CS574 HW1

Kanak Choudhury

2/13/2020

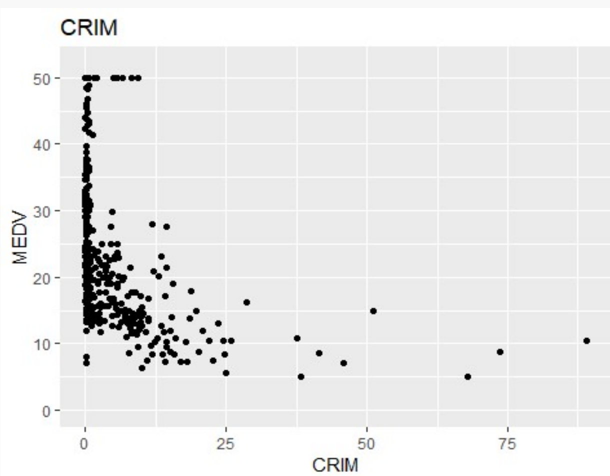
```
library(glmnet)

library(ggplot2)
```

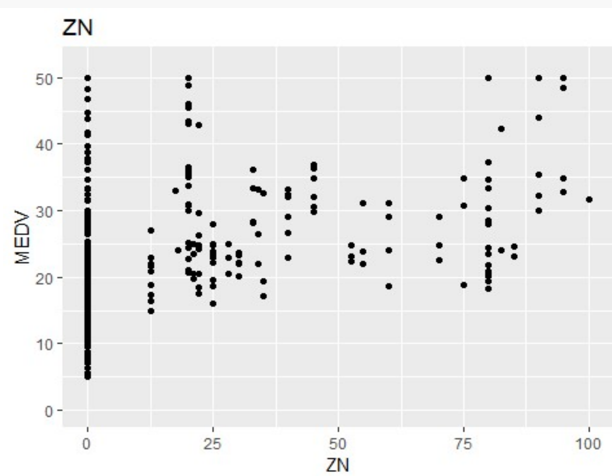
Problem 2

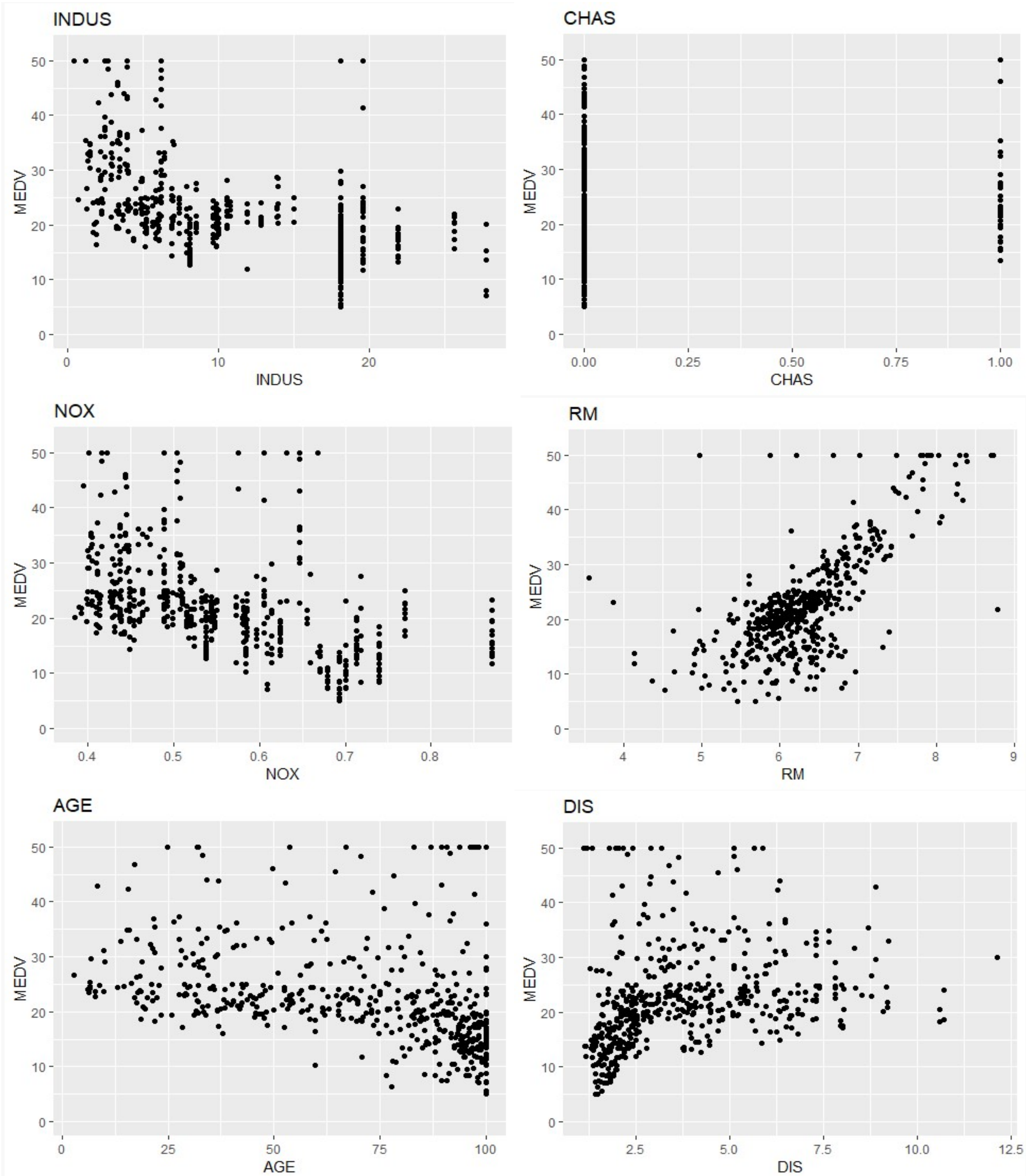
```
path = "D:\\ISU\\COMS 574 - Introduction to Machine Learning\\HW\\HW1\\"

dt = read.csv(paste(path, "housingdata.csv", sep = ""), header = T)
varname = names(dt)
```

(a)

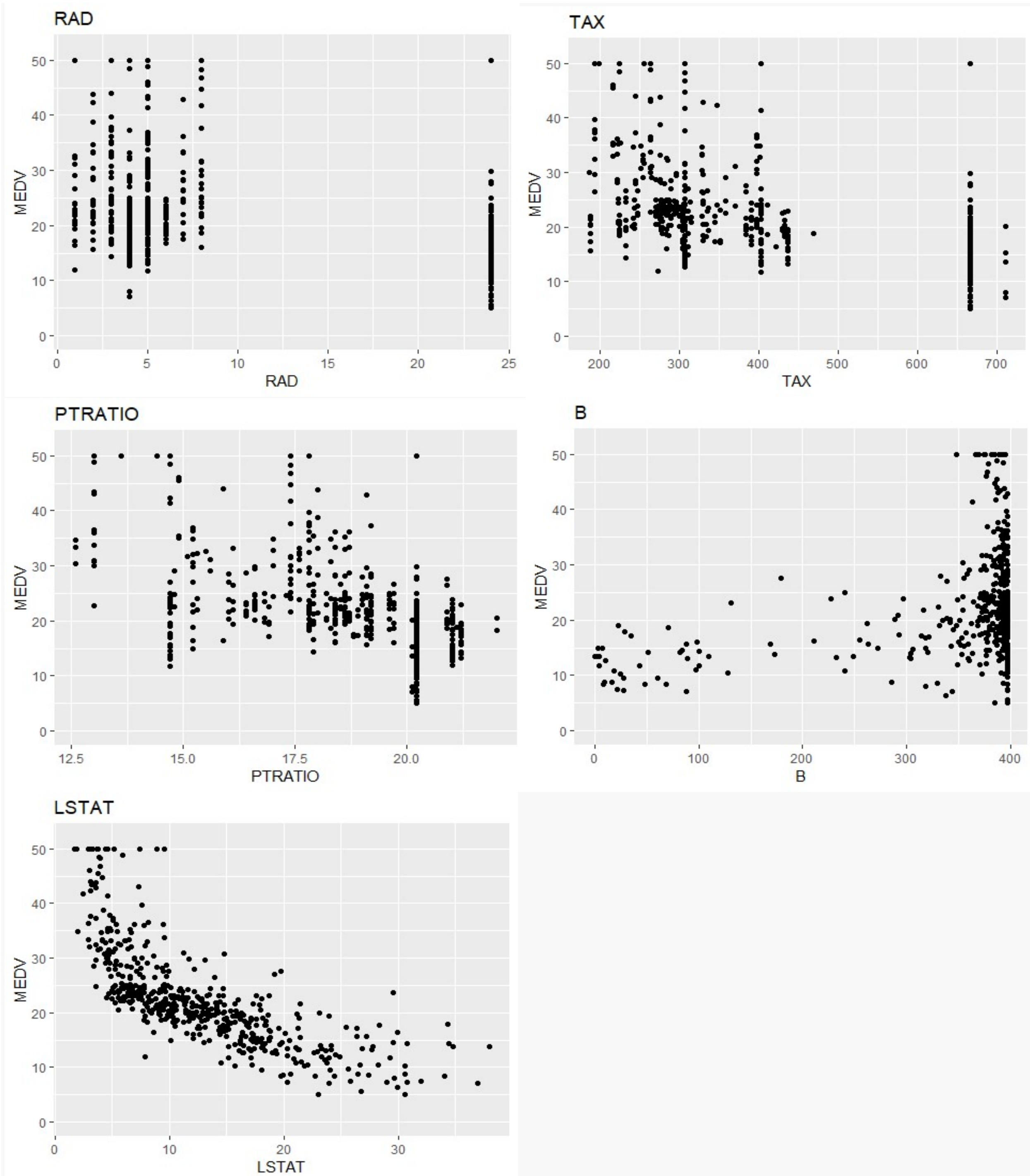
```
for (i in c(1:13)){
 print(ggplot(dt, aes_string(x=varname[i], y=varname[14])) +
         ylim(0,max(dt$MEDV+2))+
         ggtitle(varname[i]) +
         geom_point())
}
```







INDUS
CHAS
NOX
RM
AGE
DIS
MEDV
0
10
20
30
40
50
INDUS
0.00
0.25
0.50
0.75
1.00
CHAS
0.4
0.5
0.6
0.7
0.8
NOX
4
5
6
7
8
9
RM
0
25
50
75
100
AGE
2.5
5.0
7.5
10.0
12.5
DIS



Based on scatter plot, LSTAT (% lower status of the population) (negative relation) and RM (average number of rooms per dwelling) (positive relation) have moderate (highest among the features) linear association (-0.74 and 0.69) with MEDV (Median value of owner-occupied homes in \$1000's). On the other hand, B ($1000(Bk - 0.63)^2$ where Bk is the proportion of black residents by town) has the least relation with the response variable.

Though, CHAS is a binary variable, it seems that there is a high relation with response variable. It is clear from the scatter plot that there is nonlinear or quadratic relation between DIS (weighted distances to five Boston employment centres) and MEDV. All other variables have some relation with response variable. However, with respect to high order space (interaction with other variables or higher order) there might have some strong relation with the response variable.

(b)

```
dt_train = dt[1:400,]
dt_test = dt[401:dim(dt)[1],]
varn = c("AGE", "INDUS", "NOX", "RM", "TAX")

reg_res = list()
tr_mse = list()
k = 1
for (i in c(0:length(varn))){
  comb = combn(varn, i)
  for (j in c(1:dim(comb)[2])){
    if (dim(comb)[1]==0){
      formu = "MEDV ~ 1"
      comb = matrix(c("Intercept"), nrow = 1, ncol = 1)
    } else{
      formu = paste("MEDV ~ 1", paste((comb[,j]), collapse = "+"), sep = "+")
    }

    reg_res[[k]] = lm(as.formula(formu), data = dt_train)
    tr_mse[[k]] = list(i = i, var = paste((comb[,j]), collapse = ","),
                       tr_mse = mean(reg_res[[k]]$residuals^2),
                       vel_mse = mean((predict(reg_res[[k]], newdata = dt_tes
t)-dt_test$MEDV)^2))
    k = k+1
  }

}

res = do.call(rbind.data.frame, tr_mse)
names(res) = c("subset", "variables", "tr_mse", "ts_mse")

trset = NULL
for (i in c(0:length(varn))){
  aa = which(res$tr_mse == res[res$subset == i,][which.min(res[res$subset ==
i,3]),3])
  trset = rbind(trset, res[aa,])
  res1 = reg_res[[aa]]
  len1 = length(res1$coefficients)
```

```r
  if (i==0){
    print(paste("For subset model ", i, ":   y^hat = ", round(res1$coefficien
ts[1], 3) ,
                "   with MSE = ", round(res[aa,3],3)))
  } else {

    print(paste("For subset model ", i, ":   y^hat = ", round(res1$coefficien
ts[1], 3), " + " ,
              paste(round(res1$coefficients[2:len1], 3),
                    names(res1$coefficients)[2:len1], sep = " * ", collapse =
" + "),
            "   with training MSE = ", round(res[aa, 3],3)))
  }
}

print(trset)

aa = which.min(res$tr_mse)
res1 = reg_res[[aa]]
len1 = length(res1$coefficients)
print(paste("For subset model ", res[aa,]$subset, ":   y^hat = ", round(res1$
coefficients[1], 3),
            " + " , paste(round(res1$coefficients[2:len1], 3),
                  names(res1$coefficients)[2:len1], sep = " * ", collapse = "
+ "),
        "   with training MSE = ", round(res[aa, 3],3), " and testing MSE = ", ro
und(res[aa, 4],3)))

aa = which.min(res$ts_mse)
res1 = reg_res[[aa]]
len1 = length(res1$coefficients)
print(paste("For subset model ", res[aa,]$subset, ":   y^hat = ", round(res1$
coefficients[1], 3),
            " + " ,
            paste(round(res1$coefficients[2:len1], 3),
                  names(res1$coefficients)[2:len1], sep = " * ", collapse = "
+ "),
        "   with training MSE = ", round(res[aa, 3],3), " and testing MSE = ", ro
und(res[aa, 4],3)))
```

```
## [1] "For subset model  0 :   y^hat =  24.334    with MSE =  83.807"
## [1] "For subset model  1 :   y^hat =  -35.261  +  9.406 * RM    with train
ing MSE =  36.303"
## [1] "For subset model  2 :   y^hat =  -31.076  +  -0.034 * AGE + 9.09 * RM
with training MSE =  35.381"
## [1] "For subset model  3 :   y^hat =  -29.222  +  -0.028 * AGE + 8.944 * R
M + -0.004 * TAX    with training MSE =  35.193"
## [1] "For subset model  4 :   y^hat =  -30.586  +  -0.036 * AGE + 3.747 * N
OX + 8.991 * RM + -0.005 * TAX    with training MSE =  35.125"
```

```
## [1] "For subset model  5 :   y^hat =  -30.628  +  -0.034 * AGE + -0.031 *
INDUS + 4.573 * NOX + 8.947 * RM + -0.005 * TAX    with training MSE =  35.11
"
```

```
##    subset            variables   tr_mse    ts_mse
## 2       0            Intercept 83.80701 102.22659
## 5       1                   RM 36.30331  79.62583
## 9       2               AGE,RM 35.38101  67.97003
## 22      3           AGE,RM,TAX 35.19273  57.58572
## 30      4       AGE,NOX,RM,TAX 35.12503  57.85750
## 32      5 AGE,INDUS,NOX,RM,TAX 35.11013  57.66880
```

```
## [1] "For subset model  5 :   y^hat =  -30.628  +  -0.034 * AGE + -0.031 *
INDUS + 4.573 * NOX + 8.947 * RM + -0.005 * TAX    with training MSE =  35.11
and testing MSE =  57.669"
```

```
## [1] "For subset model  3 :   y^hat =  35.763  +  -0.04 * AGE + -7.286 * NO
X + -0.014 * TAX    with training MSE =  73.333  and testing MSE =  30.002"
```

The best fitting linear model for every subset of AGE, INDUS, NOX, RM, TAX using the first n = 400 samples based on training MSE is the model with all five features. However, based on test MSE, the based model is the model with variables AGE, NOX and TAX.

(c)

(i)

```r
valset = NULL
for (i in c(0:length(varn))){
  aa = which(res$ts_mse == res[res$subset == i,][which.min(res[res$subset ==
i,4]),4])
  valset = rbind(valset, res[aa,])
  res1 = reg_res[[aa]]
  len1 = length(res1$coefficients)
  if (i==0){
    print(paste("For subset model ", i, ":   y^hat = ", round(res1$coefficien
ts[1], 3) ,
                "   with MSE = ", round(res[aa,4],3)))
  } else {
```

```r
    print(paste("For subset model ", i, ":   y^hat = ", round(res1$coefficien
ts[1], 3), " + " ,
                paste(round(res1$coefficients[2:len1], 3),
                      names(res1$coefficients)[2:len1], sep = " * ", collapse
= " + "),
                "   with validation MSE = ", round(res[aa,4],3)))
  }
}
```

```
## [1] "For subset model  0 :   y^hat =  24.334    with MSE =  102.227"
## [1] "For subset model  1 :   y^hat =  32.103  +  -0.022 * TAX    with vali
dation MSE =  31.616"
## [1] "For subset model  2 :   y^hat =  33.753  +  -0.056 * AGE + -0.017 * T
AX    with validation MSE =  30.207"
## [1] "For subset model  3 :   y^hat =  35.763  +  -0.04 * AGE + -7.286 * NO
X + -0.014 * TAX    with validation MSE =  30.002"
## [1] "For subset model  4 :   y^hat =  31.131  +  -0.026 * AGE + -0.388 * I
NDUS + 3.887 * NOX + -0.01 * TAX    with validation MSE =  31.431"
## [1] "For subset model  5 :   y^hat =  -30.628  +  -0.034 * AGE + -0.031 *
INDUS + 4.573 * NOX + 8.947 * RM + -0.005 * TAX    with validation MSE =  57.
669"
```

```r
print(valset)
```

```
##    subset            variables   tr_mse    ts_mse
## 2       0            Intercept 83.80701 102.22659
## 6       1                  TAX 75.69373  31.61631
## 10      2              AGE,TAX 73.59263  30.20671
## 21      3          AGE,NOX,TAX 73.33258  30.00196
## 28      4    AGE,INDUS,NOX,TAX 70.78130  31.43145
## 32      5 AGE,INDUS,NOX,RM,TAX 35.11013  57.66880
```

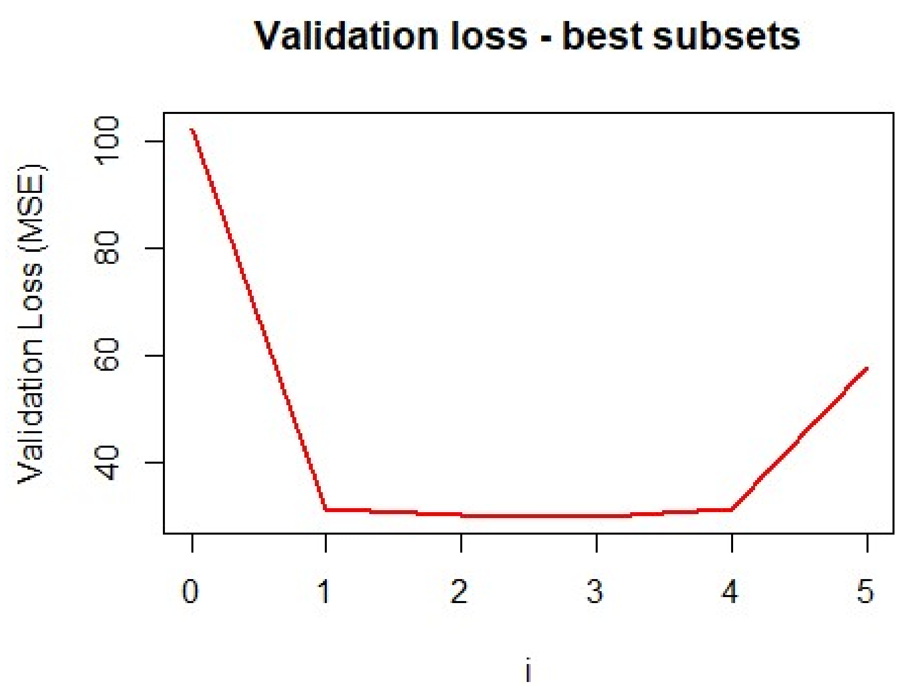
The best subset was for each i is given in the above table.

These six models are completely nested with largest order model. If we consider zero for the appropriate parameter(s), we can get any lower order model.

(ii)

```r
plot(trset$subset, trset$tr_mse, type = "l", xlab = "i", lwd = 2, col = 2,
     ylab = "Training Loss (MSE)")
title("Training loss - best subsets")
```


```
plot(valset$subset, valset$ts_mse, type = "l", xlab = "i", lwd = 2, col = 2,
     ylab = "Validation Loss (MSE)")
title("Validation loss - best subsets")
```

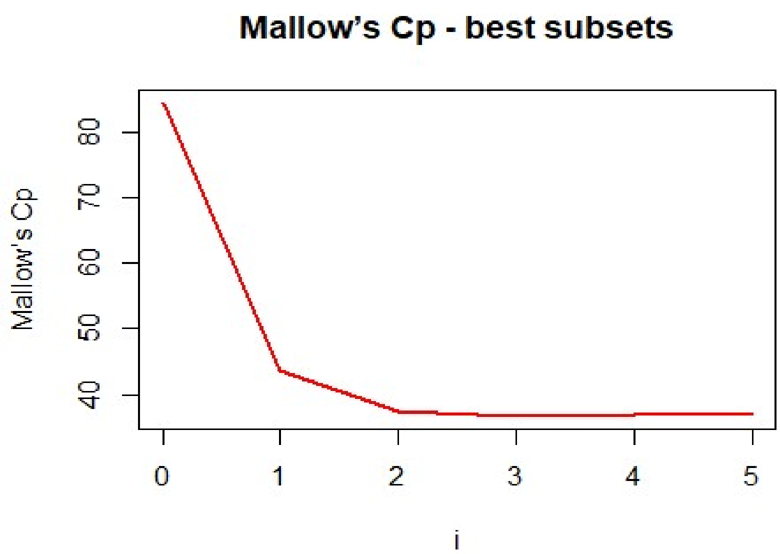


As the number of features increases in the model, the training MSE decreases. However, based on the validation loss, it looks like "U" shaped. That indicates that more complex model is not always better for prediction.

(iii)

```
reg_res_cp = list()
tr_mse_cp = list()
k = 1
for (i in c(0:length(varn))){
  comb = combn(varn, i)
  for (j in c(1:dim(comb)[2])){
    if (dim(comb)[1]==0){
      formu = "MEDV ~ 1"
      comb = matrix(c("Intercept"), nrow = 1, ncol = 1)
    } else{
      formu = paste("MEDV ~ 1", paste((comb[,j]), collapse = "+"), sep = "+")
    }

    reg_res_cp[[k]] = lm(as.formula(formu), data = dt)
    trmse = mean(reg_res_cp[[k]]$residuals^2)
    tr_mse_cp[[k]] = list(i = i, var = paste((comb[,j]), collapse = ","),
                          tr_mse_cp = trmse)
    k = k+1
  }

}

res_cp = do.call(rbind.data.frame, tr_mse_cp)
names(res_cp) = c("subset", "variables", "tr_mse")
res_cp$cp = res_cp$tr_mse + 2*res_cp$subset*res_cp$tr_mse[nrow(res_cp)] / nro
w(dt)


trset_cp = NULL
for (i in c(0:length(varn))){
  aa = which(res_cp$cp == res_cp[res_cp$subset == i,][which.min(res_cp[res_cp
$subset == i,4]),4])
  trset_cp = rbind(trset_cp, res_cp[aa,])
  res_cp1 = reg_res_cp[[aa]]
  len1 = length(res_cp1$coefficients)
  if (i==0){
    print(paste("For subset model ", i, ":   y^hat = ", round(res_cp1$coeffic
ients[1], 3) ,
                "   with MSE = ", round(res_cp[aa,3],3)))
  } else {

    print(paste("For subset model ", i, ":   y^hat = ", round(res_cp1$coeffic
ients[1], 3), " + " ,
                paste(round(res_cp1$coefficients[2:len1], 3),
                      names(res_cp1$coefficients)[2:len1], sep = " * ", colla
pse = " + "),
```

```
                "   with Mallow’s Cp = ", round(res_cp[aa, 4],3)))
  }
}

aa = which.min(res_cp$cp)
res_cp1 = reg_res_cp[[aa]]
len1 = length(res_cp1$coefficients)
print(paste("For subset model ", res_cp$subset[aa], ":   y^hat = ", round(res
_cp1$coefficients[1], 3), " + " ,
            paste(round(res_cp1$coefficients[2:len1], 3),
                  names(res_cp1$coefficients)[2:len1], sep = " * ", collapse
= " + "),
            "   with Mallow’s Cp = ", round(res_cp[aa, 4],3)))

## [1] "For subset model  3 :   y^hat =  -18.955  +  -0.037 * AGE + 7.843 * R
M + -0.013 * TAX    with Mallow’s Cp =  36.73"

plot(trset_cp$subset, trset_cp$cp, type = "l", xlab = "i", lwd = 2, col = 2,
     ylab = "Mallow’s Cp")
title("Mallow’s Cp - best subsets")



## [1] "For subset model  0 :   y^hat =  22.533    with MSE =  84.42"
## [1] "For subset model  1 :   y^hat =  -34.671  +  9.102 * RM    with Mallo
w’s Cp =  43.744"
## [1] "For subset model  2 :   y^hat =  -21.233  +  7.993 * RM + -0.016 * TA
X    with Mallow’s Cp =  37.384"
## [1] "For subset model  3 :   y^hat =  -18.955  +  -0.037 * AGE + 7.843 * R
M + -0.013 * TAX    with Mallow’s Cp =  36.73"
## [1] "For subset model  4 :   y^hat =  -18.803  +  -0.035 * AGE + -0.022 *
INDUS + 7.808 * RM + -0.012 * TAX    with Mallow’s Cp =  36.866"
## [1] "For subset model  5 :   y^hat =  -18.497  +  -0.034 * AGE + -0.016 *
INDUS + -0.942 * NOX + 7.807 * RM + -0.012 * TAX    with Mallow’s Cp =  37.00
5"
```



The best fitting linear model for every subset of AGE, INDUS, NOX, RM, TAX using all the samples based on C_p is the model with variables AGE, RM and TAX.

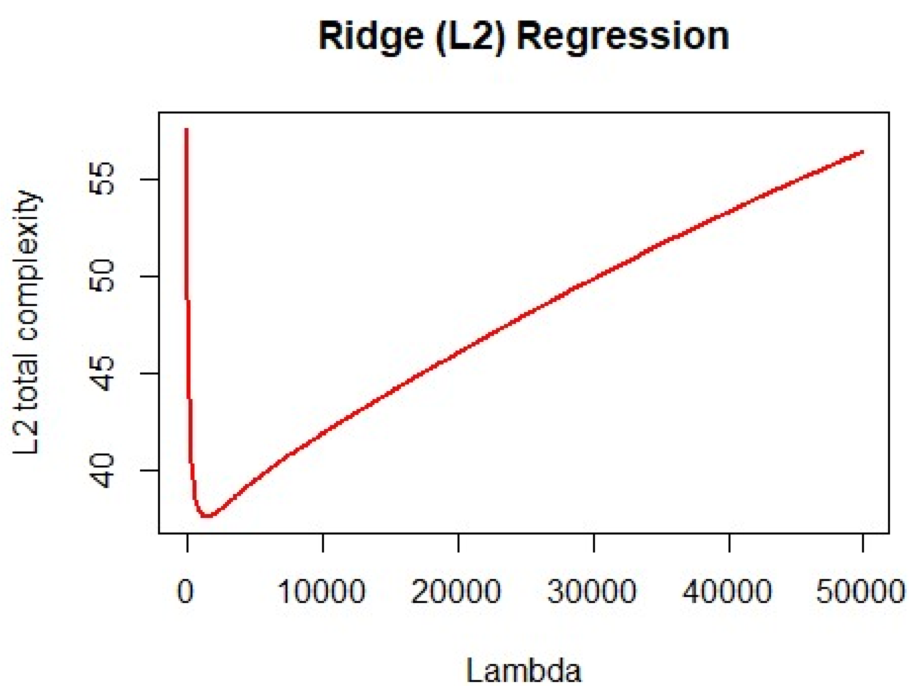
Plot using total complexity C_p and plot using validation set MSE show the same pattern. C_p gives more weight to the model with higher complexity. However, That does not mean C_p and validation set MSE behave the same way. It actually depends on data. For some data set, these two methods can provide the same result. However, if prediction is the main purpose of study, model section based on validation set total loss would be better.

(iv)

(i) - (iii)

```
dt_train = dt[1:400,]
dt_test = dt[401:dim(dt)[1],]
varn = c("AGE", "INDUS", "NOX", "RM", "TAX")

lambda = 0
l2_res = NULL

loop = TRUE

while (loop) {
  a = glmnet(x = as.matrix(dt_train[,varn]), y = dt_train[,"MEDV"], standardi
ze=FALSE,
             alpha = 0, lambda = lambda)
  pred = predict(a, as.matrix(dt_test[,varn]))
```

```r
  l2 = mean((dt_test$MEDV - pred)^2) + lambda * sum(a$beta^2)
  l2_res = rbind(l2_res, c(a$lambda, l2))
  if (lambda > 1400 & lambda <1600){
    lambda = lambda + 0.05
  }else {
    lambda = lambda + 100
  }

  if(sum(a$beta^2) < 1e-5 | lambda > 50000) loop = FALSE
}

l2_res = as.data.frame(l2_res)
plot(l2_res$V1, l2_res$V2, type = "l", xlab = "Lambda", lwd = 2, col = 2,
     ylab = "L2 total complexity")
title("Ridge (L2) Regression")
```




```r
aa = which.min(l2_res$V2)
a = glmnet(x = as.matrix(dt_train[,varn]), y = dt_train[,"MEDV"], standardize
=FALSE,
             alpha = 0, lambda = l2_res[aa,1])
pred = predict(a, as.matrix(dt_test[,varn]))
l2 = mean((dt_test$MEDV - pred)^2)

print(paste("Model with lambda = ", round(l2_res[aa,1],3), ":   y^hat = ", ro
und(a$a0, 3),
            " + " ,
```

```
            paste(round(a$beta, 3),
                  row.names(a$beta), sep = " * ", collapse = " + "),
    "   with testing MSE = ", round(l2,3)))
```

```
## [1] "Model with lambda =  1503.3 :   y^hat =  33.047  +  -0.042 * AGE + -0
.045 * INDUS + 0 * NOX + 0.026 * RM + -0.017 * TAX    with testing MSE =  30.
591"
```

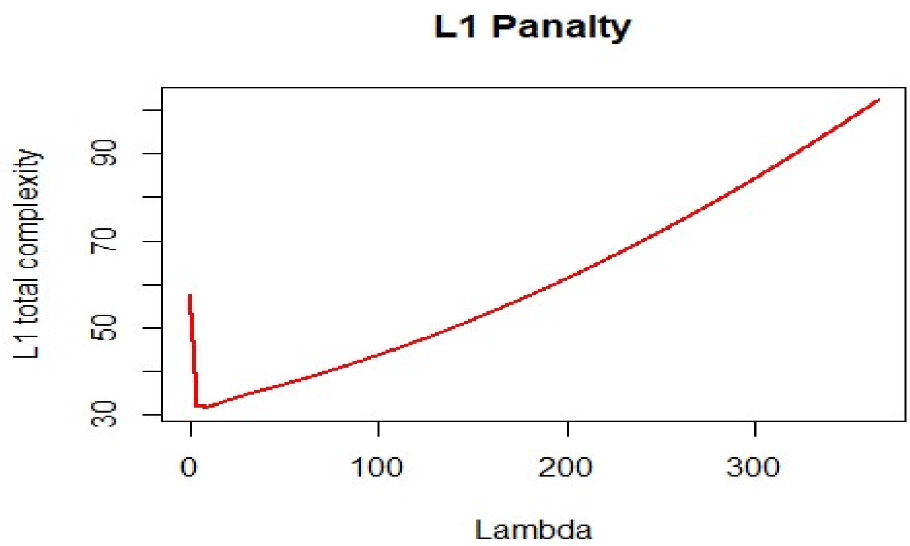
```
print(paste("Model with lambda = ", l2_res[aa,1], " has lowest validation MSE
, ", round(l2_res[aa,2],3)))
```

```
## [1] "Model with lambda =  1503.3  has lowest validation MSE,  37.628"
```

From this plot, we can see that model with $\lambda = 1503.3$ with AGE, INDUS, RM, TAX variables has the smallest validation set total complexity. However, using C_p and validation set MSE, we got smaller set model though both were different.

(iv)

```
lambda = 0
l1_res = NULL

loop = TRUE

while (loop) {
  a = glmnet(x = as.matrix(dt_train[,varn]), y = dt_train[,"MEDV"], standardi
ze=FALSE,
             alpha = 1, lambda = lambda)
  pred = predict(a, as.matrix(dt_test[,varn]))
  l1 = mean((dt_test$MEDV - pred)^2) + lambda * sum(abs(a$beta))
  l1_res = rbind(l1_res, c(a$lambda, l1))
  lambda = lambda + 0.05
  if(sum(abs(a$beta)) < 1e-8) loop = FALSE
}

l1_res = as.data.frame(l1_res)
plot(l1_res$V1, l1_res$V2, type = "l", xlab = "Lambda", lwd = 2, col = 2,
     ylab = "L1 total complexity")
title("L1 Panalty")
```




```
aa = which.min(l1_res$V2)
a = glmnet(x = as.matrix(dt_train[,varn]), y = dt_train[,"MEDV"], standardize
=FALSE,
             alpha = 1, lambda = l1_res[aa,1])
pred = predict(a, as.matrix(dt_test[,varn]))
mmse = mean((dt_test$MEDV - pred)^2)

print(paste("Model with lambda = ", round(l1_res[aa,1],3), ":   y^hat = ", ro
und(a$a0, 3),
            " + " ,
            paste(round(a$beta, 3),
                  row.names(a$beta), sep = " * ", collapse = " + "),
    "   with testing MSE = ", round(mmse,3)))

## [1] "Model with lambda =  8.6 :   y^hat =  33.229  +  -0.044 * AGE + -0.00
1 * INDUS + 0 * NOX + 0 * RM + -0.017 * TAX     with testing MSE =  31.106"

print(paste("Model with lambda = ", round(l1_res[aa,1], 3), " has lowest vali
dation MSE, ", round(mmse,3)))

## [1] "Model with lambda =  8.6  has lowest validation MSE,  31.106"
```

LASSO model also shows the same patter as Ridge regression. However, LASSO model selects smaller set variables that the Ridge regression. It includes only AGE, INDUS, TAX variables. It is important to mention that model selection based on different criteria selects different sets of variables in the respective best model.

(v)

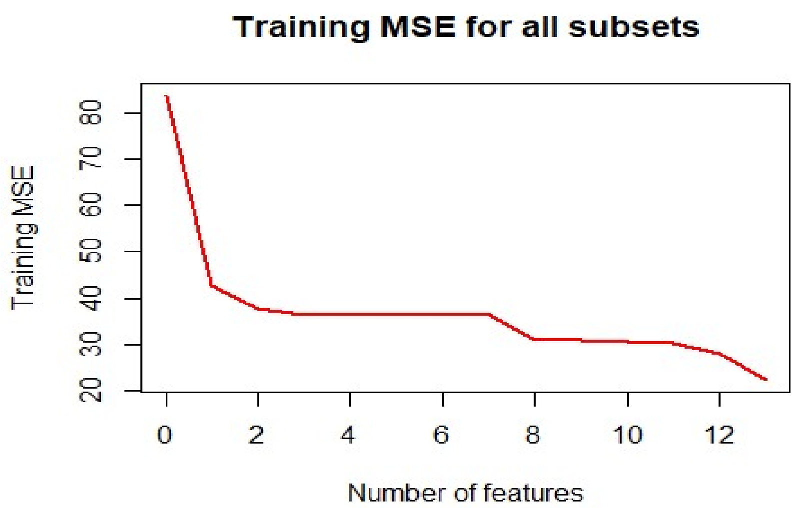
It is important to normalize all features when using L1 or L2 penalty for model estimation. Because, estimated parameters have a same unit as the corresponding variables and when we use these estimated parameters as the penalty factor, it will have high influence if the variable represents with very high values. As a result, prediction based on unnormalize features might be misleading.

(d)

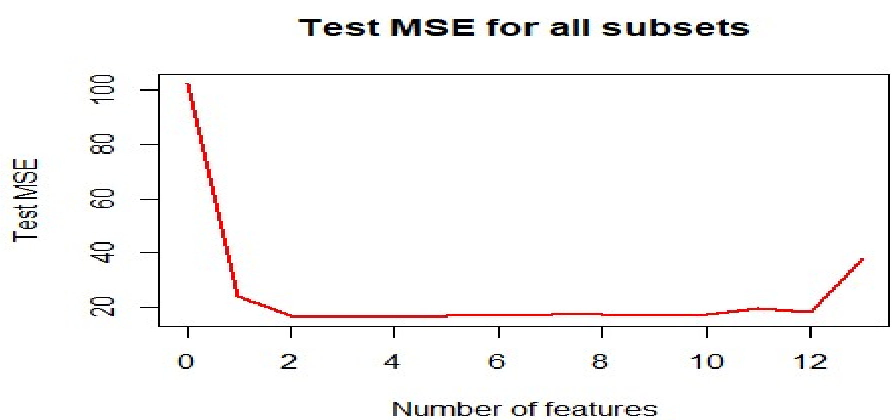
(i)

```
dt_train = dt[1:400,]
dt_test = dt[401:dim(dt)[1],]
varn = c("CRIM",    "ZN",      "INDUS",   "CHAS",    "NOX",     "RM",
         "AGE",     "DIS",     "RAD",     "TAX",     "PTRATIO", "B" ,      "L
STAT")


varinclude = c()
varexlude = varn
fi_res = list()
fi_reg = list()
k=2

formu = "MEDV ~ 1"
tem_reg = lm(as.formula(formu), data = dt_train)
temp_res = list(0, "intercept",
                     mean(tem_reg$residuals^2),
                     mean((predict(tem_reg, newdata = dt_test)-dt_test$MEDV)^
2),
                     formu)


fi_reg[[1]] = tem_reg
fi_res[[1]] = c(temp_res)


while (!is.null(varexlude) & length(varexlude)>0) {
  tem_reg = list()
  temp_res = list()
```

```
  for (i in c(1:length(varexlude))){

    if (is.null(varinclude)){
      formu = paste("MEDV ~ 1", varexlude[i], sep = "+")
    } else {
      formu = paste("MEDV ~ 1", paste(varinclude, collapse = "+"), varexlude[
i], sep = "+")
    }

    tem_reg[[i]] = lm(as.formula(formu), data = dt_train)
    temp_res[[i]] = list(length(varinclude)+1, varexlude[i],
                         mean(tem_reg[[i]]$residuals^2),
                         mean((predict(tem_reg[[i]], newdata = dt_test)-dt_test$MEDV)
^2),
                         formu)

  }

  temp_res = do.call(rbind.data.frame, temp_res)
  names(temp_res) = c("nvar", "var_include", "tr_mse", "ts_mse", "formula")
  temp_res$var_include = as.character(temp_res$var_include)
  temp_res$formula = as.character(temp_res$formula)
  varinclude = c(varinclude, as.character(temp_res[which.min(temp_res$ts_mse)
, 2]))
  varexlude = varexlude[!(varexlude %in% varinclude)]

  fi_reg[[k]] = tem_reg[[which.min(temp_res$ts_mse)]]
  fi_res[[k]] = c(temp_res[which.min(temp_res$ts_mse),])

  k=k+1
}

fi_res = do.call(rbind.data.frame, fi_res)
names(fi_res) = c("nvar", "var_include", "tr_mse", "ts_mse", "formula")
forward_res = fi_res

plot(fi_res$nvar, fi_res$tr_mse, type = "l", xlab = "Number of features", lwd
= 2, col = 2,
     ylab = "Training MSE")
title("Training MSE for all subsets")
```




```
plot(fi_res$nvar, fi_res$ts_mse, type = "l", xlab = "Number of features", lwd
= 2, col = 2,
     ylab = "Test MSE")
title("Test MSE for all subsets")
```




```
print(fi_res)

print(fi_res[which.min(fi_res$ts_mse),])

aa = which.min(fi_res$ts_mse)
res_131 = fi_reg[[aa]]
len1 = length(res_131$coefficients)
print(paste("Forward Best Subset Model :", fi_res[aa,]$formula, ":   y^hat =
", round(res_131$coefficients[1], 3), " + " ,
            paste(round(res_131$coefficients[2:len1], 3),
                  names(res_131$coefficients)[2:len1], sep = " * ", collapse
= " + "),
            "   with training MSE = ", round(fi_res[aa,3],3),
            " and validation MSE = ", round(fi_res[aa,4],3)))
```

```
##    nvar var_include   tr_mse    ts_mse
## 2     0   intercept 83.80701 102.22659
## 21    1       LSTAT 42.62158  23.92312
## 3     2     PTRATIO 37.65518  16.76650
## 4     3        CHAS 36.43344  16.55229
## 5     4          ZN 36.42852  16.61204
## 6     5       INDUS 36.41458  16.87186
## 7     6         TAX 36.38863  17.15157
## 8     7         NOX 36.37764  17.36296
## 9     8         DIS 30.84108  17.31292
## 10    9        CRIM 30.83273  16.81341
## 11   10         AGE 30.51697  17.53081
## 12   11           B 30.28257  19.59739
## 13   12         RAD 28.17532  18.49887
## 14   13          RM 22.30523  37.89378
##                                                                         formula
## 2                                                                      MEDV ~ 1
## 21                                                               MEDV ~ 1+LSTAT
## 3                                                        MEDV ~ 1+LSTAT+PTRATIO
## 4                                                   MEDV ~ 1+LSTAT+PTRATIO+CHAS
## 5                                                MEDV ~ 1+LSTAT+PTRATIO+CHAS+ZN
## 6                                          MEDV ~ 1+LSTAT+PTRATIO+CHAS+ZN+INDUS
## 7                                      MEDV ~ 1+LSTAT+PTRATIO+CHAS+ZN+INDUS+TAX
## 8                                  MEDV ~ 1+LSTAT+PTRATIO+CHAS+ZN+INDUS+TAX+NOX
## 9                              MEDV ~ 1+LSTAT+PTRATIO+CHAS+ZN+INDUS+TAX+NOX+DIS
## 10                        MEDV ~ 1+LSTAT+PTRATIO+CHAS+ZN+INDUS+TAX+NOX+DIS+CRIM
## 11                    MEDV ~ 1+LSTAT+PTRATIO+CHAS+ZN+INDUS+TAX+NOX+DIS+CRIM+AGE
## 12                  MEDV ~ 1+LSTAT+PTRATIO+CHAS+ZN+INDUS+TAX+NOX+DIS+CRIM+AGE+B
## 13          MEDV ~ 1+LSTAT+PTRATIO+CHAS+ZN+INDUS+TAX+NOX+DIS+CRIM+AGE+B+RAD
## 14 MEDV ~ 1+LSTAT+PTRATIO+CHAS+ZN+INDUS+TAX+NOX+DIS+CRIM+AGE+B+RAD+RM
```

```
##   nvar var_include   tr_mse   ts_mse                     formula
## 4    3        CHAS 36.43344 16.55229 MEDV ~ 1+LSTAT+PTRATIO+CHAS

## [1] "Forward Best Subset Model : MEDV ~ 1+LSTAT+PTRATIO+CHAS :   y^hat = 
51.786  +  -0.853 * LSTAT + -1.011 * PTRATIO + 3.923 * CHAS    with training 
MSE =  36.433  and validation MSE =  16.552"
```

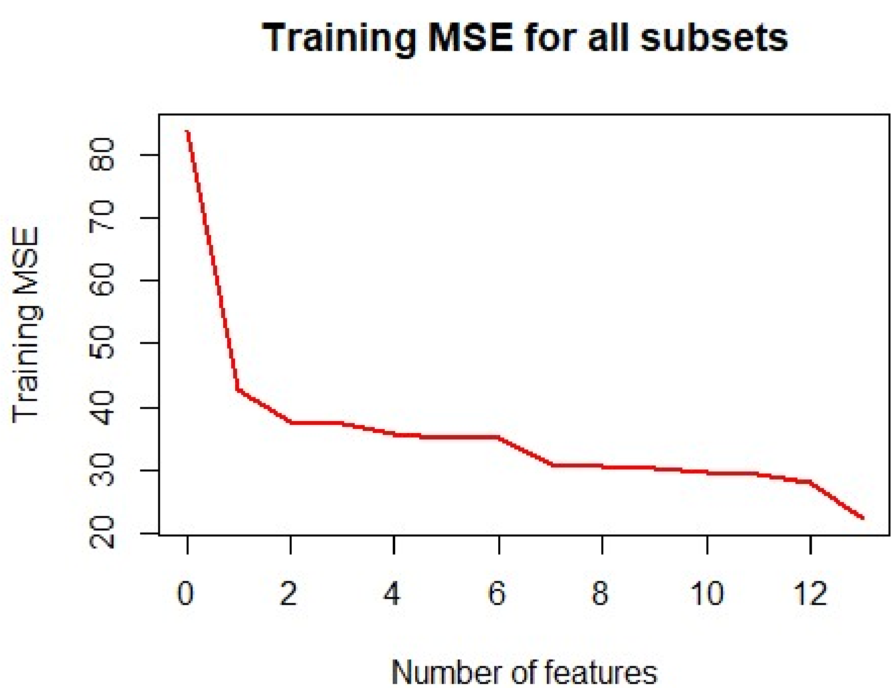
Using forward search, based on validation MSE, the best model includes three variables (LSTAT, PTRATIO, CHAS).

(d)

(ii)

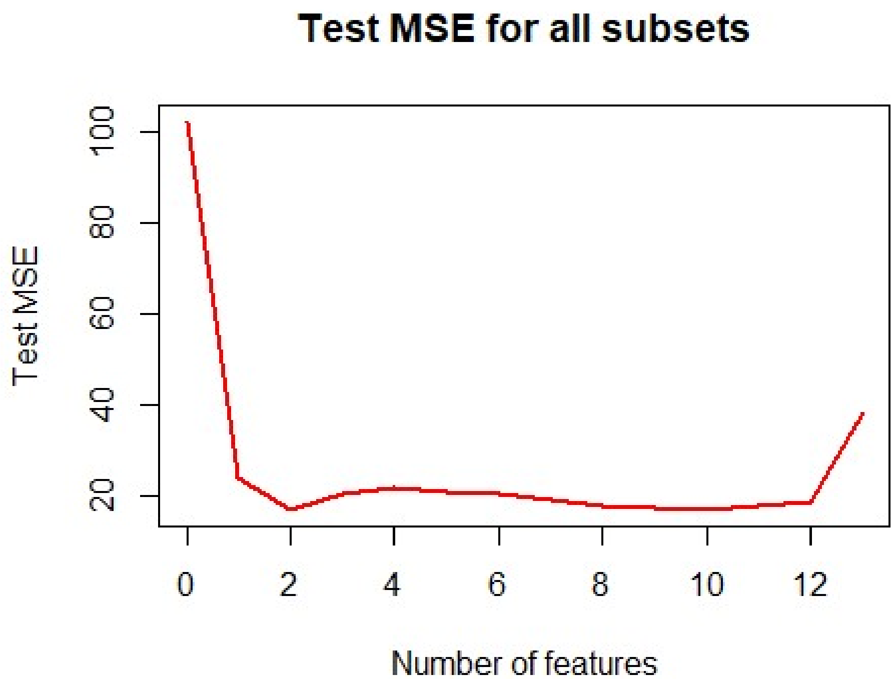
```
dt_train = dt[1:400,]
dt_test = dt[401:dim(dt)[1],]
varn = c("CRIM",     "ZN",       "INDUS",    "CHAS",     "NOX",      "RM",
         "AGE",      "DIS",      "RAD",      "TAX",      "PTRATIO",  "B" ,       "L
STAT")

varinclude = varn
varexlude = c()
fi_res = list()
fi_reg = list()
k=2

formu = paste("MEDV ~ 1", paste(varinclude, collapse = "+"), sep = "+")
tem_reg = lm(as.formula(formu), data = dt_train)
temp_res = list(13, "None",
                mean(tem_reg$residuals^2),
                mean((predict(tem_reg, newdata = dt_test)-dt_test$MEDV)^2),
                formu)

fi_reg[[1]] = tem_reg
fi_res[[1]] = c(temp_res)

while (!is.null(varinclude) & length(varinclude)>0) {
  tem_reg = list()
  temp_res = list()

  for (i in c(1:length(varinclude))){

    if (is.null(varinclude) | length(varinclude)==1){
      formu = "MEDV ~ 1"
    } else {
      formu = paste("MEDV ~ 1", paste(varinclude[-i], collapse = "+"), sep =
"+")
    }

    tem_reg[[i]] = lm(as.formula(formu), data = dt_train)
    temp_res[[i]] = list(length(varinclude)-1, varinclude[i],
                         mean(tem_reg[[i]]$residuals^2),
                         mean((predict(tem_reg[[i]], newdata = dt_test)-

                               dt_test$MEDV)^2),   formu)
  }
```

```
  temp_res = do.call(rbind.data.frame, temp_res)
  names(temp_res) = c("nvar", "var_exclude", "tr_mse", "ts_mse", "formula")
  temp_res$var_exclude = as.character(temp_res$var_exclude)
  temp_res$formula = as.character(temp_res$formula)
  varexlude = c(varexlude, as.character(temp_res[which.min(temp_res$ts_mse),
2]))
  varinclude = varinclude[!(varinclude %in% varexlude)]

  fi_reg[[k]] = tem_reg[[which.min(temp_res$ts_mse)]]
  fi_res[[k]] = c(temp_res[which.min(temp_res$ts_mse),])

  k=k+1
}

fi_res = do.call(rbind.data.frame, fi_res)
names(fi_res) = c("nvar", "var_include", "tr_mse", "ts_mse", "formula")
backward_res = fi_res

plot(fi_res$nvar, fi_res$tr_mse, type = "l", xlab = "Number of features", lwd
= 2, col = 2,
     ylab = "Training MSE")
title("Training MSE for all subsets")
```




```
plot(fi_res$nvar, fi_res$ts_mse, type = "l", xlab = "Number of features", lwd
= 2, col = 2,
```

```
     ylab = "Test MSE")
title("Test MSE for all subsets")
```




```
print(fi_res)

print(fi_res[which.min(fi_res$ts_mse),])

aa = which.min(fi_res$ts_mse)
res_131 = fi_reg[[aa]]
len1 = length(res_131$coefficients)
print(paste("For subset model ", fi_res[aa,]$formula, ":   y^hat = ", round(r
es_131$coefficients[1], 3), " + " ,
            paste(round(res_131$coefficients[2:len1], 3),
                  names(res_131$coefficients)[2:len1], sep = " * ", collapse
= " + "),
            "   with training MSE = ", round(fi_res[aa,3],3),
            " and validation MSE = ", round(fi_res[aa,4],3)))



##    nvar var_include   tr_mse    ts_mse
## 2    13        None 22.30523  37.89378
## 21   12          RM 28.17532  18.49887
## 3    11         TAX 29.25133  17.51549
## 4    10         AGE 29.58767  16.93445
## 5     9          ZN 30.43122  17.26423
## 6     8        CRIM 30.72554  17.80309
## 7     7       INDUS 30.90940  18.75269
```

```
## 8      6         DIS 34.93004  20.21552
## 9      5         NOX 34.94184  20.86509
## 10     4        CHAS 35.82823  21.82590
## 11     3         RAD 37.32106  20.31659
## 12     2           B 37.65518  16.76650
## 13     1     PTRATIO 42.62158  23.92312
## 14     0       LSTAT 83.80701 102.22659
##                                                               formula
## 2   MEDV ~ 1+CRIM+ZN+INDUS+CHAS+NOX+RM+AGE+DIS+RAD+TAX+PTRATIO+B+LSTAT
## 21     MEDV ~ 1+CRIM+ZN+INDUS+CHAS+NOX+AGE+DIS+RAD+TAX+PTRATIO+B+LSTAT
## 3          MEDV ~ 1+CRIM+ZN+INDUS+CHAS+NOX+AGE+DIS+RAD+PTRATIO+B+LSTAT
## 4              MEDV ~ 1+CRIM+ZN+INDUS+CHAS+NOX+DIS+RAD+PTRATIO+B+LSTAT
## 5                 MEDV ~ 1+CRIM+INDUS+CHAS+NOX+DIS+RAD+PTRATIO+B+LSTAT
## 6                      MEDV ~ 1+INDUS+CHAS+NOX+DIS+RAD+PTRATIO+B+LSTAT
## 7                            MEDV ~ 1+CHAS+NOX+DIS+RAD+PTRATIO+B+LSTAT
## 8                                MEDV ~ 1+CHAS+NOX+RAD+PTRATIO+B+LSTAT
## 9                                    MEDV ~ 1+CHAS+RAD+PTRATIO+B+LSTAT
## 10                                        MEDV ~ 1+RAD+PTRATIO+B+LSTAT
## 11                                            MEDV ~ 1+PTRATIO+B+LSTAT
## 12                                              MEDV ~ 1+PTRATIO+LSTAT
## 13                                                      MEDV ~ 1+LSTAT
## 14                                                            MEDV ~ 1
```

```
##    nvar var_include   tr_mse  ts_mse                formula
## 12    2           B 37.65518 16.7665 MEDV ~ 1+PTRATIO+LSTAT
```

```
## [1] "For subset model  MEDV ~ 1+PTRATIO+LSTAT :   y^hat =  52.79  +  -1.05
* PTRATIO + -0.85 * LSTAT    with training MSE =  37.655  and validation MSE
=  16.766"
```

Using backward search, based on validation MSE, the best model includes only two variables (LSTAT, PTRATIO).

(d)

(iii)

```
print(forward_res)
```

```
##    nvar var_include   tr_mse    ts_mse
## 2     0   intercept 83.80701 102.22659
## 21    1       LSTAT 42.62158  23.92312
## 3     2     PTRATIO 37.65518  16.76650
## 4     3        CHAS 36.43344  16.55229
```

```
## 5      4          ZN 36.42852  16.61204
## 6      5       INDUS 36.41458  16.87186
## 7      6         TAX 36.38863  17.15157
## 8      7         NOX 36.37764  17.36296
## 9      8         DIS 30.84108  17.31292
## 10     9        CRIM 30.83273  16.81341
## 11    10         AGE 30.51697  17.53081
## 12    11           B 30.28257  19.59739
## 13    12         RAD 28.17532  18.49887
## 14    13          RM 22.30523  37.89378
##                                                                 formula
## 2                                                              MEDV ~ 1
## 21                                                       MEDV ~ 1+LSTAT
## 3                                                MEDV ~ 1+LSTAT+PTRATIO
## 4                                           MEDV ~ 1+LSTAT+PTRATIO+CHAS
## 5                                        MEDV ~ 1+LSTAT+PTRATIO+CHAS+ZN
## 6                                  MEDV ~ 1+LSTAT+PTRATIO+CHAS+ZN+INDUS
## 7                              MEDV ~ 1+LSTAT+PTRATIO+CHAS+ZN+INDUS+TAX
## 8                          MEDV ~ 1+LSTAT+PTRATIO+CHAS+ZN+INDUS+TAX+NOX
## 9                      MEDV ~ 1+LSTAT+PTRATIO+CHAS+ZN+INDUS+TAX+NOX+DIS
## 10                MEDV ~ 1+LSTAT+PTRATIO+CHAS+ZN+INDUS+TAX+NOX+DIS+CRIM
## 11            MEDV ~ 1+LSTAT+PTRATIO+CHAS+ZN+INDUS+TAX+NOX+DIS+CRIM+AGE
## 12          MEDV ~ 1+LSTAT+PTRATIO+CHAS+ZN+INDUS+TAX+NOX+DIS+CRIM+AGE+B
## 13      MEDV ~ 1+LSTAT+PTRATIO+CHAS+ZN+INDUS+TAX+NOX+DIS+CRIM+AGE+B+RAD
## 14 MEDV ~ 1+LSTAT+PTRATIO+CHAS+ZN+INDUS+TAX+NOX+DIS+CRIM+AGE+B+RAD+RM
```

```
print(backward_res)
```

```
##    nvar var_include   tr_mse    ts_mse
## 2    13        None 22.30523  37.89378
## 21   12          RM 28.17532  18.49887
## 3    11         TAX 29.25133  17.51549
## 4    10         AGE 29.58767  16.93445
## 5     9          ZN 30.43122  17.26423
## 6     8        CRIM 30.72554  17.80309
## 7     7       INDUS 30.90940  18.75269
## 8     6         DIS 34.93004  20.21552
## 9     5         NOX 34.94184  20.86509
## 10    4        CHAS 35.82823  21.82590
## 11    3         RAD 37.32106  20.31659
## 12    2           B 37.65518  16.76650
## 13    1     PTRATIO 42.62158  23.92312
## 14    0       LSTAT 83.80701 102.22659
##                                                                 formula
## 2  MEDV ~ 1+CRIM+ZN+INDUS+CHAS+NOX+RM+AGE+DIS+RAD+TAX+PTRATIO+B+LSTAT
## 21    MEDV ~ 1+CRIM+ZN+INDUS+CHAS+NOX+AGE+DIS+RAD+TAX+PTRATIO+B+LSTAT
## 3         MEDV ~ 1+CRIM+ZN+INDUS+CHAS+NOX+AGE+DIS+RAD+PTRATIO+B+LSTAT
## 4             MEDV ~ 1+CRIM+ZN+INDUS+CHAS+NOX+DIS+RAD+PTRATIO+B+LSTAT
## 5                MEDV ~ 1+CRIM+INDUS+CHAS+NOX+DIS+RAD+PTRATIO+B+LSTAT
## 6                     MEDV ~ 1+INDUS+CHAS+NOX+DIS+RAD+PTRATIO+B+LSTAT
```

```
## 7                              MEDV ~ 1+CHAS+NOX+DIS+RAD+PTRATIO+B+LSTAT
## 8                                  MEDV ~ 1+CHAS+NOX+RAD+PTRATIO+B+LSTAT
## 9                                      MEDV ~ 1+CHAS+RAD+PTRATIO+B+LSTAT
## 10                                          MEDV ~ 1+RAD+PTRATIO+B+LSTAT
## 11                                              MEDV ~ 1+PTRATIO+B+LSTAT
## 12                                                MEDV ~ 1+PTRATIO+LSTAT
## 13                                                        MEDV ~ 1+LSTAT
## 14                                                              MEDV ~ 1
```

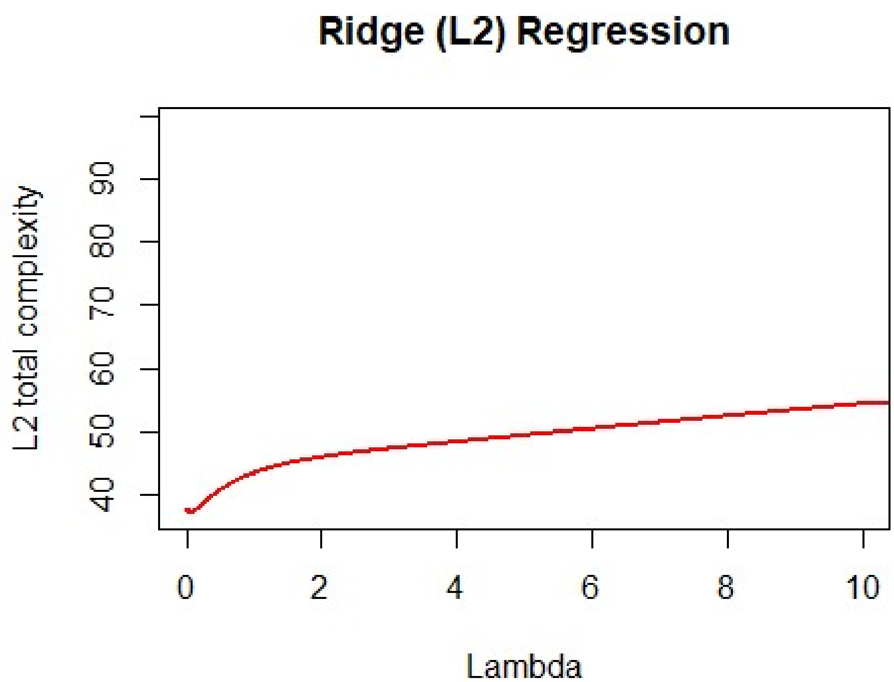
All lower order models are nested in the full model with all features. The best model found from backward search is nested within model found from forward search because forward model includes LSTAT, PTRATIO, CHAS variables and backward model includes only LSTAT, PTRATIO that can be found by considering CHAS coefficient equal zero.

(d)

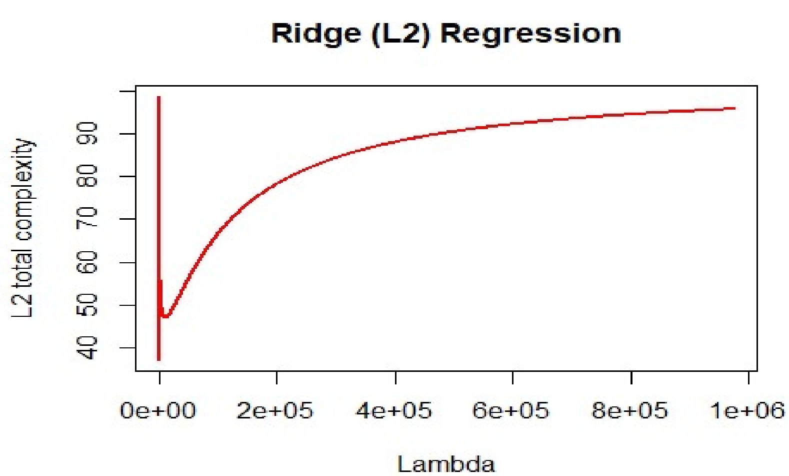
(iv)

```
# Ridge

dt_train = dt[1:400,]
dt_test = dt[401:dim(dt)[1],]
varn = c("CRIM",     "ZN",       "INDUS",    "CHAS",     "NOX",      "RM",
         "AGE",      "DIS",      "RAD",      "TAX",      "PTRATIO", "B" ,       "L
STAT")

lambda = 0
l2_res = NULL

loop = TRUE

while (loop) {
  a = glmnet(x = as.matrix(dt_train[,varn]), y = dt_train[,"MEDV"], standardi
ze=FALSE,
             alpha = 0, lambda = lambda)
  pred = predict(a, as.matrix(dt_test[,varn]))
  l2 = mean((dt_test$MEDV - pred)^2) + lambda * sum(a$beta^2)
  l2_res = rbind(l2_res, c(a$lambda, l2))
  if (lambda <10){
    lambda = lambda + .005
  }else {
    lambda = lambda + 100
  }

  if(sum(a$beta^2) < 1e-5 | lambda > 1500000) loop = FALSE
```

```
}

l2_res = as.data.frame(l2_res)
plot(l2_res$V1, l2_res$V2, type = "l", xlab = "Lambda", lwd = 2, col = 2, xli
m = c(0, 10),
     ylab = "L2 total complexity")
title("Ridge (L2) Regression")
```




```
plot(l2_res$V1, l2_res$V2, type = "l", xlab = "Lambda", lwd = 2, col = 2,
     ylab = "L2 total complexity")
title("Ridge (L2) Regression")
```



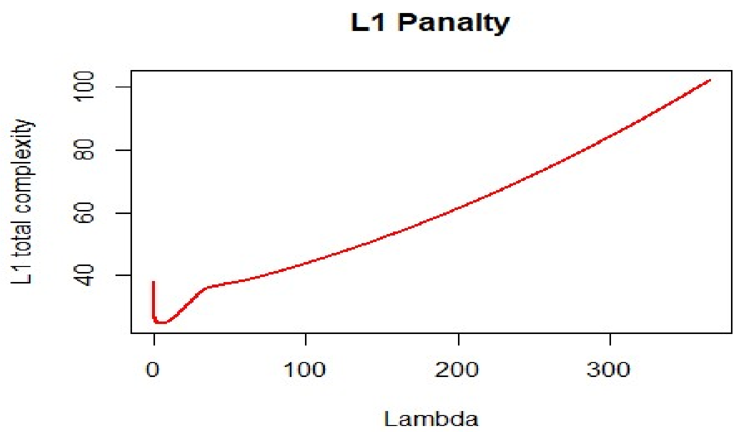
```r
aa = which.min(l2_res$V2)
a = glmnet(x = as.matrix(dt_train[,varn]), y = dt_train[,"MEDV"], standardize
=FALSE,
             alpha = 0, lambda = l2_res[aa,1])
pred = predict(a, as.matrix(dt_test[,varn]))
l2 = mean((dt_test$MEDV - pred)^2)

print(paste("Model with lambda = ", round(l2_res[aa,1],3), ":   y^hat = ", ro
und(a$a0, 3),
            " + " ,
            paste(round(a$beta, 3),
                  row.names(a$beta), sep = " * ", collapse = " + "),
    "   with testing MSE = ", round(l2,3)))
```

```
## [1] "Model with lambda =  0.065 :   y^hat =  21.105   +  -0.184 * CRIM + 0.
047 * ZN + 0.006 * INDUS + 1.548 * CHAS + -4.246 * NOX + 4.849 * RM + -0.005
* AGE + -1.159 * DIS + 0.46 * RAD + -0.017 * TAX + -0.696 * PTRATIO + 0.002 *
B + -0.539 * LSTAT    with testing MSE =  34.081"
```

```r
print(paste("Model with lambda = ", l2_res[aa,1], " has lowest validation MSE
, ", round(l2_res[aa,2],3)))
```

```
## [1] "Model with lambda =  0.065  has lowest validation MSE,  37.091"
```

```r
# LASSO

lambda = 0
l1_res = NULL

loop = TRUE

while (loop) {
  a = glmnet(x = as.matrix(dt_train[,varn]), y = dt_train[,"MEDV"], standardi
ze=FALSE,
             alpha = 1, lambda = lambda)
  pred = predict(a, as.matrix(dt_test[,varn]))
  l1 = mean((dt_test$MEDV - pred)^2) + lambda * sum(abs(a$beta))
  l1_res = rbind(l1_res, c(a$lambda, l1))
  lambda = lambda + 0.05
  if(sum(abs(a$beta)) < 1e-8) loop = FALSE
}

l1_res = as.data.frame(l1_res)
plot(l1_res$V1, l1_res$V2, type = "l", xlab = "Lambda", lwd = 2, col = 2,
     ylab = "L1 total complexity")
title("L1 Panalty")
```




```
aa = which.min(l1_res$V2)
a = glmnet(x = as.matrix(dt_train[,varn]), y = dt_train[,"MEDV"], standardize
=FALSE,
             alpha = 1, lambda = l1_res[aa,1])
pred = predict(a, as.matrix(dt_test[,varn]))
mmse = mean((dt_test$MEDV - pred)^2)

print(paste("Model with lambda = ", round(l1_res[aa,1],3), ":   y^hat = ", ro
und(a$a0, 3),
            " + " ,
            paste(round(a$beta, 5),
                  row.names(a$beta), sep = " * ", collapse = " + "),
    "   with testing MSE = ", round(mmse,3)))

## [1] "Model with lambda =  5.1 :   y^hat =  29.115  +  0 * CRIM + 0.04001 *
ZN + 0 * INDUS + 0 * CHAS + 0 * NOX + 0 * RM + 0.05182 * AGE + 0 * DIS + 0 *
RAD + -0.00581 * TAX + 0 * PTRATIO + 0.00771 * B + -0.85077 * LSTAT    with t
esting MSE =  19.898"

print(paste("Model with lambda = ", round(l1_res[aa,1], 3), " has lowest vali
dation MSE, ", round(mmse,3)))

## [1] "Model with lambda =  5.1  has lowest validation MSE,  19.898"
```

Using Ridge regression best model found for $\lambda = 0.065$ which includes all variables with validation $MSE = 37.091$. On the other hand, best LASSO model found fro $\lambda = 5.1$ with validation $MSE = 19.898$ which includes only ZN, AGE, TAX, B and LSTAT and it is mach lower that Ridge regression. However, forward search model includes only LSTAT, PTRATIO, CHAS variables with validation $MSE = 16.552$ and it is the lowest MSE among forward, backward, Ridge and LASSO models.