Linear Model Selection And Regularization

Ravi Kumar Tiwari

13 July 2016

Linear model selection and regularization

It is often the case that some or many of the variables used in a multiple regression model are in fact not associated with the response. By removing these variables, we can obtain a model that is more easily interpreted.

In this module, we see some approaches for automatically performing feature selection for excluding irrelevant variables from a multiple regression model

1. *Subset Selection*: This approach involves identifying 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.
2. *Shrinkage*: This approach involves fitting a model involving all p predictors. However, the estimated coefficients are shrunken towards zero relative to the least squares estimates. Depending on what type of shrinkage (also known as *regularization*) is performed, some of the coefficients may be estimated to be exactly zero. Hence, shrinkage methods can also perform variable selection.

Subset selection

Best subset selection

To perform best subset selection, we fit a separate least squares regression for each possible combination of the p predictors. That is, we fit all p models that contain exactly one predictor, all $C(n, 2)$ = p(p-1)/2 models that contain exactly two predictors, and so forth. We then look at all of the resulting models, with the goal of identifying the one that is best

1. Model creation

```
#install.packages("leaps")
library(leaps)
full <- regsubsets(mpg ~ ., data = mtcars)
fullSum <- summary(full)
names(fullSum)
```

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

2. Best model that contain a given number of predictors

```
fullSum$outmat
```

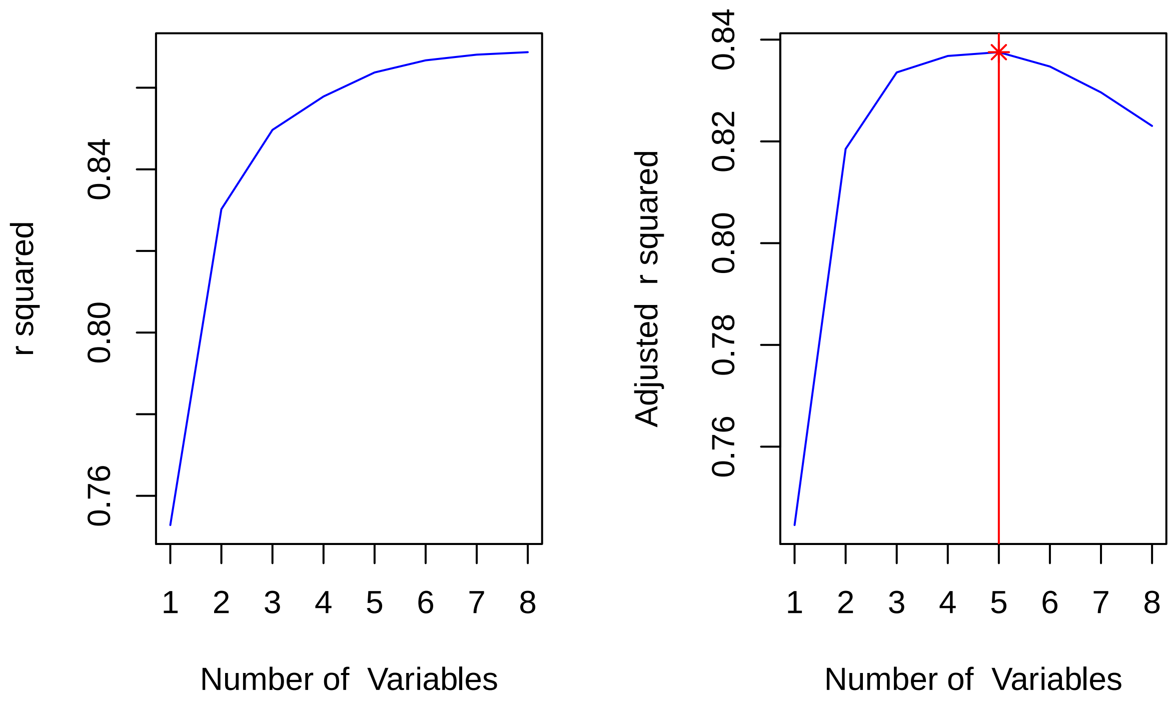
```
##          cyl disp hp  drat wt  qsec vs  am  gear carb
## 1  ( 1 ) " " " "  " " " "  "*" " "  " " " " " "  " " 
## 2  ( 1 ) "*" " "  " " " "  "*" " "  " " " " " "  " " 
## 3  ( 1 ) " " " "  " " " "  "*" "*"  " " "*" " "  " " 
## 4  ( 1 ) " " " "  "*" " "  "*" "*"  " " "*" " "  " "
## 5  ( 1 ) " " "*"  "*" " "  "*" "*"  " " "*" " "  " "
## 6  ( 1 ) " " "*"  "*" "*"  "*" "*"  " " "*" " "  " "
## 7  ( 1 ) " " "*"  "*" "*"  "*" "*"  " " "*" "*"  " "
## 8  ( 1 ) " " "*"  "*" "*"  "*" "*"  " " "*" "*"  "*"
```

3. The best overall model

```
ind <- which.max(fullSum$adjr2)
coef(full, ind)
```

```
## (Intercept)        disp          hp          wt        qsec          am
## 14.36190396  0.01123765 -0.02117055 -4.08433206  1.00689683  3.47045340
```

4. Visualization



Stepwise selection

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.

1. 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.

1. Model creation

```
#install.packages("leaps")
library(leaps)
fwdSelection <- regsubsets(mpg ~ ., data = mtcars, method = "forward")
sumFwdSel <- summary(fwdSelection)
names(sumFwdSel)
```

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

2. Best model that contain a given number of predictors

```
sumFwdSel$outmat
```

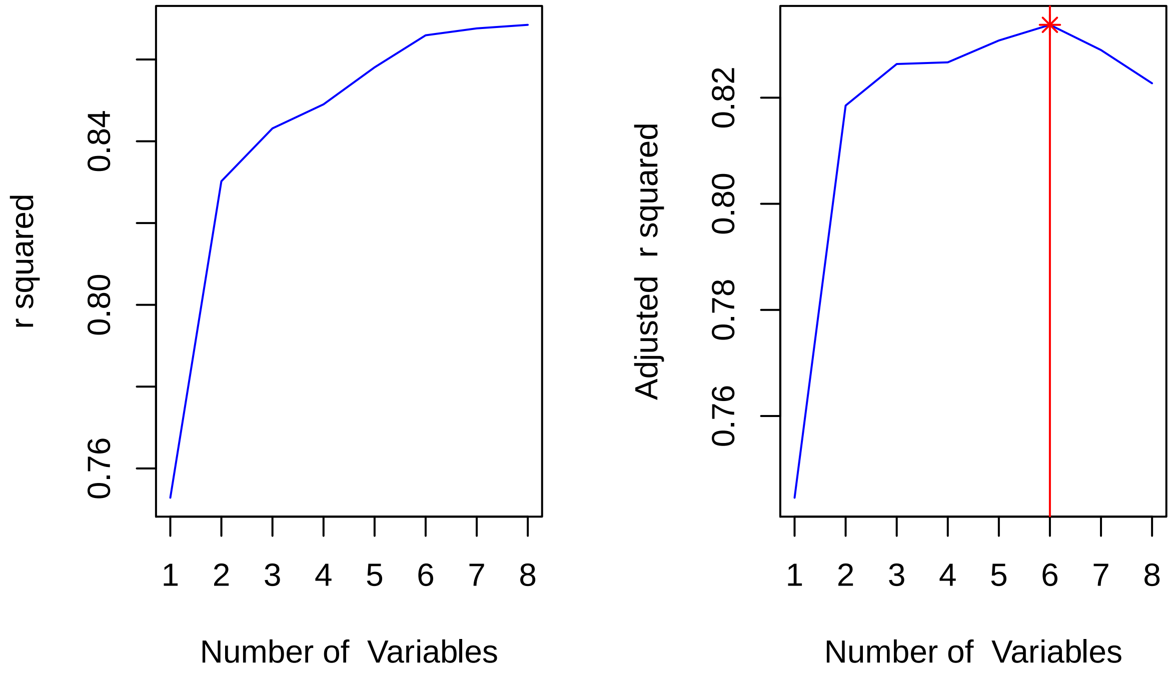
```
##          cyl disp hp  drat wt  qsec vs  am  gear carb
## 1  ( 1 ) " " " "  " " " "  "*" " "  " " " " " "  " " 
## 2  ( 1 ) "*" " "  " " " "  "*" " "  " " " " " "  " " 
## 3  ( 1 ) "*" " "  "*" " "  "*" " "  " " " " " "  " " 
## 4  ( 1 ) "*" " "  "*" " "  "*" " "  " " "*" " "  " " 
## 5  ( 1 ) "*" " "  "*" " "  "*" "*"  " " "*" " "  " " 
## 6  ( 1 ) "*" "*"  "*" " "  "*" "*"  " " "*" " "  " " 
## 7  ( 1 ) "*" "*"  "*" "*"  "*" "*"  " " "*" " "  " " 
## 8  ( 1 ) "*" "*"  "*" "*"  "*" "*"  " " "*" "*"  " " 
```

3. The best overall model

```
ind <- which.max(sumFwdSel$adjr2)
coef(fwdSelection, ind)
```

```
## (Intercept)         cyl        disp          hp          wt        qsec 
## 20.05169952 -0.50206577  0.01396099 -0.01956054 -3.99773180  0.81017782 
##          am 
##  2.94074955 
```

4. Visualization



2. Backward stepwise selection

Backward stepwise selection begins with the full least squares model containing all p predictors, and then iteratively removes the least useful predictor, one-at-a-time.

1. Model creation

```
#install.packages("leaps")
library(leaps)
bwdSelection <- regsubsets(mpg ~ ., data = mtcars, method = "backward")
sumbwdSel <- summary(bwdSelection)
names(sumbwdSel)
```

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

2. Best model that contain a given number of predictors

```
sumbwdSel$outmat
```

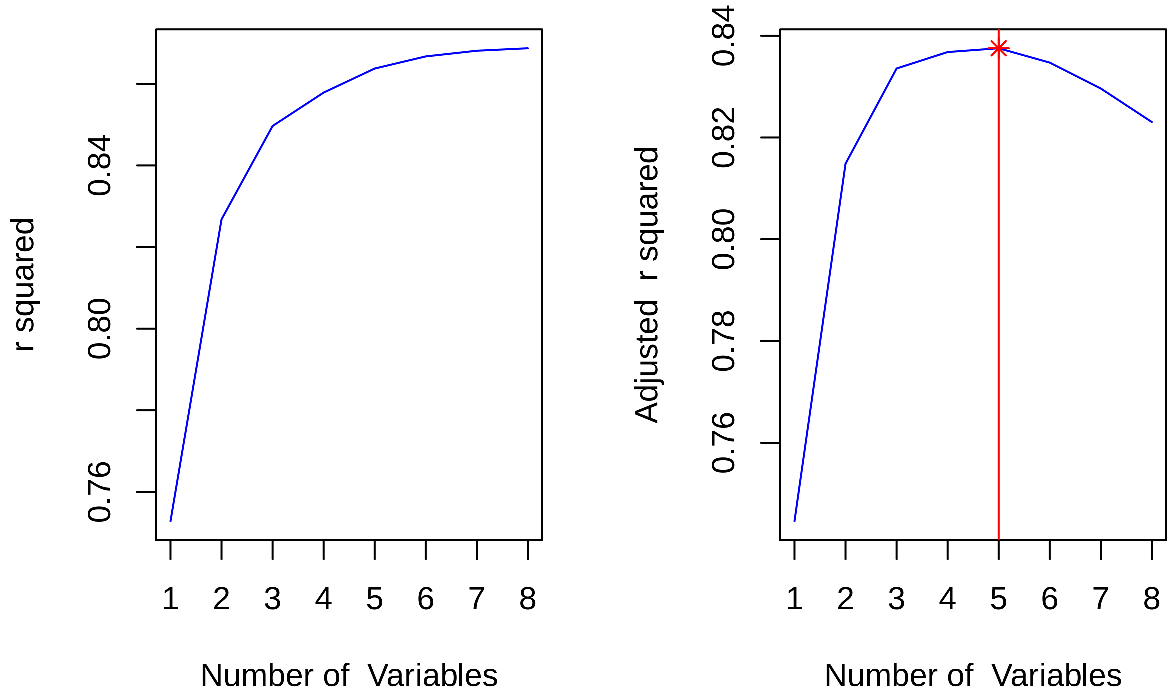
```
##          cyl disp hp  drat wt  qsec vs  am  gear carb
## 1  ( 1 ) " " " "  " " " "  "*" " "  " " " " " "  " " 
## 2  ( 1 ) " " " "  "*" " "  "*" " "  " " " " " "  " " 
## 3  ( 1 ) " " " "  " " " "  "*" "*"  " " "*" " "  " " 
## 4  ( 1 ) " " " "  "*" " "  "*" "*"  " " "*" " "  " "
## 5  ( 1 ) " " "*"  "*" " "  "*" "*"  " " "*" " "  " "
## 6  ( 1 ) " " "*"  "*" "*"  "*" "*"  " " "*" " "  " "
## 7  ( 1 ) " " "*"  "*" "*"  "*" "*"  " " "*" "*"  " "
## 8  ( 1 ) " " "*"  "*" "*"  "*" "*"  " " "*" "*"  "*"
```

3. The best overall model

```r
ind <- which.max(sumbwdSel$adjr2)
coef(bwdSelection, ind)
```

```
## (Intercept)        disp          hp          wt        qsec          am
## 14.36190396  0.01123765 -0.02117055 -4.08433206  1.00689683  3.47045340
```

4. Visualization



2. Shrinkage Method

This approach involves fitting a model involving all p predictors. However, 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

In the case of the lasso, the penalty has the effect of forcing some of the coefficient estimates to be exactly equal to zero when the tuning parameter lambda is sufficiently large. Hence, much like best subset selection, the lasso performs variable selection. As a result, models generated from the lasso are generally much easier to interpret than those produced by ridge regression

Lasso Regression

1. Model Building

```
#install.packages("glmnet")
library(glmnet)
x <- model.matrix(mpg ~ ., mtcars)[,-1]
y <- mtcars$mpg

grid <- 10^seq(10,-2, length = 100)  # lambda values

lassoMod <- glmnet(x, y, alpha = 1,lambda = grid) # alpha = 0 (ridge)
dim(coef(lassoMod))
```

2. Accessing model parameters

```
lassoMod$lambda[10]
```

```
## [1] 811130831
```

```
coef(lassoMod)[,10]
```

```
## (Intercept)         cyl        disp          hp        drat          wt
##    20.09062     0.00000     0.00000     0.00000     0.00000     0.00000
##        qsec          vs          am        gear        carb
##     0.00000     0.00000     0.00000     0.00000     0.00000
```

```
lassoMod$lambda[100]
```

```
## [1] 0.01
```

```
coef(lassoMod)[,100]
```

```
##  (Intercept)          cyl         disp           hp         drat
## 13.045397106 -0.080511987  0.009608113 -0.019084736  0.814125259
##           wt         qsec           vs           am         gear
## -3.413559917  0.759037821  0.271406624  2.474100299  0.639686217
##         carb
## -0.302312344
```

```
predict(lassoMod, s = 50, type = "coefficients")[1:10,] # lambda = 50
```

```
## (Intercept)         cyl        disp          hp        drat          wt
##    20.09062     0.00000     0.00000     0.00000     0.00000     0.00000
##        qsec          vs          am        gear
##     0.00000     0.00000     0.00000     0.00000
```

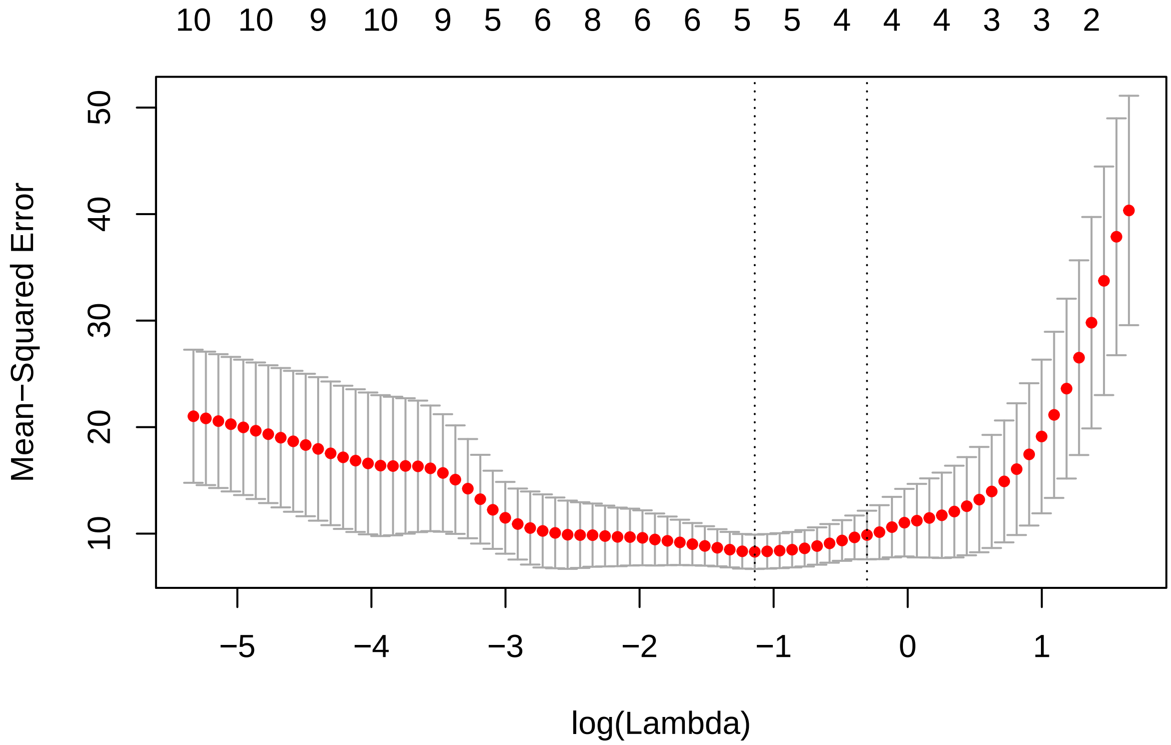
3. Best model selection

a. Create the data partition

```
set.seed(1)
train <- sample(1: nrow(x), nrow(x)/2)
test <- (-train)
yTest <- y[test]
```

b. Use cross validation to find the best value of lambda

```
cv.out <- cv.glmnet(x[train,],y[train], alpha=1, nfolds = 5)
plot(cv.out)
```



```
bestlam  <- cv.out$lambda.min
lasso.pred <- predict(cv.out ,s=bestlam ,newx=x[test,])
mean((lasso.pred -yTest)^2)
```

```
## [1] 13.24478
```