    " "    " "    " "
## 7  ( 1 ) " "   "*"  " "   " "  " " " "*"   " "    "*"    "*"    "*"
## 8  ( 1 ) "*"   "*"  " "   " "  " " " "*"   " "    " "    " "    "*"
##          CRBI CWalks LeagueN DivisionW PutOuts Assists Errors NewLeagueN
## 1  ( 1 ) "*"  " "    " "     " "       " "     " "     " "    " "
## 2  ( 1 ) "*"  " "    " "     " "       " "     " "     " "    " "
## 3  ( 1 ) "*"  " "    " "     " "       "*"     " "     " "    " "
## 4  ( 1 ) "*"  " "    " "     "*"       "*"     " "     " "    " "
## 5  ( 1 ) "*"  " "    " "     "*"       "*"     " "     " "    " "
## 6  ( 1 ) "*"  " "    " "     "*"       "*"     " "     " "    " "
## 7  ( 1 ) " "  " "    " "     "*"       "*"     " "     " "    " "
## 8  ( 1 ) " "  "*"    " "     "*"       "*"     " "     " "    " "
```

- An asterisk indicates that a given variable is included in the corresponding model. For instance, this output indicates that the best two-variable model contains only *Hits* and *CRBI*.
- By default,regsubsets() only reports results up to the best eight-variable model. The nvmax option can be used in order to return as many variables as are desired.
- Here we fit up to a 19-variable model:

```
regfit.full = regsubsets (Salary ~ .,data = Hitters, nvmax = 19)
reg.summary = summary(regfit.full)
```

- The summary() function also returns R^2, *RSS*, adjusted R^2, C_p, and *BIC*.
- We can examine these to try to select the best overall model.

```
 names(reg.summary)
```

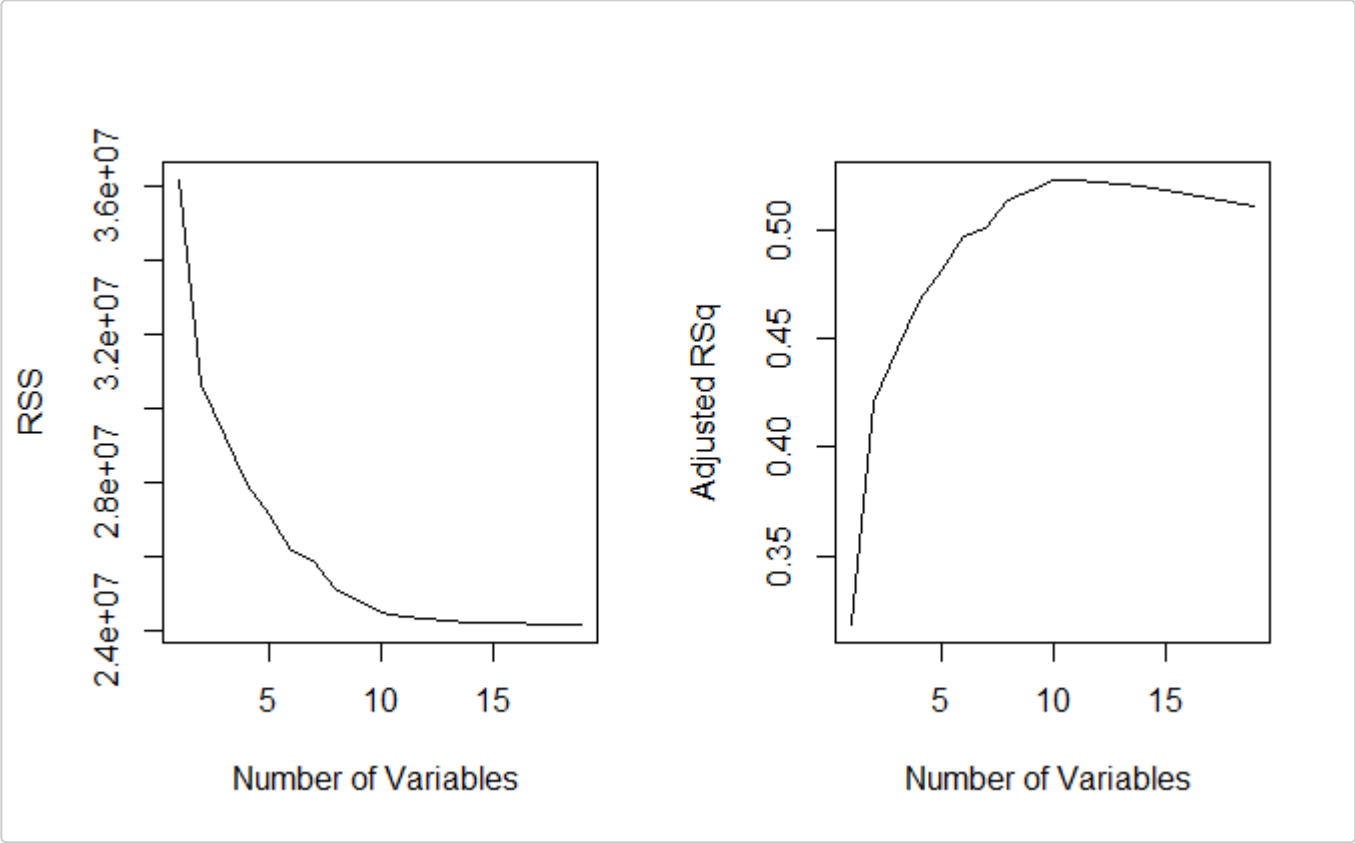
```
## [1] "which"  "rsq"    "rss"    "adjr2"  "cp"     "bic"    "outmat" "obj"
```

```
reg.summary$rsq
```

```
##  [1] 0.3214501 0.4252237 0.4514294 0.4754067 0.4908036 0.5087146 0.5141227
##  [8] 0.5285569 0.5346124 0.5404950 0.5426153 0.5436302 0.5444570 0.5452164
## [15] 0.5454692 0.5457656 0.5459518 0.5460945 0.5461159
```

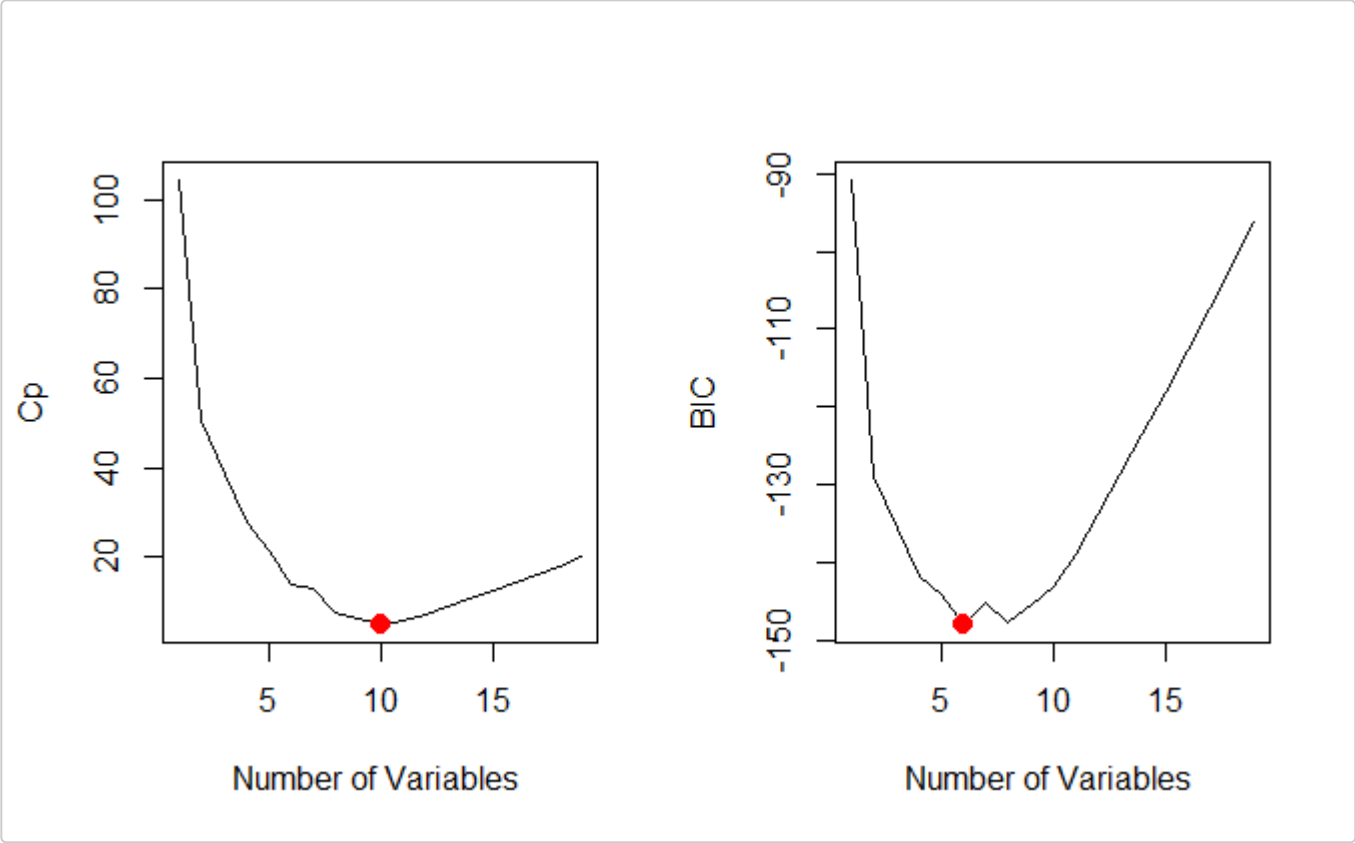
- We see that the R^2 statistic increases from 32%, when only one variable is included in the model, to almost 55%, when all variables are included.
- Plotting RSS, adjusted R^2, C_p, and *BIC* for all of the models at once will help us decide which model to select.

```
par(mfrow=c(1,2))
plot(reg.summary$rss ,xlab="Number of Variables ",ylab="RSS", type="l")
plot(reg.summary$adjr2 ,xlab="Number of Variables ", ylab="Adjusted RSq",type="l")
```



- As the number of features included in the models increases, *RSS* decreases monotonically, and the R^2 increases monotonically.
- If we use these statistics to select the best model, then we will always end up with a model involving all of the variables.
- The problem is that a low *RSS* or a high R^2 indicates a model with a low *training error*, whereas we wish to choose a model that has a low *test error*.
- Therefore, we use *cross-validated prediction*, *AIC*, *BIC*, or adjusted R^2 for model selection.

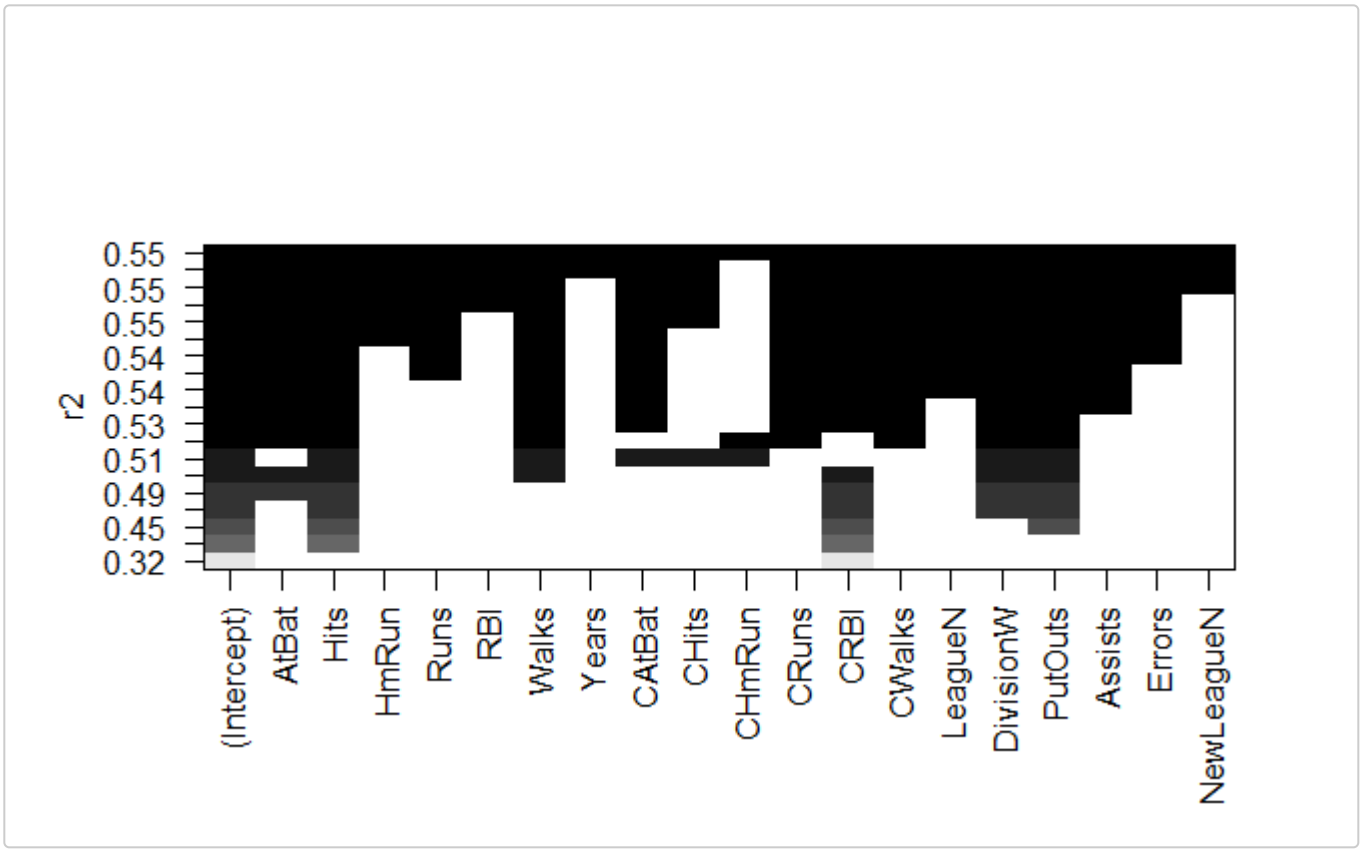
```
par(mfrow=c(1,2))
plot(reg.summary$cp ,xlab="Number of Variables ",ylab="Cp", type = 'l')
which.min(reg.summary$cp )
points(10,reg.summary$cp[10],col="red",cex = 2,pch = 20)
which.min(reg.summary$bic )
plot(reg.summary$bic ,xlab = "Number of Variables ",ylab = "BIC", type = 'l')
points(6,reg.summary$bic[6],col="red",cex = 2,pch = 20)
```



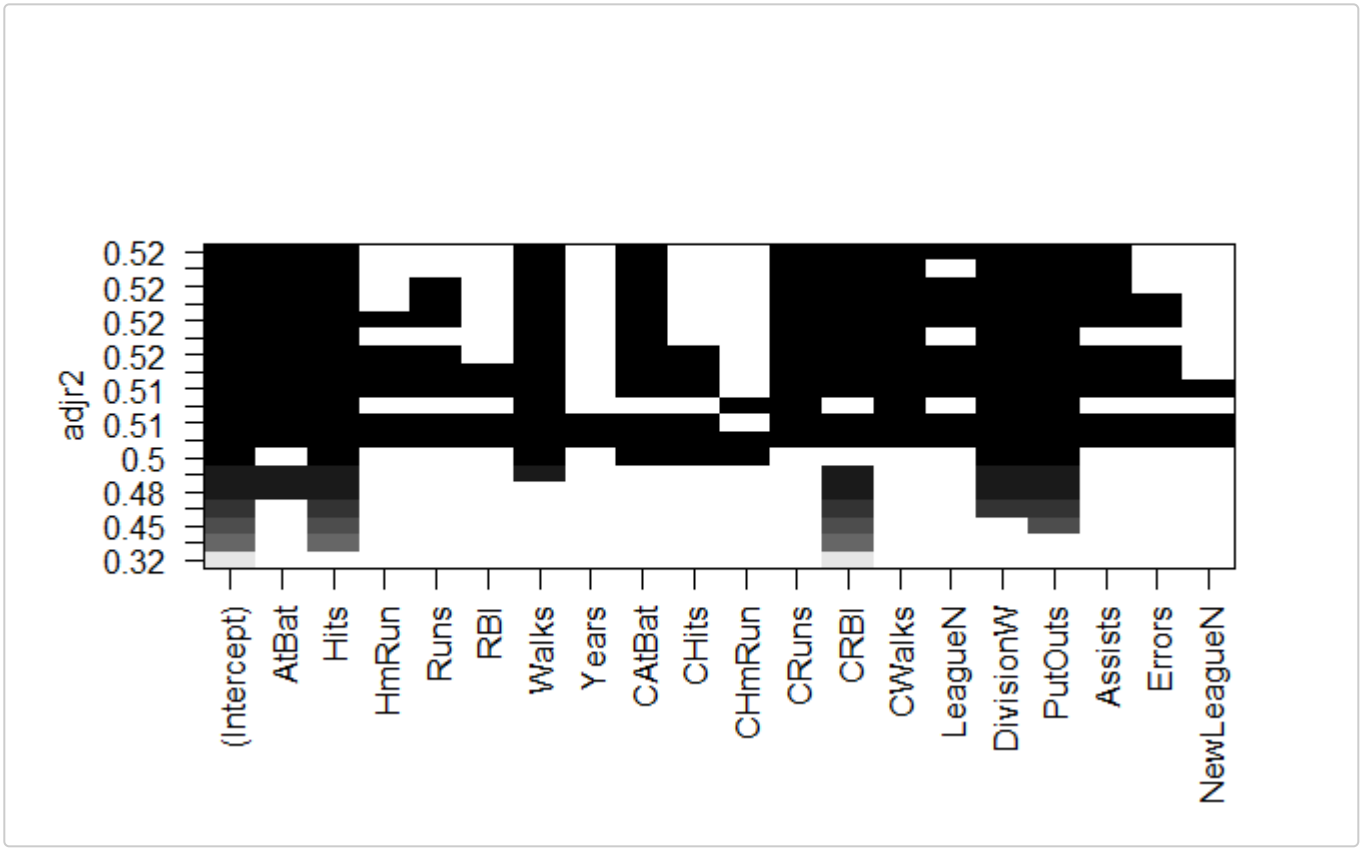

```
## [1] 10
## [1] 6
```

- The regsubsets() function has a built-in plot() command which can be used to display the selected variables for the best model with a given number of predictors, ranked according to the *BIC*, C_p, adjusted R^2, or *AIC*.

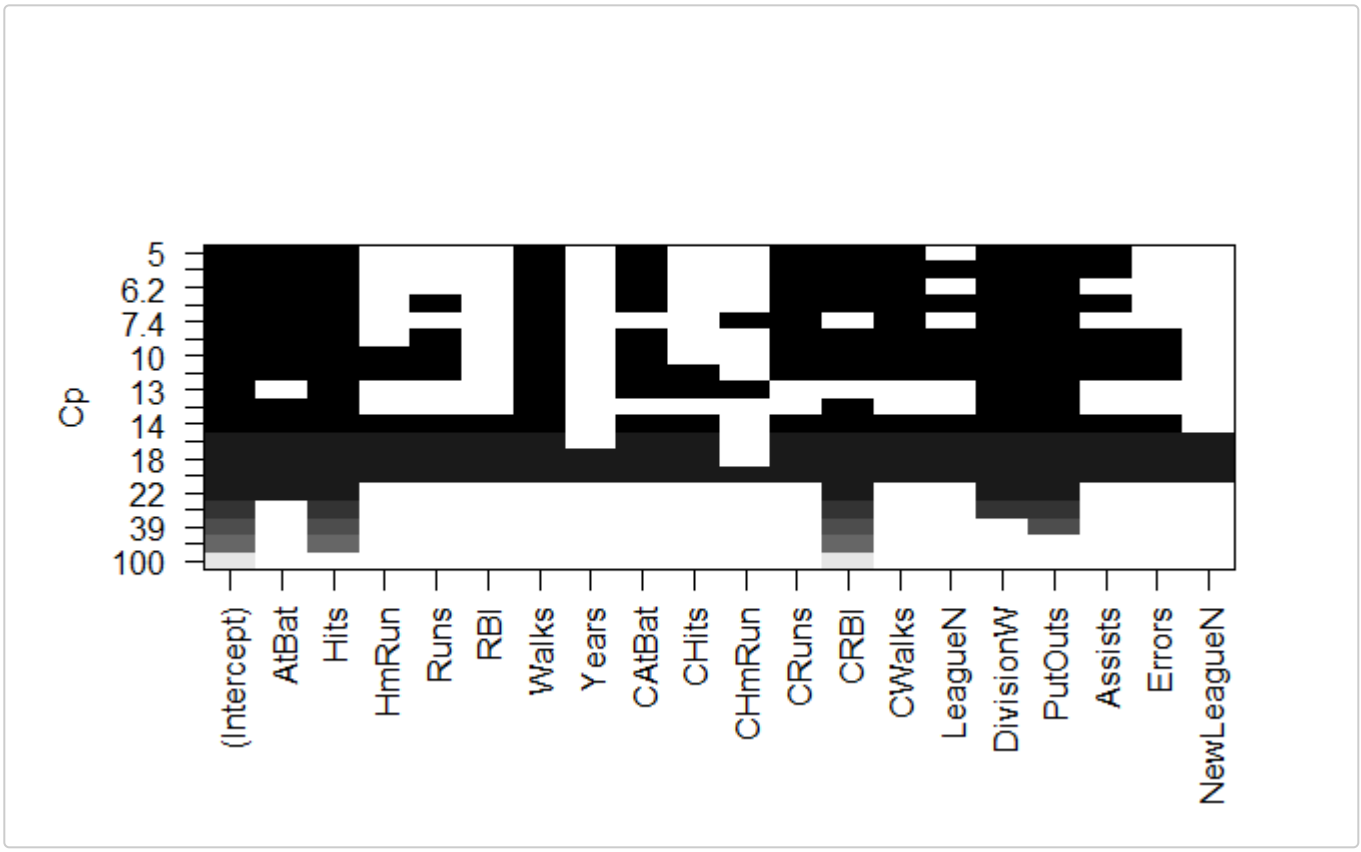
```
plot(regfit.full,scale="r2")
```



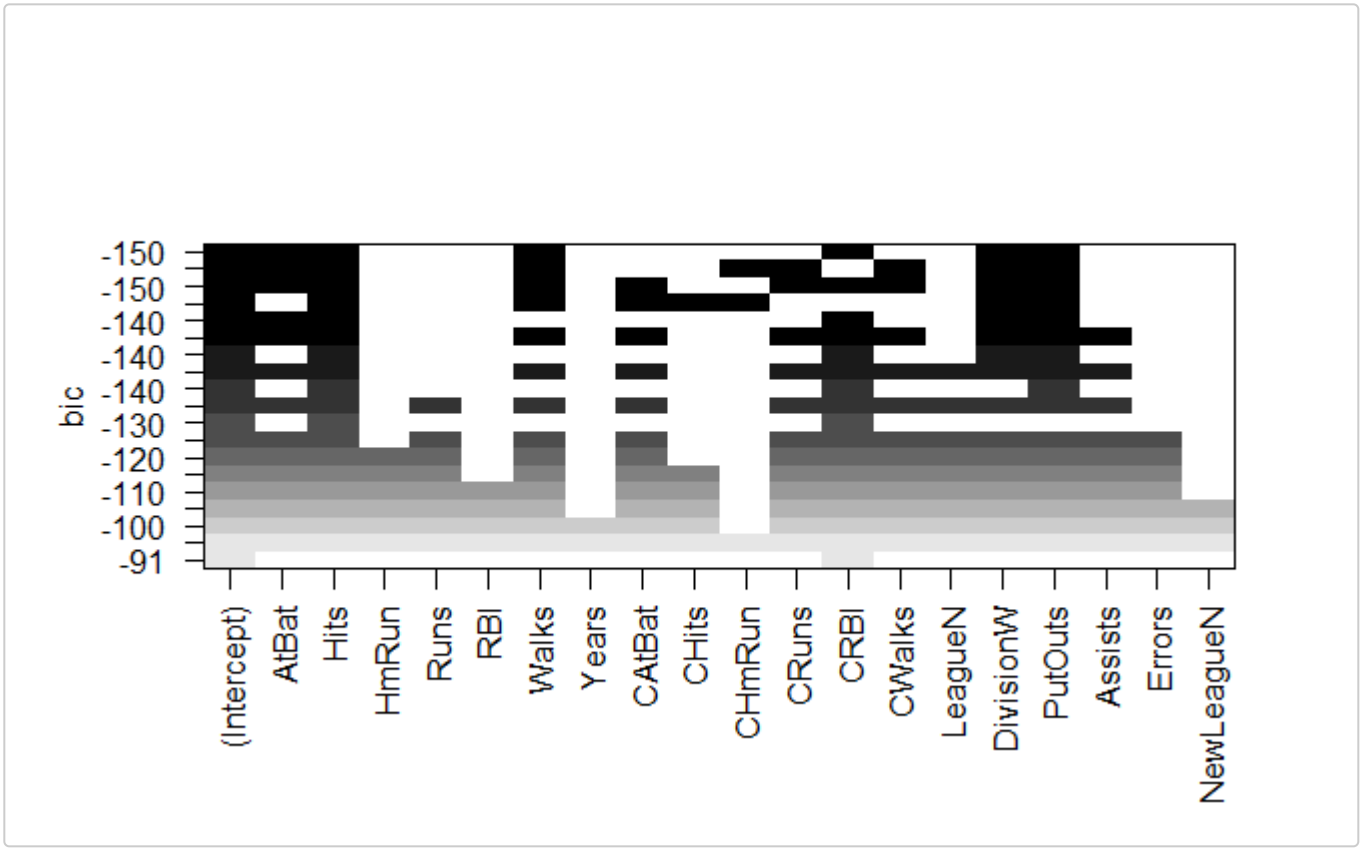

```
plot(regfit.full,scale="adjr2")
```




```
plot(regfit.full,scale="Cp")
```




```
plot(regfit.full,scale="bic")
```



- The top row of each plot contains a black square for each variable selected according to the optimal model associated with that statistic.

- For example, we see that several models share a *BIC* close to 150. However, the model with the lowest BIC is the six-variable model that contains only *AtBat*, *Hits*, *Walks*, *CRBI*, *DivisionW*, *andPutOuts*.
- We can use thecoef() function to see the coefficient estimates associated with this model.

```
coef(regfit.full ,6)
```

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

- For computational reasons, best subset selection cannot be applied with very large *p*.
- Best subset selection may also suffer from statistical problems when p is large.
- The larger the search space, the higher the chance of finding models that look good on the training data, even though they might not have any predictive power on future data.
- Thus an enormous search space can lead to overfitting and high variance of the coefficient estimates.

Forward and Backward Stepwise Selection

- Forward stepwise selection begins with a model containing no predictors, and then adds predictors to the model, one-at-a-time, until all of the predictors are in the model.
- In particular, at each step the variable that gives the greatest additional improvement to the fit is added to the model.
- Backward stepwise selection, it begins with the full least squares model containing all *p* predictors, and then iteratively removes the least useful predictor, one-at-a-time.
- Backward selection requires that the number of samples *n* is larger than the number of variables *p* (so that the full model can be fit).
- In contrast, forward stepwise can be used even when $n < p$, and so is the only viable subset method when *p* is very large.
- We can also use the regsubsets() function to perform forward stepwise or backward stepwise selection, using the argument method = “forward” or method = “backward”.

```
regfit.fwd = regsubsets (Salary ~ .,data = Hitters ,nvmax = 19, method = "forward")
summary(regfit.fwd)
```

```
## Subset selection object
## Call: regsubsets.formula(Salary ~ ., data = Hitters, 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
## 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 ) "*"   "*"  "*"   "*"  "*" "*"   "*"   "*"    "*"   "*"    "*"
##           CRBI 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 ) "*"  "*"    "*"     "*"       "*"     "*"     "*"    "*"
```

```
regfit.bwd = regsubsets (Salary ~ .,data = Hitters ,nvmax = 19, method = "backward")
summary(regfit.bwd)
```

```
## Subset selection object
## Call: regsubsets.formula(Salary ~ ., data = Hitters, 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
## 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 ) "*"   "*"  "*"   "*"  "*" "*"   "*"   "*"    "*"   "*"    "*"
##           CRBI 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 ) "*"  "*"    "*"     "*"       "*"     "*"     "*"    "*"
```

- We see that using forward stepwise selection, the best one variable model contains only *CRBI*, and the best two-variable model additionally includes *Hits*.
- For this data, the best one-variable through sixvariable models are each identical for best subset and forward selection.
- However, the best seven-variable models identified by forward stepwise selection, backward stepwise selection, and best subset selection are different.

```
coef(regfit.full ,7)
coef(regfit.fwd ,7)
coef(regfit.bwd ,7)
```

```
##  (Intercept)         Hits        Walks       CAtBat        CHits
##   79.4509472    1.2833513    3.2274264   -0.3752350    1.4957073
##       CHmRun    DivisionW      PutOuts
##    1.4420538 -129.9866432    0.2366813
##  (Intercept)        AtBat         Hits        Walks         CRBI
##  109.7873062   -1.9588851    7.4498772    4.9131401    0.8537622
##       CWalks    DivisionW      PutOuts
##   -0.3053070 -127.1223928    0.2533404
##  (Intercept)        AtBat         Hits        Walks        CRuns
##  105.6487488   -1.9762838    6.7574914    6.0558691    1.1293095
##       CWalks    DivisionW      PutOuts
##   -0.7163346 -116.1692169    0.3028847
```

- Both forward stepwise selection and backward stepwise selection are not guaranteed to yield the best model containing a subset of the p predictors.

Choosing the Optimal Model

- Best subset selection, forward selection, and backward selection result in the creation of a set of models, each of which contains a subset of the p predictors.
- In order to implement these methods, we need a way to determine which of these models is best.
- The model containing all of the predictors will always have the smallest *RSS* and the largest R^2, since these quantities are related to the training error.
- Also, *RSS* and R^2 are not suitable for selecting the best model among a collection of models with diferent numbers of predictors.
- Here, we wish to choose a model with a low test error.

Validation Set Approach

- In order to use the validation set approach, we begin by splitting the observations into a training set and a test set.

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

- Now, we apply regsubsets() to the training set in order to perform best subset selection.

```
regfit.best = regsubsets (Salary ~ .,data = Hitters[train,], nvmax = 19)
```

- We now compute the validation set error for the best model of each model size.
- We first make a model matrix from the test data.

```
test.mat = model.matrix(Salary ~ .,data = Hitters[test,])
```

- The model.matrix() function is used in many regression packages for building an "X" matrix from data.
- Now we run a loop, and for each size *i*, we extract the coefficients from regfit.best for the best model of that size, multiply them into the appropriate columns of the test model matrix to form the predictions, and compute the test MSE.

```
val.errors = rep(NA,19)
for(i in 1:19){
  coefi = coef(regfit.best ,id = i)
  pred = test.mat[,names(coefi)] %*% coefi
  val.errors[i] = mean((Hitters$Salary[test]-pred)^2)
}
```

We find that the best model is the one that contains ten variables.

```
val.errors
which.min(val.errors)
coef(regfit.best ,10)
```

```
##  [1] 164377.3 144405.5 152175.7 145198.4 137902.1 139175.7 126849.0
##  [8] 136191.4 132889.6 135434.9 136963.3 140694.9 140690.9 141951.2
## [15] 141508.2 142164.4 141767.4 142339.6 142238.2
## [1] 7
##  (Intercept)        AtBat         Hits        HmRun        Walks
##   71.8074075   -1.5038124    5.9130470  -11.5241809    8.4349759
##       CAtBat        CRuns         CRBI       CWalks    DivisionW
##   -0.1654850    1.7064330    0.7903694   -0.9107515 -109.5616997
##      PutOuts
##    0.2426078
```

- Finally, we perform best subset selection on the full data set, and select the best ten-variable model.
- It is important that we make use of the full data set in order to obtain more accurate coefficient estimates.
- Note that we perform best subset selection on the full data set and select the best tenvariable model, rather than simply using the variables that were obtained from the training set, because the best ten-variable model on the full data set may differ from the corresponding model on the training set.

```
regfit.best = regsubsets (Salary ~.,data = Hitters,nvmax = 19)
coef(regfit.best ,10)
```

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

- Note, the best ten-variable model on the full data set has a different set of variables than the best ten-variable model on the training set.

Cross-Validation Method

- We now try to choose among the models of different sizes using crossvalidation.
- This approach is somewhat involved, as we must perform best subset selection within each of the k training sets.
- First, we create a vector that allocates each observation to one of k = 10 folds, and we create a matrix in which we will store the results.

```
k = 10
set.seed(1)
folds = sample(1:k,nrow(Hitters),replace=TRUE)
cv.errors = matrix(NA,k,19, dimnames =list(NULL , paste(1:19)))
```

- Now we write a for loop that performs cross-validation.

- The elements of the jth fold constitute the test set, and the remainder are in the training set.
- We make our predictions for each model size (using our new predict() method), compute the test errors on the appropriate subset, and store them in the appropriate slot in the matrix cv.errors.

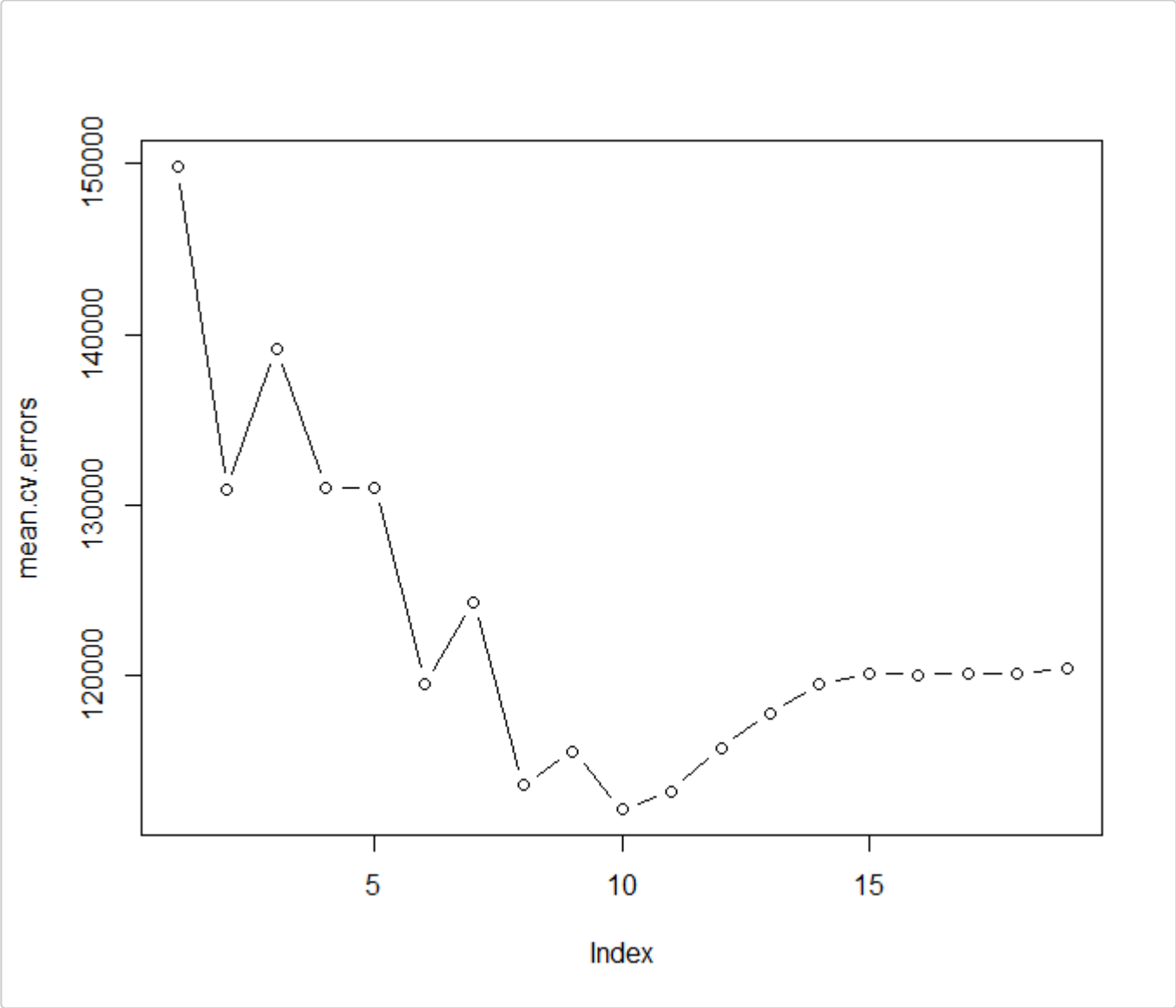
```
# defining predict function

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


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

- This has given us a 10×19 matrix, of which the (i,j)th element corresponds to the test MSE for the ith cross-validation fold for the best j-variable model.
- We use the apply() function to average over the columns of this matrix in order to obtain a vector for which the jth element is the crossvalidation error for the j-variable model.

```
mean.cv.errors = apply(cv.errors,2,mean)
mean.cv.errors
par(mfrow=c(1,1))
plot(mean.cv.errors,type = 'b')
```




```
##        1        2        3        4        5        6        7        8
## 149821.1 130922.0 139127.0 131028.8 131050.2 119538.6 124286.1 113580.0
##        9       10       11       12       13       14       15       16
## 115556.5 112216.7 113251.2 115755.9 117820.8 119481.2 120121.6 120074.3
##       17       18       19
## 120084.8 120085.8 120403.5
```

- We see that cross-validation selects an 10-variable model.
- We now perform best subset selection on the full data set in order to obtain the 10-variable model.

```
reg.best = regsubsets (Salary ~ .,data = Hitters, nvmax = 19)
coef(reg.best ,10)
```

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

```
##      Assists
##    0.2831680
```

Exercise

Fit a stepwise model using a new dataset: Salaries for Professors. First we need to modify the dataset.

```
library(car)
data(Salaries)
help(Salaries)
```

Salaries R Documentation

Salaries for Professors

Description

The 2008-09 nine-month academic salary for Assistant Professors, Associate Professors and Professors in a college in the U.S. The data were collected as part of the on-going effort of the college's administration to monitor salary differences between male and female faculty members.

Usage

```
Salaries
```

Format

A data frame with 397 observations on the following 6 variables.

rank

a factor with levels `AssocProf` `AsstProf` `Prof`

discipline

a factor with levels `A` ("theoretical" departments) or `B` ("applied" departments).

yrs.since.phd

years since PhD.

yrs.service

years of service.

sex

a factor with levels `Female` `Male`

salary

nine-month salary, in dollars.

References

Fox J. and Weisberg, S. (2011) *An R Companion to Applied Regression*, Second Edition Sage.

We will delete the “rownames” column after we use it to rename the rows. This is easy to see but hard to explain! Then we remove missing data. Then re-label the levels of discipline.

```
Salaries$rownames = rownames(Salaries)
Salaries$rownames = NULL
Salaries.old = Salaries
Salaries = na.omit(Salaries)
levels(Salaries$discipline) = c("Theorectical", "Applied")
some(Salaries)
```

	rank	discipline	yrs.since.phd	yrs.service	sex	salary
103	Prof	Applied	16	5	Male	153303
162	Prof	Applied	26	19	Male	176500
180	AsstProf	Applied	3	3	Female	92000
185	Prof	Applied	23	23	Male	101000
208	Prof	Applied	18	18	Male	120000
219	AssocProf	Applied	14	7	Female	109650
253	Prof	Theorectical	31	12	Male	132000
324	Prof	Applied	24	15	Female	161101
358	Prof	Theorectical	39	35	Male	107309
383	AssocProf	Theorectical	8	5	Male	86895

Exercise 1.

Perform stepwise regression starting with the full model using all the predictors of salary.

Exercise 2.

Compare the coefficients of the stepwise model and the full model.

Exercise 3.

Which variable of variables did stepwise drop from the full model?

Exercise 4.

Perform a cross-validation of the stepwise model.

Exercise 5.

Compare the two models using the mse's from the cross-validations with number of folds equal to 3. Which model gives the better mse?