Chapter 6 in ISLR: Part 1 Model Selection

Laura Kapitula

library(ISLR) #contains the credit data  
library(tidyverse)  
library(moderndive)  
theme\_set(theme\_classic())

some of the data analysis below is adapted from <http://www.science.smith.edu/~jcrouser/SDS293/labs/lab8-r.html> ## Model Selection and Shrinkage

Recall the linear model

* How can we find a sweet spot in the variance bias trade-off?
* We consider some approaches for model selection and regularization (ie shrinkage).

## Three classes of methods

* **Subset Selection**. We identify a subset of the p predictors that we believe to be related to the response. We then fit a model using least squares on the reduced set of variables.
* **Shrinkage**. We fit a model involving all p predictors, but the estimated coefficients are shrunken towards zero relative to the least squares estimates. This shrinkage (also known as regularization) has the effect of reducing variance and can also perform variable selection.
* **Dimension Reduction**. This is also sometimes called automatic feature creation. We project the p predictors into a M-dimensional subspace, where M < p. This is achieved by computing M different linear combinations, or projections, of the variables. Then these M projections are used as predictors to fit a linear regression model by least squares.

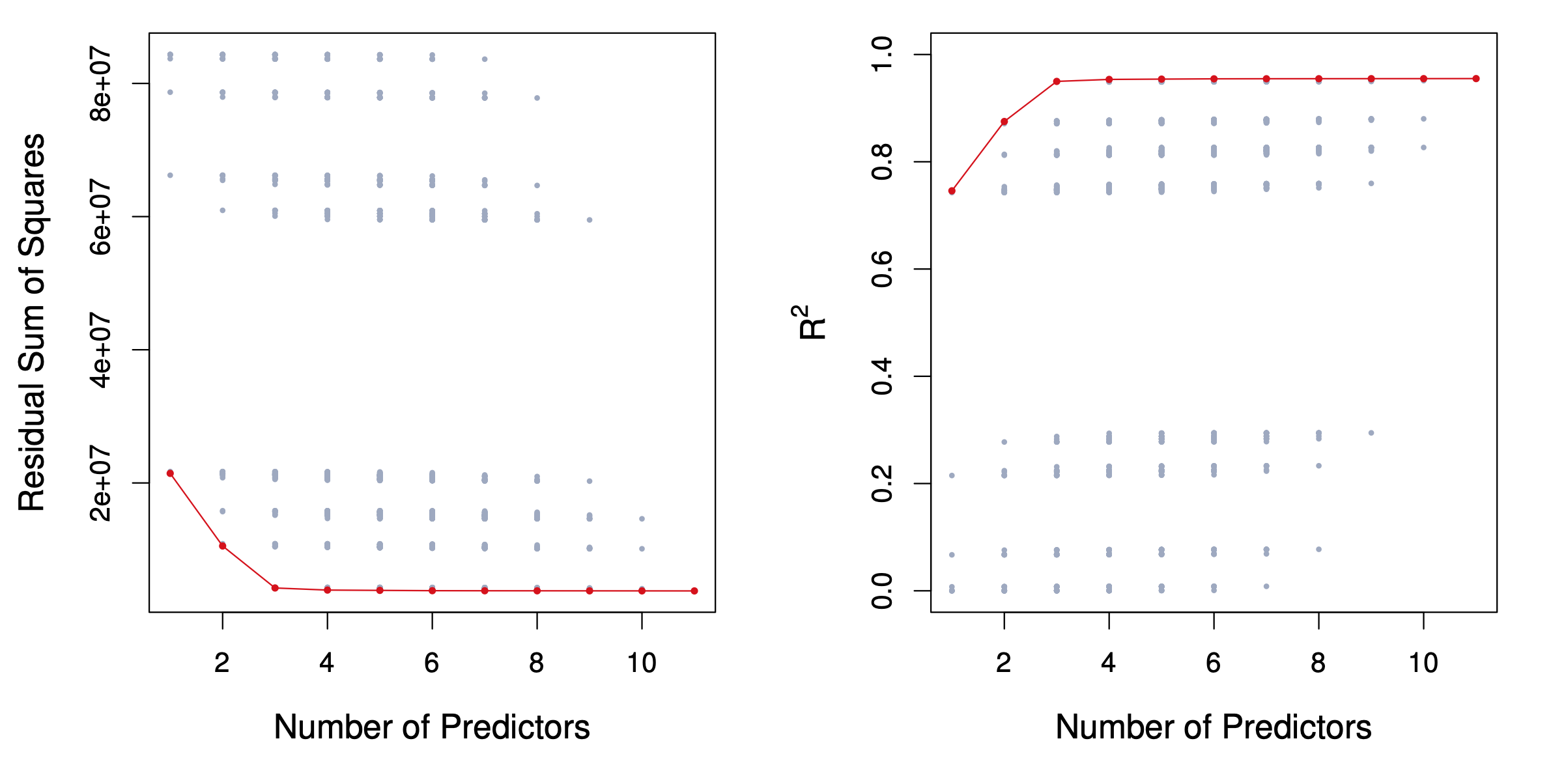
## Subset Selection

*Best subset and stepwise model selection procedures*

## Best Subset Selection

1. Let denote the null model, which contains no predictors. This model simply predicts the sample mean for each observation.
2. For :
3. Fit all combinations of models that contain exactly models that contain exactly k predictors.
4. Pick the best among these models and call it . Here best can be defined as having the smallest , or equivalently largest .
5. Select a single best model from among using cross-validated prediction error, (AIC), BIC, or adjusted .

## Example- Credit data set

 For each possible model containing a subset of the ten predictors in the Credit data set, the RSS and are displayed. The red frontier tracks the best model for a given number of predictors, according to RSS and . Though the data set contains only ten predictors, the x-axis ranges from 1 to 11, since one of the variables is categorical and takes on three values, leading to the creation of two dummy variables

Best subset with regsubsets

data(Credit, package = "ISLR")  
library(leaps) # where regsubsets is found  
regfit\_full= regsubsets(Balance ~ ., data=Credit[2:12]) #id is in column 1, so I leave it off  
reg\_summary=summary(regfit\_full)  
reg\_summary

## Subset selection object  
## Call: regsubsets.formula(Balance ~ ., data = Credit[2:12])  
## 11 Variables (and intercept)  
## Forced in Forced out  
## Income FALSE FALSE  
## Limit FALSE FALSE  
## Rating FALSE FALSE  
## Cards FALSE FALSE  
## Age FALSE FALSE  
## Education FALSE FALSE  
## GenderFemale FALSE FALSE  
## StudentYes FALSE FALSE  
## MarriedYes FALSE FALSE  
## EthnicityAsian FALSE FALSE  
## EthnicityCaucasian FALSE FALSE  
## 1 subsets of each size up to 8  
## Selection Algorithm: exhaustive  
## Income Limit Rating Cards Age Education GenderFemale StudentYes  
## 1 ( 1 ) " " " " "\*" " " " " " " " " " "   
## 2 ( 1 ) "\*" " " "\*" " " " " " " " " " "   
## 3 ( 1 ) "\*" " " "\*" " " " " " " " " "\*"   
## 4 ( 1 ) "\*" "\*" " " "\*" " " " " " " "\*"   
## 5 ( 1 ) "\*" "\*" "\*" "\*" " " " " " " "\*"   
## 6 ( 1 ) "\*" "\*" "\*" "\*" "\*" " " " " "\*"   
## 7 ( 1 ) "\*" "\*" "\*" "\*" "\*" " " "\*" "\*"   
## 8 ( 1 ) "\*" "\*" "\*" "\*" "\*" " " "\*" "\*"   
## MarriedYes EthnicityAsian EthnicityCaucasian  
## 1 ( 1 ) " " " " " "   
## 2 ( 1 ) " " " " " "   
## 3 ( 1 ) " " " " " "   
## 4 ( 1 ) " " " " " "   
## 5 ( 1 ) " " " " " "   
## 6 ( 1 ) " " " " " "   
## 7 ( 1 ) " " " " " "   
## 8 ( 1 ) " " "\*" " "

Best model with 4 variables includes: Income, Limit, Cards, Student.

## Choosing k

Naturally, RSS and improve as we increase k.

To optimize k, we want to minimize the test error, not the training error.

We could use cross-validation, or alternative estimates of test error:

* Akaike Information Criterion (AIC) (closely related to Mallow’s ) given an estimate of the irreducible error :
* Bayesian Information Criterion (BIC):

\* Adjusted R2:

Notice that rather than letting the results of our call to the summary() function print to the screen, we’ve saved the results to a variable called reg\_summary. That way, we can access just the parts we need. Let’s see what’s in there:

names(reg\_summary)

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

Excellent! In addition to the verbose output we get when we print the summary to the screen, the summary() function also returns $R^2 (\tt{rsq})$, RSS, adjusted , , and BIC. We can examine these to try to select the best overall model. Let’s start by looking at :

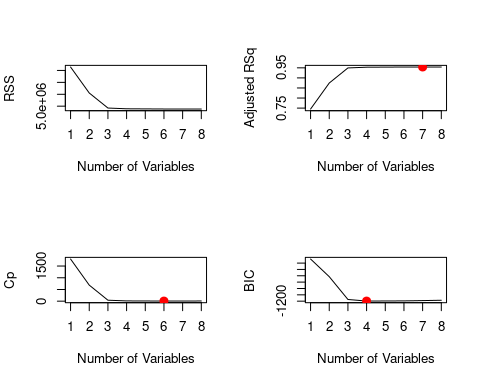
reg\_summary$rsq

## [1] 0.7458484 0.8751179 0.9498788 0.9535800 0.9541606 0.9546879 0.9548167  
## [8] 0.9548880

We see that the statistic increases from 75% when only one variable is included in the model to over 95% when all variables are included. As expected, the statistic increases monotonically as more variables are included.

Plotting RSS, adjusted , , and BIC for all of the models at once will help us decide which model to select. Note the type="l" option tells R to connect the plotted points with lines:

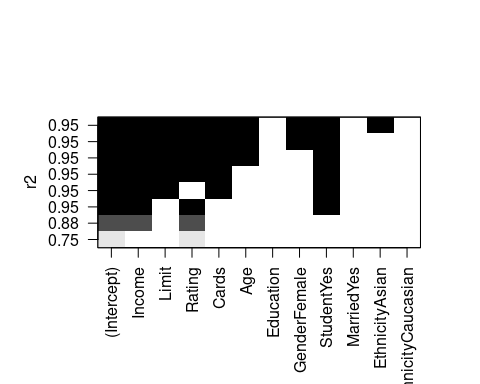
# code below from http://www.science.smith.edu/~jcrouser/SDS293/labs/lab8-r.html  
# Set up a 2x2 grid so we can look at 4 plots at once  
par(mfrow = c(2,2))  
plot(reg\_summary$rss, xlab = "Number of Variables", ylab = "RSS", type = "l")  
plot(reg\_summary$adjr2, xlab = "Number of Variables", ylab = "Adjusted RSq", type = "l")  
  
# We will now plot a red dot to indicate the model with the largest adjusted R^2 statistic.  
# The which.max() function can be used to identify the location of the maximum point of a vector  
adj\_r2\_max = which.max(reg\_summary$adjr2)   
# The points() command works like the plot() command, except that it puts points   
# on a plot that has already been created instead of creating a new plot  
points(adj\_r2\_max, reg\_summary$adjr2[adj\_r2\_max], col ="red", cex = 2, pch = 20)  
  
# We'll do the same for C\_p and BIC, this time looking for the models with the SMALLEST statistic  
plot(reg\_summary$cp, xlab = "Number of Variables", ylab = "Cp", type = "l")  
cp\_min = which.min(reg\_summary$cp)   
points(cp\_min, reg\_summary$cp[cp\_min], col = "red", cex = 2, pch = 20)  
  
plot(reg\_summary$bic, xlab = "Number of Variables", ylab = "BIC", type = "l")  
bic\_min = which.min(reg\_summary$bic)   
points(bic\_min, reg\_summary$bic[bic\_min], col = "red", cex = 2, pch = 20)

 Recall that in the second step of our selection process, we narrowed the field down to just one model on any predictors. We see that according to BIC, the best performer is the model with 4 variables. According to , 6 variables. Adjusted suggests that 7 might be best. No one measure and no one model will necessarily be flagged as best.

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

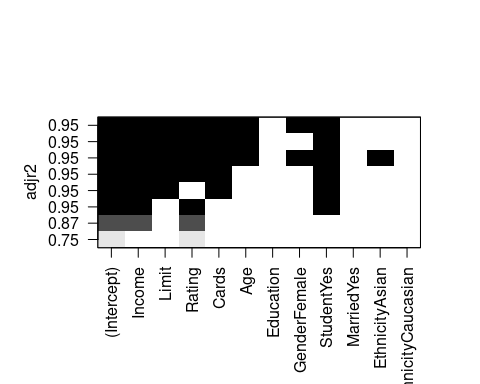
To find out more about this function, type ?plot.regsubsets.

plot(regfit\_full, scale="r2")



As expected, is maximized by the model that contains all the predictors.

plot(regfit\_full, scale="adjr2")

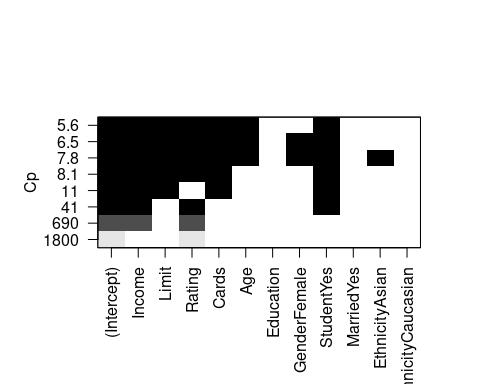


Adjusted downselects to 7 predictors. We can use the coef() function to see which predictors made the cut:

coef(regfit\_full, 7)

## (Intercept) Income Limit Rating Cards Age   
## -488.6158695 -7.8036338 0.1936237 1.0940490 18.1091708 -0.6206538   
## GenderFemale StudentYes   
## -10.4531521 426.5812620

plot(regfit\_full, scale="Cp")

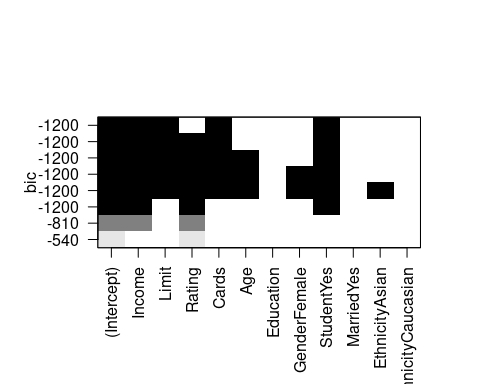


downselects further to 6 variables.

coef(regfit\_full, 6)

## (Intercept) Income Limit Rating Cards Age   
## -493.7341870 -7.7950824 0.1936914 1.0911874 18.2118976 -0.6240560   
## StudentYes   
## 425.6099369

plot(regfit\_full, scale="bic")



We see that several models share a BIC close to -1200. However, the model with the lowest BIC is the four-variable model given below.

coef(regfit\_full, 4)

## (Intercept) Income Limit Cards StudentYes   
## -499.7272117 -7.8392288 0.2666445 23.1753794 429.6064203

# Forward and Backward Stepwise Selection

For computational reasons, best subset selection cannot be applied with very large p. Why not?

* Best subset selection may also suffer from statistical problems when p is large: 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.
* For both of these reasons, stepwise methods, which explore a far more restricted set of models, are attractive alternatives to best subset selection.

There are simulations that have been done that have shown Stepwise methods will often not work well at finding important predictive variables so do remember just because a stepwise method drops or brings in a variable it does NOT mean that variable is necessarily important or not-important. If your only goal is prediction and you use cross-validation to evaluate the models suggested by stepwise methods they can be used to find a decent predictive model, but I would recommend using a shrinkage method over stepwise. I would NEVER recommend using stepwise using p-value for model selection. Remember once you do model selection your p-values are basically meaningless because all the theory assumes you know the correct model before you start.

## Forward 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.
* In Detail

1. Let denote the null model, which contains no predictors. This model simply predicts the sample mean for each observation.
2. For :
3. Consider all all models that augment the predictors in with one additional predictor.
4. Choose the *best* among these models and call it . Here *best* is defined as having the smallest , or equivalently largest .
5. Select a single best model from among using cross-validated prediction error, (AIC), BIC, or adjusted .

That forward stepwise selection has a computational advantage over best subset selection is clear. However, it is not guaranteed to find the best possible model out of all models ubsets of the p predictors.

We can also use the regsubsets() function to perform forward stepwise or backward stepwise selection, using the argument method="forward" or method="backward".

# Forward  
regfit\_fwd = regsubsets(Balance ~ ., data=Credit[2:12], nvmax=11, method="forward")  
summary(regfit\_fwd)

## Subset selection object  
## Call: regsubsets.formula(Balance ~ ., data = Credit[2:12], nvmax = 11,   
## method = "forward")  
## 11 Variables (and intercept)  
## Forced in Forced out  
## Income FALSE FALSE  
## Limit FALSE FALSE  
## Rating FALSE FALSE  
## Cards FALSE FALSE  
## Age FALSE FALSE  
## Education FALSE FALSE  
## GenderFemale FALSE FALSE  
## StudentYes FALSE FALSE  
## MarriedYes FALSE FALSE  
## EthnicityAsian FALSE FALSE  
## EthnicityCaucasian FALSE FALSE  
## 1 subsets of each size up to 11  
## Selection Algorithm: forward  
## Income Limit Rating Cards Age Education GenderFemale StudentYes  
## 1 ( 1 ) " " " " "\*" " " " " " " " " " "   
## 2 ( 1 ) "\*" " " "\*" " " " " " " " " " "   
## 3 ( 1 ) "\*" " " "\*" " " " " " " " " "\*"   
## 4 ( 1 ) "\*" "\*" "\*" " " " " " " " " "\*"   
## 5 ( 1 ) "\*" "\*" "\*" "\*" " " " " " " "\*"   
## 6 ( 1 ) "\*" "\*" "\*" "\*" "\*" " " " " "\*"   
## 7 ( 1 ) "\*" "\*" "\*" "\*" "\*" " " "\*" "\*"   
## 8 ( 1 ) "\*" "\*" "\*" "\*" "\*" " " "\*" "\*"   
## 9 ( 1 ) "\*" "\*" "\*" "\*" "\*" " " "\*" "\*"   
## 10 ( 1 ) "\*" "\*" "\*" "\*" "\*" " " "\*" "\*"   
## 11 ( 1 ) "\*" "\*" "\*" "\*" "\*" "\*" "\*" "\*"   
## MarriedYes EthnicityAsian EthnicityCaucasian  
## 1 ( 1 ) " " " " " "   
## 2 ( 1 ) " " " " " "   
## 3 ( 1 ) " " " " " "   
## 4 ( 1 ) " " " " " "   
## 5 ( 1 ) " " " " " "   
## 6 ( 1 ) " " " " " "   
## 7 ( 1 ) " " " " " "   
## 8 ( 1 ) " " "\*" " "   
## 9 ( 1 ) "\*" "\*" " "   
## 10 ( 1 ) "\*" "\*" "\*"   
## 11 ( 1 ) "\*" "\*" "\*"

Compare the models with 1-4 predictors here to what we got for best subsets. What do you notice? First three are the same, but 4 predictor model is different.

## Backward Stepwise Selection

* Like forward stepwise selection, backward stepwise selection provides an efficient alternative to best subset selection.
* However, unlike forward 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.
* In Detail

1. Let denote the full model, which contains all the predictors.
2. For $k =p, p-1, …, 1 $:
3. Consider all all models that contain all but one of the predictors in for a total of predictors.
4. Choose the *best* among these models and call it . Here *best* is defined as having the smallest , or equivalently largest .
5. Select a single best model from among using cross-validated prediction error, (AIC), BIC, or adjusted .

* Like forward stepwise selection, the backward selection approach searches through only models, and so can be applied in settings where p is too large to apply best subset selection
* Like forward stepwise selection, backward stepwise selection is not guaranteed to yield the *best* model containing a subset of the predictors.
* 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 , and so is the only viable subset method when is very large.

For the Credit data.

# Backward  
regfit\_bwd = regsubsets(Balance ~ ., data=Credit[2:12], nvmax=11, method="backward")  
summary(regfit\_bwd)

## Subset selection object  
## Call: regsubsets.formula(Balance ~ ., data = Credit[2:12], nvmax = 11,   
## method = "backward")  
## 11 Variables (and intercept)  
## Forced in Forced out  
## Income FALSE FALSE  
## Limit FALSE FALSE  
## Rating FALSE FALSE  
## Cards FALSE FALSE  
## Age FALSE FALSE  
## Education FALSE FALSE  
## GenderFemale FALSE FALSE  
## StudentYes FALSE FALSE  
## MarriedYes FALSE FALSE  
## EthnicityAsian FALSE FALSE  
## EthnicityCaucasian FALSE FALSE  
## 1 subsets of each size up to 11  
## Selection Algorithm: backward  
## Income Limit Rating Cards Age Education GenderFemale StudentYes  
## 1 ( 1 ) " " "\*" " " " " " " " " " " " "   
## 2 ( 1 ) "\*" "\*" " " " " " " " " " " " "   
## 3 ( 1 ) "\*" "\*" " " " " " " " " " " "\*"   
## 4 ( 1 ) "\*" "\*" " " "\*" " " " " " " "\*"   
## 5 ( 1 ) "\*" "\*" "\*" "\*" " " " " " " "\*"   
## 6 ( 1 ) "\*" "\*" "\*" "\*" "\*" " " " " "\*"   
## 7 ( 1 ) "\*" "\*" "\*" "\*" "\*" " " "\*" "\*"   
## 8 ( 1 ) "\*" "\*" "\*" "\*" "\*" " " "\*" "\*"   
## 9 ( 1 ) "\*" "\*" "\*" "\*" "\*" " " "\*" "\*"   
## 10 ( 1 ) "\*" "\*" "\*" "\*" "\*" " " "\*" "\*"   
## 11 ( 1 ) "\*" "\*" "\*" "\*" "\*" "\*" "\*" "\*"   
## MarriedYes EthnicityAsian EthnicityCaucasian  
## 1 ( 1 ) " " " " " "   
## 2 ( 1 ) " " " " " "   
## 3 ( 1 ) " " " " " "   
## 4 ( 1 ) " " " " " "   
## 5 ( 1 ) " " " " " "   
## 6 ( 1 ) " " " " " "   
## 7 ( 1 ) " " " " " "   
## 8 ( 1 ) " " "\*" " "   
## 9 ( 1 ) "\*" "\*" " "   
## 10 ( 1 ) "\*" "\*" "\*"   
## 11 ( 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 six-variable 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.

models=  
for (k in 1:5){  
full=coef(regfit\_full, k)  
fwd=coef(regfit\_fwd, k)  
bwd=coef(regfit\_bwd, k)  
print(full)  
print(fwd)  
print(bwd)}

## (Intercept) Rating   
## -390.84634 2.56624   
## (Intercept) Rating   
## -390.84634 2.56624   
## (Intercept) Limit   
## -292.7904955 0.1716373   
## (Intercept) Income Rating   
## -534.812150 -7.672124 3.949265   
## (Intercept) Income Rating   
## -534.812150 -7.672124 3.949265   
## (Intercept) Income Limit   
## -385.1792604 -7.6633230 0.2643216   
## (Intercept) Income Rating StudentYes   
## -581.078888 -7.874931 3.987472 418.760284   
## (Intercept) Income Rating StudentYes   
## -581.078888 -7.874931 3.987472 418.760284   
## (Intercept) Income Limit StudentYes   
## -432.3374179 -7.9016203 0.2675379 427.0232667   
## (Intercept) Income Limit Cards StudentYes   
## -499.7272117 -7.8392288 0.2666445 23.1753794 429.6064203   
## (Intercept) Income Limit Rating StudentYes   
## -516.7182606 -7.9446339 0.1216825 2.1904387 422.6683787   
## (Intercept) Income Limit Cards StudentYes   
## -499.7272117 -7.8392288 0.2666445 23.1753794 429.6064203   
## (Intercept) Income Limit Rating Cards StudentYes   
## -526.1555233 -7.8749239 0.1944093 1.0879014 17.8517307 426.8501456   
## (Intercept) Income Limit Rating Cards StudentYes   
## -526.1555233 -7.8749239 0.1944093 1.0879014 17.8517307 426.8501456   
## (Intercept) Income Limit Rating Cards StudentYes   
## -526.1555233 -7.8749239 0.1944093 1.0879014 17.8517307 426.8501456

# Model selection using the Validation Set Approach

We see above that it is possible to choose among a set of models of different sizes using , BIC, and adjusted . We will now consider how to do this using the validation set and cross-validation approaches.

How do these criteria above compare to cross validation?

* They are much less expensive to compute.
* They are motivated by asymptotic arguments and rely on model assumptions (eg. normality of the errors).
* Equivalent concepts for other models (e.g. logistic regression).

## Important Point about using Validation Approach

In order for these approaches to yield accurate estimates of the test error, we must use *only the training observations* to perform all aspects of model-fitting including variable selection. Therefore, the determination of which model of a given size is best must be made using *only the training observations*. If the full data set is used to perform the best subset selection step, the validation set errors and cross-validation errors that we obtain will not be accurate estimates of the test error.

In order to use the validation set approach, we begin by splitting the observations into a training set and a test set as before. Here, we’ve decided to split the data into three-quarters for training and one-quarter for test using the sample\_frac() method:

set.seed(1)  
  
train = Credit[2:12] %>%  
 sample\_frac(0.75)  
  
test = Credit[2:12] %>%  
 setdiff(train)

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

regfit\_best\_train = regsubsets(Balance~., data = train, nvmax = 11) #do best subsets on the training data.  
summary(regfit\_best\_train)

## Subset selection object  
## Call: regsubsets.formula(Balance ~ ., data = train, nvmax = 11)  
## 11 Variables (and intercept)  
## Forced in Forced out  
## Income FALSE FALSE  
## Limit FALSE FALSE  
## Rating FALSE FALSE  
## Cards FALSE FALSE  
## Age FALSE FALSE  
## Education FALSE FALSE  
## GenderFemale FALSE FALSE  
## StudentYes FALSE FALSE  
## MarriedYes FALSE FALSE  
## EthnicityAsian FALSE FALSE  
## EthnicityCaucasian FALSE FALSE  
## 1 subsets of each size up to 11  
## Selection Algorithm: exhaustive  
## Income Limit Rating Cards Age Education GenderFemale StudentYes  
## 1 ( 1 ) " " "\*" " " " " " " " " " " " "   
## 2 ( 1 ) "\*" " " "\*" " " " " " " " " " "   
## 3 ( 1 ) "\*" "\*" " " " " " " " " " " "\*"   
## 4 ( 1 ) "\*" "\*" " " "\*" " " " " " " "\*"   
## 5 ( 1 ) "\*" "\*" " " "\*" "\*" " " " " "\*"   
## 6 ( 1 ) "\*" "\*" " " "\*" "\*" " " "\*" "\*"   
## 7 ( 1 ) "\*" "\*" " " "\*" "\*" " " "\*" "\*"   
## 8 ( 1 ) "\*" "\*" "\*" "\*" "\*" " " "\*" "\*"   
## 9 ( 1 ) "\*" "\*" "\*" "\*" "\*" " " "\*" "\*"   
## 10 ( 1 ) "\*" "\*" "\*" "\*" "\*" " " "\*" "\*"   
## 11 ( 1 ) "\*" "\*" "\*" "\*" "\*" "\*" "\*" "\*"   
## MarriedYes EthnicityAsian EthnicityCaucasian  
## 1 ( 1 ) " " " " " "   
## 2 ( 1 ) " " " " " "   
## 3 ( 1 ) " " " " " "   
## 4 ( 1 ) " " " " " "   
## 5 ( 1 ) " " " " " "   
## 6 ( 1 ) " " " " " "   
## 7 ( 1 ) " " "\*" " "   
## 8 ( 1 ) " " "\*" " "   
## 9 ( 1 ) "\*" "\*" " "   
## 10 ( 1 ) "\*" "\*" "\*"   
## 11 ( 1 ) "\*" "\*" "\*"

Notice that we use the training data here. 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 (Balance~., data = test)

The model.matrix() function is used in many regression packages for building an matrix from data. Now we run a loop, and for each size , 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.

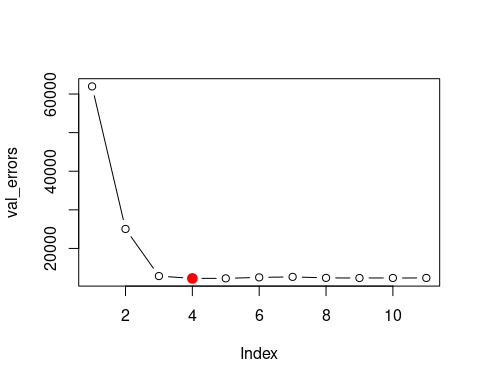
#we use 11 because we have 11 predictors possible.  
val\_errors = rep(NA,11)  
  
# Iterates over each size i  
for(i in 1:11){  
   
 # Extract the vector of predictors in the best fit model on i predictors  
 coefi = coef(regfit\_best\_train, id = i)  
   
 # Make predictions using matrix multiplication of the test matirx and the coefficients vector  
 pred = test\_mat[,names(coefi)]%\*%coefi  
   
 # Calculate the MSE  
 val\_errors[i] = mean((test$Balance-pred)^2)  
}

Now let’s plot the errors, and find the model that minimizes it:

# Find the model with the smallest error  
min = which.min(val\_errors)  
min

## [1] 4

# Plot the errors for each model size  
plot(val\_errors, type = 'b')  
points(min, val\_errors[min][1], col = "red", cex = 2, pch = 20)



Viola! We find that the best model (according to the validation set approach) is the one that contains 4 predictors.

This was a little tedious, partly because there is no predict() method for regsubsets(). Since we will be using this function again, we can capture our steps above and write our own predict() method:

predict.regsubsets = function(object,newdata,id,...){  
 form = as.formula(object$call[[2]]) # Extract the formula used when we called regsubsets()  
 mat = model.matrix(form,newdata) # Build the model matrix  
 coefi = coef(object,id=id) # Extract the coefficiants of the ith model  
 xvars = names(coefi) # Pull out the names of the predictors used in the ith model  
 mat[,xvars]%\*%coefi # Make predictions using matrix multiplication  
}

This function pretty much mimics what we did above. The one tricky part is how we extracted the formula used in the call to regsubsets(), but you don’t need to worry too much about the mechanics of this right now. We’ll use this function to make our lives a little easier when we do cross-validation.

Now that we know what we’re looking for, let’s perform best subset selection on the full dataset (up to 4 predictors) and select the best 4-predictor 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 4-predictor model, rather than simply using the predictors that we obtained from the training set, because the best 4-predictor model on the **full data set** may differ from the corresponding model on the training set.

regfit\_best = regsubsets(Balance~., data = Credit[2:12], nvmax = 4)

Here they are the same.

coef(regfit\_best, 4)

## (Intercept) Income Limit Cards StudentYes   
## -499.7272117 -7.8392288 0.2666445 23.1753794 429.6064203

coef(regfit\_best\_train, 4)

## (Intercept) Income Limit Cards StudentYes   
## -521.5910597 -7.8556330 0.2705757 24.2629771 419.5997841

# Model selection using Cross-Validation

Now let’s try to choose among the models of different sizes using cross-validation. This approach is somewhat involved, as we must perform best subset selection\* within each of the training sets. Despite this, we see that with its clever subsetting syntax, R makes this job quite easy. First, we create a vector that assigns each observation to one of folds, and we create a matrix in which we will store the results:

\* or forward selection / backward selection

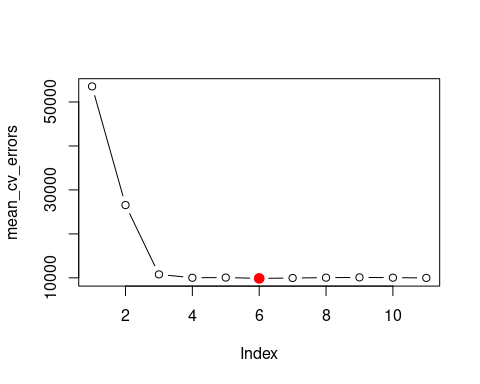
k = 10 # number of folds   
set.seed(1) # set the random seed so we all get the same results  
  
# Assign each observation to a single fold  
folds = sample(1:k, nrow(Credit), replace = TRUE)  
  
# Create a matrix to store the results of our upcoming calculations,   
#we use 11 because we have 11 possible predictors (10 variables 1 with 3 levels)  
cv\_errors = matrix(NA, k, 11, dimnames = list(NULL, paste(1:11)))

Now let’s write a for loop that performs cross-validation. In the fold, the elements of folds that equal are in the test set, and the remainder are in the training set. We make our predictions for each model size (using our new method), compute the test errors on the appropriate subset, and store them in the appropriate slot in the matrix cv.errors.

# Outer loop iterates over all folds  
for(j in 1:k){  
   
 # The perform best subset selection on the full dataset, minus the jth fold  
 best\_fit = regsubsets(Balance~., data = Credit[folds!=j,2:12], nvmax=11)  
   
 # Inner loop iterates over each size i  
 for(i in 1:11){  
   
 # Predict the values of the current fold from the "best subset" model on i predictors  
 pred = predict(best\_fit, Credit[folds==j,2:12], id=i)  
   
 # Calculate the MSE, store it in the matrix we created above  
 cv\_errors[j,i] = mean((Credit$Balance[folds==j]-pred)^2)  
 }  
}

This has filled up the cv\_.\_errors matrix such that the element corresponds to the test MSE for the cross-validation fold for the best -variable model. We can then use the apply() function to take the mean over the columns of this matrix. This will give us a vector for which the element is the cross-validation error for the -variable model.

# Take the mean of over all folds for each model size  
mean\_cv\_errors = apply(cv\_errors, 2, mean)  
  
# Find the model size with the smallest cross-validation error  
min = which.min(mean\_cv\_errors)  
  
# Plot the cross-validation error for each model size, highlight the min  
plot(mean\_cv\_errors, type='b')  
points(min, mean\_cv\_errors[min][1], col = "red", cex = 2, pch = 20)



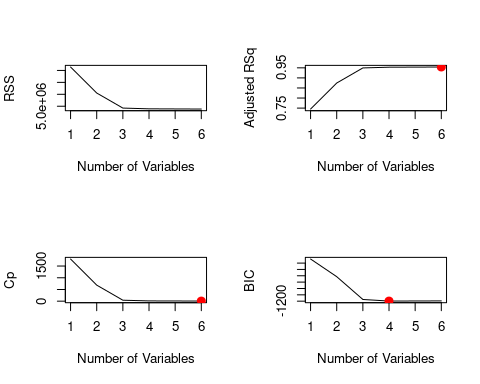
We see that cross-validation selects 6-predictor model. Now let’s use best subset selection on the full data set in order to obtain the 6-predictor model.

reg\_best = regsubsets(Balance~., data = Credit[2:12], nvmax = 6)  
coef(reg\_best, 6)

## (Intercept) Income Limit Rating Cards Age   
## -493.7341870 -7.7950824 0.1936914 1.0911874 18.2118976 -0.6240560   
## StudentYes   
## 425.6099369

For comparison, let’s also take a look at the statistics from above:

par(mfrow=c(2,2))  
  
reg\_summary = summary(reg\_best)  
  
# Plot RSS  
plot(reg\_summary$rss, xlab = "Number of Variables", ylab = "RSS", type = "l")  
  
# Plot Adjusted R^2, highlight max value  
plot(reg\_summary$adjr2, xlab = "Number of Variables", ylab = "Adjusted RSq", type = "l")  
max = which.max(reg\_summary$adjr2)  
points(max, reg\_summary$adjr2[max], col = "red", cex = 2, pch = 20)  
  
# Plot Cp, highlight min value  
plot(reg\_summary$cp, xlab = "Number of Variables", ylab = "Cp", type = "l")  
min = which.min(reg\_summary$cp)  
points(min,reg\_summary$cp[min], col = "red", cex = 2, pch = 20)  
  
# Plot BIC, highlight min value  
plot(reg\_summary$bic, xlab = "Number of Variables", ylab = "BIC", type = "l")  
min = which.min(reg\_summary$bic)  
points(min, reg\_summary$bic[min], col = "red", cex = 2, pch = 20)



Notice how some of the indicators agree with the cross-validated model, and others are very different?

Next time we will walk through using Shrinkage Methods and dimension reduction when you have a lot of predictors.