Question 10.1

Using the crime data set uscrime.txt from Questions 8.2, 9.1, and 10.1, build a regression model using:

1. Stepwise regression
2. Lasso
3. Elastic net For Parts 2 and 3, remember to scale the data first – otherwise, the regression coefficients will be on different scales and the constraint won't have the desired effect.

For Parts 2 and 3, use the glmnet function in R.

Notes on R: • For the elastic net model, what we called λ in the videos, glmnet calls "alpha"; you can get a range of results by varying alpha from 1 (lasso) to 0 (ridge regression) [and, of course, other values of alpha in between]. • In a function call like glmnet(x,y,family="mgaussian",alpha=1) the predictors x need to be in R's matrix format, rather than data frame format. You can convert a data frame to a matrix using as.matrix – for example, x <- as.matrix(data[,1:n-1]) • Rather than specifying a value of T, glmnet returns models for a variety of values of T.

In [4]:

```
# Clear environment

rm(list = ls())
set.seed(1)
```

In [44]:

```
##########LIBRARY##########
library(caret) # To use K fold
#install.packages("glmnet")
library(glmnet) #for lasso regression
#library(tidyr) # to use tibble function
#library(DAAG)
#library("ggplot2")
#install.packages("devtools")
#install.packages("corrplot")
#library("devtools")
#library("corrplot") # to use Correlation plot
#library(plyr) #to use arrange function
```

```
Loading required package: Matrix

Loaded glmnet 4.1-2
```

In [10]:
```
##########INGEST FILE##########

#crime<- read.table("C:\\Preethi\\R\\USCrimes.txt",header=TRUE,stringsAsFactors = FALSE,sep="\t")
#head(data,10)
#colClasses = c("numeric", "numeric", "numeric")
#colClasses
data = read.table("C:\\Preethi\\R\\USCrimes.txt",header=TRUE,stringsAsFactors = FALSE)
#%>% as_tibble()
head(data,2) #last field is the response
```

A data.frame: 2 × 16

	M	So	Ed	Po1	Po2	LF	M.F	Pop	NW	U1	U2	Wealth	Ineq	Prob	Time	Crime
	<dbl>	<int>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<int>	<dbl>	<dbl>	<dbl>	<int>	<dbl>	<dbl>	<dbl>	<int>
1	15.1	1	9.1	5.8	5.6	0.510	95.0	33	30.1	0.108	4.1	3940	26.1	0.084602	26.2011	791
2	14.3	0	11.3	10.3	9.5	0.583	101.2	13	10.2	0.096	3.6	5570	19.4	0.029599	25.2999	1635

1. Approach for Stepwise backward regression

1. use k fold and establish a backward Stepwise linear regression. This provided multiple combination of factors and AIC. I selected "M + Ed + Po1 + M.F + U2 + Ineq + Time" which had good tradeoff of lesser factors and less AIC (AIC=398.75) (just by using combination of 7 factors)
2. Using these factors, I received Adjusted R2 value : 0.69 when fitting a model
3. Used leave one out cross validation to understand the R2 square value to interpret the efficiency of the model Using K fold, the model comes up with .61 R2 value for the same 7 factors
4. Looking at model' summary # Noticed that the p-value for M.F and TIME is above 0.1. So created a new regression model removing these 2 factors, yielding a R2 sqaure value of 0.69 and K-fold R2 error is better than tha previous k-fold value (0.61 R2) CONCLUSION: Concluding to use model with 5 factors with better R squared value and more simpler model #So we wll be eliminating 2 more factors from the model (M + Ed + Po1 + U2 + Ineq) ONLY

In [12]:
```
#SCALING IS NOT REQUIRED for CLASSIC MODELS!
#a=c(1,3,4,5,6,7,8,9,10,11,12,13,14,15)
#as.data.frame(scale(data[,a])) #Excluding Int field (So) a
```

In [78]:
```
# Now using the code below to perform 5 fold CV
ctrl <- trainControl(method = "repeatedcv", number = 5, repeats = 5)# k fold validation
#lmFit_Step <- train(Crime ~ ., data = data, "lmStepAIC", scope =
#                  list(lower = Crime~1, upper = Crime~.), direction = "backward",trControl=ctrl #backward Regression
#Step:  AIC=425.99
```

I have placed results one the chosen stpe ONLY as the remainder of the results made the results run for many pages(for sake of pdf attachement

Step: AIC=425.99 .outcome ~ M + Ed + Po1 + Po2 + LF + M.F + Pop + NW + U1 + U2 + Wealth + Ineq + Prob + Time

```
       Df Sum of Sq     RSS    AIC
```

- Pop 1 2237 1277226 424.06
- LF 1 6851 1281839 424.20
- Wealth 1 7581 1282569 424.22
- Time 1 11129 1286117 424.32
- NW 1 24307 1299296 424.71
- M.F 1 25868 1300856 424.76
- Po2 1 44428 1319417 425.29 1274989 425.99
- U1 1 76748 1351736 426.21
- M 1 127443 1402432 427.61
- Po1 1 133488 1408477 427.78
- Prob 1 142421 1417410 428.02
- U2 1 161153 1436141 428.52
- Ineq 1 299562 1574551 432.01
- Ed 1 378483 1653472 433.87

The above step wise regression provides multiple combination of factors and AIC.Efficient of Step regression combination using AIC with combination of less factors(7) is below

Step: AIC=398.75 Factors used: M + Ed + Po1 + M.F + U2 + Ineq + Time (7 factors)

```
       Df Sum of Sq     RSS    AIC
```

1150254 398.75

- M.F 1 79771 1230025 399.23
- U2 1 80546 1230800 399.25
- M 1 243554 1393807 403.86
- Time 1 245502 1395756 403.91
- Ed 1 551044 1701297 411.23
- Ineq 1 742524 1892778 415.18
- Po1 1 2258821 3409074 436.95

In [20]:
```
#Using the factors from backward stepwise regression , fit  a new model with these 7 factors
mod_Step = lm(Crime ~ M + Ed + Po1 + M.F + U2 + Ineq + Time, data = data)
summary(mod_Step)
#Adjusted R2 value : 0.69
```

```
Call:
lm(formula = Crime ~ M + Ed + Po1 + M.F + U2 + Ineq + Time, data = data)

Residuals:
    Min      1Q  Median      3Q     Max 
-460.60 -106.75  -20.63  119.27  537.06 

Coefficients:
             Estimate Std. Error t value Pr(>|t|)    
(Intercept) -6498.308   1317.977  -4.931 1.56e-05 ***
M              87.421     36.712   2.381 0.022232 *  
Ed            182.806     56.781   3.219 0.002589 ** 
Po1           120.875     14.531   8.318 3.58e-10 ***
M.F            16.180     13.940   1.161 0.252845    
U2             77.169     45.121   1.710 0.095165 .  
Ineq           61.707     14.820   4.164 0.000167 ***
Time            6.901      5.077   1.359 0.181832    
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 212 on 39 degrees of freedom
Multiple R-squared:  0.7452,	Adjusted R-squared:  0.6995 
F-statistic: 16.29 on 7 and 39 DF,  p-value: 8.248e-10
```

In [23]:
```
#BACKWISE STEP REGRESSION +CROSS VALIDATION
# Now let's use cross-validation to see how good this model
# really is.  Because we only have 47 data points, let's use
# 47-fold cross-validation (equivalently, leave-one-out
# cross-validation).

SStot <- sum((data$Crime - mean(data$Crime))^2)
SStot
```

6880927.65957447

In [29]:
```
totsse <- 0
for(i in 1:nrow(data)) {
  mod_Step_i = lm(Crime ~ M + Ed + Po1 + M.F + U2 + Ineq + Time, data = data[-i,])
  pred_i <- predict(mod_Step_i,newdata=data[i,])
  totsse <- totsse + ((pred_i - data[i,16])^2)
}
R2_kfold <- 1 - totsse/SStot
R2_kfold
```

1: 0.611730795689203

In [30]:
```
#Using K fold, the model comes up with .61 R2 value for the same 7 factors
```

```
Call:
lm(formula = Crime ~ M + Ed + Po1 + M.F + U2 + Ineq + Time, data = data[-i,
    ])

Residuals:
    Min      1Q  Median      3Q     Max
-443.65 -115.24  -14.55  121.69  545.79

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -6799.521   1352.235  -5.028 1.22e-05 ***
M              87.557     36.715   2.385 0.022183 *
Ed            186.567     56.910   3.278 0.002238 **
Po1           119.853     14.568   8.227 5.75e-10 ***
M.F            18.921     14.210   1.332 0.190940
U2             83.775     45.608   1.837 0.074064 .
Ineq           61.330     14.825   4.137 0.000188 ***
Time            6.472      5.095   1.270 0.211738
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 212 on 38 degrees of freedom
Multiple R-squared:  0.7516,    Adjusted R-squared:  0.7058
F-statistic: 16.42 on 7 and 38 DF,  p-value: 9.709e-10
```

In [32]:
```
#USING P VALUE TO ELIMIATE MORE FACTORS
summary(mod_Step_i)
```

```
Call:
lm(formula = Crime ~ M + Ed + Po1 + M.F + U2 + Ineq + Time, data = data[-i,
    ])

Residuals:
    Min      1Q  Median      3Q     Max 
-443.65 -115.24  -14.55  121.69  545.79 

Coefficients:
             Estimate Std. Error t value Pr(>|t|)    
(Intercept) -6799.521   1352.235  -5.028 1.22e-05 ***
M              87.557     36.715   2.385 0.022183 *  
Ed            186.567     56.910   3.278 0.002238 ** 
Po1           119.853     14.568   8.227 5.75e-10 ***
M.F            18.921     14.210   1.332 0.190940    
U2             83.775     45.608   1.837 0.074064 .  
Ineq           61.330     14.825   4.137 0.000188 ***
Time            6.472      5.095   1.270 0.211738    
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 212 on 38 degrees of freedom
Multiple R-squared:  0.7516,	Adjusted R-squared:  0.7058 
F-statistic: 16.42 on 7 and 38 DF,  p-value: 9.709e-10
```

In [36]:

```
# Notice that in the model above, the p-value for M.F and TIME is above 0.1.
# We might keep it in the model, because it's close to 0.1 and
# might be important.  That's what we tested above.
# Or, we might remove it, and re-run the model without it.
#MODEL WITHOUT M.F with high P value
mod_p_val = lm(Crime ~ M + Ed + Po1 + U2 + Ineq , data = data)
summary(mod_p_val)
#noticed higher adjusted P value (0.69)
```

```
Call:
lm(formula = Crime ~ M + Ed + Po1 + U2 + Ineq, data = data)

Residuals:
    Min      1Q  Median      3Q     Max 
-453.44  -98.59  -18.07  106.03  629.64 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept) -5243.74     951.16  -5.513 2.13e-06 ***
M             101.98      35.32   2.887 0.006175 ** 
Ed            203.08      47.42   4.283 0.000109 ***
Po1           123.31      14.16   8.706 7.26e-11 ***
U2             91.36      43.41   2.105 0.041496 *  
Ineq           63.49      14.68   4.324 9.56e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 213 on 41 degrees of freedom
Multiple R-squared:  0.7296,	Adjusted R-squared:  0.6967 
F-statistic: 22.13 on 5 and 41 DF,  p-value: 1.105e-10
```

In [38]:

```
# the above model looks good with M.F and Time, so now let's see how it cross-validates:

SStot <- sum((data$Crime - mean(data$Crime))^2)
totsse <- 0
for(i in 1:nrow(data)) {
  mod_Step_i = lm(Crime ~ M + Ed + Po1 + U2 + Ineq, data = data[-i,])
  pred_i <- predict(mod_Step_i,newdata=data[i,])
  totsse <- totsse + ((pred_i - data[i,16])^2)
}
R2_mod <- 1 - totsse/SStot
R2_mod
```

1: 0.637093804597545

In [ ]:
```
#CONCLUSION
# The above k  fold validation retrieves
#####a  better R squared value than the previous model
#####More simpler model
#So we wll be eliminating 2 more factors from the model
# going with a 5 factor model(M + Ed + Po1 + U2 + Ineq) ONLY
#with original R2 value =0.69
#with K fold R2 value =0.63
```

2.LASSO REGRESSION

Approach

SCALING IS IMPORTANT FOR GLOBAL REFINED LASSO & ELASITC NET. So scaled the data without response and categorical value and then added them back to the data

LASSO:

1.using alpha=1 for Lasso regression

1. Observed that Lasso has made PO2 as zero just like stated in the lectures from previous assignment, we know Po1 and Po2 were highly correlated factors. Lasso removed one of the highligly correlated factors
2. Fitting with 9 Non zero co-eff variables (So+M+Ed+Po1+M.F+NW+U2+Ineq+Prob), we get R2 value :0.72 better than Stepwise Backward regression & p values are all low for the selected predictors
3. Cross validation of this model give 0.62 R2

RIDGE:

1.using alpha=0 for Lasso regression 2.Observed that highly corelated co-efficients PO1 and PO2 are both retained in lasso but thier co-efficients are underestimated from 300 to 100

1. Using 7 variables M+Ed+Po1+Po2+U1+U2+Wealth+Ineq+Prob the highest of R2 (0.73) square for model fitting is from Ridge regression
2. Running another test model using most influential 6 factors (Ed+Po1+Po2+U2+Ineq+Prob) 0.66 R2 still better with 6 factors ONLY!

In [46]:
```
#SCALING IS IMPORTANT FOR GLOBAL REFINED LASSO & ELASITC NET
#Scaling the data except the response variable and categorical

scaledData = as.data.frame(scale(data[,c(1,3,4,5,6,7,8,9,10,11,12,13,14,15)]))
scaledData <- cbind(data[,2],scaledData,data[,16]) # Add column 2 back in
colnames(scaledData)[1] <- "So"
colnames(scaledData)[16] <- "Crime"
head(scaledData,2)
```

A data.frame: 2 × 16

	So	M	Ed	Po1	Po2	LF	M.F	Pop	NW	U1	U2	Wealt
	<int>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>
1	1	0.9886930	-1.3085099	-0.9085105	-0.8666988	-1.2667456	-1.1206050	-0.09500679	1.943738564	0.69510600	0.8313680	-1.361
2	0	0.3521372	0.6580587	0.6056737	0.5280852	0.5396568	0.9834175	-0.62033844	0.008483424	0.02950365	0.2393332	0.3276

In [48]:
```
#building lasso
#using alpha=1 for Lasso regression
XP=data.matrix(scaledData[,-16])
YP=data.matrix(scaledData$Crime)
lasso=cv.glmnet(x=as.matrix(scaledData[,-16]),y=as.matrix(scaledData$Crime),alpha=1,
                nfolds = 5,type.measure="mse",family="gaussian")
summary(lasso)
```

```
           Length Class  Mode
lambda     89     -none- numeric
cvm        89     -none- numeric
cvsd       89     -none- numeric
cvup       89     -none- numeric
cvlo       89     -none- numeric
nzero      89     -none- numeric
call        7     -none- call
name        1     -none- character
glmnet.fit 12     elnet  list
lambda.min  1     -none- numeric
lambda.1se  1     -none- numeric
index       2     -none- numeric
```

In [51]:
```
#Output the coefficients of the variables selected by lasso
coef(lasso, s=lasso$lambda.min)
```

```
16 x 1 sparse Matrix of class "dgCMatrix"
                    s1
(Intercept) 888.180264
So           49.657975
M            80.599625
Ed          106.706431
Po1         311.884001
Po2           .
LF            3.638231
M.F          47.254311
Pop           .
NW            2.441652
U1           -4.152300
U2           36.142316
Wealth        .
Ineq        169.915175
Prob        -78.892733
Time          .
```

In [ ]:
```
#Observe that Lasso has made PO2 as zero just like stated in the lectures
#from previous assignment, we know P)1 and PO2 were highly correlated factors
# lasso removed one of the highligh correlated factors
```

In [52]:
```
#Fitting a new model with these 9 variables

mod_lasso = lm(Crime ~So+M+Ed+Po1+M.F+NW+U2+Ineq+Prob, data = scaledData)
summary(mod_lasso)
#R2 value :0.72 better than Stepwise Backward regression and the p values are all low for the selected predictors
```

```
Call:
lm(formula = Crime ~ So + M + Ed + Po1 + M.F + NW + U2 + Ineq +
    Prob, data = scaledData)

Residuals:
    Min      1Q  Median      3Q     Max
-415.05 -122.24    0.05  114.69  557.46

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  871.394     50.552  17.238  < 2e-16 ***
So            98.967    119.891   0.825 0.414395
M            109.922     48.931   2.246 0.030730 *
Ed           197.053     62.019   3.177 0.002998 **
Po1          333.443     48.855   6.825 4.85e-08 ***
M.F           40.508     38.974   1.039 0.305384
NW             4.098     57.400   0.071 0.943465
U2            63.300     36.911   1.715 0.094719 .
Ineq         237.685     66.071   3.597 0.000935 ***
Prob        -101.929     39.465  -2.583 0.013892 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 204.5 on 37 degrees of freedom
Multiple R-squared:  0.7752,	Adjusted R-squared:  0.7205
F-statistic: 14.17 on 9 and 37 DF,  p-value: 1.541e-09
```

In [53]:

```
# Lasso regression and Cross Validation

# Now let's see how it cross-validates:

SStot <- sum((data$Crime - mean(data$Crime))^2)
totsse <- 0
for(i in 1:nrow(scaledData)) {
  mod_lasso_i = lm(Crime ~ So+M+Ed+Po1+M.F+NW+U2+Ineq+Prob, data = scaledData[-i,])
  pred_i <- predict(mod_lasso_i,newdata=scaledData[i,])
  totsse <- totsse + ((pred_i - data[i,16])^2)
}
R2_mod <- 1 - totsse/SStot
R2_mod
# with k fold validation , we see R value as 0.62 in Lasso regression
```

1: 0.62029104686309

2.(b) RIDGE REGRESSION

In [50]:

```
#CURIOUS TO FIND RIDGE REGRESSIOn
#using alpha=0 for Ridge regression
ridge=cv.glmnet(x=as.matrix(scaledData[,-16]),y=as.matrix(scaledData$Crime),alpha=0,
                nfolds = 5,type.measure="mse",family="gaussian")
summary(ridge)
```

```
           Length Class  Mode
lambda     100    -none- numeric
cvm        100    -none- numeric
cvsd       100    -none- numeric
cvup       100    -none- numeric
cvlo       100    -none- numeric
nzero      100    -none- numeric
call         7    -none- call
name         1    -none- character
glmnet.fit  12    elnet  list
lambda.min   1    -none- numeric
lambda.1se   1    -none- numeric
index        2    -none- numeric
```

In [54]:
```
#Output the coefficients of the variables selected by RIDGE
coef(ridge, s=lasso$lambda.min)
```

```
16 x 1 sparse Matrix of class "dgCMatrix"
                    s1
(Intercept) 877.904372
So           79.843406
M            88.772431
Ed          128.555609
Po1         163.678598
Po2         105.495298
LF           16.942841
M.F          67.201484
Pop          -6.949854
NW           28.991305
U1          -60.101677
U2          100.377104
Wealth       43.865388
Ineq        175.277509
Prob        -91.645910
Time          2.003333
```

In [ ]:
```
#Observe that highly corelated co-efficients PO1 and PO2 are both retained in lasso
# but thier co-efficients are underestimated from 300 to 100
```

In [56]:

```
#Fitting a new model with  variables from ridge regression
#(removing co-efficients with less impact)

mod_lasso = lm(Crime ~M+Ed+Po1+Po2+U1+U2+Wealth+Ineq+Prob, data = scaledData)
summary(mod_lasso)
#the highest of R2 square for model fitting is from Ridge regression
#0.73 Adj. R2 (but with 7 variables)
```

```
Call:
lm(formula = Crime ~ M + Ed + Po1 + Po2 + U1 + U2 + Wealth +
    Ineq + Prob, data = scaledData)

Residuals:
    Min      1Q  Median      3Q     Max
-422.04 -107.96   -6.03  121.06  497.52

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   905.09      29.22  30.976  < 2e-16 ***
M             140.95      42.43   3.322 0.002021 **
Ed            229.05      56.72   4.039 0.000260 ***
Po1           502.22     270.38   1.857 0.071215 .
Po2          -220.54     271.85  -0.811 0.422400
U1            -58.33      55.37  -1.053 0.299009
U2            122.49      61.98   1.976 0.055610 .
Wealth         92.34      90.70   1.018 0.315241
Ineq          301.53      71.44   4.220 0.000152 ***
Prob          -75.47      35.66  -2.116 0.041123 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 200.3 on 37 degrees of freedom
Multiple R-squared:  0.7842,	Adjusted R-squared:  0.7318
F-statistic: 14.94 on 9 and 37 DF,  p-value: 7.466e-10
```

In [57]:
```
#considering factors with more than 100 threshold from ridge regression
mod_lasso_100 = lm(Crime ~Ed+Po1+Po2+U2+Ineq+Prob, data = scaledData)
summary(mod_lasso_100) #0.66 R2 still better with 6 factors ONLY!
```

```
Call:
lm(formula = Crime ~ Ed + Po1 + Po2 + U2 + Ineq + Prob, data = scaledData)

Residuals:
    Min      1Q  Median      3Q     Max 
-548.80  -93.42   -1.94  146.33  474.07 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)   905.09      32.48  27.869  < 2e-16 ***
Ed            189.17      54.46   3.474  0.00125 ** 
Po1           556.75     297.18   1.873  0.06833 .  
Po2          -229.66     301.12  -0.763  0.45012    
U2             35.23      36.05   0.977  0.33436    
Ineq          315.68      59.60   5.297 4.58e-06 ***
Prob          -81.43      38.54  -2.113  0.04091 *  
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 222.6 on 40 degrees of freedom
Multiple R-squared:  0.7118,	Adjusted R-squared:  0.6686 
F-statistic: 16.47 on 6 and 40 DF,  p-value: 1.896e-09
```

3. Elastic Net!

Approach 1) vary alpha in steps of 0.1 from 0 to 1 and calculate the resultant R-Squared values 2) Identify the best alpha 3) Using best alpha , fit a elasticnet model 4) The Elastic Net selects 13 variables (more when compared to Stepwise and Lasso regression) 5) Based on best alpha, R is 0.72 (R2 is better than Stepwise and lasso) 6)With cross-validates and cco-efficients with p value > 0.05, I get 0.66 R2 squared value
(similar to ridge regression; R square value)

In [70]:
```
#We vary alpha in steps of 0.1 from 0 to 1 and calculate the resultant R-Squared values

R2=c()
for (i in 0:10) {
  mod_elastic = cv.glmnet(x=as.matrix(scaledData[,-16]),y=as.matrix(scaledData$Crime),
                    alpha=i/10,nfolds = 5,type.measure="mse",family="gaussian")
#The deviance(dev.ratio ) shows the percentage of deviance explained,
#(equivalent to r squared in case of regression)
  R2 = cbind(R2,mod_elastic$glmnet.fit$dev.ratio[which(mod_elastic$glmnet.fit$lambda == mod_elastic$lambda.min)])
}
R2
#Highest R2
```

A matrix: 1 × 11 of type dbl

0.7305131	0.7575969	0.7824825	0.7727509	0.7754023	0.7921574	0.7651323	0.6506664	0.7408659	0.7365694	0.7451488

In [71]:
```
#Best value of alpha
alpha_best = (which.max(R2)-1)/10
alpha_best (best value of alpha is 0.5)
```

0.5

In [73]:
```
#Lets build the model using this alpha value.
Elastic_net=cv.glmnet(x=as.matrix(scaledData[,-16]),y=as.matrix(scaledData$Crime),alpha=alpha_best,
                      nfolds = 5,type.measure="mse",family="gaussian")
#Output the coefficients of the variables selected by Elastic Net
coef(Elastic_net, s=Elastic_net$lambda.min)
```

```
16 x 1 sparse Matrix of class "dgCMatrix"
                   s1
(Intercept) 890.51243
So           42.80724
M            96.24185
Ed          156.03106
Po1         266.57855
Po2          18.97955
LF             .
M.F          60.50018
Pop          -7.39804
NW           17.91207
U1          -63.24503
U2          102.37302
Wealth       41.09751
Ineq        216.57041
Prob        -88.79877
Time           .
```

In [74]:
```
# The Elastic Net selects 13 variables
#(more when compared to Stepwise and Lasso regression)
mod_Elastic_net = lm(Crime ~So+M+Ed+Po1+Po2+M.F+Pop+NW+U1+U2+Wealth+Ineq+Prob, data = scaledData)
summary(mod_Elastic_net)
```

```
Call:
lm(formula = Crime ~ So + M + Ed + Po1 + Po2 + M.F + Pop + NW + 
    U1 + U2 + Wealth + Ineq + Prob, data = scaledData)

Residuals:
    Min      1Q  Median      3Q     Max 
-389.63  -94.25    7.83  109.20  491.62 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)   893.38      52.51  17.012  < 2e-16 ***
So             34.40     127.12   0.271  0.78840    
M             109.87      49.82   2.205  0.03451 *  
Ed            202.41      64.00   3.163  0.00335 ** 
Po1           501.63     287.30   1.746  0.09012 .  
Po2          -215.08     288.65  -0.745  0.46148    
M.F            43.45      48.99   0.887  0.38162    
Pop           -36.21      46.10  -0.785  0.43784    
NW             24.91      58.61   0.425  0.67360    
U1            -86.62      66.24  -1.308  0.20002    
U2            136.97      67.41   2.032  0.05027 .  
Wealth         82.03      96.17   0.853  0.39983    
Ineq          275.77      86.79   3.177  0.00322 ** 
Prob          -95.16      41.52  -2.292  0.02843 *  
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 204 on 33 degrees of freedom
Multiple R-squared:  0.8005,	Adjusted R-squared:  0.7219 
F-statistic: 10.19 on 13 and 33 DF,  p-value: 4.088e-08
```

In [77]:
```
#R2 is better than Stepwise and lasso
# Now let's see how it cross-validates
#(Looking at p values of the co-eff, i removed P-values > 0.05)

SStot <- sum((data$Crime - mean(data$Crime))^2)
totsse <- 0
for(i in 1:nrow(scaledData)) {
  mod_lasso_i = lm(Crime ~ M+Ed+Po1+U2+Ineq+Prob, data = scaledData[-i,])
  pred_i <- predict(mod_lasso_i,newdata=scaledData[i,])
  totsse <- totsse + ((pred_i - data[i,16])^2)
}
R2_mod <- 1 - totsse/SStot
R2_mod
#But cross validation yields  0.66 R2 squared value when compared to Lasso and Stepwise regression
```

1: 0.666163842867471

In [ ]:
```
#########CONCLUSION OF WEEK 8 VARIABLE SELECTION#########
```