STAT 542 / CS 598: Homework 2

Pushpit Saxena (netId: pushpit2)

Contents

Loading the necessary libraries

```
library(mlbench)
library(glmnet)
```

```
## Loading required package: Matrix
```

```
## Loading required package: foreach
```

```
## Loaded glmnet 2.0-18
```

```
library(zoo)
```

```
##
## Attaching package: 'zoo'
```

```
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```r
library(magrittr)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```r
library(caret)
```

```
## Loading required package: lattice

## Loading required package: ggplot2
```

Question 1: Linear Model Selection

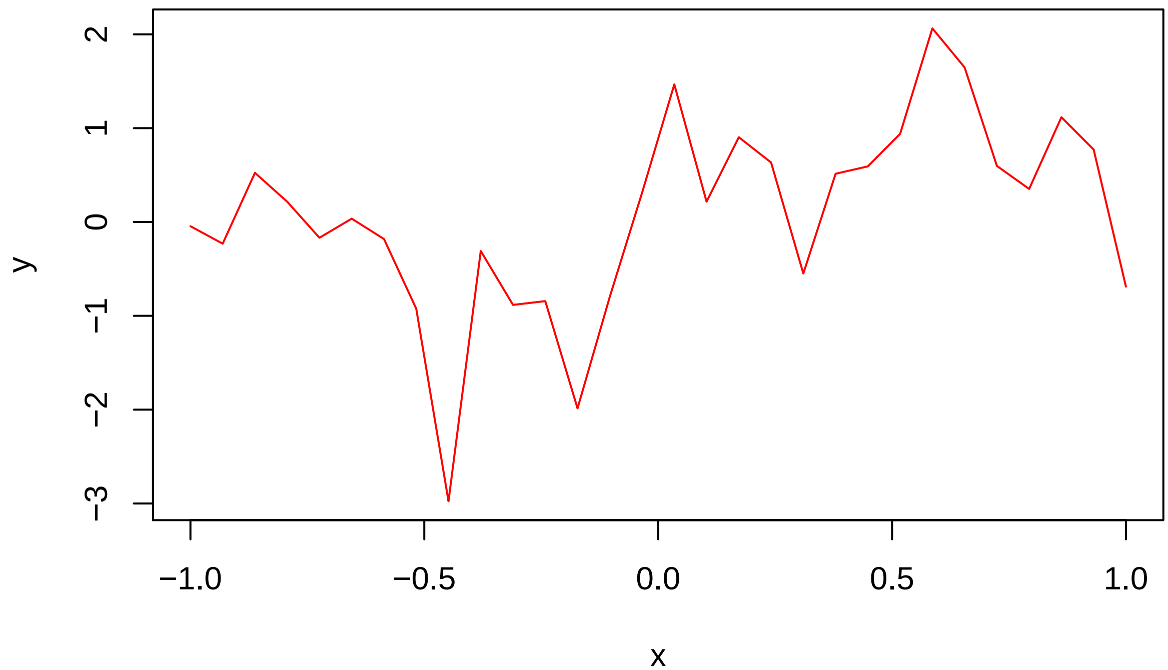
Prepare the Boston Housing Data

```r
data(BostonHousing2)
BH = BostonHousing2[, !(colnames(BostonHousing2) %in% c("medv", "town", "tract"))]
full.model <- lm(cmedv~., data = BH)
summary(full.model)
```

```
##
## Call:
## lm(formula = cmedv ~ ., data = BH)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -15.5831  -2.7643  -0.5994   1.7482  26.0822
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -4.350e+02  3.032e+02  -1.435 0.152029
## lon         -3.935e+00  3.372e+00  -1.167 0.243770
## lat          4.495e+00  3.669e+00   1.225 0.221055
## crim        -1.045e-01  3.261e-02  -3.206 0.001436 **
## zn           4.657e-02  1.374e-02   3.390 0.000755 ***
```

```
## indus        1.524e-02  6.175e-02   0.247 0.805106
## chas1        2.578e+00  8.650e-01   2.980 0.003024 **
## nox         -1.582e+01  4.005e+00  -3.951 8.93e-05 ***
## rm           3.754e+00  4.166e-01   9.011  < 2e-16 ***
## age          2.468e-03  1.335e-02   0.185 0.853440
## dis         -1.400e+00  2.088e-01  -6.704 5.61e-11 ***
## rad          3.067e-01  6.658e-02   4.607 5.23e-06 ***
## tax         -1.289e-02  3.727e-03  -3.458 0.000592 ***
## ptratio     -8.771e-01  1.363e-01  -6.436 2.92e-10 ***
## b            9.176e-03  2.663e-03   3.446 0.000618 ***
## lstat       -5.374e-01  5.042e-02 -10.660  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.7 on 490 degrees of freedom
## Multiple R-squared:  0.7458, Adjusted R-squared:  0.738
## F-statistic: 95.82 on 15 and 490 DF,  p-value: < 2.2e-16
```

```r
set.seed(1)
n=30
mean = 0
std = 1
x = sort(seq(from = -1, to = 1, length.out = n))
x_r = sort(runif(n, min=-1, max=1))
y = sin(pi * x) + rnorm(n)
y_r = sin(pi * x_r) + rnorm(n)
plot(x, y, col="red", type='l')
```



Dimension of boston housing data

```r
dim(BH)
```

```
## [1] 506  16
```

Question 1.a: Report the most significant variable from this full model with all features.

Answer:

To answer this question based on the full model fitted on unscaled data, I have taken a look at the P-value of each of the predictors and picked the predictor with the least *P-value*: **lstat**

```r
sort(summary(full.model)$coefficients[,4])[1]
```

```
##        lstat
## 5.27442e-24
```

Question 1.b: Starting from this full model, use stepwise regression with both forward and backward and BIC criterion to select the best model. Which variables are removed from the full model?

Answer:

This can be done using the step function. I have taken the sample size of 506 as n and passed log(n) as k parameter to step function. Also setup the direction as "both" to have forward and backward. The variables that are removed are: **lon, lat, indus, age**

```r
n = dim(BH)[1]
stepBIC = step(full.model, direction = "both", k=log(n), trace = 0)
```

Model selected based on stepwise regression starting at full model (direction=both, criterion=BIC):

```r
summary(stepBIC)
```

```
##
## Call:
## lm(formula = cmedv ~ crim + zn + chas + nox + rm + dis + rad +
##     tax + ptratio + b + lstat, data = BH)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -15.566  -2.686  -0.552   1.790  26.167
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  36.244827   5.022209   7.217 2.02e-12 ***
## crim         -0.106657   0.032487  -3.283 0.001099 **
## zn            0.047099   0.013402   3.514 0.000481 ***
## chas1         2.727209   0.846606   3.221 0.001360 **
## nox         -17.316823   3.503652  -4.943 1.06e-06 ***
## rm            3.778662   0.402685   9.384  < 2e-16 ***
## dis          -1.520270   0.184071  -8.259 1.35e-15 ***
## rad           0.296555   0.062836   4.720 3.08e-06 ***
## tax          -0.012077   0.003342  -3.613 0.000333 ***
## ptratio      -0.917035   0.127912  -7.169 2.77e-12 ***
## b             0.009202   0.002650   3.473 0.000561 ***
## lstat        -0.528441   0.047001 -11.243  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.694 on 494 degrees of freedom
## Multiple R-squared:  0.7444, Adjusted R-squared:  0.7387
## F-statistic: 130.8 on 11 and 494 DF,  p-value: < 2.2e-16
```

Comparison with the full model:

```r
anova(full.model, stepBIC)
```

```
## Analysis of Variance Table
```

```
##
## Model 1: cmedv ~ lon + lat + crim + zn + indus + chas + nox + rm + age +
##     dis + rad + tax + ptratio + b + lstat
## Model 2: cmedv ~ crim + zn + chas + nox + rm + dis + rad + tax + ptratio +
##     b + lstat
##   Res.Df   RSS Df Sum of Sq      F Pr(>F)
## 1    490 10825
## 2    494 10884 -4    -59.22 0.6702 0.6129
```

Question 1.c: Starting from this full model, use the best subset selection and list the best model of each model size.

Answer:

I have used regsubsets function from leaps library to find the best model of each model size. I have used the which attribute of the summary of regsubsets object to list the best model of each model size. The matrix shown in the output shows the best model for each model size (Please note that True indicate the predictor is included in the model)

```r
p = 15
library(leaps)
b = regsubsets(cmedv ~ ., data = BH, nvmax = p, nbest = 1)
rs = summary(b, matrix = T)
# Listing the best model of each model size.
rs$which
```

```
##    (Intercept)   lon   lat  crim    zn indus chas1   nox    rm   age   dis
## 1         TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## 2         TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE
## 3         TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE
## 4         TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE FALSE  TRUE
## 5         TRUE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE  TRUE FALSE  TRUE
## 6         TRUE FALSE FALSE FALSE FALSE FALSE  TRUE  TRUE  TRUE FALSE  TRUE
## 7         TRUE FALSE FALSE FALSE FALSE FALSE  TRUE  TRUE  TRUE FALSE  TRUE
## 8         TRUE FALSE FALSE FALSE  TRUE FALSE  TRUE  TRUE  TRUE FALSE  TRUE
## 9         TRUE FALSE FALSE FALSE FALSE FALSE  TRUE  TRUE  TRUE FALSE  TRUE
## 10        TRUE FALSE FALSE  TRUE  TRUE FALSE FALSE  TRUE  TRUE FALSE  TRUE
## 11        TRUE FALSE FALSE  TRUE  TRUE FALSE  TRUE  TRUE  TRUE FALSE  TRUE
## 12        TRUE FALSE  TRUE  TRUE  TRUE FALSE  TRUE  TRUE  TRUE FALSE  TRUE
## 13        TRUE  TRUE  TRUE  TRUE  TRUE FALSE  TRUE  TRUE  TRUE FALSE  TRUE
## 14        TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE FALSE  TRUE
## 15        TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
##      rad   tax ptratio     b lstat
## 1  FALSE FALSE   FALSE FALSE  TRUE
## 2  FALSE FALSE   FALSE FALSE  TRUE
## 3  FALSE FALSE    TRUE FALSE  TRUE
## 4  FALSE FALSE    TRUE FALSE  TRUE
## 5  FALSE FALSE    TRUE FALSE  TRUE
## 6  FALSE FALSE    TRUE FALSE  TRUE
## 7  FALSE FALSE    TRUE  TRUE  TRUE
## 8  FALSE FALSE    TRUE  TRUE  TRUE
## 9   TRUE  TRUE    TRUE  TRUE  TRUE
## 10  TRUE  TRUE    TRUE  TRUE  TRUE
```

```
## 11  TRUE  TRUE    TRUE  TRUE  TRUE
## 12  TRUE  TRUE    TRUE  TRUE  TRUE
## 13  TRUE  TRUE    TRUE  TRUE  TRUE
## 14  TRUE  TRUE    TRUE  TRUE  TRUE
## 15  TRUE  TRUE    TRUE  TRUE  TRUE
```

Coefficients vector for each of these models can be obtained as (the following code can be uncommented to see the coefficient information, I have commented it out just to keep the output in the pdf file manageable):

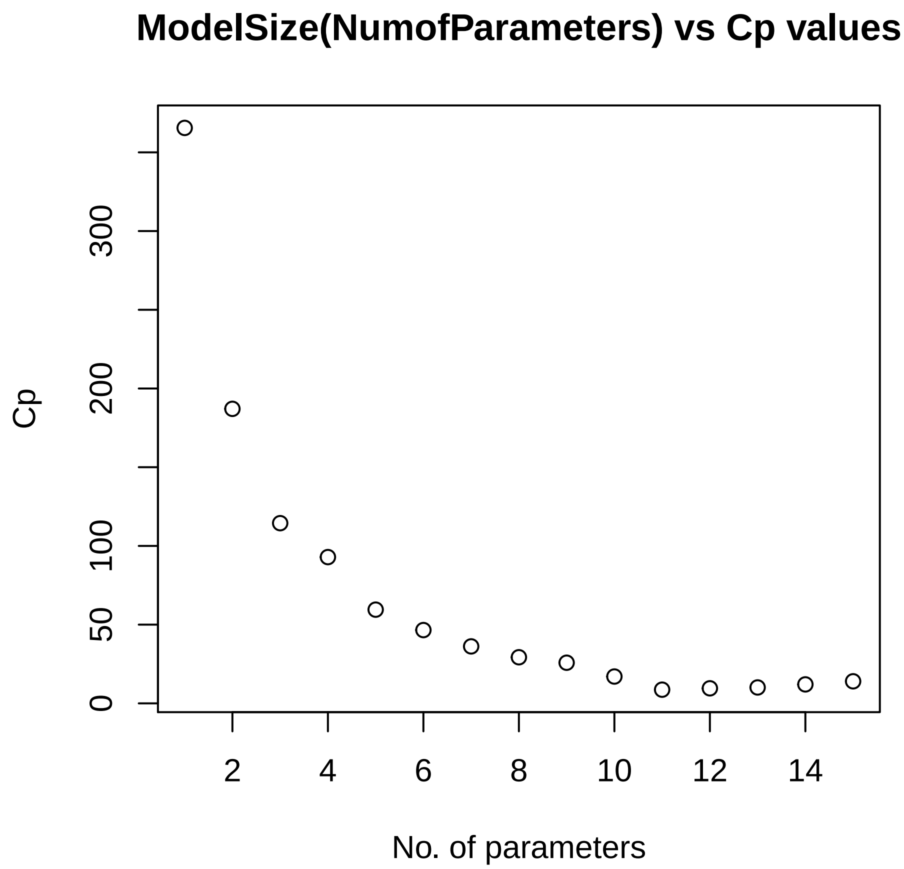
```
#coef(rs$obj, 1:15)
```

Question 1.d: Use the Cp criterion to select the best model from part c). Which variables are removed from the full model? What is the most significant variable?

Answer:

As apparant from the plot below the best model as per the CP criterion is model with 11 predictors. Based on the models from part c; in the best model with 11 predcitors, the predictors that are dropped are: **lon, lat, indus, age**. The most significant variable (based on **P-value** similar to part a) is **lstat**

```
#msize = apply(rs$which, 1, sum)
msize = 1:15
#par(mfrow=c(1,2))
CP = rs$rss/(summary(full.model)$sigma^2) + 2 * msize - n
AIC = n*log(rs$rss/n) + 2*msize;
BIC = n*log(rs$rss/n) + msize*log(n);
#cbind(CP, rs$cp)
plot(msize, CP, xlab="No. of parameters", ylab="Cp", main="ModelSize(NumofParameters) vs Cp values")
```




```r
cat("Number of parameters in the best model as per Cp criterion is ", which.min(CP))
```

```
## Number of parameters in the best model as per Cp criterion is  11
```

The most significant predictor for best model with 11 predictors (based on **P-value** similar to part a) is : **lstat**

```r
lm_11_pred_model <- lm(cmedv ~ . -lon -lat -age -indus, data=BH)
sort(summary(lm_11_pred_model)$coefficients[,4])[1]
```

```
##        lstat 
## 2.855042e-26
```

Question 2: Code your own Lasso

Data Preparation

```r
library(MASS)
```

```
##
## Attaching package: 'MASS'

## The following object is masked from 'package:dplyr':
##
##     select
```

```r
set.seed(1)
n = 200
p = 200

# generate data
V = matrix(0.2, p, p)
diag(V) = 1
X = as.matrix(mvrnorm(n, mu = rep(0, p), Sigma = V))
y = X[, 1] + 0.5*X[, 2] + 0.25*X[, 3] + rnorm(n)

# we will use a scaled version
X = scale(X)
Y = scale(y)
```

Question 2.a: Hence, we need first to write a function that updates just one parameter, which is also known as the soft-thresholding function. Construct the function in the form of soft_th <- function(b, lambda), where b is a number that represents the one-dimensional linear regression solution, and lambda is the penalty level. The function should output a scaler, which is the minimizer of

$$(\mathbf{X} - b)^2 + \lambda |b|$$

Answer:

Followed the mathematical derivation for soft-thresholding operator shown in the lectures slides and implemented it as below:

```r
soft_th <- function(b, lambda) {
  if (b > lambda/2) {
    return (b - (lambda / 2))
  } else if (abs(b) <= lambda/2) {
    return (0)
  } else if (b < -lambda/2) {
    return (b + (lambda/2))
  }
}
```

Question 2.b: Single iteration

Answer:

I have implemented a single loop of coordinate descent algorithm updating all the parameters one by one. The function is named **coord_descent** . This function will be used later in full implementation of mylasso function.

```
coord_descent <- function(X, Y, b, r, lambda) {
  for(j in 1:p) {

       # calculating partial residuals
       r <- r + X[,j]*b[j]

       # updating beta and soft-thresholding
       xr <- sum(X[,j]*r)
       xx <- sum(X[,j]^2)
       b[j] <- xr/xx
       b[j] <- soft_th(b[j], lambda = lambda)

       # Re calculting residual (Gauss-Seidel style coordinate descent)
       # r <- Y - X%*%b
       r <- r - X[,j]*b[j]
       # print(b)
  }
  return(list("b" = b, "r" = r))
}
lambda = 0.7
b = rep(0, p)
r = Y - X%*%b
obj = coord_descent(X, Y, b, r, lambda)
cat("First 3 observations in r after single loop:\n",
    paste(obj$r[1:3], collapse="\n "))
```

```
## First 3 observations in r after single loop:
##  -0.0760433820215123
##  0.146774031079523
##  0.156256770280221
```

```
cat("\nNonzero entries in the updated beta_new vector:\n",
    paste(obj$b[which(obj$b != 0)], collapse="\n "))
```

```
##
## Nonzero entries in the updated beta_new vector:
##  0.352963431429547
##  0.0902926012878801
```

Question 2.c: My own implementation of lasso

Answer:

```r
mylasso <- function(X, Y, lambda, tol, maxitr) {
  p = dim(X)[2]
  b <- rep(0, p)
  r <- Y - X%*%b
  final_itr = maxitr
  for (itr in 1:maxitr) {
    b_old = b
    obj = coord_descent(X, Y, b, r, lambda)
    b = obj$b
    r = obj$r
    l1norm <- dist(rbind(b, b_old), method="manhattan")
    if (l1norm < tol) {
      #print(sprintf("Final iteration: %d, Final l1 distance: %f", itr, l1norm))
      final_itr = itr
      break
    }
    #print(sprintf("Iteration: %d, tol: %f, l1 distance: %f", itr, tol, l1norm))
  }

  # print(r[1:3])
  # print(b[which(b != 0)])
  return (list("final_itr" = final_itr, "b" = b, "r" = r))
}
```

Running the method with lambda = 0.3, tol = 1e-5 and maxItr = 100

```r
lassoObj <- mylasso(X, Y, 0.3, 1e-5, 100)
```

i) The number of iterations took:

```r
lassoObj$final_itr
```

```
## [1] 9
```

ii) The nonzero entries in the final beta parameter estimate:

```r
lassoObj$b[which(lassoObj$b != 0)]
```

```
## [1] 0.457802236 0.226116017 0.114399954 0.001018992 0.011551407 0.004669249
```

iii) The first three observations of the residual:

```r
lassoObj$r[1:3]
```

```
## [1] -0.1757378  0.2262848  0.1912103
```

Question 2.d: Comparison with glmnet

Answer:

I have used the glmnet to do the lasso regression (setting alpha = 1) with lambda = 0.3 / 2 . The accuracy of our own implementation is almost similar to the one we get from glmnet. I have calculated the MSE for mylasso and glmnet model (shown below). Also, the beta-vectors differ by less than 0.005.

```r
# setting the glmnet lambda to be at half i.e. 0.15

mse_mylasso <- mean((Y - X%*%lassoObj$b)^2)
glmnetlasso <- glmnet(X, Y, alpha=1, lambda = 0.15)
mse_gl <- mean((Y -glmnetlasso %>% predict(X) %>% as.vector())^2)
l1norm2 <- dist(rbind(lassoObj$b, glmnetlasso$beta[,1]), method="manhattan")
cat("MSE (mylasso): ", mse_mylasso, "MSE(glmnet): ", mse_gl, "
    Diff between MSE: ", abs(mse_mylasso - mse_gl))
```

```
## MSE (mylasso):  0.3758599 MSE(glmnet):  0.3761872
##       Diff between MSE:  0.0003273428
```

```r
if (l1norm2 < 0.005) {
  cat("The distance between beta vector generated by mylasso
      and glmnet (less the 0.005) is: ", l1norm2)
}
```

```
## The distance between beta vector generated by mylasso
##         and glmnet (less the 0.005) is:  0.001095256
```