Table of Contents

Stepwise Regression Calculations

```
#Clear the environment and set up the necessary libraries.
rm(list = ls())
library(sme)
library(glmnet)
library(DAAG)
#Set a seed and pull in the data
setseed(10)
Crime_data <- read.table("uscrime.txt", stringsAsFactors = FALSE, header = TRUE)
>
```

Here I am setting up the code and setting a seed for replicable results.

```
#Function to calculate R-squared
Rsquared <- function(yhat_data,datatable){
  regs <- sum((yhat_data - datatable$Crime)^2)
  total <- sum((datatable$Crime - mean(datatable$Crime))^2)
  R2 <- 1 - regs/total
  return(R2)
}
>
```

This function will calculate R-squared value for my models which I will use throughout my solution.

```
#Setting up lm values to calculate backwards regression
Crime_back <- lm(Crime~., data = Crime_data)
step(Crime_back, direction="backward")
```

```
Start:  AIC=515
Crime ~ M + So + Ed + Po1 + Po2 + LF + M.F + Pop + NW + U1 +
    U2 + Wealth + Ineq + Prob + Time

         Df Sum of Sq     RSS AIC
- So      1        29 1354974 513
- LF      1      8917 1363862 513
- Time    1     10304 1365250 513
- Pop     1     14122 1369068 513
- NW      1     18395 1373341 513
- M.F     1     31967 1386913 514
- Wealth  1     37613 1392558 514
- Po2     1     37919 1392865 514
<none>                1354946 515
- U1      1     83722 1438668 515
- Po1     1    144306 1499252 517
- U2      1    181536 1536482 519
- M       1    193770 1548716 519
- Prob    1    199538 1554484 519
- Ed      1    402117 1757063 525
- Ineq    1    423031 1777977 525

Step:  AIC=513
Crime ~ M + Ed + Po1 + Po2 + LF + M.F + Pop + NW + U1 + U2 +
    Wealth + Ineq + Prob + Time

         Df Sum of Sq     RSS AIC
- Time    1     10341 1365315 511
- LF      1     10878 1365852 511
- Pop     1     14127 1369101 511
- NW      1     21626 1376600 511
- M.F     1     32449 1387423 512
- Po2     1     37954 1392929 512
- Wealth  1     39223 1394197 512
<none>                1354974 513
- U1      1     96420 1451395 514
- Po1     1    144302 1499277 515
- U2      1    189859 1544834 517
- M       1    195084 1550059 517
- Prob    1    204463 1559437 517
- Ed      1    403140 1758114 523
- Ineq    1    488834 1843808 525

Step:  AIC=511
Crime ~ M + Ed + Po1 + Po2 + LF + M.F + Pop + NW + U1 + U2 +
    Wealth + Ineq + Prob

         Df Sum of Sq     RSS AIC
- LF      1     10533 1375848 509
- NW      1     15482 1380797 510
- Pop     1     21846 1387161 510
```

This is a snip of the output, which I will condense & summarize in the next step.

```
#Supresses output
back_model <- step(Crime_back, direction="backward", trace=FALSE)
back_model
```

```
Coefficients:
(Intercept)            M           Ed          Po1
    -6426.1         93.3        180.1        102.7
        M.F           U1           U2         Ineq
       22.3      -6086.6        187.3         61.3
       Prob
    -3796.0
```

This is the results of the backwards step regression with the results summarized.

```
summary(back_model)
```

```
Call:
lm(formula = Crime ~ M + Ed + Po1 + M.F + U1 + U2 + Ineq + Prob,
    data = Crime_data)

Residuals:
   Min     1Q Median     3Q    Max
 -445   -111      3    122    483

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  -6426.1     1194.6   -5.38  4.0e-06 ***
M               93.3       33.5    2.79   0.0083 **
Ed             180.1       52.8    3.41   0.0015 **
Po1            102.7       15.5    6.61  8.3e-08 ***
M.F             22.3       13.6    1.64   0.1087
U1           -6086.6     3339.3   -1.82   0.0762 .
U2             187.3       72.5    2.58   0.0137 *
Ineq            61.3       14.0    4.39  8.6e-05 ***
Prob         -3796.0     1490.6   -2.55   0.0151 *
---
Signif. codes:
0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 196 on 38 degrees of freedom
Multiple R-squared:  0.789,        Adjusted R-squared:  0.744
F-statistic: 17.7 on 8 and 38 DF,  p-value: 1.16e-10
```

From the summary, we are able to see the coefficients of the model, which I will use in a comparison in the next few steps.

```
#Creating the forward regression
Crime_forward <- lm(Crime~1, data=Crime_data)
forward_model <- step(Crime_forward,
    scope=formula(lm(Crime~.,data=Crime_data)),
    direction = "forward", trace=FALSE)
forward_model
```

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

Coefficients:
(Intercept)         Po1        Ineq          Ed
   -5040.5       115.0        67.7       196.5
       M        Prob          U2
    105.0    -3801.8        89.4
```

I have created the forward regression to test how well it performs against my stepwise regression.

```
#Summary of the forward regression model
summary(forward_model)
```

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

Residuals:
   Min     1Q Median     3Q    Max
-470.7  -78.4  -19.7  133.1  556.2

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  -5040.5     899.8   -5.60  1.7e-06 ***
Po1            115.0      13.8    8.36  2.6e-10 ***
Ineq            67.7      13.9    4.85  1.9e-05 ***
Ed             196.5      44.8    4.39  8.1e-05 ***
M              105.0      33.3    3.15   0.0031 **
Prob         -3801.8    1528.1   -2.49   0.0171 *
U2              89.4      40.9    2.18   0.0348 *
---
Signif. codes:
0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 201 on 40 degrees of freedom
Multiple R-squared:  0.766,        Adjusted R-squared:  0.731
F-statistic: 21.8 on 6 and 40 DF,  p-value: 3.42e-11
```

Here I can see the coefficients that resulted from the forward regression and they are different than the results of the backward regression.

```
#Implementing the stepwise regression
Crime_combine <- lm(Crime~., data = Crime_data)
combine_model <- step(Crime_combine,
    scope=list(lower=formula(lm(Crime~1,data=Crime_data)),
           upper = formula(lm(Crime~., data=Crime_data))),
    direction="both", trace=FALSE)
combine_model
```

```
Call:
lm(formula = Crime ~ M + Ed + Po1 + M.F + U1 + U2 + Ineq + Prob,
   data = Crime_data)

Coefficients:
(Intercept)          M         Ed        Po1
   -6426.1        93.3      180.1      102.7
      M.F         U1         U2       Ineq
     22.3     -6086.6      187.3       61.3
     Prob
   -3796.0
```

Here are the results of the stepwise regression in a summary form with the list of resulting coefficients.

```
#Summary of the stepwise regression
summary(combine_model)
```

```
Call:
lm(formula = Crime ~ M + Ed + Po1 + M.F + U1 + U2 + Ineq + Prob,
    data = Crime_data)

Residuals:
   Min     1Q Median     3Q    Max
 -445   -111      3    122    483

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  -6426.1     1194.6   -5.38  4.0e-06 ***
M               93.3       33.5    2.79   0.0083 **
Ed             180.1       52.8    3.41   0.0015 **
Po1            102.7       15.5    6.61  8.3e-08 ***
M.F             22.3       13.6    1.64   0.1087
U1           -6086.6     3339.3   -1.82   0.0762 .
U2             187.3       72.5    2.58   0.0137 *
Ineq            61.3       14.0    4.39  8.6e-05 ***
Prob         -3796.0     1490.6   -2.55   0.0151 *
---
Signif. codes:
0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 196 on 38 degrees of freedom
Multiple R-squared:  0.789,        Adjusted R-squared:  0.744
F-statistic: 17.7 on 8 and 38 DF,  p-value: 1.16e-10
```

Here is a summary of the stepwise regression along with extra details such as the adjusted R-squared value of 0.744.

Stepwise Regression Conclusions

```
#Calculating the AIC of the backward regression
AIC_back <- AIC(lm(formula = Crime_data$Crime~M +Ed+Po1+M.F+U1+U2+Ineq+Prob,
data=Crime_data))
AIC_back
```

```
[1] 639
```

This is the Akaike Information Criterion value of the backward regression.

```
#Calculating the AIC of the forward regression
AIC_forward <- AIC(lm(formula = Crime_data$Crime~Po1+Ineq+Ed+M+Prob+U2, data=Crime_data))
AIC_forward
```

```
[1] 640
```

This is the Akaike Information Criterion value of the forward regression.

```
#Calculating the AIC of the stepwise regression
AIC_combine <- AIC(lm(formula = Crime_data$Crime~M +Ed+Po1+M.F+U1+U2+Ineq+Prob,
data=Crime_data))
AIC_combine
```

```
[1] 639
```

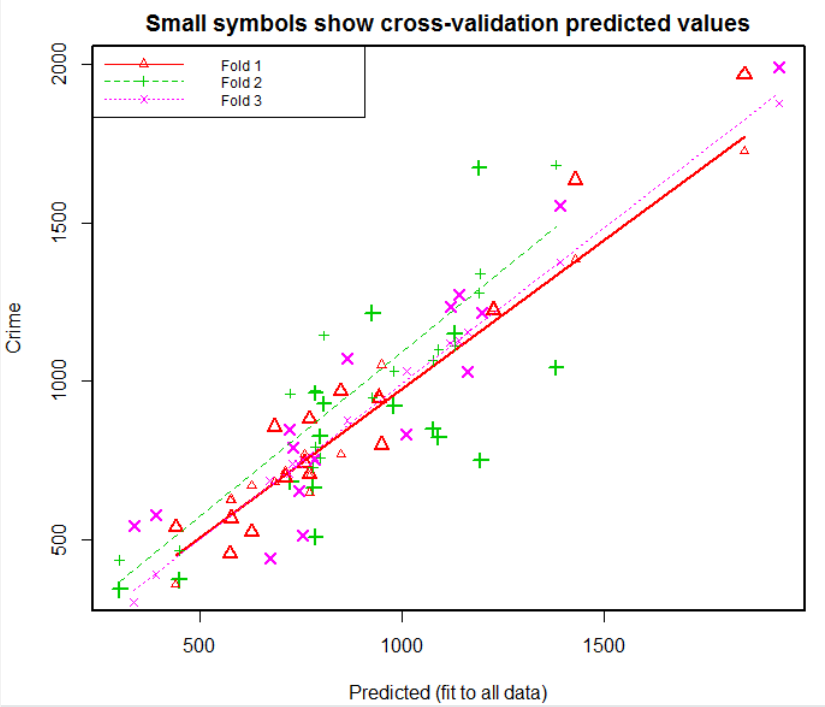
There is no significant changes in the AIC value between the different models, so I based on this, I will use a different method to evaluate the results.

```
#Viewing the AICc of the stepwise regression
AICc_combine <- AICc(lm(formula = Crime_data$Crime~M +Ed+Po1+M.F+U1+U2+Ineq+Prob,
data=Crime_data))
AICc_combine
#Viewing the BIC of the stepwise regression
BIC_combine <- BIC(lm(formula = Crime_data$Crime~M +Ed+Po1+M.F+U1+U2+Ineq+Prob,
data=Crime_data))
BIC_combine
```

```
[1] 645
[1] 658
```

Viewing the AICc and BIC value of the stepwise regression out of curiosity sake. I decided to measure the models' effectiveness based on R-squared values instead.

```
#Cross Validate
combine_model <- lm(formula = Crime~M +Ed+Po1+M.F+U1+U2+Ineq+Prob, data=Crime_data)
combine_cv <- cv.lm(data=Crime_data, form.lm=combine_model)
```



Calculating the cross validation model for the stepwise regression.

```
#Calculating the R-suqared value of the stepwise regression
combine_yhat <- as.data.frame(combine_cv$cvpred)
combine_R2 <- Rsquared(combine_yhat, Crime_data)
combine_R2
```

```
[1] 0.661
```

The stepwise regression is concluded with an R-squared value of 0.661 which is will save for a comparison against the Lasso and Elastic Net comparison. I did not include the R-squared values for the forward and backward regression because they will not be used in this homework assignment for comparison.

Lasso Calculations

```
#Scale the data
Crime_lasso <- cv.glmnet(x=as.matrix(Crime_data[,-16]),
                         y=as.matrix(Crime_data[,16]),
                         alpha=1,
                         nfolds=8,
                         nlambda=20,
                         type.measure="mse",
                         family="gaussian",
                         standardize=TRUE)

>
```

Standardize = TRUE to scale the data.

```
Crime_lasso$cvm
```

```
[1] 146563 111248  95144  90923  78927  70973  68940  69679
 [9]  67977  67673  69124  70635  71783  72439  72882  73198
[17]  73410  73535
```

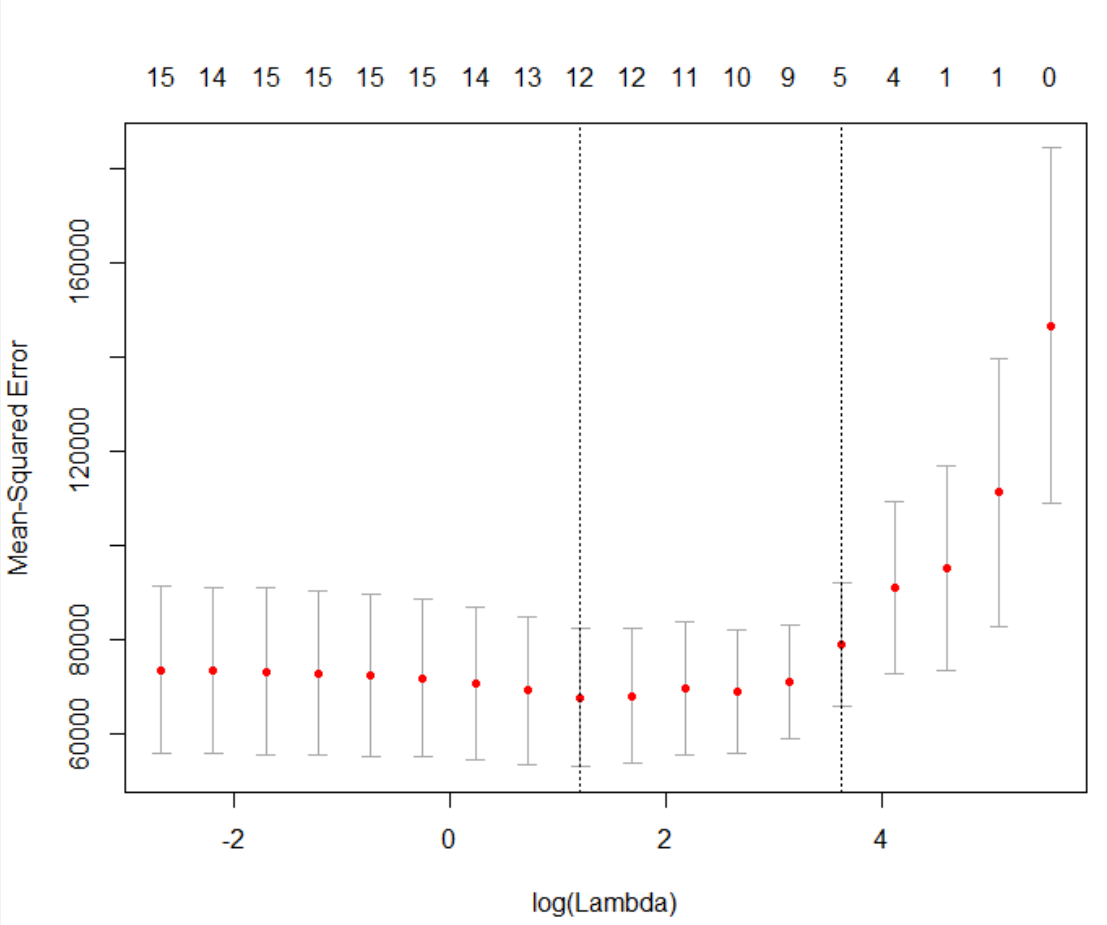
Calculates the cvm of the lasso model.

```
Crime_lasso$cvm
```

```
s15 s16 s17
 15  14  15

$name
                mse
"Mean-Squared Error"

$glmnet.fit

Call:  glmnet(x = as.matrix(week8_data[, -16]), y = as.matrix(week8_data[,      16]), alpha = 1, nlambda = 20, family = "gaussian", standardize
= TRUE)

      Df  %Dev   Lambda
 [1,]  0 0.000 263.0000
 [2,]  1 0.293 162.0000
 [3,]  1 0.405  99.8000
 [4,]  4 0.477  61.5000
 [5,]  5 0.620  37.8000
 [6,]  9 0.699  23.3000
 [7,] 10 0.747  14.4000
 [8,] 11 0.774   8.8400
 [9,] 12 0.788   5.4400
[10,] 12 0.794   3.3500
[11,] 13 0.796   2.0600
[12,] 14 0.797   1.2700
[13,] 15 0.800   0.7830
[14,] 15 0.802   0.4820
[15,] 15 0.803   0.2970
[16,] 15 0.803   0.1830
[17,] 14 0.803   0.1130
[18,] 15 0.803   0.0694
[19,] 15 0.803   0.0427
[20,] 15 0.803   0.0263

$lambda.min
[1] 3.35

$lambda.1se
[1] 37.8

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

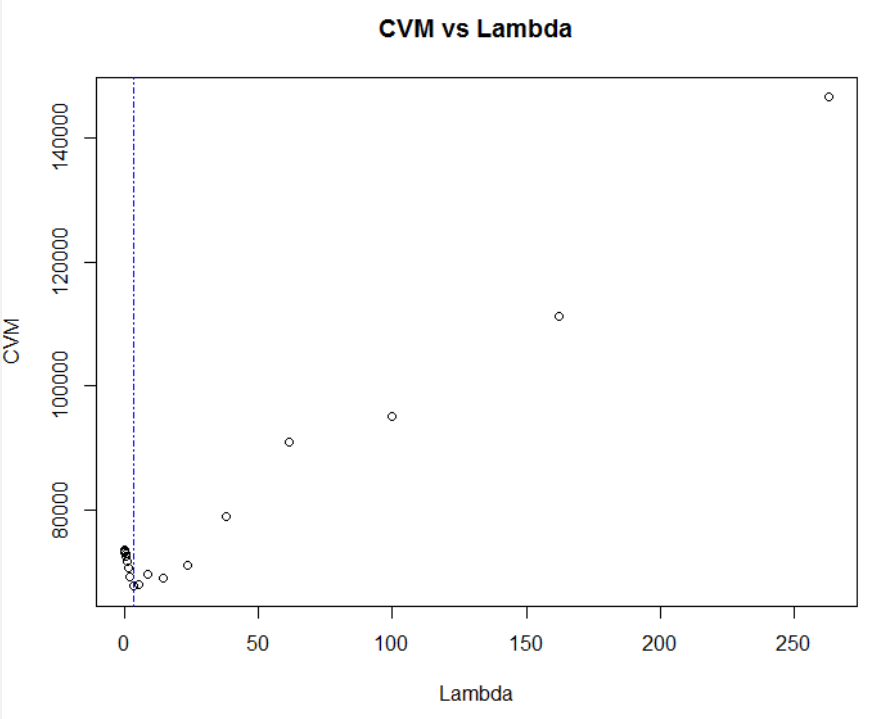
Checking the output of the lasso model to see if it matches the setup.

```
plot(Crime_lasso)
```



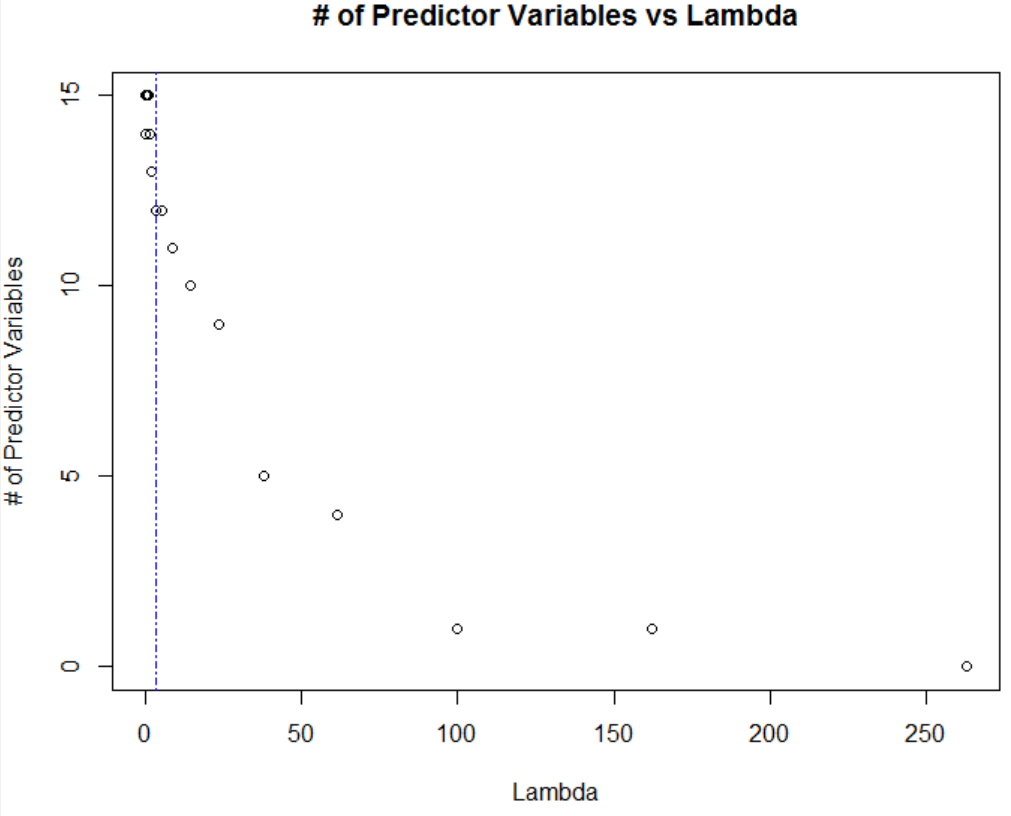
This shows the lower bounds and the upper bounds of the Lasso model. The dotted lines also shows the thresholds.

```
#Error rate
plot(Crime_lasso$lambda, Crime_lasso$cvm,
    main="CVM vs Lambda",xlab="Lambda",ylab="CVM")
abline(v=Crime_lasso$lambda.min,col="blue", lty=4)
```



This plot shows the CVM versus Lambda plot and marks the point where Lambda is at the lowest. This means that error decreases at first then increases. The min lambda is our point of interest.

```
#Error rate
plot(Crime_lasso$lambda, Crime_lasso$cvm,
    main="CVM vs Lambda",xlab="Lambda",ylab="CVM")
abline(v=Crime_lasso$lambda.min,col="blue", lty=4)
```



This plots the number of predictor variables used in the Lambda and we can see that 3-4 predictor variables are optimal.

```
#Smallest value in csm is the value for lamdba
Crime_lasso$lambda.min
```

```
[1] 3.35
```

The minimum lambda.

```
#Combining lambda and cross validation.
cbind(Crime_lasso$lambda, Crime_lasso$cvm, Crime_lasso$nzero)
```

```
      [,1]   [,2] [,3]
s0  263.0954 146563    0
s1  162.0268 111248    1
s2   99.7839  95144    1
s3   61.4518  90923    4
s4   37.8450  78927    5
s5   23.3067  70973    9
s6   14.3534  68940   10
s7    8.8395  69679   11
s8    5.4438  67977   12
s9    3.3526  67673   12
s10   2.0647  69124   13
s11   1.2715  70635   14
s12   0.7831  71783   15
s13   0.4822  72439   15
s14   0.2970  72882   15
s15   0.1829  73198   15
s16   0.1126  73410   14
s17   0.0694  73535   15
```

Here I am combining the 3 values together into one table. Lambda, CVM, and nzero.

```
#Coeffieients for the variables selected by Lasso
coef(Crime_lasso, s=Crime_lasso$lambda.min)
```

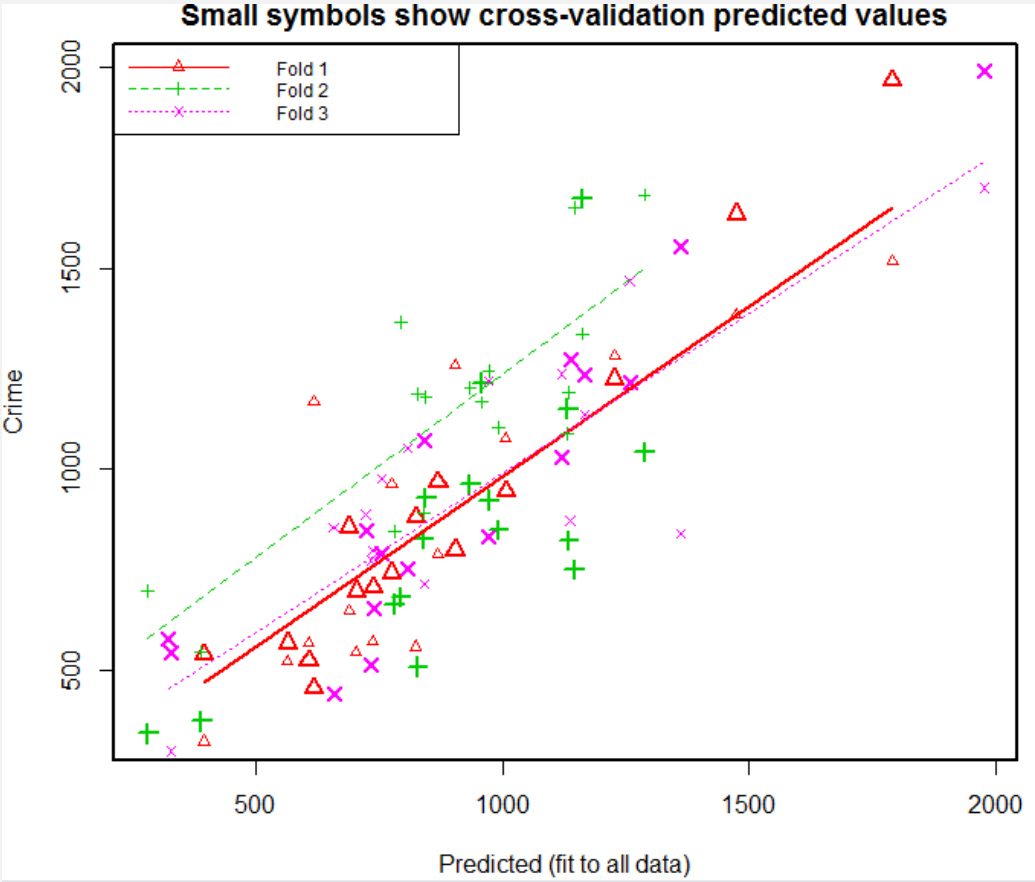
```
16 x 1 sparse Matrix of class "dgCMatrix"
                1
(Intercept) -5.91e+03
M            8.30e+01
So           3.18e+01
Ed           1.56e+02
Po1          9.96e+01
Po2          .
LF           .
M.F          1.78e+01
Pop         -4.95e-01
NW           1.39e+00
U1          -3.93e+03
U2           1.37e+02
Wealth       5.59e-02
Ineq         6.27e+01
Prob        -3.92e+03
Time         .
```

Here we see the coefficients used in the Lasso model.

```
lasso_model <- lm(formula =
Crime~M+So+Ed+Po1+Po2+LF+M.F+Pop+NW+U1+U2+Wealth+Ineq+Prob+Time, data=Crime_data)
summary(lasso_model)

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

Residuals:
   Min     1Q Median     3Q    Max
-395.7  -98.1   -6.7  113.0  512.7

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -5.98e+03   1.63e+03   -3.68  0.00089 ***
M            8.78e+01   4.17e+01    2.11  0.04344 *
So          -3.80e+00   1.49e+02   -0.03  0.97977
Ed           1.88e+02   6.21e+01    3.03  0.00486 **
Po1          1.93e+02   1.06e+02    1.82  0.07889 .
Po2         -1.09e+02   1.17e+02   -0.93  0.35883
LF          -6.64e+02   1.47e+03   -0.45  0.65465
M.F          1.74e+01   2.04e+01    0.86  0.39900
Pop         -7.33e-01   1.29e+00   -0.57  0.57385
NW           4.20e+00   6.48e+00    0.65  0.52128
U1          -5.83e+03   4.21e+03   -1.38  0.17624
U2           1.68e+02   8.23e+01    2.04  0.05016 .
Wealth       9.62e-02   1.04e-01    0.93  0.36075
Ineq         7.07e+01   2.27e+01    3.11  0.00398 **
Prob        -4.86e+03   2.27e+03   -2.14  0.04063 *
Time        -3.48e+00   7.17e+00   -0.49  0.63071
---
Signif. codes:
0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 209 on 31 degrees of freedom
Multiple R-squared:  0.803,        Adjusted R-squared:  0.708
F-statistic: 8.43 on 15 and 31 DF,  p-value: 3.54e-07
```

I used the coefficients from above to create the LM model.

```
lasso_cv <- cv.lm(data=Crime_data, form.lm=lasso_model)
```



Creating the cross validation so that I can use it to find yhat and calculate the R-squared.

Lasso Conclusions

```
#Creating the yhat value and R squared calculations
lasso_yhat <- as.data.frame(lasso_cv$cvpred)
lasso_R2 <- Rsquared(lasso_yhat, Crime_data)
lasso_R2
```

```
[1] 0.128
```

The R squares is 0.128 which is relatively low. Let's see if we can do better with the elastic net.

Elastic Net Calculations

```
#Creating a function to scale and find the elastic net and plotting the model
Crime_elasnet <- function(temp_var){
  temp_elasnet <- cv.glmnet(x=as.matrix(Crime_data[,-16]),
                            y=as.matrix(Crime_data[,16]),
                            alpha=temp_var,
                            nfolds=8,
                            nlambda=20,
                            type.measure="mse",
                            family="gaussian",
                            standardize=TRUE)
  plot(temp_elasnet$lambda, temp_elasnet$cvm,
       main="CVM vs Lambda",xlab="Lambda",ylab="CVM")
  abline(v=temp_elasnet$lambda.min,col="blue", lty=4)
  return(temp_elasnet)
}
```

```
>
```

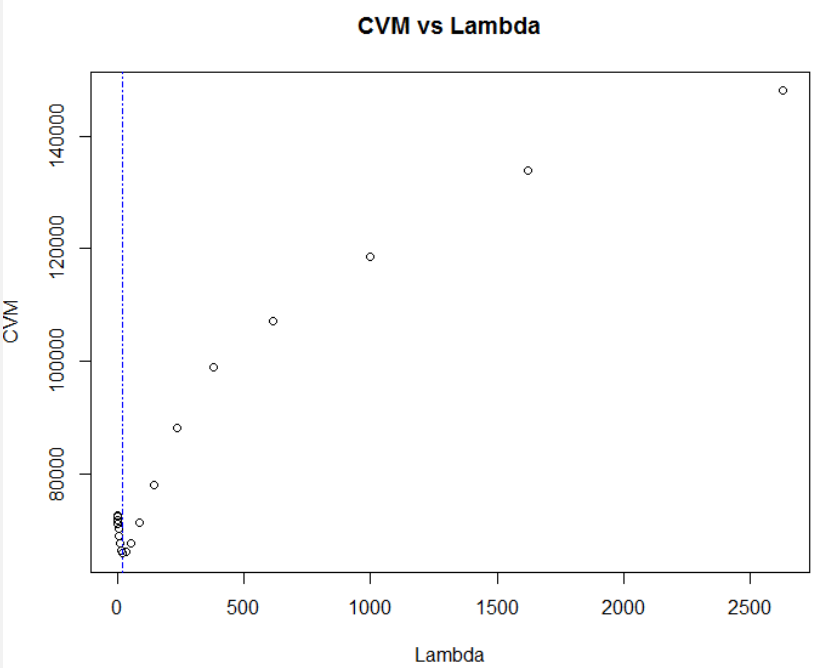
Creating a function to make it easier to compare different alphas so that we can use elastic net.

```
#Creating a function to calculate the yhat and R squared calling our first R-squared function
elasnet_R2 <- function(temp_model){
  temp_cv <- cv.lm(data=Crime_data, form.lm=temp_model)
  temp_yhat <- as.data.frame(temp_cv$cvpred)
  temp_R2 <- Rsquared(temp_yhat, Crime_data)
  return(temp_R2)
}
```

```
>
```

This function will do most of the work for me. Because I will need to figure out the coefficients of the model and use that to create a model, I cannot automate that process.

```
#Model with 0.1 alpha value
elasnet_1 <- Crime_elasnet(0.1)
```



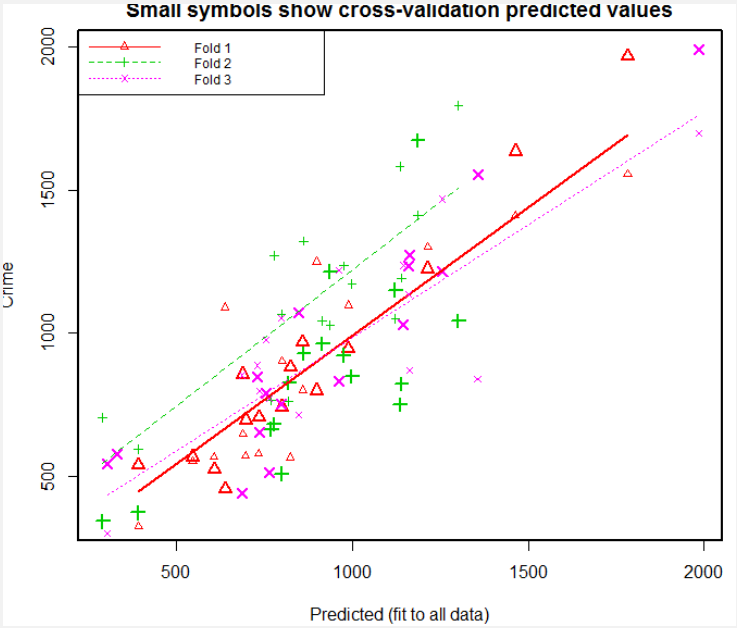
Looks normal, we will have to see how well this performs against other models.

```
coef(elasnet_1, s=elasnet_1$lambda.min)
```

```
16 x 1 sparse Matrix of class "dgCMatrix"
                    1
(Intercept) -5.54e+03
M            7.20e+01
So           6.91e+01
Ed           1.19e+02
Po1          5.93e+01
Po2          3.49e+01
LF           3.06e+02
M.F          2.23e+01
Pop         -1.10e-01
NW           2.42e+00
U1          -3.26e+03
U2           1.16e+02
Wealth       4.17e-02
Ineq         4.60e+01
Prob        -3.99e+03
Time         .
```

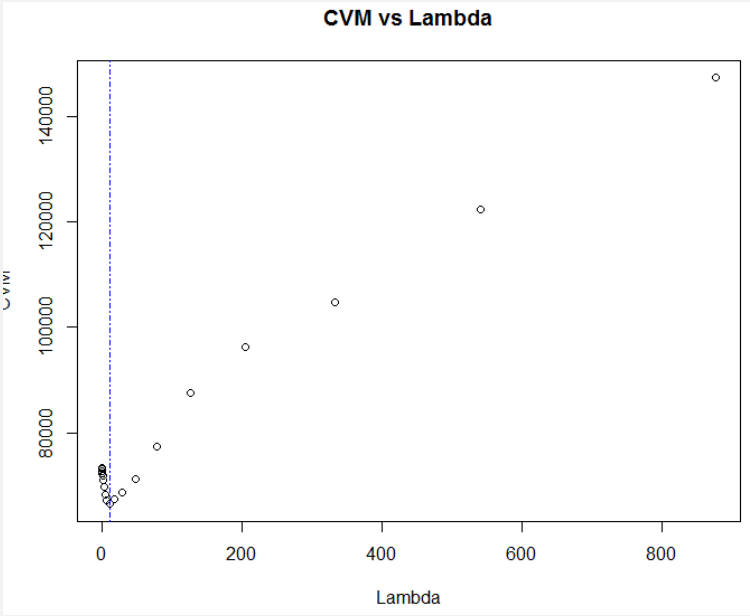
Running through the coefficients to find the model I want to use.

```
elasnet_1model <- lm(formula =
Crime~M+So+Ed+Po1+Po2+LF+M.F+Pop+NW+U1+U2+Wealth+Ineq+Prob, data=Crime_data)
model1_r2 <- elasnet_R2(elasnet_1model)
```



This is the first model. I will save the output of the R squared value for comparison.

```
#Model with 0.3 alpha value
elasnet_2 <-Crime_elasnet(0.3)
```



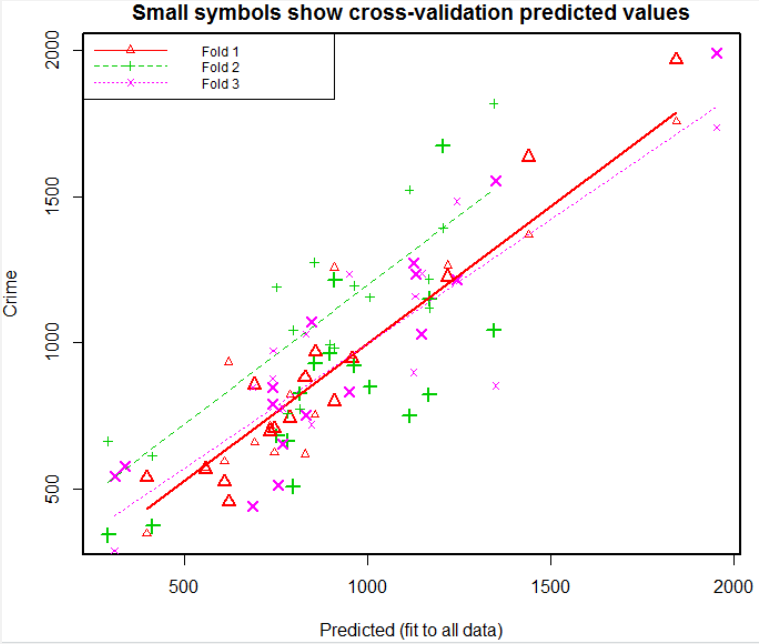
This is the cvm vs lambda plot for alpha equal 0.3.

```
coef(elasnet_2, s=elasnet_2$lambda.min)
```

```
16 x 1 sparse Matrix of class "dgCMatrix"
                    1
(Intercept) -5.71e+03
M            7.61e+01
So           4.91e+01
Ed           1.36e+02
Po1          7.53e+01
Po2          2.09e+01
LF           7.17e+01
M.F          2.13e+01
Pop         -2.01e-01
NW           1.97e+00
U1          -3.61e+03
U2           1.24e+02
Wealth       4.51e-02
Ineq         5.29e+01
Prob        -3.96e+03
Time         .
```

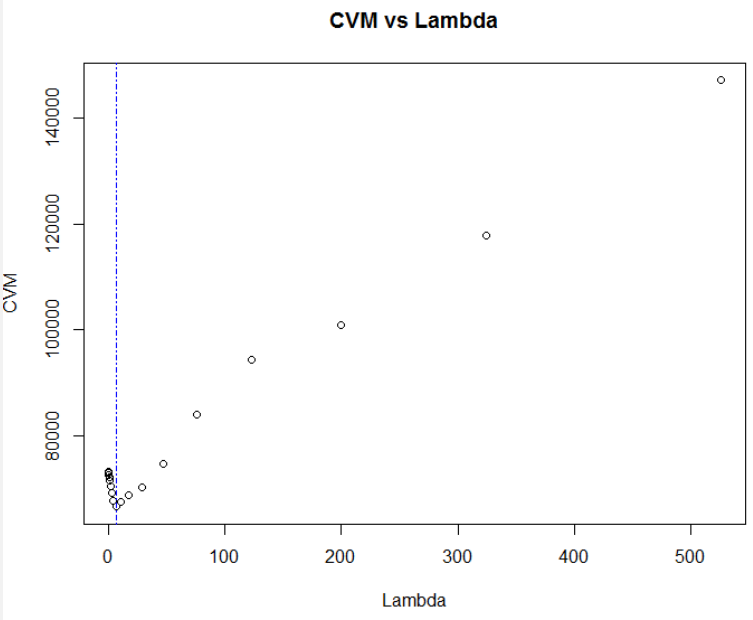
These are the coefficients of alpha equal 0.3.

```
elasnet_2model <- lm(formula = Crime~M+So+Ed+Po1+Po2+LF+M.F+NW+U1+U2+Wealth+Ineq+Prob, data=Crime_data)
model2_r2 <- elasnet_R2(elasnet_2model)
```



This is the plot of the variance. Of course I used the data in my analysis.

```
#Model with 0.5 alpha value
elasnet_3 <-Crime_elasnet(0.5)
```



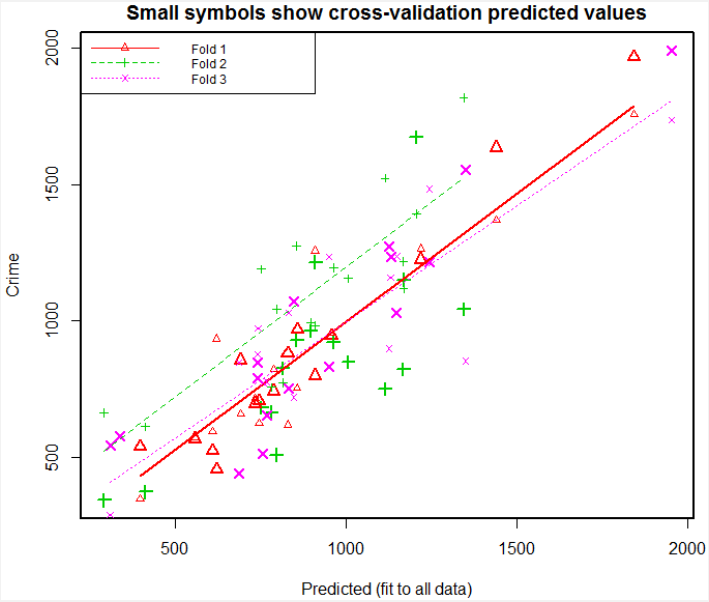
This is the plot for alpha equal 0.5.

```
coef(elasnet_3, s=elasnet_3$lambda.min)
```

```
16 x 1 sparse Matrix of class "dgCMatrix"
                    1
(Intercept) -5813.274
M             79.256
So            39.836
Ed           147.766
Po1           96.356
Po2            .
LF             .
M.F           19.671
Pop           -0.346
NW             1.834
U1         -3841.129
U2           131.296
Wealth         0.052
Ineq          57.785
Prob       -3961.445
Time           .
```

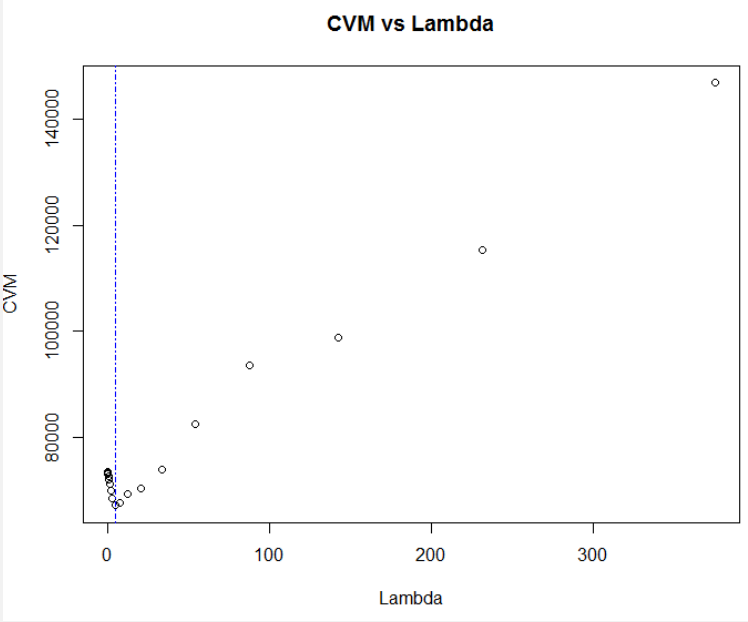
This is the list of coefficients used when alpha equals 0.5.

```
elasnet_3model <- lm(formula = Crime~M+So+Ed+Po1+Po2+LF+M.F+NW+U1+U2+Wealth+Ineq+Prob, data=Crime_data)
model3_r2 <- elasnet_R2(elasnet_3model)
```



Here is the plot for alpha 0.5.

```
#Model with 0.7 alpha value
elasnet_4 <-Crime_elasnet(0.7)
```



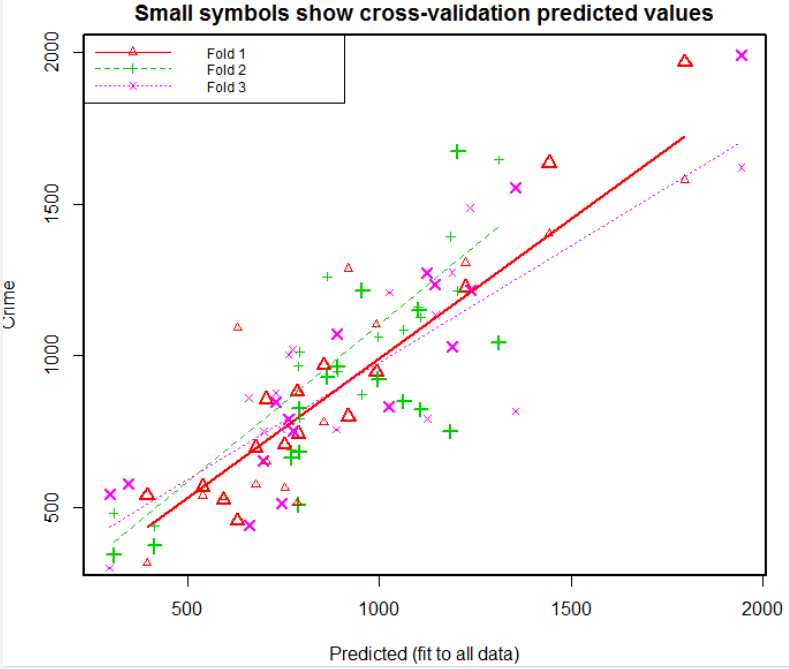
Here is the cvm vs lambda plot for alpha equal 0.7.

```
coef(elasnet_4, s=elasnet_4$lambda.min)
```

```
16 x 1 sparse Matrix of class "dgCMatrix"
                    1
(Intercept) -5.87e+03
M            8.13e+01
So           3.56e+01
Ed           1.53e+02
Po1          9.82e+01
Po2          .
LF           .
M.F          1.86e+01
Pop         -4.28e-01
NW           1.59e+00
U1          -3.90e+03
U2           1.35e+02
Wealth       5.37e-02
Ineq         6.04e+01
Prob        -3.94e+03
Time         .
```

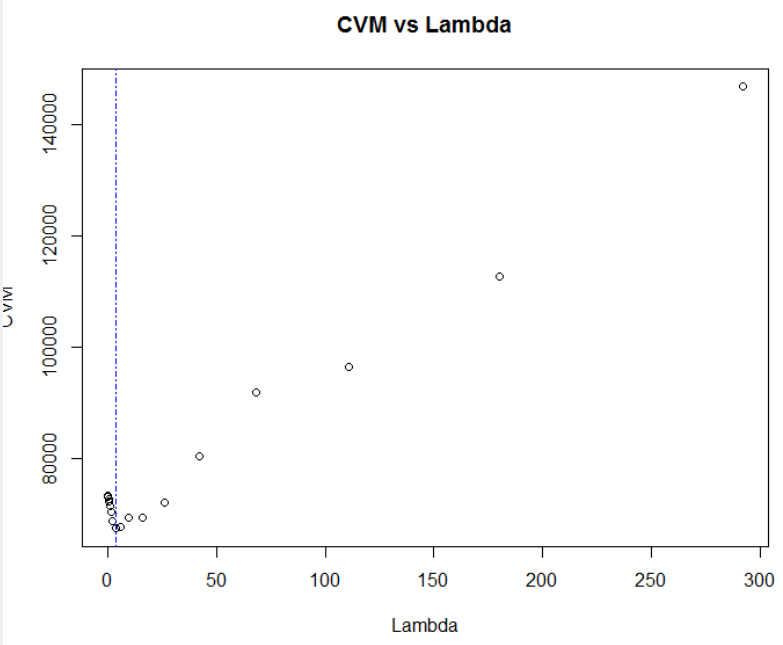
Here are the coefficients from the alpha equal 0.7 model.

```
coef(elasnet_4, s=elasnet_4$lambda.min)
```



Here is the plot of cv predicted values when alpha equals 0.7.

```
#Model with 0.9 alpha value
elasnet_5 <-Crime_elasnet(0.9)
```



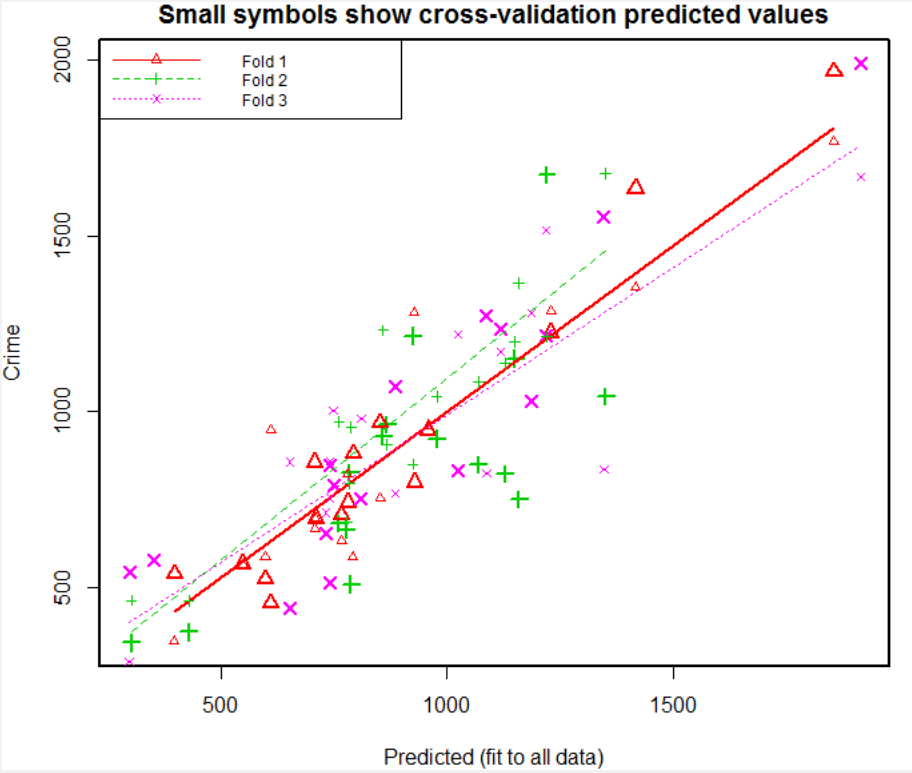
Here is the cvm vs lambda plot when alpha equals 0.9.

```
coef(elasnet_5, s=elasnet_5$lambda.min)
```

```
16 x 1 sparse Matrix of class "dgCMatrix"
                    1
(Intercept) -5.90e+03
M            8.26e+01
So           3.28e+01
Ed           1.55e+02
Po1          9.93e+01
Po2          .
LF           .
M.F          1.80e+01
Pop         -4.77e-01
NW           1.45e+00
U1          -3.92e+03
U2           1.37e+02
Wealth       5.53e-02
Ineq         6.21e+01
Prob        -3.93e+03
Time         .
```

Here are the coefficients when alpha equals 0.9.

```
elasnet_5model <- lm(formula = Crime~M+So+Ed+Po1+M.F+NW+U1+U2+Wealth+Ineq+Prob,
data=Crime_data)
model5_r2 <- elasnet_R2(elasnet_5model)
```



Here is the cross validation plot for alpha equal 0.9.

Elastic Net Conclusion

```
#Alpha = 0.1
model1_r2
```

```
[1] 0.18
```

```
#Alpha = 0.3
model2_r2
```

```
[1] 0.276
```

```
#Alpha = 0.5
model3_r2
```

```
[1] 0.276
```

```
#Alpha = 0.7
model4_r2
```

```
[1] 0.299
```

```
#Alpha = 0.9
model5_r2
```

```
[1] 0.37
```

These are the different R squared values calculated in the previous Elastic Net Calculations section. As we can see the best model output appears to be model 5 which was when I used alpha equal 0.9. The default value for alpha is 1 ,which we saw in the lasso method.

Overall Conclusion

The R squared value of the Stepwise Regression was 0.661 (page 10) which is pretty high. This means that the model fit the data pretty well. This may have resulted in a bit of overfitting. The Lasso model performed relatively low at 0.128 (page 18). And with Elastic net we were able to get as high as 0.37 (page 28) for the R squared value.

Overall I would pick the Elastic Net solution because it performed relatively well without having a sense of overfitting. Also, Elastic Net went through a more rigorous analysis.