Boston Housing Price Prediction

Spring 2020

Background

The Boston Housing Price Dataset was obtained from the StatLib library which is maintained at Carnegie Mellon University. It contains US census data concerning houses in various areas around the city of Boston.

The dataset consists of 506 observations of 14 attributes. Below is a brief description of each feature and the outcome in our dataset:

1. *crim* - per capita crime rate by town
2. *zn* - proportion of residential land zoned for lots over 25,000 sq.ft
3. *indus* - proportion of non-retail business acres per town
4. *chas* - Charles River dummy variable (1 if tract bounds river; else 0)
5. *nox* - nitric oxides concentration (parts per 10 million)
6. *rm* - average number of rooms per dwelling
7. *age* - proportion of owner-occupied units built prior to 1940
8. *dis* - weighted distances to five Boston employment centres
9. *rad* - index of accessibility to radial highways
10. *tax* - full-value property-tax rate per $10,000
11. *ptratio* - pupil-teacher ratio by town
12. *black* - 1000(Bk - 0.63)^2 where Bk is the proportion of blacks by town
13. *lstat* - % lower status of the population
14. *medv* - Median value of owner-occupied homes in $1000's

Please load the dataset "Boston" and then split the dataset into a train and test set in a 80:20 ratio. Use the training set to build the models in Questions 1-6. Use the test set to help evaluate model performance in Question 7. Please make sure that you are using R version 3.6.X.

Read Data

```
set.seed(100)

fullData = read.csv("Boston.csv",header=TRUE)
testRows = sample(nrow(fullData),0.2*nrow(fullData))
testData = fullData[testRows, ]
trainData = fullData[-testRows, ]
```

Question 1: Full Model

(a) Fit a standard linear regression with the variable *medv* as the response and the other variables as predictors. Call it *model1*. Display the model summary.

```
m1=lm(medv~., trainData)
summary(m1)
```

```
##
## Call:
## lm(formula = medv ~ ., data = trainData)
```

```
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -16.1428  -2.6946  -0.5747   1.9087  27.2846
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  3.392e+01  5.928e+00   5.723 2.09e-08 ***
## crim        -1.170e-01  3.791e-02  -3.087  0.00217 **
## zn           5.029e-02  1.482e-02   3.394  0.00076 ***
## indus        3.611e-02  6.811e-02   0.530  0.59625
## chas         3.173e+00  9.688e-01   3.275  0.00115 **
## nox         -1.855e+01  4.450e+00  -4.168 3.79e-05 ***
## rm           4.080e+00  5.003e-01   8.156 4.77e-15 ***
## age         -4.247e-04  1.512e-02  -0.028  0.97760
## dis         -1.424e+00  2.220e-01  -6.415 4.06e-10 ***
## rad          3.062e-01  7.553e-02   4.054 6.08e-05 ***
## tax         -1.262e-02  4.198e-03  -3.007  0.00281 **
## ptratio     -9.102e-01  1.427e-01  -6.376 5.12e-10 ***
## black        8.180e-03  3.089e-03   2.648  0.00843 **
## lstat       -4.754e-01  5.782e-02  -8.222 2.98e-15 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.704 on 391 degrees of freedom
## Multiple R-squared:  0.7411, Adjusted R-squared:  0.7325
## F-statistic: 86.08 on 13 and 391 DF,  p-value: < 2.2e-16
```

(b) Which regression coefficients are significant at the 95% confidence level? At the 99% confidence level?

```
set.seed(100)
p=summary(m1)$coef[,4]
a95=0.05
a99=0.01

p<=a95
```

```
## (Intercept)        crim          zn       indus        chas         nox
##        TRUE        TRUE        TRUE       FALSE        TRUE        TRUE
##          rm         age         dis         rad         tax     ptratio
##        TRUE       FALSE        TRUE        TRUE        TRUE        TRUE
##       black       lstat
##        TRUE        TRUE
```

```
p<=a99
```

```
## (Intercept)        crim          zn       indus        chas         nox
##        TRUE        TRUE        TRUE       FALSE        TRUE        TRUE
##          rm         age         dis         rad         tax     ptratio
##        TRUE       FALSE        TRUE        TRUE        TRUE        TRUE
##       black       lstat
##        TRUE        TRUE
```

coefficients are significant at the 95% confidence level: crim,zn,chas,nox, rm, dis, rad,tax,ptratio, black, lstat

coefficients are significant at the 99% confidence level:crim, zn,chas,nox,rm, dis, rad, tax, ptraio, black, lstat

(c) What are the 10-fold and leave one out cross-validation scores for this model?

```
set.seed(100)
library(boot)
n=nrow(trainData)
attach(trainData)
glm.fit=glm(medv~.,data=trainData)
c(cv.glm(trainData,glm.fit,K=10)$delta[1],cv.glm(trainData,glm.fit,K=nrow(trainData))$delta[1])
```

```
## [1] 23.85361 23.82173
```

10-fold CV: 23.85361

leave one out cross-validation:23.82173

(d) What are the Mallow's Cp, AIC, and BIC criterion values for this model?

```
set.seed(100)
library(CombMSC)
n=nrow(trainData)
c(Cp(m1,S2=4.704^2),AIC(m1,k=2),AIC(m1,k=log(n)))
```

```
## [1]   13.97495 2419.28127 2479.33957
```

Mallow's Cp:13.97495

AIC: 2419.28127

BIC: 2479.33957

(e) Build a new model on the training data with only the variables which coefficients were found to be statistically significant at the 99% confident level. Call it *model2*. Perform an ANOVA test to compare this new model with the full model. Which one would you prefer? Is it good practice to select variables based on statistical significance of individual coefficients? Explain.

```
set.seed(100)

m2=lm(medv~crim+zn+chas+nox+rm+dis+rad+tax+ptratio+black+lstat )
summary(m2)
```

```
##
## Call:
## lm(formula = medv ~ crim + zn + chas + nox + rm + dis + rad +
##     tax + ptratio + black + lstat)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -16.1233  -2.7088  -0.6016   1.8937  27.2943
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  33.772638   5.846894   5.776 1.56e-08 ***
## crim         -0.117770   0.037788  -3.117 0.001964 **
## zn            0.049737   0.014578   3.412 0.000712 ***
## chas          3.240501   0.956878   3.387 0.000779 ***
## nox         -17.946001   4.012830  -4.472 1.01e-05 ***
## rm            4.049937   0.483386   8.378 9.63e-16 ***
## dis          -1.447350   0.209868  -6.896 2.14e-11 ***
## rad           0.296828   0.072912   4.071 5.66e-05 ***
## tax          -0.011748   0.003851  -3.051 0.002436 **
```

```
## ptratio      -0.899458   0.139980  -6.426 3.80e-10 ***
## black         0.008115   0.003077   2.638 0.008683 **
## lstat        -0.473518   0.053260  -8.891  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.694 on 393 degrees of freedom
## Multiple R-squared:  0.7409, Adjusted R-squared:  0.7336
## F-statistic: 102.1 on 11 and 393 DF,  p-value: < 2.2e-16
```

```
anova(m1,m2)
```

```
## Analysis of Variance Table
##
## Model 1: medv ~ crim + zn + indus + chas + nox + rm + age + dis + rad +
##     tax + ptratio + black + lstat
## Model 2: medv ~ crim + zn + chas + nox + rm + dis + rad + tax + ptratio +
##     black + lstat
##   Res.Df    RSS Df Sum of Sq      F Pr(>F)
## 1    391 8651.3
## 2    393 8657.6 -2   -6.2491 0.1412 0.8683
```

After apply Anova test, p-value of 0.8683 is close to 1 means statstically not significant by reducing varibles which are not significant at 99% CI, therefor I prefer model1. It's not a good practice to select varibles based on statistical significance of individual coedfficients, because there could be other factors like complete sarperration or outliers that have influence on significance of predictor which still have very strong predicting power.

Question 2: Full Model Search

(a) Compare all possible models using Mallow's Cp. What is the total number of possible models with the full set of variables? Display a table indicating the variables included in the best model of each size and the corresponding Mallow's Cp value.

Hint: The table must include 13 models. You can use nbest parameter.

```
set.seed(100)
library(leaps)
attach(trainData)
outM=leaps(trainData[,-c(14)],medv,method = "Cp")
cbind(as.matrix(outM$which),outM$Cp)[71:121,]
```

```
##    1 2 3 4 5 6 7 8 9 A B C D
## 8  0 1 0 1 1 1 0 1 0 0 1 1 1 25.32215
## 8  1 1 0 1 1 1 0 1 0 0 1 0 1 26.50659
## 8  1 0 0 1 1 1 0 1 1 0 1 0 1 29.56004
## 8  0 0 0 1 1 1 0 1 1 0 1 1 1 29.74207
## 8  1 0 0 1 1 1 0 1 0 0 1 1 1 31.06765
## 8  0 1 0 1 1 1 0 1 0 1 1 0 1 32.42551
## 8  0 1 0 1 1 1 0 1 1 0 1 0 1 32.99574
## 8  0 0 0 1 1 1 0 1 1 1 1 0 1 33.01005
## 8  0 0 0 1 1 1 1 1 0 0 1 1 1 33.32976
## 8  0 1 1 1 1 1 0 1 0 0 1 0 1 33.34250
## 9  1 1 0 1 1 1 0 1 0 0 1 1 1 22.81194
## 9  1 1 0 1 1 1 0 1 1 0 1 0 1 23.29242
## 9  1 0 0 1 1 1 0 1 1 0 1 1 1 23.82741
## 9  0 1 0 1 1 1 0 1 1 0 1 1 1 24.86961
```

```
## 9  1 0 0 1 1 1 0 1 1 1 1 0 1 24.95445
## 9  0 1 0 1 1 1 0 1 1 1 1 0 1 25.19470
## 9  1 1 0 0 1 1 0 1 1 1 1 0 1 25.40038
## 9  0 0 0 1 1 1 0 1 1 1 1 1 1 25.81434
## 9  0 1 0 1 1 1 0 1 0 1 1 1 1 27.04366
## 9  0 1 1 1 1 1 0 1 0 0 1 1 1 27.16816
## 10 1 1 0 1 1 1 0 1 1 1 1 0 1 15.20861
## 10 1 1 0 1 1 1 0 1 1 0 1 1 1 17.55010
## 10 0 1 0 1 1 1 0 1 1 1 1 1 1 17.95331
## 10 1 1 0 0 1 1 0 1 1 1 1 1 1 19.70092
## 10 1 0 0 1 1 1 0 1 1 1 1 1 1 19.87214
## 10 1 1 1 1 1 1 0 1 1 0 1 0 1 24.55769
## 10 1 1 0 1 1 1 1 1 0 0 1 1 1 24.65809
## 10 1 1 1 1 1 1 0 1 0 0 1 1 1 24.67387
## 10 1 1 0 1 1 1 0 1 0 1 1 1 1 24.78352
## 10 1 1 0 1 1 1 1 1 1 0 1 0 1 25.28618
## 11 1 1 0 1 1 1 0 1 1 1 1 1 1 10.28243
## 11 1 1 1 1 1 1 0 1 1 1 1 0 1 17.02162
## 11 1 1 0 1 1 1 1 1 1 1 1 0 1 17.19893
## 11 1 1 1 1 1 1 0 1 1 0 1 1 1 19.04609
## 11 0 1 1 1 1 1 0 1 1 1 1 1 1 19.53095
## 11 1 1 0 1 1 1 1 1 1 0 1 1 1 19.54626
## 11 0 1 0 1 1 1 1 1 1 1 1 1 1 19.95112
## 11 1 1 1 0 1 1 0 1 1 1 1 1 1 20.74356
## 11 1 0 0 1 1 1 1 1 1 1 1 1 1 21.58278
## 11 1 1 0 0 1 1 1 1 1 1 1 1 1 21.68644
## 12 1 1 1 1 1 1 0 1 1 1 1 1 1 12.00079
## 12 1 1 0 1 1 1 1 1 1 1 1 1 1 12.28114
## 12 1 1 1 1 1 1 1 1 1 1 1 0 1 19.01049
## 12 1 1 1 1 1 1 1 1 1 0 1 1 1 21.04130
## 12 0 1 1 1 1 1 1 1 1 1 1 1 1 21.52786
## 12 1 1 1 0 1 1 1 1 1 1 1 1 1 22.72704
## 12 1 0 1 1 1 1 1 1 1 1 1 1 1 23.51614
## 12 1 1 1 1 1 1 1 1 0 1 1 1 1 28.43349
## 12 1 1 1 1 0 1 1 1 1 1 1 1 1 29.36967
## 12 1 1 1 1 1 1 1 1 1 1 0 1 1 52.65938
## 13 1 1 1 1 1 1 1 1 1 1 1 1 1 14.00000
```

```
bestM=which(outM$Cp==min(outM$Cp))
cbind(as.matrix(outM$which),outM$Cp)[bestM,]
```

```
##        1        2        3        4        5        6        7        8
##  1.00000  1.00000  0.00000  1.00000  1.00000  1.00000  0.00000  1.00000
##        9        A        B        C        D
##  1.00000  1.00000  1.00000  1.00000  1.00000 10.28243
```

total possible number of model: 2^13

1 2 3 4 5 6 7 8 9 A B C D
1 0 0 0 0 0 0 0 0 0 0 0 0 1 298.07927 2 0 0 0 0 0 1 0 0 0 0 0 0 1 146.92399 3 0 0 0 0 0 1 0 0 0 0 1 0 1 87.73031 4 0 0 0 0 0 1 0 0 0 0 1 1 1 77.39864 5 0 0 0 0 1 1 0 1 0 0 1 0 1 50.68647 6 0 0 0 1 1 1 0 1 0 0 1 0 1 37.41118 7 0 1 0 1 1 1 0 1 0 0 1 0 1 31.71986 8 0 1 0 1 1 1 0 1 0 0 1 1 1 25.32215 9 1 1 0 1 1 1 0 1 0 0 1 1 1 22.81194 10 1 1 0 1 1 1 0 1 1 1 1 0 1 15.20861 11 1 1 0 1 1 1 0 1 1 1 1 1 1 10.28243 12 1 1 1 1 1 1 0 1 1 1 1 1 1 12.00079 13 1 1 1 1 1 1 1 1 1 1 1 1 1 14.00000

(b) How many variables are in the model with the lowest Mallow's Cp value? Which variables are they?

Fit this model and call it *model3*.

```
set.seed(100)

m3=lm(medv~ crim+zn+chas+nox+rm+dis+rad+tax+ptratio+black+lstat)
summary(m3)
```

```
##
## Call:
## lm(formula = medv ~ crim + zn + chas + nox + rm + dis + rad +
##     tax + ptratio + black + lstat)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -16.1233  -2.7088  -0.6016   1.8937  27.2943
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  33.772638   5.846894   5.776 1.56e-08 ***
## crim         -0.117770   0.037788  -3.117 0.001964 **
## zn            0.049737   0.014578   3.412 0.000712 ***
## chas          3.240501   0.956878   3.387 0.000779 ***
## nox         -17.946001   4.012830  -4.472 1.01e-05 ***
## rm            4.049937   0.483386   8.378 9.63e-16 ***
## dis          -1.447350   0.209868  -6.896 2.14e-11 ***
## rad           0.296828   0.072912   4.071 5.66e-05 ***
## tax          -0.011748   0.003851  -3.051 0.002436 **
## ptratio      -0.899458   0.139980  -6.426 3.80e-10 ***
## black         0.008115   0.003077   2.638 0.008683 **
## lstat        -0.473518   0.053260  -8.891  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.694 on 393 degrees of freedom
## Multiple R-squared:  0.7409, Adjusted R-squared:  0.7336
## F-statistic: 102.1 on 11 and 393 DF,  p-value: < 2.2e-16
```

1 varibles model has the lowest Cp value 10.28243 , they are every predicting varibles excpet 3. indus, and 7. age.

Question 3: Stepwise Regression

(a) Perform backward stepwise regression using BIC. Allow the minimum model to be the model with only an intercept, and the full model to be *model1*. Display the model summary of your final model. Call it *model4*

```
set.seed(100)
mini=lm(medv~1)

m4=step(m1,scope=list(lower=mini,upper=m1),direction = "backward",k=log(n))
```

```
## Start:  AIC=1324
## medv ~ crim + zn + indus + chas + nox + rm + age + dis + rad +
##     tax + ptratio + black + lstat
##
##           Df Sum of Sq     RSS    AIC
```

```
## - age      1      0.02  8651.4 1318.0
## - indus    1      6.22  8657.6 1318.3
## <none>                  8651.3 1324.0
## - black    1    155.12  8806.5 1325.2
## - tax      1    200.05  8851.4 1327.2
## - crim     1    210.82  8862.2 1327.7
## - chas     1    237.35  8888.7 1329.0
## - zn       1    254.81  8906.2 1329.8
## - rad      1    363.61  9015.0 1334.7
## - nox      1    384.32  9035.7 1335.6
## - ptratio  1    899.64  9551.0 1358.1
## - dis      1    910.68  9562.0 1358.5
## - rm       1   1471.83 10123.2 1381.6
## - lstat    1   1495.84 10147.2 1382.6
##
## Step:  AIC=1317.99
## medv ~ crim + zn + indus + chas + nox + rm + dis + rad + tax +
##     ptratio + black + lstat
##
##           Df Sum of Sq     RSS    AIC
## - indus    1      6.23  8657.6 1312.3
## <none>                  8651.4 1318.0
## - black    1    155.34  8806.7 1319.2
## - tax      1    200.14  8851.5 1321.2
## - crim     1    210.87  8862.2 1321.7
## - chas     1    237.70  8889.1 1323.0
## - zn       1    261.18  8912.5 1324.0
## - rad      1    366.72  9018.1 1328.8
## - nox      1    433.43  9084.8 1331.8
## - ptratio  1    911.69  9563.1 1352.6
## - dis      1    966.15  9617.5 1354.9
## - rm       1   1549.90 10201.3 1378.7
## - lstat    1   1745.96 10397.3 1386.4
##
## Step:  AIC=1312.28
## medv ~ crim + zn + chas + nox + rm + dis + rad + tax + ptratio +
##     black + lstat
##
##           Df Sum of Sq     RSS    AIC
## <none>                  8657.6 1312.3
## - black    1    153.25  8810.8 1313.4
## - tax      1    205.06  8862.7 1315.8
## - crim     1    213.98  8871.6 1316.2
## - chas     1    252.65  8910.2 1317.9
## - zn       1    256.44  8914.0 1318.1
## - rad      1    365.11  9022.7 1323.0
## - nox      1    440.59  9098.2 1326.4
## - ptratio  1    909.57  9567.2 1346.7
## - dis      1   1047.76  9705.3 1352.5
## - rm       1   1546.37 10204.0 1372.8
## - lstat    1   1741.27 10398.9 1380.5
```

```
summary(m4)
```

```
##
```

```
## Call:
## lm(formula = medv ~ crim + zn + chas + nox + rm + dis + rad +
##     tax + ptratio + black + lstat, data = trainData)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -16.1233  -2.7088  -0.6016   1.8937  27.2943
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  33.772638   5.846894   5.776 1.56e-08 ***
## crim         -0.117770   0.037788  -3.117 0.001964 **
## zn            0.049737   0.014578   3.412 0.000712 ***
## chas          3.240501   0.956878   3.387 0.000779 ***
## nox         -17.946001   4.012830  -4.472 1.01e-05 ***
## rm            4.049937   0.483386   8.378 9.63e-16 ***
## dis          -1.447350   0.209868  -6.896 2.14e-11 ***
## rad           0.296828   0.072912   4.071 5.66e-05 ***
## tax          -0.011748   0.003851  -3.051 0.002436 **
## ptratio      -0.899458   0.139980  -6.426 3.80e-10 ***
## black         0.008115   0.003077   2.638 0.008683 **
## lstat        -0.473518   0.053260  -8.891  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.694 on 393 degrees of freedom
## Multiple R-squared:  0.7409, Adjusted R-squared:  0.7336
## F-statistic: 102.1 on 11 and 393 DF,  p-value: < 2.2e-16
```

(b) How many variables are in *model4*? Which regression coefficients are significant at the 99% confidence level?

```
set.seed(100)
p=summary(m4)$coef[,4]

a99=0.01


p<=a99
```

```
## (Intercept)        crim          zn        chas         nox          rm
##        TRUE        TRUE        TRUE        TRUE        TRUE        TRUE
##         dis         rad         tax     ptratio       black       lstat
##        TRUE        TRUE        TRUE        TRUE        TRUE        TRUE
```

11 varibles in m4. All coefficients including intercept are siginificant at 99% CI.

(c) Perform forward stepwise selection with AIC. Allow the minimum model to be the model with onlyan intercept, and the full model to be *model1*. Display the model summary of your final model. Call it *model5*. Do the variables included in *model5* differ from the variables in *model4*?

```
set.seed(100)
m5=step(mini,scope=list(lower=mini,upper=m1),direction = "forward")
```

```
## Start:  AIC=1789.17
## medv ~ 1
##
```

```
##           Df Sum of Sq   RSS    AIC
## + lstat    1   17943.1 15468 1479.3
## + rm       1   16903.9 16507 1505.6
## + ptratio  1    9083.3 24328 1662.7
## + indus    1    7874.6 25536 1682.3
## + tax      1    7286.3 26125 1691.5
## + nox      1    6106.6 27304 1709.4
## + age      1    5228.9 28182 1722.2
## + crim     1    5156.1 28255 1723.3
## + rad      1    4938.9 28472 1726.4
## + zn       1    4673.0 28738 1730.2
## + black    1    3399.2 30012 1747.7
## + dis      1    2110.7 31300 1764.7
## + chas     1    1311.0 32100 1775.0
## <none>                 33411 1789.2
##
## Step:  AIC=1479.27
## medv ~ lstat
##
##           Df Sum of Sq   RSS    AIC
## + rm       1    3388.7 12079 1381.1
## + ptratio  1    2141.3 13327 1420.9
## + chas     1     786.1 14682 1460.2
## + dis      1     528.7 14939 1467.2
## + tax      1     200.3 15268 1476.0
## + zn       1     189.3 15279 1476.3
## + age      1     158.3 15310 1477.1
## + black    1     143.1 15325 1477.5
## + crim     1      98.4 15370 1478.7
## + indus    1      79.0 15389 1479.2
## <none>                 15468 1479.3
## + rad      1       9.5 15458 1481.0
## + nox      1       0.0 15468 1481.3
##
## Step:  AIC=1381.12
## medv ~ lstat + rm
##
##           Df Sum of Sq   RSS    AIC
## + ptratio  1   1353.98 10725 1335.0
## + chas     1    560.55 11519 1363.9
## + black    1    362.57 11717 1370.8
## + tax      1    362.24 11717 1370.8
## + crim     1    319.22 11760 1372.3
## + dis      1    190.91 11888 1376.7
## + rad      1    167.69 11912 1377.5
## + zn       1     73.21 12006 1380.7
## <none>                 12079 1381.1
## + indus    1     46.37 12033 1381.6
## + nox      1     34.48 12045 1382.0
## + age      1      0.11 12079 1383.1
##
## Step:  AIC=1334.97
## medv ~ lstat + rm + ptratio
##
```

```
##         Df Sum of Sq   RSS    AIC
## + chas   1    410.07 10315 1321.2
## + black  1    272.85 10452 1326.5
## + dis    1    258.15 10467 1327.1
## + crim   1    161.47 10564 1330.8
## + tax    1     64.04 10661 1334.5
## + nox    1     56.19 10669 1334.8
## <none>               10725 1335.0
## + age    1     10.51 10715 1336.6
## + zn     1      0.36 10725 1337.0
## + rad    1      0.21 10725 1337.0
## + indus  1      0.06 10725 1337.0
##
## Step:  AIC=1321.18
## medv ~ lstat + rm + ptratio + chas
##
##         Df Sum of Sq   RSS    AIC
## + black  1   247.499 10068 1313.3
## + dis    1   186.709 10128 1315.8
## + crim   1   145.710 10170 1317.4
## + nox    1    85.404 10230 1319.8
## + tax    1    62.169 10253 1320.7
## <none>               10315 1321.2
## + indus  1     6.616 10309 1322.9
## + zn     1     1.532 10314 1323.1
## + rad    1     0.998 10314 1323.1
## + age    1     0.795 10314 1323.2
##
## Step:  AIC=1313.34
## medv ~ lstat + rm + ptratio + chas + black
##
##         Df Sum of Sq     RSS    AIC
## + dis    1   238.702  9829.0 1305.6
## + crim   1    70.135  9997.5 1312.5
## <none>               10067.7 1313.3
## + nox    1    37.957 10029.7 1313.8
## + rad    1    15.959 10051.7 1314.7
## + tax    1    11.589 10056.1 1314.9
## + age    1     2.237 10065.4 1315.2
## + zn     1     0.722 10066.9 1315.3
## + indus  1     0.081 10067.6 1315.3
##
## Step:  AIC=1305.63
## medv ~ lstat + rm + ptratio + chas + black + dis
##
##         Df Sum of Sq    RSS    AIC
## + nox    1    517.23 9311.7 1285.7
## + zn     1    181.19 9647.8 1300.1
## + indus  1    118.87 9710.1 1302.7
## + crim   1    114.72 9714.2 1302.9
## + age    1    107.76 9721.2 1303.2
## + tax    1     83.17 9745.8 1304.2
## <none>               9829.0 1305.6
## + rad    1      0.14 9828.8 1307.6
```

```
##
## Step:  AIC=1285.73
## medv ~ lstat + rm + ptratio + chas + black + dis + nox
##
##         Df Sum of Sq    RSS    AIC
## + zn     1   188.613 9123.1 1279.5
## + rad    1    90.817 9220.9 1283.8
## + crim   1    61.487 9250.2 1285.0
## <none>               9311.7 1285.7
## + age    1    11.435 9300.3 1287.2
## + indus  1     2.566 9309.2 1287.6
## + tax    1     0.876 9310.9 1287.7
##
## Step:  AIC=1279.45
## medv ~ lstat + rm + ptratio + chas + black + dis + nox + zn
##
##         Df Sum of Sq    RSS    AIC
## + crim   1    99.794 9023.3 1277.0
## + rad    1    54.266 9068.9 1279.0
## <none>               9123.1 1279.5
## + tax    1     6.162 9117.0 1281.2
## + indus  1     3.407 9119.7 1281.3
## + age    1     0.977 9122.1 1281.4
##
## Step:  AIC=1276.99
## medv ~ lstat + rm + ptratio + chas + black + dis + nox + zn +
##     crim
##
##         Df Sum of Sq    RSS    AIC
## + rad    1   160.677 8862.7 1271.7
## <none>               9023.3 1277.0
## + age    1     3.404 9019.9 1278.8
## + indus  1     3.055 9020.3 1278.8
## + tax    1     0.629 9022.7 1279.0
##
## Step:  AIC=1271.71
## medv ~ lstat + rm + ptratio + chas + black + dis + nox + zn +
##     crim + rad
##
##         Df Sum of Sq    RSS    AIC
## + tax    1   205.058 8657.6 1264.2
## <none>               8862.7 1271.7
## + indus  1    11.152 8851.5 1273.2
## + age    1     0.085 8862.6 1273.7
##
## Step:  AIC=1264.23
## medv ~ lstat + rm + ptratio + chas + black + dis + nox + zn +
##     crim + rad + tax
##
##         Df Sum of Sq    RSS    AIC
## <none>               8657.6 1264.2
## + indus  1    6.2317 8651.4 1265.9
## + age    1    0.0285 8657.6 1266.2
```

```
summary(m5)
```

```
##
## Call:
## lm(formula = medv ~ lstat + rm + ptratio + chas + black + dis +
##     nox + zn + crim + rad + tax)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -16.1233  -2.7088  -0.6016   1.8937  27.2943
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  33.772638   5.846894   5.776 1.56e-08 ***
## lstat        -0.473518   0.053260  -8.891  < 2e-16 ***
## rm            4.049937   0.483386   8.378 9.63e-16 ***
## ptratio      -0.899458   0.139980  -6.426 3.80e-10 ***
## chas          3.240501   0.956878   3.387 0.000779 ***
## black         0.008115   0.003077   2.638 0.008683 **
## dis          -1.447350   0.209868  -6.896 2.14e-11 ***
## nox         -17.946001   4.012830  -4.472 1.01e-05 ***
## zn            0.049737   0.014578   3.412 0.000712 ***
## crim         -0.117770   0.037788  -3.117 0.001964 **
## rad           0.296828   0.072912   4.071 5.66e-05 ***
## tax          -0.011748   0.003851  -3.051 0.002436 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.694 on 393 degrees of freedom
## Multiple R-squared:  0.7409, Adjusted R-squared:  0.7336
## F-statistic: 102.1 on 11 and 393 DF,  p-value: < 2.2e-16
```

(d) Compare the adjusted R^2, Mallow's Cp, AICs and BICs of the full model(*model1*), the model found in Question 2 (*model3*), and the model found using backward selection with BIC (*model4*). Which model is preferred based on these criteria and why?

```
set.seed(100)
c(Cp(m2,S2=4.694^2),AIC(m2,k=2),AIC(m2,k=log(n)))
```

```
## [1]   11.92619 2415.57370 2467.62424
```

```
c(Cp(m3,S2=4.694^2),AIC(m3,k=2),AIC(m3,k=log(n)))
```

```
## [1]   11.92619 2415.57370 2467.62424
```

```
c(Cp(m4,S2=4.694^2),AIC(m4,k=2),AIC(m4,k=log(n)))
```

```
## [1]   11.92619 2415.57370 2467.62424
```

m1: adujusted r^2: 0.7325 , Mallow's Cp:13.97495, AIC: 2419.28127, BIC: 2479.33957

M2: a-r^2: 0.7336 , cp:11.92619 AIC: 2415.57370 BIC:2440.43000

m3: a-r^2: 0.7336 cp:11.92619 AIC: 2415.57370 BIC:2440.43000

m4: a-r^2: 0.7336 , cp:11.92619 AIC: 2415.57370 BIC:2440.43000

Model 2, 3 ,4 are identical. Model 2/3/4 will be choose because it has larger adjusted r suqred and smaller AIC and BIC.

Question 4: Ridge Regression

(a) Perform ridge regression on the training set. Use cv.glmnet() to find the lambda value that minimizes the cross-validation error using 10 fold CV.

```
set.seed(100)

library(MASS)
library(glmnet)
```

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

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

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

```
#scale  data
#attach(testData)
vari1=trainData[,-c(3,7,14)]
vari=scale(vari1)



sdv=scale(trainData$medv)
ridge=cv.glmnet(vari,sdv,alpha=0,nfolds=10)
ridge
```

```
## $lambda
##   [1] 731.92481794 666.90263444 607.65684251 553.67428344 504.48738613
##   [6] 459.67011721 418.83429094 381.62620692 347.72358654 316.83278150
##  [11] 288.68623045 263.04014141 239.67238024 218.38054656 198.98022070
##  [16] 181.30336632 165.19687497 150.52123991 137.14934782 124.96537777
##  [21] 113.86379804 103.74845205  94.53172552  86.13378757  78.48189929
##  [26]  71.50978367  65.15705158  59.36867870  54.09452892  49.28892007
##  [31]  44.91022826  40.92052736  37.28526049  33.97294072  30.95487830
##  [36]  28.20493223  25.69928379  23.41623025  21.33599689  19.44056574
##  [41]  17.71351946  16.13989921  14.70607505  13.39962787  12.20924187
##  [46]  11.12460647  10.13632709   9.23584373   8.41535683   7.66775972
##  [51]   6.98657708   6.36590883   5.80037904   5.28508935   4.81557658
##  [56]   4.38777403   3.99797628   3.64280708   3.31919014   3.02432244
##  [61]   2.75564997   2.51084562   2.28778902   2.08454815   1.89936263
##  [66]   1.73062849   1.57688423   1.43679818   1.30915699   1.19285509
##  [71]   1.08688513   0.99032925   0.90235113   0.82218874   0.74914776
##  [76]   0.68259554   0.62195563   0.56670281   0.51635849   0.47048663
##  [81]   0.42868989   0.39060626   0.35590588   0.32428818   0.29547931
##  [86]   0.26922975   0.24531212   0.22351927   0.20366243   0.18556962
##  [91]   0.16908413   0.15406316   0.14037661   0.12790594   0.11654313
##  [96]   0.10618976   0.09675615   0.08816060   0.08032866   0.07319248
##
## $cvm
##   [1] 0.9991674 0.9930290 0.9919404 0.9912088 0.9904075 0.9895303 0.9885700
##   [8] 0.9875190 0.9863692 0.9851116 0.9837365 0.9822336 0.9805915 0.9787982
##  [15] 0.9768406 0.9747047 0.9723757 0.9698377 0.9670736 0.9640656 0.9607947
##  [22] 0.9572409 0.9533833 0.9492003 0.9446693 0.9397671 0.9344704 0.9287551
##  [29] 0.9225974 0.9159740 0.9088618 0.9012393 0.8930859 0.8843837 0.8751167
##  [36] 0.8652726 0.8548423 0.8438213 0.8322099 0.8200137 0.8072442 0.7939193
##  [43] 0.7800625 0.7657044 0.7508821 0.7356387 0.7200230 0.7040888 0.6878942
```

```
##  [50] 0.6715005 0.6549714 0.6383717 0.6217663 0.6052191 0.5887921 0.5725446
##  [57] 0.5565324 0.5408074 0.5254168 0.5104048 0.4958107 0.4816696 0.4680127
##  [64] 0.4548671 0.4422559 0.4301987 0.4187112 0.4078052 0.3974878 0.3877647
##  [71] 0.3786351 0.3700955 0.3621392 0.3547519 0.3479188 0.3416207 0.3358355
##  [78] 0.3305354 0.3257007 0.3213020 0.3173049 0.3136846 0.3104029 0.3074455
##  [85] 0.3047783 0.3023719 0.3002063 0.2982498 0.2964929 0.2949168 0.2934954
##  [92] 0.2922178 0.2910675 0.2900371 0.2891125 0.2882852 0.2875464 0.2868882
##  [99] 0.2863027 0.2858149
##
## $cvsd
##   [1] 0.07848297 0.07813938 0.07799472 0.07796091 0.07792390 0.07788338
##   [7] 0.07783905 0.07779056 0.07773752 0.07767954 0.07761618 0.07754696
##  [13] 0.07747138 0.07738889 0.07729892 0.07720082 0.07709394 0.07697757
##  [19] 0.07685096 0.07671332 0.07656382 0.07640158 0.07622571 0.07603526
##  [25] 0.07582926 0.07560674 0.07536670 0.07510813 0.07483004 0.07453145
##  [31] 0.07421142 0.07386904 0.07350349 0.07311401 0.07269990 0.07226062
##  [37] 0.07179568 0.07130477 0.07078765 0.07024424 0.06967454 0.06907870
##  [43] 0.06845684 0.06780927 0.06713626 0.06643807 0.06571496 0.06496713
##  [49] 0.06419469 0.06339773 0.06257626 0.06173029 0.06085987 0.05996517
##  [55] 0.05904656 0.05810471 0.05714072 0.05615610 0.05515331 0.05413514
##  [61] 0.05310526 0.05206806 0.05102865 0.04999276 0.04896665 0.04795694
##  [67] 0.04697038 0.04601369 0.04509315 0.04421500 0.04338486 0.04260517
##  [73] 0.04188028 0.04121095 0.04059811 0.04004103 0.03953786 0.03908529
##  [79] 0.03868136 0.03832130 0.03799892 0.03771135 0.03745155 0.03721581
##  [85] 0.03700130 0.03680177 0.03661565 0.03643765 0.03626648 0.03610010
##  [91] 0.03593601 0.03577366 0.03561276 0.03545200 0.03529174 0.03513070
##  [97] 0.03496974 0.03480907 0.03464983 0.03454017
##
## $cvup
##   [1] 1.0776503 1.0711684 1.0699351 1.0691697 1.0683314 1.0674136 1.0664090
##   [8] 1.0653096 1.0641067 1.0627911 1.0613527 1.0597806 1.0580629 1.0561871
##  [15] 1.0541395 1.0519055 1.0494697 1.0468153 1.0439246 1.0407789 1.0373585
##  [22] 1.0336424 1.0296090 1.0252355 1.0204985 1.0153739 1.0098371 1.0038632
##  [29] 0.9974275 0.9905054 0.9830733 0.9751083 0.9665894 0.9574977 0.9478166
##  [36] 0.9375332 0.9266380 0.9151261 0.9029976 0.8902579 0.8769188 0.8629980
##  [43] 0.8485193 0.8335137 0.8180184 0.8020768 0.7857380 0.7690559 0.7520889
##  [50] 0.7348982 0.7175476 0.7001020 0.6826261 0.6651842 0.6478386 0.6306493
##  [57] 0.6136732 0.5969635 0.5805701 0.5645399 0.5489159 0.5337377 0.5190414
##  [64] 0.5048598 0.4912226 0.4781556 0.4656816 0.4538189 0.4425810 0.4319797
##  [71] 0.4220199 0.4127006 0.4040195 0.3959629 0.3885169 0.3816617 0.3753733
##  [78] 0.3696207 0.3643821 0.3596233 0.3553038 0.3513960 0.3478544 0.3446613
##  [85] 0.3417796 0.3391736 0.3368219 0.3346875 0.3327594 0.3310169 0.3294314
##  [92] 0.3279915 0.3266802 0.3254891 0.3244042 0.3234159 0.3225162 0.3216973
##  [99] 0.3209526 0.3203551
##
## $cvlo
##   [1] 0.9206844 0.9148896 0.9139457 0.9132479 0.9124836 0.9116469 0.9107309
##   [8] 0.9097285 0.9086317 0.9074320 0.9061204 0.9046866 0.9031201 0.9014093
##  [15] 0.8995417 0.8975039 0.8952818 0.8928601 0.8902227 0.8873523 0.8842309
##  [22] 0.8808393 0.8771576 0.8731650 0.8688400 0.8641604 0.8591037 0.8536469
##  [29] 0.8477674 0.8414425 0.8346504 0.8273702 0.8195824 0.8112697 0.8024168
##  [36] 0.7930120 0.7830466 0.7725165 0.7614223 0.7497695 0.7375697 0.7248406
##  [43] 0.7116056 0.6978951 0.6837459 0.6692007 0.6543081 0.6391217 0.6236995
##  [50] 0.6081027 0.5923951 0.5766414 0.5609064 0.5452539 0.5297455 0.5144399
```

```
##  [57] 0.4993917 0.4846513 0.4702635 0.4562696 0.4427054 0.4296016 0.4169841
##  [64] 0.4048743 0.3932892 0.3822418 0.3717408 0.3617915 0.3523947 0.3435497
##  [71] 0.3352502 0.3274903 0.3202590 0.3135410 0.3073207 0.3015797 0.2962976
##  [78] 0.2914501 0.2870194 0.2829807 0.2793060 0.2759733 0.2729513 0.2702297
##  [85] 0.2677770 0.2655701 0.2635906 0.2618122 0.2602264 0.2588167 0.2575594
##  [92] 0.2564441 0.2554547 0.2545851 0.2538207 0.2531545 0.2525767 0.2520792
##  [99] 0.2516529 0.2512748
##
## $nzero
##  s0  s1  s2  s3  s4  s5  s6  s7  s8  s9 s10 s11 s12 s13 s14 s15 s16 s17 s18 s19
##  11  11  11  11  11  11  11  11  11  11  11  11  11  11  11  11  11  11  11  11
## s20 s21 s22 s23 s24 s25 s26 s27 s28 s29 s30 s31 s32 s33 s34 s35 s36 s37 s38 s39
##  11  11  11  11  11  11  11  11  11  11  11  11  11  11  11  11  11  11  11  11
## s40 s41 s42 s43 s44 s45 s46 s47 s48 s49 s50 s51 s52 s53 s54 s55 s56 s57 s58 s59
##  11  11  11  11  11  11  11  11  11  11  11  11  11  11  11  11  11  11  11  11
## s60 s61 s62 s63 s64 s65 s66 s67 s68 s69 s70 s71 s72 s73 s74 s75 s76 s77 s78 s79
##  11  11  11  11  11  11  11  11  11  11  11  11  11  11  11  11  11  11  11  11
## s80 s81 s82 s83 s84 s85 s86 s87 s88 s89 s90 s91 s92 s93 s94 s95 s96 s97 s98 s99
##  11  11  11  11  11  11  11  11  11  11  11  11  11  11  11  11  11  11  11  11
##
## $name
##                 mse
## "Mean-Squared Error"
##
## $glmnet.fit
##
## Call:  glmnet(x = vari, y = sdv, alpha = 0)
##
##        Df      %Dev    Lambda
##   [1,] 11 4.771e-36 731.90000
##   [2,] 11 7.006e-03 666.90000
##   [3,] 11 7.682e-03 607.70000
##   [4,] 11 8.422e-03 553.70000
##   [5,] 11 9.233e-03 504.50000
##   [6,] 11 1.012e-02 459.70000
##   [7,] 11 1.109e-02 418.80000
##   [8,] 11 1.216e-02 381.60000
##   [9,] 11 1.332e-02 347.70000
##  [10,] 11 1.459e-02 316.80000
##  [11,] 11 1.598e-02 288.70000
##  [12,] 11 1.750e-02 263.00000
##  [13,] 11 1.917e-02 239.70000
##  [14,] 11 2.098e-02 218.40000
##  [15,] 11 2.296e-02 199.00000
##  [16,] 11 2.512e-02 181.30000
##  [17,] 11 2.748e-02 165.20000
##  [18,] 11 3.005e-02 150.50000
##  [19,] 11 3.285e-02 137.10000
##  [20,] 11 3.589e-02 125.00000
##  [21,] 11 3.920e-02 113.90000
##  [22,] 11 4.280e-02 103.70000
##  [23,] 11 4.670e-02  94.53000
##  [24,] 11 5.093e-02  86.13000
##  [25,] 11 5.552e-02  78.48000
```

```
##  [26,] 11 6.048e-02  71.51000
##  [27,] 11 6.584e-02  65.16000
##  [28,] 11 7.163e-02  59.37000
##  [29,] 11 7.786e-02  54.09000
##  [30,] 11 8.457e-02  49.29000
##  [31,] 11 9.176e-02  44.91000
##  [32,] 11 9.948e-02  40.92000
##  [33,] 11 1.077e-01  37.29000
##  [34,] 11 1.165e-01  33.97000
##  [35,] 11 1.259e-01  30.95000
##  [36,] 11 1.359e-01  28.20000
##  [37,] 11 1.465e-01  25.70000
##  [38,] 11 1.576e-01  23.42000
##  [39,] 11 1.694e-01  21.34000
##  [40,] 11 1.817e-01  19.44000
##  [41,] 11 1.947e-01  17.71000
##  [42,] 11 2.082e-01  16.14000
##  [43,] 11 2.222e-01  14.71000
##  [44,] 11 2.368e-01  13.40000
##  [45,] 11 2.518e-01  12.21000
##  [46,] 11 2.673e-01  11.12000
##  [47,] 11 2.831e-01  10.14000
##  [48,] 11 2.993e-01   9.23600
##  [49,] 11 3.157e-01   8.41500
##  [50,] 11 3.323e-01   7.66800
##  [51,] 11 3.491e-01   6.98700
##  [52,] 11 3.660e-01   6.36600
##  [53,] 11 3.829e-01   5.80000
##  [54,] 11 3.997e-01   5.28500
##  [55,] 11 4.164e-01   4.81600
##  [56,] 11 4.329e-01   4.38800
##  [57,] 11 4.493e-01   3.99800
##  [58,] 11 4.653e-01   3.64300
##  [59,] 11 4.810e-01   3.31900
##  [60,] 11 4.963e-01   3.02400
##  [61,] 11 5.113e-01   2.75600
##  [62,] 11 5.257e-01   2.51100
##  [63,] 11 5.397e-01   2.28800
##  [64,] 11 5.532e-01   2.08500
##  [65,] 11 5.662e-01   1.89900
##  [66,] 11 5.786e-01   1.73100
##  [67,] 11 5.904e-01   1.57700
##  [68,] 11 6.017e-01   1.43700
##  [69,] 11 6.124e-01   1.30900
##  [70,] 11 6.225e-01   1.19300
##  [71,] 11 6.320e-01   1.08700
##  [72,] 11 6.409e-01   0.99030
##  [73,] 11 6.492e-01   0.90240
##  [74,] 11 6.570e-01   0.82220
##  [75,] 11 6.643e-01   0.74910
##  [76,] 11 6.710e-01   0.68260
##  [77,] 11 6.772e-01   0.62200
##  [78,] 11 6.829e-01   0.56670
##  [79,] 11 6.881e-01   0.51640
```

```
##  [80,] 11 6.929e-01   0.47050
##  [81,] 11 6.974e-01   0.42870
##  [82,] 11 7.014e-01   0.39060
##  [83,] 11 7.051e-01   0.35590
##  [84,] 11 7.084e-01   0.32430
##  [85,] 11 7.115e-01   0.29550
##  [86,] 11 7.143e-01   0.26920
##  [87,] 11 7.169e-01   0.24530
##  [88,] 11 7.192e-01   0.22350
##  [89,] 11 7.214e-01   0.20370
##  [90,] 11 7.233e-01   0.18560
##  [91,] 11 7.251e-01   0.16910
##  [92,] 11 7.267e-01   0.15410
##  [93,] 11 7.281e-01   0.14040
##  [94,] 11 7.295e-01   0.12790
##  [95,] 11 7.307e-01   0.11650
##  [96,] 11 7.318e-01   0.10620
##  [97,] 11 7.328e-01   0.09676
##  [98,] 11 7.337e-01   0.08816
##  [99,] 11 7.345e-01   0.08033
## [100,] 11 7.353e-01   0.07319
##
## $lambda.min
## [1] 0.07319248
##
## $lambda.1se
## [1] 0.4286899
##
## attr(,"class")
## [1] "cv.glmnet"
```

```
#answer=lm.ridge(sdv~vari,lambda=lambda)
```

(b) List the value of coefficients at the optimum lambda value.

```
set.seed(100)
lambda10=cv.glmnet(vari,sdv,alpha=0,nfolds=10)
lambda100=glmnet(vari,sdv,alpha=0,nlambda=100)
coef(lambda100,s=lambda10$lambda.min)
```

```
## 12 x 1 sparse Matrix of class "dgCMatrix"
##                          1
## (Intercept)  2.613461e-16
## crim        -8.894609e-02
## zn           9.926043e-02
## chas         9.226595e-02
## nox         -1.618083e-01
## rm           3.205043e-01
## dis         -2.379269e-01
## rad          1.434241e-01
## tax         -1.132312e-01
## ptratio     -1.997721e-01
## black        7.670491e-02
## lstat       -3.428239e-01
```

(c) How many variables were selected? Give an explanation for this number.

11 varibles are selected. From question 3, model 2, 3 4, 5 are identical, so these 11 varibles are selected : crim, zn,chas,nox,rm, dis, rad, tax, ptraio, black, lstat. Ridge regression it self is not for model selection.

Question 5: Lasso Regression

(a) Perform lasso regression on the training set.Use cv.glmnet() to find the lambda value that minimizes the cross-validation error using 10 fold CV.

```
set.seed(100)
lvari=cbind(trainData$lstat,trainData$rm,trainData$ptratio,trainData$chas,trainData$black,trainData$dis


lasso= cv.glmnet(lvari,trainData$medv,alpha=1, nfolds=10)
lasso
```

```
## $lambda
##  [1] 6.656112477 6.064801790 5.526021515 5.035104994 4.587800145 4.180232625
##  [7] 3.808872280 3.470502564 3.162192682 2.881272200 2.625307919 2.392082799
## [13] 2.179576756 1.985949165 1.809522915 1.648769886 1.502297714 1.368837726
## [19] 1.247233955 1.136433128 1.035475541 0.943486748 0.859669986 0.783299274
## [25] 0.713713125 0.650308818 0.592537175 0.539897806 0.491934773 0.448232643
## [31] 0.408412891 0.372130615 0.339071557 0.308949374 0.281503163 0.256495198
## [37] 0.233708872 0.212946820 0.194029211 0.176792191 0.161086460 0.146775983
## [43] 0.133736809 0.121856000 0.111030648 0.101166992 0.092179595 0.083990614
## [49] 0.076529119 0.069730483 0.063535819 0.057891472 0.052748553 0.048062517
## [55] 0.043792776 0.039902346 0.036357532 0.033127629 0.030184662 0.027503139
## [61] 0.025059836 0.022833590 0.020805116 0.018956847 0.017272773 0.015738308
## [67] 0.014340160 0.013066219 0.011905453 0.010847805 0.009884116 0.009006038
## [73] 0.008205967 0.007476971
##
## $cvm
##  [1] 82.23213 75.32983 68.63494 62.28396 56.95520 52.53100 48.85803 45.80157
##  [9] 43.21228 40.78858 38.53912 36.66370 35.10571 33.81134 32.73592 31.84235
## [17] 31.10608 30.51967 30.06198 29.67556 29.27002 28.85791 28.47038 28.14084
## [25] 27.86390 27.64131 27.46236 27.32083 27.20492 26.98452 26.65325 26.33397
## [33] 26.06540 25.82562 25.60811 25.41792 25.23648 25.09739 24.98552 24.88320
## [41] 24.77146 24.59684 24.41030 24.24712 24.11082 23.99702 23.90217 23.82290
## [49] 23.75665 23.70118 23.65508 23.61617 23.58383 23.55665 23.53375 23.51478
## [57] 23.49861 23.48528 23.47364 23.46403 23.45606 23.44913 23.44362 23.43855
## [65] 23.43450 23.43085 23.42787 23.42539 23.42340 23.42113 23.41961 23.41918
## [73] 23.41891 23.41947
##
## $cvsd
##  [1] 6.646602 6.421729 6.057352 5.565406 5.145785 4.800390 4.515646 4.280698
##  [9] 4.090206 3.900146 3.722789 3.579940 3.466145 3.375902 3.304651 3.248645
## [17] 3.207588 3.187643 3.180173 3.192351 3.202272 3.200884 3.199337 3.199938
## [25] 3.202138 3.206432 3.212555 3.218886 3.226504 3.239037 3.236660 3.206635
## [33] 3.180232 3.155470 3.121102 3.090598 3.068041 3.045116 3.022084 3.002495
## [41] 2.983489 2.964451 2.936453 2.907769 2.881334 2.857110 2.834942 2.814618
## [49] 2.796000 2.778914 2.763380 2.749023 2.735919 2.723980 2.712994 2.703084
## [57] 2.693907 2.685644 2.677916 2.670910 2.664608 2.658810 2.653567 2.648679
## [65] 2.644346 2.640269 2.636562 2.633404 2.630415 2.627513 2.625125 2.623179
## [73] 2.621440 2.620567
##
## $cvup
```

```
##  [1] 88.87873 81.75156 74.69229 67.84937 62.10099 57.33139 53.37368 50.08227
##  [9] 47.30249 44.68872 42.26191 40.24364 38.57185 37.18724 36.04057 35.09100
## [17] 34.31367 33.70731 33.24215 32.86791 32.47229 32.05879 31.66972 31.34078
## [25] 31.06603 30.84774 30.67492 30.53972 30.43142 30.22356 29.88991 29.54061
## [33] 29.24563 28.98109 28.72921 28.50852 28.30452 28.14251 28.00761 27.88569
## [41] 27.75495 27.56129 27.34675 27.15489 26.99215 26.85413 26.73712 26.63752
## [49] 26.55265 26.48010 26.41846 26.36520 26.31975 26.28063 26.24675 26.21786
## [57] 26.19252 26.17092 26.15156 26.13494 26.12067 26.10795 26.09719 26.08723
## [65] 26.07885 26.07112 26.06443 26.05880 26.05381 26.04864 26.04474 26.04236
## [73] 26.04035 26.04004
##
## $cvlo
##  [1] 75.58553 68.90810 62.57758 56.71856 51.80942 47.73061 44.34238 41.52087
##  [9] 39.12208 36.88843 34.81633 33.08376 31.63956 30.43544 29.43127 28.59371
## [17] 27.89849 27.33203 26.88180 26.48321 26.06775 25.65702 25.27104 24.94091
## [25] 24.66176 24.43488 24.24981 24.10195 23.97841 23.74548 23.41659 23.12734
## [33] 22.88516 22.67015 22.48701 22.32733 22.16844 22.05228 21.96344 21.88070
## [41] 21.78797 21.63239 21.47385 21.33935 21.22948 21.13991 21.06723 21.00829
## [49] 20.96065 20.92227 20.89170 20.86715 20.84791 20.83267 20.82076 20.81169
## [57] 20.80470 20.79964 20.79572 20.79312 20.79145 20.79032 20.79006 20.78987
## [65] 20.79015 20.79058 20.79131 20.79199 20.79298 20.79362 20.79449 20.79600
## [73] 20.79747 20.79891
##
## $nzero
##  s0  s1  s2  s3  s4  s5  s6  s7  s8  s9 s10 s11 s12 s13 s14 s15 s16 s17 s18 s19
##   0   2   2   2   2   2   2   2   2   3   3   3   3   3   3   3   3   3   3   5
## s20 s21 s22 s23 s24 s25 s26 s27 s28 s29 s30 s31 s32 s33 s34 s35 s36 s37 s38 s39
##   5   6   6   6   6   6   6   6   7   8   8   8   9   9   9   9   9   9   9  10
## s40 s41 s42 s43 s44 s45 s46 s47 s48 s49 s50 s51 s52 s53 s54 s55 s56 s57 s58 s59
##  11  11  11  11  11  11  11  11  11  11  11  11  11  11  11  11  11  11  11  11
## s60 s61 s62 s63 s64 s65 s66 s67 s68 s69 s70 s71 s72 s73
##  11  11  11  11  11  11  11  11  11  11  11  11  11  11
##
## $name
##                 mse
## "Mean-Squared Error"
##
## $glmnet.fit
##
## Call:  glmnet(x = lvari, y = trainData$medv, alpha = 1)
##
##       Df    %Dev   Lambda
##  [1,]  0 0.00000 6.656000
##  [2,]  2 0.09305 6.065000
##  [3,]  2 0.18560 5.526000
##  [4,]  2 0.26250 5.035000
##  [5,]  2 0.32640 4.588000
##  [6,]  2 0.37940 4.180000
##  [7,]  2 0.42340 3.809000
##  [8,]  2 0.45990 3.471000
##  [9,]  2 0.49020 3.162000
## [10,]  3 0.52120 2.881000
## [11,]  3 0.54800 2.625000
## [12,]  3 0.57030 2.392000
```

```
## [13,]  3 0.58870 2.180000
## [14,]  3 0.60400 1.986000
## [15,]  3 0.61680 1.810000
## [16,]  3 0.62730 1.649000
## [17,]  3 0.63610 1.502000
## [18,]  3 0.64340 1.369000
## [19,]  3 0.64940 1.247000
## [20,]  5 0.65540 1.136000
## [21,]  5 0.66280 1.035000
## [22,]  6 0.66920 0.943500
## [23,]  6 0.67450 0.859700
## [24,]  6 0.67900 0.783300
## [25,]  6 0.68270 0.713700
## [26,]  6 0.68580 0.650300
## [27,]  6 0.68830 0.592500
## [28,]  6 0.69040 0.539900
## [29,]  7 0.69220 0.491900
## [30,]  8 0.69630 0.448200
## [31,]  8 0.70080 0.408400
## [32,]  8 0.70460 0.372100
## [33,]  9 0.70810 0.339100
## [34,]  9 0.71180 0.308900
## [35,]  9 0.71490 0.281500
## [36,]  9 0.71750 0.256500
## [37,]  9 0.71960 0.233700
## [38,]  9 0.72130 0.212900
## [39,]  9 0.72280 0.194000
## [40,] 10 0.72450 0.176800
## [41,] 11 0.72640 0.161100
## [42,] 11 0.72890 0.146800
## [43,] 11 0.73090 0.133700
## [44,] 11 0.73260 0.121900
## [45,] 11 0.73400 0.111000
## [46,] 11 0.73520 0.101200
## [47,] 11 0.73610 0.092180
## [48,] 11 0.73690 0.083990
## [49,] 11 0.73760 0.076530
## [50,] 11 0.73820 0.069730
## [51,] 11 0.73860 0.063540
## [52,] 11 0.73900 0.057890
## [53,] 11 0.73930 0.052750
## [54,] 11 0.73960 0.048060
## [55,] 11 0.73980 0.043790
## [56,] 11 0.74000 0.039900
## [57,] 11 0.74010 0.036360
## [58,] 11 0.74020 0.033130
## [59,] 11 0.74040 0.030180
## [60,] 11 0.74040 0.027500
## [61,] 11 0.74050 0.025060
## [62,] 11 0.74060 0.022830
## [63,] 11 0.74060 0.020810
## [64,] 11 0.74070 0.018960
## [65,] 11 0.74070 0.017270
## [66,] 11 0.74070 0.015740
```

```
## [67,] 11 0.74080 0.014340
## [68,] 11 0.74080 0.013070
## [69,] 11 0.74080 0.011910
## [70,] 11 0.74080 0.010850
## [71,] 11 0.74080 0.009884
## [72,] 11 0.74080 0.009006
## [73,] 11 0.74080 0.008206
## [74,] 11 0.74080 0.007477
##
## $lambda.min
## [1] 0.008205967
##
## $lambda.1se
## [1] 0.3089494
##
## attr(,"class")
## [1] "cv.glmnet"
```

```
lassom=glmnet(lvari,trainData$medv,alpha=1,nlambda=100)
lassom
```

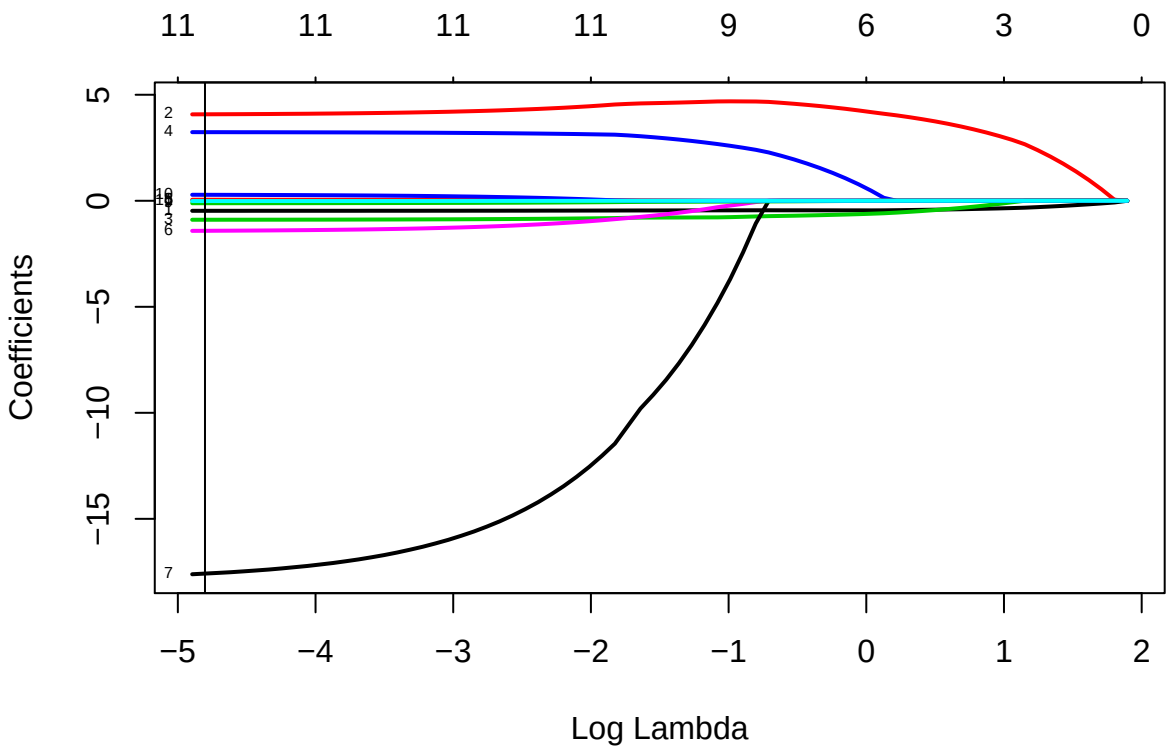
```
##
## Call:  glmnet(x = lvari, y = trainData$medv, alpha = 1, nlambda = 100)
##
##       Df    %Dev   Lambda
##  [1,]  0 0.00000 6.656000
##  [2,]  2 0.09305 6.065000
##  [3,]  2 0.18560 5.526000
##  [4,]  2 0.26250 5.035000
##  [5,]  2 0.32640 4.588000
##  [6,]  2 0.37940 4.180000
##  [7,]  2 0.42340 3.809000
##  [8,]  2 0.45990 3.471000
##  [9,]  2 0.49020 3.162000
## [10,]  3 0.52120 2.881000
## [11,]  3 0.54800 2.625000
## [12,]  3 0.57030 2.392000
## [13,]  3 0.58870 2.180000
## [14,]  3 0.60400 1.986000
## [15,]  3 0.61680 1.810000
## [16,]  3 0.62730 1.649000
## [17,]  3 0.63610 1.502000
## [18,]  3 0.64340 1.369000
## [19,]  3 0.64940 1.247000
## [20,]  5 0.65540 1.136000
## [21,]  5 0.66280 1.035000
## [22,]  6 0.66920 0.943500
## [23,]  6 0.67450 0.859700
## [24,]  6 0.67900 0.783300
## [25,]  6 0.68270 0.713700
## [26,]  6 0.68580 0.650300
## [27,]  6 0.68830 0.592500
## [28,]  6 0.69040 0.539900
## [29,]  7 0.69220 0.491900
## [30,]  8 0.69630 0.448200
```

```
## [31,]  8 0.70080 0.408400
## [32,]  8 0.70460 0.372100
## [33,]  9 0.70810 0.339100
## [34,]  9 0.71180 0.308900
## [35,]  9 0.71490 0.281500
## [36,]  9 0.71750 0.256500
## [37,]  9 0.71960 0.233700
## [38,]  9 0.72130 0.212900
## [39,]  9 0.72280 0.194000
## [40,] 10 0.72450 0.176800
## [41,] 11 0.72640 0.161100
## [42,] 11 0.72890 0.146800
## [43,] 11 0.73090 0.133700
## [44,] 11 0.73260 0.121900
## [45,] 11 0.73400 0.111000
## [46,] 11 0.73520 0.101200
## [47,] 11 0.73610 0.092180
## [48,] 11 0.73690 0.083990
## [49,] 11 0.73760 0.076530
## [50,] 11 0.73820 0.069730
## [51,] 11 0.73860 0.063540
## [52,] 11 0.73900 0.057890
## [53,] 11 0.73930 0.052750
## [54,] 11 0.73960 0.048060
## [55,] 11 0.73980 0.043790
## [56,] 11 0.74000 0.039900
## [57,] 11 0.74010 0.036360
## [58,] 11 0.74020 0.033130
## [59,] 11 0.74040 0.030180
## [60,] 11 0.74040 0.027500
## [61,] 11 0.74050 0.025060
## [62,] 11 0.74060 0.022830
## [63,] 11 0.74060 0.020810
## [64,] 11 0.74070 0.018960
## [65,] 11 0.74070 0.017270
## [66,] 11 0.74070 0.015740
## [67,] 11 0.74080 0.014340
## [68,] 11 0.74080 0.013070
## [69,] 11 0.74080 0.011910
## [70,] 11 0.74080 0.010850
## [71,] 11 0.74080 0.009884
## [72,] 11 0.74080 0.009006
## [73,] 11 0.74080 0.008206
## [74,] 11 0.74080 0.007477
```

(b) Plot the regression coefficient path.

```
set.seed(100)

plot(lassom,xvar="lambda",lwd=2,label=T)
abline(v=log(lasso$lambda.min),col="black")
```



(c) How many variables were selected? Which are they?

```
set.seed(100)
coef(lassom,s=lasso$lambda.min)
```

```
## 12 x 1 sparse Matrix of class "dgCMatrix"
##                          1
## (Intercept)  33.091624028
## V1           -0.472760296
## V2            4.078342939
## V3           -0.894486502
## V4            3.235918045
## V5            0.008038558
## V6           -1.415833221
## V7          -17.581885229
## V8            0.048368313
## V9           -0.114638457
## V10           0.281318072
## V11          -0.011110876
```

11 varibles are selected. crim+zn+chas+nox+rm+dis+rad+tax+ptratio+black+lstat

Question 6: Elastic Net

(a) Perform elastic net regression on the training set. Use cv.glmnet() to find the lambda value that minimizes the cross-validation error using 10 fold CV. Give equal weight to both penalties.

```
set.seed(100)

ecv=cv.glmnet(lvari, trainData$medv,alpha=0.5,nfolds=10)
em=glmnet(lvari, trainData$medv,alpha=0.5,nlambda=100)
ecv
```

```
## $lambda
##  [1] 13.31222495 12.12960358 11.05204303 10.07020999  9.17560029  8.36046525
##  [7]  7.61774456  6.94100513  6.32438536  5.76254440  5.25061584  4.78416560
## [13]  4.35915351  3.97189833  3.61904583  3.29753977  3.00459543  2.73767545
## [19]  2.49446791  2.27286626  2.07095108  1.88697350  1.71933997  1.56659855
## [25]  1.42742625  1.30061764  1.18507435  1.07979561  0.98386955  0.89646529
## [31]  0.81682578  0.74426123  0.67814311  0.61789875  0.56300633  0.51299040
## [37]  0.46741774  0.42589364  0.38805842  0.35358438  0.32217292  0.29355197
## [43]  0.26747362  0.24371200  0.22206130  0.20233398  0.18435919  0.16798123
## [49]  0.15305824  0.13946097  0.12707164  0.11578294  0.10549711  0.09612503
## [55]  0.08758555  0.07980469  0.07271506  0.06625526  0.06036932  0.05500628
## [61]  0.05011967  0.04566718  0.04161023  0.03791369  0.03454555  0.03147662
## [67]  0.02868032  0.02613244  0.02381091  0.02169561  0.01976823  0.01801208
## [73]  0.01641193  0.01495394
##
## $cvm
##  [1] 82.35882 77.68616 71.90774 66.60605 61.87753 57.68414 53.95887 50.51962
##  [9] 47.18439 44.25892 41.73375 39.56121 37.70034 36.12277 34.77872 33.63404
## [17] 32.67012 31.89216 31.20962 30.56050 29.95479 29.39459 28.91245 28.51128
## [25] 28.17841 27.90252 27.67437 27.48463 27.32076 27.08719 26.76918 26.46161
## [33] 26.18540 25.93142 25.70524 25.51337 25.32821 25.17572 25.05636 24.95015
## [41] 24.83458 24.66440 24.48070 24.31977 24.18200 24.06408 23.96344 23.87734
## [49] 23.80417 23.74251 23.69022 23.64574 23.60817 23.57686 23.55010 23.52763
## [57] 23.50888 23.49295 23.47956 23.46847 23.45931 23.45120 23.44466 23.43930
## [65] 23.43442 23.43086 23.42760 23.42471 23.42284 23.42079 23.41951 23.41914
## [73] 23.41652 23.41804
##
## $cvsd
##  [1] 6.592421 6.497775 6.098167 5.709053 5.363470 5.059827 4.798263 4.562912
##  [9] 4.308031 4.086388 3.900136 3.744715 3.616085 3.511815 3.426780 3.357765
## [17] 3.304238 3.275087 3.260300 3.254875 3.252564 3.243719 3.232550 3.224688
## [25] 3.220442 3.219390 3.220101 3.221777 3.226537 3.235415 3.229525 3.204482
## [33] 3.182259 3.158279 3.130175 3.103988 3.084207 3.064950 3.044399 3.024841
## [41] 3.007103 2.990591 2.962826 2.935756 2.910228 2.886434 2.864262 2.843587
## [49] 2.824347 2.806504 2.789790 2.774401 2.760078 2.746877 2.734676 2.723307
## [57] 2.712914 2.703282 2.694447 2.686411 2.678860 2.671955 2.665749 2.659919
## [65] 2.654614 2.649779 2.645310 2.641279 2.637634 2.634158 2.631311 2.628970
## [73] 2.625866 2.624465
##
## $cvup
##  [1] 88.95124 84.18393 78.00591 72.31510 67.24100 62.74397 58.75714 55.08253
##  [9] 51.49242 48.34531 45.63389 43.30592 41.31643 39.63458 38.20550 36.99181
## [17] 35.97436 35.16724 34.46992 33.81538 33.20735 32.63831 32.14500 31.73597
## [25] 31.39885 31.12191 30.89447 30.70641 30.54730 30.32261 29.99871 29.66609
## [33] 29.36766 29.08970 28.83541 28.61736 28.41241 28.24067 28.10076 27.97499
## [41] 27.84168 27.65499 27.44353 27.25552 27.09223 26.95052 26.82770 26.72093
## [49] 26.62851 26.54901 26.48001 26.42014 26.36824 26.32374 26.28478 26.25094
## [57] 26.22179 26.19623 26.17400 26.15488 26.13817 26.12316 26.11041 26.09922
## [65] 26.08904 26.08064 26.07291 26.06598 26.06048 26.05495 26.05082 26.04811
## [73] 26.04239 26.04250
##
## $cvlo
##  [1] 75.76640 71.18838 65.80958 60.89699 56.51406 52.62431 49.16061 45.95671
##  [9] 42.87636 40.17253 37.83362 35.81649 34.08425 32.61095 31.35194 30.27628
```

```
## [17] 29.36588 28.61707 27.94932 27.30562 26.70223 26.15087 25.67990 25.28660
## [25] 24.95797 24.68313 24.45427 24.26285 24.09422 23.85178 23.53966 23.25712
## [33] 23.00315 22.77314 22.57506 22.40938 22.24400 22.11077 22.01196 21.92531
## [41] 21.82748 21.67381 21.51787 21.38401 21.27178 21.17765 21.09917 21.03376
## [49] 20.97982 20.93600 20.90043 20.87134 20.84809 20.82999 20.81542 20.80432
## [57] 20.79596 20.78967 20.78511 20.78206 20.78045 20.77925 20.77891 20.77938
## [65] 20.77981 20.78108 20.78229 20.78343 20.78521 20.78663 20.78820 20.79017
## [73] 20.79066 20.79357
##
## $nzero
##  s0  s1  s2  s3  s4  s5  s6  s7  s8  s9 s10 s11 s12 s13 s14 s15 s16 s17 s18 s19
##   0   2   2   2   2   2   2   3   3   3   3   3   3   3   3   4   4   4   6   7
## s20 s21 s22 s23 s24 s25 s26 s27 s28 s29 s30 s31 s32 s33 s34 s35 s36 s37 s38 s39
##   7   7   7   7   8   8   8   8   8   9   8   8   9   9   9   9   9   9   9  10
## s40 s41 s42 s43 s44 s45 s46 s47 s48 s49 s50 s51 s52 s53 s54 s55 s56 s57 s58 s59
##  11  11  11  11  11  11  11  11  11  11  11  11  11  11  11  11  11  11  11  11
## s60 s61 s62 s63 s64 s65 s66 s67 s68 s69 s70 s71 s72 s73
##  11  11  11  11  11  11  11  11  11  11  11  11  11  11
##
## $name
##                 mse
## "Mean-Squared Error"
##
## $glmnet.fit
##
## Call:  glmnet(x = lvari, y = trainData$medv, alpha = 0.5)
##
##       Df    %Dev   Lambda
##  [1,]  0 0.00000 13.31000
##  [2,]  2 0.06671 12.13000
##  [3,]  2 0.13830 11.05000
##  [4,]  2 0.20250 10.07000
##  [5,]  2 0.25960  9.17600
##  [6,]  2 0.31030  8.36000
##  [7,]  2 0.35510  7.61800
##  [8,]  3 0.39790  6.94100
##  [9,]  3 0.43860  6.32400
## [10,]  3 0.47390  5.76300
## [11,]  3 0.50440  5.25100
## [12,]  3 0.53060  4.78400
## [13,]  3 0.55320  4.35900
## [14,]  3 0.57250  3.97200
## [15,]  3 0.58890  3.61900
## [16,]  4 0.60310  3.29800
## [17,]  4 0.61550  3.00500
## [18,]  4 0.62590  2.73800
## [19,]  6 0.63550  2.49400
## [20,]  7 0.64540  2.27300
## [21,]  7 0.65430  2.07100
## [22,]  7 0.66180  1.88700
## [23,]  7 0.66810  1.71900
## [24,]  7 0.67340  1.56700
## [25,]  8 0.67790  1.42700
## [26,]  8 0.68170  1.30100
```

```
## [27,]  8 0.68500  1.18500
## [28,]  8 0.68770  1.08000
## [29,]  8 0.69000  0.98390
## [30,]  9 0.69360  0.89650
## [31,]  8 0.69810  0.81680
## [32,]  8 0.70190  0.74430
## [33,]  9 0.70580  0.67810
## [34,]  9 0.70940  0.61790
## [35,]  9 0.71250  0.56300
## [36,]  9 0.71520  0.51300
## [37,]  9 0.71740  0.46740
## [38,]  9 0.71940  0.42590
## [39,]  9 0.72100  0.38810
## [40,] 10 0.72240  0.35360
## [41,] 11 0.72480  0.32220
## [42,] 11 0.72710  0.29360
## [43,] 11 0.72920  0.26750
## [44,] 11 0.73100  0.24370
## [45,] 11 0.73250  0.22210
## [46,] 11 0.73380  0.20230
## [47,] 11 0.73490  0.18440
## [48,] 11 0.73580  0.16800
## [49,] 11 0.73660  0.15310
## [50,] 11 0.73730  0.13950
## [51,] 11 0.73790  0.12710
## [52,] 11 0.73830  0.11580
## [53,] 11 0.73870  0.10550
## [54,] 11 0.73910  0.09613
## [55,] 11 0.73940  0.08759
## [56,] 11 0.73960  0.07980
## [57,] 11 0.73980  0.07272
## [58,] 11 0.74000  0.06626
## [59,] 11 0.74010  0.06037
## [60,] 11 0.74030  0.05501
## [61,] 11 0.74040  0.05012
## [62,] 11 0.74040  0.04567
## [63,] 11 0.74050  0.04161
## [64,] 11 0.74060  0.03791
## [65,] 11 0.74060  0.03455
## [66,] 11 0.74070  0.03148
## [67,] 11 0.74070  0.02868
## [68,] 11 0.74070  0.02613
## [69,] 11 0.74080  0.02381
## [70,] 11 0.74080  0.02170
## [71,] 11 0.74080  0.01977
## [72,] 11 0.74080  0.01801
## [73,] 11 0.74080  0.01641
## [74,] 11 0.74080  0.01495
##
## $lambda.min
## [1] 0.01641193
##
## $lambda.1se
## [1] 0.6178987
```

```
##
## attr(,"class")
## [1] "cv.glmnet"
```

```
em
```

```
##
## Call:  glmnet(x = lvari, y = trainData$medv, alpha = 0.5, nlambda = 100)
##
##       Df    %Dev   Lambda
##  [1,]  0 0.00000 13.31000
##  [2,]  2 0.06671 12.13000
##  [3,]  2 0.13830 11.05000
##  [4,]  2 0.20250 10.07000
##  [5,]  2 0.25960  9.17600
##  [6,]  2 0.31030  8.36000
##  [7,]  2 0.35510  7.61800
##  [8,]  3 0.39790  6.94100
##  [9,]  3 0.43860  6.32400
## [10,]  3 0.47390  5.76300
## [11,]  3 0.50440  5.25100
## [12,]  3 0.53060  4.78400
## [13,]  3 0.55320  4.35900
## [14,]  3 0.57250  3.97200
## [15,]  3 0.58890  3.61900
## [16,]  4 0.60310  3.29800
## [17,]  4 0.61550  3.00500
## [18,]  4 0.62590  2.73800
## [19,]  6 0.63550  2.49400
## [20,]  7 0.64540  2.27300
## [21,]  7 0.65430  2.07100
## [22,]  7 0.66180  1.88700
## [23,]  7 0.66810  1.71900
## [24,]  7 0.67340  1.56700
## [25,]  8 0.67790  1.42700
## [26,]  8 0.68170  1.30100
## [27,]  8 0.68500  1.18500
## [28,]  8 0.68770  1.08000
## [29,]  8 0.69000  0.98390
## [30,]  9 0.69360  0.89650
## [31,]  8 0.69810  0.81680
## [32,]  8 0.70190  0.74430
## [33,]  9 0.70580  0.67810
## [34,]  9 0.70940  0.61790
## [35,]  9 0.71250  0.56300
## [36,]  9 0.71520  0.51300
## [37,]  9 0.71740  0.46740
## [38,]  9 0.71940  0.42590
## [39,]  9 0.72100  0.38810
## [40,] 10 0.72240  0.35360
## [41,] 11 0.72480  0.32220
## [42,] 11 0.72710  0.29360
## [43,] 11 0.72920  0.26750
## [44,] 11 0.73100  0.24370
## [45,] 11 0.73250  0.22210
```

```
## [46,] 11 0.73380  0.20230
## [47,] 11 0.73490  0.18440
## [48,] 11 0.73580  0.16800
## [49,] 11 0.73660  0.15310
## [50,] 11 0.73730  0.13950
## [51,] 11 0.73790  0.12710
## [52,] 11 0.73830  0.11580
## [53,] 11 0.73870  0.10550
## [54,] 11 0.73910  0.09613
## [55,] 11 0.73940  0.08759
## [56,] 11 0.73960  0.07980
## [57,] 11 0.73980  0.07272
## [58,] 11 0.74000  0.06626
## [59,] 11 0.74010  0.06037
## [60,] 11 0.74030  0.05501
## [61,] 11 0.74040  0.05012
## [62,] 11 0.74040  0.04567
## [63,] 11 0.74050  0.04161
## [64,] 11 0.74060  0.03791
## [65,] 11 0.74060  0.03455
## [66,] 11 0.74070  0.03148
## [67,] 11 0.74070  0.02868
## [68,] 11 0.74070  0.02613
## [69,] 11 0.74080  0.02381
## [70,] 11 0.74080  0.02170
## [71,] 11 0.74080  0.01977
## [72,] 11 0.74080  0.01801
## [73,] 11 0.74080  0.01641
## [74,] 11 0.74080  0.01495
```

(b) List the coefficient values at the optimal lambda. How many variables were selected? How do these variables compare to those from Lasso in Question 5?

```
set.seed(100)

coef(em,s=ecv$lambda.min)
```

```
## 12 x 1 sparse Matrix of class "dgCMatrix"
##                            1
## (Intercept)  32.953461538
## V1           -0.472434473
## V2            4.083219031
## V3           -0.894112852
## V4            3.238734815
## V5            0.008046398
## V6           -1.408555738
## V7          -17.509221899
## V8            0.048094591
## V9           -0.114259874
## V10           0.278462305
## V11          -0.010976811
```

11 varibles are selected, they are identical to question 5.

Question 7: Model comparison

(a) Predict *medv* for each of the rows in the test data using the full model, and the models found using backward stepwise regression with BIC, ridge regression, lasso regression, and elastic net.

```
set.seed(100)
m1p=predict(m1,testData[,-14],interval="prediction")
m1p
```

```
##           fit        lwr      upr
## 202 29.166871 19.7351632 38.59858
## 503 22.075833 12.7198498 31.43182
## 358 22.951092 13.4023333 32.49985
## 112 26.264912 16.9510267 35.57880
## 499 21.023442 11.7163298 30.33055
## 473 22.454064 13.0966259 31.81150
## 206 22.503593 13.1434844 31.86370
## 492 13.880188  4.1633529 23.59702
## 407  7.579798 -1.9173901 17.07699
## 470 18.337630  8.9742002 27.70106
## 454 22.687526 13.2773377 32.09771
## 4   28.327498 18.9883379 37.66666
## 311 18.064771  8.6526843 27.47686
## 326 24.415240 15.0394615 33.79102
## 98  35.661512 26.2544130 45.06861
## 391 16.950690  7.6150095 26.28637
## 7   22.784342 13.4550991 32.11358
## 183 33.470721 24.0924049 42.84904
## 299 29.018160 19.6475480 38.38877
## 466 17.740230  8.3608803 27.11958
## 445 11.709863  2.3897577 21.02997
## 396 20.228480 10.9018552 29.55511
## 483 28.414381 18.9913035 37.83746
## 307 35.491829 26.0944134 44.88924
## 456 16.265452  6.8098750 25.72103
## 146 12.405441  2.7868205 22.02406
## 281 38.645691 29.2764291 48.01495
## 258 43.031651 33.5126103 52.55069
## 435 16.014541  6.6255763 25.40351
## 324 19.231448  9.9263148 28.53658
## 68  20.953309 11.6225764 30.28404
## 436 13.917065  4.5039744 23.33016
## 48  18.258469  8.8858629 27.63107
## 288 27.018212 17.6912056 36.34522
## 341 21.169318 11.8363959 30.50224
## 347 14.826354  5.4020312 24.25068
## 167 36.896156 27.4046922 46.38762
## 377 17.991903  8.6340431 27.34976
## 272 26.987064 17.6302465 36.34388
## 459 17.220446  7.8957843 26.54511
## 116 20.271004 10.9462488 29.59576
## 450 17.363339  8.0436033 26.68308
## 496 16.812541  7.3767742 26.24831
## 93  28.905672 19.5370883 38.27426
## 301 30.817220 21.4346461 40.19979
```

```
## 158 32.733551 23.2941497 42.17295
## 461 19.229072  9.8815861 28.57656
## 221 33.446264 23.9815273 42.91100
## 87  22.054993 12.7464530 31.36353
## 95  27.040632 17.6534048 36.42786
## 353 16.999584  7.5143900 26.48478
## 223 32.426121 22.9758650 41.87638
## 413  2.379576 -7.1923444 11.95150
## 220 29.904481 20.4404789 39.36848
## 251 24.186626 14.8456716 33.52758
## 31  11.782785  2.4301558 21.13541
## 182 27.328613 17.9713166 36.68591
## 425 14.879788  5.3469002 24.41268
## 297 27.237019 17.8149175 36.65912
## 408 19.468832 10.0846555 28.85301
## 171 22.337147 12.9326545 31.74164
## 490  8.528255 -1.1803728 18.23688
## 191 30.653459 21.3024148 40.00450
## 393  9.816725  0.4597161 19.17373
## 148  8.403780 -1.1799992 17.98756
## 398 16.205968  6.8778739 25.53406
## 363 17.531133  8.0674913 26.99478
## 334 21.911883 12.5246932 31.29907
## 88  25.542861 16.2353638 34.85036
## 387  5.943988 -3.4798052 15.36778
## 366 12.992582  3.1047341 22.88043
## 420 15.272923  5.7824049 24.76344
## 371 34.508905 24.9351970 44.08261
## 434 17.286804  7.8828325 26.69078
## 411 14.724212  4.6956089 24.75281
## 430 13.681104  4.2519260 23.11028
## 431 18.543096  9.1153521 27.97084
## 254 30.368341 20.7577427 39.97894
## 47  20.373674 11.0381318 29.70922
## 196 40.918309 31.5112751 50.32534
## 12  21.424092 12.0396598 30.80852
## 121 21.975597 12.3849169 31.56628
## 16  18.981132  9.6527154 28.30955
## 406  7.668637 -2.5104558 17.84773
## 131 20.043162 10.6405652 29.44576
## 133 19.993626 10.5888722 29.39838
## 44  24.409025 15.0193507 33.79870
## 472 22.869576 13.4574273 32.28172
## 156 20.457199 10.6609367 30.25346
## 245 16.376548  6.9589261 25.79417
## 480 21.568047 12.2285469 30.90755
## 42  27.857289 18.4441665 37.27041
## 143 14.939001  5.2477066 24.63029
## 185 22.354333 12.9534501 31.75522
## 298 19.630291 10.1833937 29.07719
## 332 19.984130 10.6632365 29.30502
## 154 16.863001  7.3454572 26.38054
## 359 22.277289 12.7119245 31.84265
## 280 34.932373 25.5785140 44.28623
```

```
## 485 19.363403  9.9670144 28.75979
## 137 15.976245  6.6016587 25.35083
```

```
predata=testData[,-c(3,7,14)]
BICp=predict(m4,predata, interval = "prediction")
BICp
```

```
##           fit        lwr      upr
## 202 29.265590 19.8665655 38.66461
## 503 22.081281 12.7459574 31.41660
## 358 23.037509 13.5171484 32.55787
## 112 26.359023 17.0740203 35.64403
## 499 21.156651 11.8937845 30.41952
## 473 22.343932 13.0159701 31.67189
## 206 22.405610 13.1397267 31.67149
## 492 13.722625  4.0470626 23.39819
## 407  7.643015 -1.8190723 17.10510
## 470 18.236815  8.9099220 27.56371
## 454 22.651113 13.2626792 32.03955
## 4   28.389660 19.0741414 37.70518
## 311 18.117981  8.7664886 27.46947
## 326 24.381362 15.0925635 33.67016
## 98  35.780822 26.4093927 45.15225
## 391 16.968967  7.6567339 26.28120
## 7   22.763629 13.4610355 32.06622
## 183 33.601476 24.2881081 42.91484
## 299 29.036705 19.6870558 38.38635
## 466 17.684159  8.3650626 27.00326
## 445 11.767048  2.4715142 21.06258
## 396 20.223814 10.9184349 29.52919
## 483 28.229597 18.8622447 37.59695
## 307 35.576345 26.2098086 44.94288
## 456 16.276816  6.8442246 25.70941
## 146 12.400637  2.8334992 21.96777
## 281 38.597687 29.2603926 47.93498
## 258 43.155917 33.6690277 52.64281
## 435 16.028248  6.6629985 25.39350
## 324 19.273715 10.0015587 28.54587
## 68  20.962442 11.6662361 30.25865
## 436 13.953392  4.5674589 23.33933
## 48  18.230178  8.9045437 27.55581
## 288 26.914317 17.6169214 36.21171
## 341 21.233616 11.9243879 30.54284
## 347 14.925943  5.5316972 24.32019
## 167 36.601829 27.2118439 45.99181
## 377 17.983838  8.6516564 27.31602
## 272 26.923688 17.6438273 36.20355
## 459 17.209086  7.9066063 26.51157
## 116 20.410774 11.1258329 29.69571
## 450 17.372421  8.0733257 26.67152
## 496 16.941205  7.6767184 26.20569
## 93  28.603983 19.3226768 37.88529
## 301 30.832072 21.4946982 40.16945
## 158 32.476189 23.1377850 41.81459
## 461 19.216186  9.8899531 28.54242
```

```
## 221 33.579709 24.1589592 43.00046
## 87  22.134139 12.8608652 31.40741
## 95  26.760597 17.4752517 36.04594
## 353 17.094594  7.6369386 26.55225
## 223 32.548053 23.1309553 41.96515
## 413  2.427053 -7.1199633 11.97407
## 220 29.785531 20.3743652 39.19670
## 251 24.125227 14.8325005 33.41795
## 31  11.908442  2.5939682 21.22292
## 182 27.488414 18.1710273 36.80580
## 425 14.842723  5.3319433 24.35350
## 297 26.909582 17.5892760 36.22989
## 408 19.472387 10.1300392 28.81474
## 171 22.122042 12.7901560 31.45393
## 490  8.404663 -1.2715567 18.08088
## 191 30.716840 21.4001408 40.03354
## 393  9.872073  0.5379207 19.20623
## 148  8.421760 -1.1103628 17.95388
## 398 16.231527  6.9255160 25.53754
## 363 17.587697  8.1547859 27.02061
## 334 21.912678 12.5540560 31.27130
## 88  25.630870 16.3495655 34.91218
## 387  6.014684 -3.3846058 15.41397
## 366 13.075356  3.2440045 22.90671
## 420 15.292520  5.8593693 24.72567
## 371 34.503659 24.9689574 44.03836
## 434 17.296694  7.9138542 26.67953
## 411 14.671489  4.7237737 24.61920
## 430 13.698462  4.2907535 23.10617
## 431 18.485059  9.0825475 27.88757
## 254 30.211822 20.6854692 39.73818
## 47  20.334914 11.0448833 29.62495
## 196 40.926950 31.5414517 50.31245
## 12  21.396612 12.0719552 30.72127
## 121 21.402149 12.0718199 30.73248
## 16  19.051829  9.7493502 28.35431
## 406  7.662961 -2.4923494 17.81827
## 131 19.846842 10.4984893 29.19519
## 133 19.792255 10.4462595 29.13825
## 44  24.318975 15.0323500 33.60560
## 472 22.735557 13.3859271 32.08519
## 156 20.484325 10.7407208 30.22793
## 245 16.370658  7.0481710 25.69315
## 480 21.502111 12.1910490 30.81317
## 42  27.744355 18.4528534 37.03586
## 143 15.008734  5.3705602 24.64691
## 185 22.542398 13.2124074 31.87239
## 298 19.336572  9.9773884 28.69576
## 332 19.982664 10.6903588 29.27497
## 154 16.836806  7.3428625 26.33075
## 359 22.358881 12.8228845 31.89488
## 280 34.916201 25.5929498 44.23945
## 485 19.239575  9.9038875 28.57526
## 137 15.806171  6.4736416 25.13870
```

```
ridgep=predict(m4,predata,interval = "prediction")
ridgep
```

```
##           fit        lwr      upr
## 202 29.265590 19.8665655 38.66461
## 503 22.081281 12.7459574 31.41660
## 358 23.037509 13.5171484 32.55787
## 112 26.359023 17.0740203 35.64403
## 499 21.156651 11.8937845 30.41952
## 473 22.343932 13.0159701 31.67189
## 206 22.405610 13.1397267 31.67149
## 492 13.722625  4.0470626 23.39819
## 407  7.643015 -1.8190723 17.10510
## 470 18.236815  8.9099220 27.56371
## 454 22.651113 13.2626792 32.03955
## 4   28.389660 19.0741414 37.70518
## 311 18.117981  8.7664886 27.46947
## 326 24.381362 15.0925635 33.67016
## 98  35.780822 26.4093927 45.15225
## 391 16.968967  7.6567339 26.28120
## 7   22.763629 13.4610355 32.06622
## 183 33.601476 24.2881081 42.91484
## 299 29.036705 19.6870558 38.38635
## 466 17.684159  8.3650626 27.00326
## 445 11.767048  2.4715142 21.06258
## 396 20.223814 10.9184349 29.52919
## 483 28.229597 18.8622447 37.59695
## 307 35.576345 26.2098086 44.94288
## 456 16.276816  6.8442246 25.70941
## 146 12.400637  2.8334992 21.96777
## 281 38.597687 29.2603926 47.93498
## 258 43.155917 33.6690277 52.64281
## 435 16.028248  6.6629985 25.39350
## 324 19.273715 10.0015587 28.54587
## 68  20.962442 11.6662361 30.25865
## 436 13.953392  4.5674589 23.33933
## 48  18.230178  8.9045437 27.55581
## 288 26.914317 17.6169214 36.21171
## 341 21.233616 11.9243879 30.54284
## 347 14.925943  5.5316972 24.32019
## 167 36.601829 27.2118439 45.99181
## 377 17.983838  8.6516564 27.31602
## 272 26.923688 17.6438273 36.20355
## 459 17.209086  7.9066063 26.51157
## 116 20.410774 11.1258329 29.69571
## 450 17.372421  8.0733257 26.67152
## 496 16.941205  7.6767184 26.20569
## 93  28.603983 19.3226768 37.88529
## 301 30.832072 21.4946982 40.16945
## 158 32.476189 23.1377850 41.81459
## 461 19.216186  9.8899531 28.54242
## 221 33.579709 24.1589592 43.00046
## 87  22.134139 12.8608652 31.40741
## 95  26.760597 17.4752517 36.04594
```

```
## 353 17.094594  7.6369386 26.55225
## 223 32.548053 23.1309553 41.96515
## 413  2.427053 -7.1199633 11.97407
## 220 29.785531 20.3743652 39.19670
## 251 24.125227 14.8325005 33.41795
## 31  11.908442  2.5939682 21.22292
## 182 27.488414 18.1710273 36.80580
## 425 14.842723  5.3319433 24.35350
## 297 26.909582 17.5892760 36.22989
## 408 19.472387 10.1300392 28.81474
## 171 22.122042 12.7901560 31.45393
## 490  8.404663 -1.2715567 18.08088
## 191 30.716840 21.4001408 40.03354
## 393  9.872073  0.5379207 19.20623
## 148  8.421760 -1.1103628 17.95388
## 398 16.231527  6.9255160 25.53754
## 363 17.587697  8.1547859 27.02061
## 334 21.912678 12.5540560 31.27130
## 88  25.630870 16.3495655 34.91218
## 387  6.014684 -3.3846058 15.41397
## 366 13.075356  3.2440045 22.90671
## 420 15.292520  5.8593693 24.72567
## 371 34.503659 24.9689574 44.03836
## 434 17.296694  7.9138542 26.67953
## 411 14.671489  4.7237737 24.61920
## 430 13.698462  4.2907535 23.10617
## 431 18.485059  9.0825475 27.88757
## 254 30.211822 20.6854692 39.73818
## 47  20.334914 11.0448833 29.62495
## 196 40.926950 31.5414517 50.31245
## 12  21.396612 12.0719552 30.72127
## 121 21.402149 12.0718199 30.73248
## 16  19.051829  9.7493502 28.35431
## 406  7.662961 -2.4923494 17.81827
## 131 19.846842 10.4984893 29.19519
## 133 19.792255 10.4462595 29.13825
## 44  24.318975 15.0323500 33.60560
## 472 22.735557 13.3859271 32.08519
## 156 20.484325 10.7407208 30.22793
## 245 16.370658  7.0481710 25.69315
## 480 21.502111 12.1910490 30.81317
## 42  27.744355 18.4528534 37.03586
## 143 15.008734  5.3705602 24.64691
## 185 22.542398 13.2124074 31.87239
## 298 19.336572  9.9773884 28.69576
## 332 19.982664 10.6903588 29.27497
## 154 16.836806  7.3428625 26.33075
## 359 22.358881 12.8228845 31.89488
## 280 34.916201 25.5929498 44.23945
## 485 19.239575  9.9038875 28.57526
## 137 15.806171  6.4736416 25.13870
```

```
mla=lm(medv~ crim+zn+chas+nox+rm+dis+rad+tax+ptratio+black+lstat, testData)
```

```r
lassp=predict(mla,predata,interval="prediction" )
lassp
```

```
##           fit         lwr       upr
## 202 28.175393  17.2930686 39.057716
## 503 22.920920  12.5216028 33.320238
## 358 21.307342  10.6428369 31.971847
## 112 28.090740  17.9946346 38.186845
## 499 22.185933  12.1715909 32.200274
## 473 22.660336  12.4838398 32.836831
## 206 23.438919  13.4330740 33.444764
## 492 14.051237   2.8017402 25.300734
## 407  8.482821  -1.9445994 18.910242
## 470 19.380251   9.2474274 29.513075
## 454 22.186070  11.8165839 32.555556
## 4   30.027410  19.8222191 40.232601
## 311 19.974344   9.7619893 30.186698
## 326 25.921351  15.8030343 36.039668
## 98  36.864087  26.5179719 47.210202
## 391 17.970149   7.8537953 28.086503
## 7   24.260152  14.0965814 34.423723
## 183 35.467753  25.2658604 45.669646
## 299 28.682220  18.1400018 39.224438
## 466 18.760379   8.6524606 28.868296
## 445 10.457065   0.3619204 20.552209
## 396 20.562556  10.4279831 30.697129
## 483 29.817522  19.5230901 40.111954
## 307 34.657777  24.1194068 45.196148
## 456 14.718360   4.3360859 25.100634
## 146 11.933634   1.1741961 22.693072
## 281 40.037145  29.7617390 50.312550
## 258 45.832022  34.9853386 56.678705
## 435 16.079527   5.8418741 26.317180
## 324 19.673080   9.6353377 29.710821
## 68  21.491599  11.3816574 31.601540
## 436 11.452130   1.1552112 21.749049
## 48  16.689388   6.4006290 26.978147
## 288 26.126819  15.8729926 36.380645
## 341 21.728495  11.5328660 31.924125
## 347 14.718975   4.2351592 25.202791
## 167 40.743742  30.1449090 51.342576
## 377 16.433368   6.1207355 26.746000
## 272 27.481996  17.3887827 37.575210
## 459 17.378176   7.3038021 27.452551
## 116 21.091315  10.9972088 31.185422
## 450 16.775025   6.6870846 26.862965
## 496 16.904947   6.8796070 26.930286
## 93  28.521442  18.3747380 38.668147
## 301 29.787634  19.2796764 40.295592
## 158 37.236262  26.7465596 47.725963
## 461 18.838958   8.7117410 28.966174
## 221 31.415579  20.7323882 42.098769
## 87  22.206644  12.1504199 32.262867
## 95  26.183844  16.0155409 36.352148
```

```
## 353 15.202222   4.2742984 26.130146
## 223 30.338175  19.6830743 40.993277
## 413 -2.149040 -13.1533859  8.855306
## 220 28.498233  17.9246588 39.071807
## 251 24.043295  13.9161784 34.170412
## 31   8.830438  -1.5512938 19.212170
## 182 28.993054  18.8131325 39.172975
## 425 13.751929   3.0974087 24.406449
## 297 29.068315  18.7879080 39.348723
## 408 21.717274  11.5740300 31.860517
## 171 24.708192  14.3785995 35.037785
## 490  7.407426  -3.8641458 18.678997
## 191 31.388781  21.1334035 41.644158
## 393  9.231612  -1.0331340 19.496357
## 148  9.495029  -1.3629282 20.352985
## 398 16.426955   6.2960239 26.557887
## 363 21.025648  10.5351281 31.516168
## 334 22.872871  12.4787378 33.267005
## 88  26.931821  16.8696734 36.993968
## 387  5.377182  -5.0810493 15.835413
## 366 18.729614   7.3329516 30.126276
## 420 12.451002   2.0387265 22.863277
## 371 34.670814  23.8172653 45.524362
## 434 16.605521   6.3457492 26.865292
## 411 16.681741   4.8319755 28.531507
## 430 10.656419   0.2874884 21.025350
## 431 16.885128   6.5250630 27.245193
## 254 29.008768  18.2287239 39.788813
## 47  20.413393  10.3023288 30.524458
## 196 39.612281  28.7010465 50.523515
## 12  22.583512  12.3443179 32.822706
## 121 21.618492  11.3227327 31.914251
## 16  19.945575   9.7320039 30.159146
## 406  8.685769  -3.8852855 21.256823
## 131 20.011221   9.6319984 30.390443
## 133 20.392147  10.0164291 30.767866
## 44  25.811907  15.7247452 35.899069
## 472 23.422002  13.1842151 33.659789
## 156 19.854533   8.6322387 31.076827
## 245 15.073147   4.8256619 25.320632
## 480 22.891274  12.7884675 32.994080
## 42  29.422025  19.3077407 39.536310
## 143 13.112399   2.1010925 24.123706
## 185 23.189069  12.9728144 33.405324
## 298 19.781546   9.3595086 30.203583
## 332 19.266970   9.1319184 29.402021
## 154 20.482051   9.7225934 31.241508
## 359 21.402925  10.7068715 32.098979
## 280 37.082706  26.8444602 47.320953
## 485 20.138974   9.9670324 30.310916
## 137 15.174834   4.8376032 25.512064
```

```
me=lm(medv~ crim+zn+chas+nox+rm+dis+rad+tax+ptratio+black+lstat, testData)
mep=predict(me,predata,interval="prediction" )
```

```
mep
```

```
##           fit         lwr       upr
## 202 28.175393  17.2930686 39.057716
## 503 22.920920  12.5216028 33.320238
## 358 21.307342  10.6428369 31.971847
## 112 28.090740  17.9946346 38.186845
## 499 22.185933  12.1715909 32.200274
## 473 22.660336  12.4838398 32.836831
## 206 23.438919  13.4330740 33.444764
## 492 14.051237   2.8017402 25.300734
## 407  8.482821  -1.9445994 18.910242
## 470 19.380251   9.2474274 29.513075
## 454 22.186070  11.8165839 32.555556
## 4   30.027410  19.8222191 40.232601
## 311 19.974344   9.7619893 30.186698
## 326 25.921351  15.8030343 36.039668
## 98  36.864087  26.5179719 47.210202
## 391 17.970149   7.8537953 28.086503
## 7   24.260152  14.0965814 34.423723
## 183 35.467753  25.2658604 45.669646
## 299 28.682220  18.1400018 39.224438
## 466 18.760379   8.6524606 28.868296
## 445 10.457065   0.3619204 20.552209
## 396 20.562556  10.4279831 30.697129
## 483 29.817522  19.5230901 40.111954
## 307 34.657777  24.1194068 45.196148
## 456 14.718360   4.3360859 25.100634
## 146 11.933634   1.1741961 22.693072
## 281 40.037145  29.7617390 50.312550
## 258 45.832022  34.9853386 56.678705
## 435 16.079527   5.8418741 26.317180
## 324 19.673080   9.6353377 29.710821
## 68  21.491599  11.3816574 31.601540
## 436 11.452130   1.1552112 21.749049
## 48  16.689388   6.4006290 26.978147
## 288 26.126819  15.8729926 36.380645
## 341 21.728495  11.5328660 31.924125
## 347 14.718975   4.2351592 25.202791
## 167 40.743742  30.1449090 51.342576
## 377 16.433368   6.1207355 26.746000
## 272 27.481996  17.3887827 37.575210
## 459 17.378176   7.3038021 27.452551
## 116 21.091315  10.9972088 31.185422
## 450 16.775025   6.6870846 26.862965
## 496 16.904947   6.8796070 26.930286
## 93  28.521442  18.3747380 38.668147
## 301 29.787634  19.2796764 40.295592
## 158 37.236262  26.7465596 47.725963
## 461 18.838958   8.7117410 28.966174
## 221 31.415579  20.7323882 42.098769
## 87  22.206644  12.1504199 32.262867
## 95  26.183844  16.0155409 36.352148
## 353 15.202222   4.2742984 26.130146
```

```
## 223 30.338175  19.6830743 40.993277
## 413 -2.149040 -13.1533859  8.855306
## 220 28.498233  17.9246588 39.071807
## 251 24.043295  13.9161784 34.170412
## 31   8.830438  -1.5512938 19.212170
## 182 28.993054  18.8131325 39.172975
## 425 13.751929   3.0974087 24.406449
## 297 29.068315  18.7879080 39.348723
## 408 21.717274  11.5740300 31.860517
## 171 24.708192  14.3785995 35.037785
## 490  7.407426  -3.8641458 18.678997
## 191 31.388781  21.1334035 41.644158
## 393  9.231612  -1.0331340 19.496357
## 148  9.495029  -1.3629282 20.352985
## 398 16.426955   6.2960239 26.557887
## 363 21.025648  10.5351281 31.516168
## 334 22.872871  12.4787378 33.267005
## 88  26.931821  16.8696734 36.993968
## 387  5.377182  -5.0810493 15.835413
## 366 18.729614   7.3329516 30.126276
## 420 12.451002   2.0387265 22.863277
## 371 34.670814  23.8172653 45.524362
## 434 16.605521   6.3457492 26.865292
## 411 16.681741   4.8319755 28.531507
## 430 10.656419   0.2874884 21.025350
## 431 16.885128   6.5250630 27.245193
## 254 29.008768  18.2287239 39.788813
## 47  20.413393  10.3023288 30.524458
## 196 39.612281  28.7010465 50.523515
## 12  22.583512  12.3443179 32.822706
## 121 21.618492  11.3227327 31.914251
## 16  19.945575   9.7320039 30.159146
## 406  8.685769  -3.8852855 21.256823
## 131 20.011221   9.6319984 30.390443
## 133 20.392147  10.0164291 30.767866
## 44  25.811907  15.7247452 35.899069
## 472 23.422002  13.1842151 33.659789
## 156 19.854533   8.6322387 31.076827
## 245 15.073147   4.8256619 25.320632
## 480 22.891274  12.7884675 32.994080
## 42  29.422025  19.3077407 39.536310
## 143 13.112399   2.1010925 24.123706
## 185 23.189069  12.9728144 33.405324
## 298 19.781546   9.3595086 30.203583
## 332 19.266970   9.1319184 29.402021
## 154 20.482051   9.7225934 31.241508
## 359 21.402925  10.7068715 32.098979
## 280 37.082706  26.8444602 47.320953
## 485 20.138974   9.9670324 30.310916
## 137 15.174834   4.8376032 25.512064
```

(b) Compare the predictions using mean squared prediction error. Which model performed the best?

```
set.seed(100)
```

```r
a1=predict(m1, testData, interval='prediction')[,1]

a4=predict(m4, predata, interval='prediction')[,1]

al=predict(mla,predata,interval = 'prediction')[,1]

m1m=mean((a1-testData$medv)^2)
m4m=mean((a4-testData$medv)^2)
alm=mean((al-testData$medv)^2)
c(m1m,m4m,alm)
```

```
## [1] 24.51305 24.47241 21.67312
```

Since m4 and ridge model are identical, lasso and elastic are identical, only three MSE need to be calculated.

MSE (m1)=24.51305

MSE(BIC,Ridge)=24.47241

MSE(lasso, elastic)=22.81797

Lasso and Elastic has the smallest MSE, so they perform better.

(c) Provide a table listing each method described in Question 7a and the variables selected by each method (see Unit 5.2.3 for an example). Which variables were selected consistently?

	Backward Stepwise	Ridge	Lasso	Elastic Net
crim	x	x	x	x
zn	x	x	x	x
indus				
chas	x	x	x	x
nox	x	x	x	x
rm	x	x	x	x
age				
dis	x	x	x	x
rad	x	x	x	x
tax	x	x	x	x
ptratio	x	x	x	x
black	x	x	x	x
lstat	x	x	x	x

crim+zn+chas+nox+rm+dis+rad+tax+ptratio+black+lstat were selected constantly