

## Question 11.1

### 1. Stepwise Regression

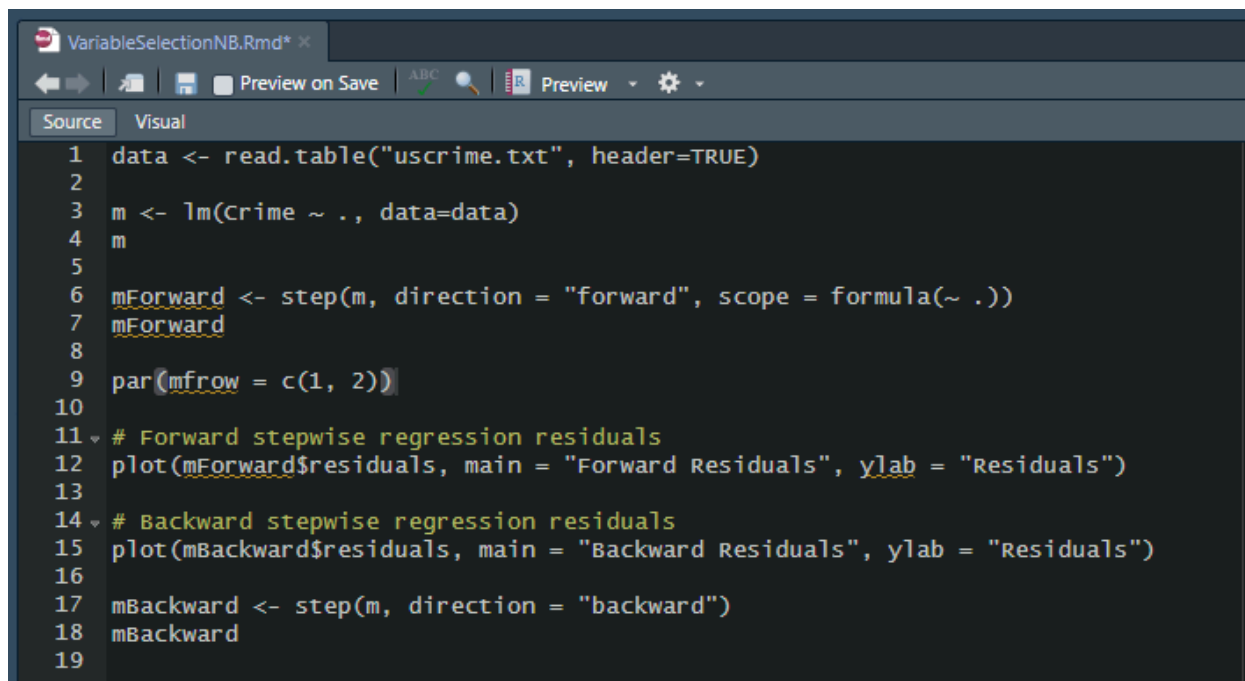
I ran the stepwise regression model both forwards and backwards to determine the best course of action for variable selection. Figure 1 depicts the code used to run these models and details a little more specifically on what that entails. The forward regression model (Figure 2) shows absolutely no difference between running the normal linear regression model, and the model again after forward stepwise variable selection. However, the backward stepwise regression showed a much more promising solution as it excluded variables for the purpose of combating overfitting, and it produced a set with a smaller AIC value of 503.93 compared to its forward counterpart of 514.65. I believe that starting with all of the values in the backwards stepwise regression can lead to better results because the selection has all of the information on the variables already, and drops the least influential ones. Whereas in forward stepwise regression, it selects factors based on the variables chosen before it. So there is a chance it may leave out important variables because of the configuration it selects before coming to a certain variable. Or, it may decide that all variables contribute in some way that helps the data and includes all of them, which is what is seen in Figure 2. Finally, Figure 4 is the graph of the residuals for both the forward and backward iteration of the regression models, the figure caption goes into a little more detail on how to interpret these results.

### 2. Lasso

The lasso approach was used to identify and resolve the multicollinearity between the predictor values of the uscrime dataset. This approach came to fruition as visualized in Figures 7 and 7.1. Before I get into the analysis of the coefficients, it is important to note that the approach used in Figure 7 with k-fold cross validation attempted to find the best lambda possible by minimizing the mean squared error. This follows what Dr. Sokol mentioned in the lectures, and it seems to have worked solely based on the clustering of data points around Value = 0.0 in Figure 7.1. Many of the predictors that are in the uscrime dataset are not present in this model. That is because of how the lasso method had determined that their relevance to the model is essentially negligible, and thus in the regression their coefficient value is set to 0.

### 3. Elastic Net

Similarly to the lasso approach used above, the elastic net attempts to use the same function, but with the inclusion of the squares of the coefficients. This helps to include some variables that may have been set to 0 by the lasso method, but not all of them. And this is based on the magnitude of the alpha value selected for this model. For this assignment, I used three different alpha values (0.25, 0.5, 0.75) to visualize the difference between the models, and see how the coefficients were dealt with. Also similar to the lasso method, the optimal lambda value was found in the same k-fold cross validation procedure, as shown in Figure 8. As the alpha values trended upwards, the more the variable selection chose to set values equal to 0 rather than shrink them towards 0. This is to be expected as an alpha value of 1 is tied to the lasso method, and thus the diamond shape that correlates with setting predictors equal to 0. And an alpha value of 0 is indicative of ridge regression, which does set values equal to 0, but shrinks them towards it. Of course differing alpha values affects coefficients in the different ways and based on the graphs generated in the Figures 9.1, 10.1, and 11.1, it would seem that an alpha value of 0.5 (Figure 9.1) would be best suited for this model as it clusters the coefficients more closely around the Value = 0.0.

The image shows a screenshot of the RStudio interface. The top pane displays the source code for a file named 'VariableSelectionNB.Rmd'. The code is written in R and performs the following steps: 1. Reads a table from 'uscrime.txt' with headers. 2. Fits a linear model 'm' with 'Crime' as the response variable. 3. Performs forward stepwise regression using the 'step' function, creating a model 'mForward'. 4. Plots the residuals of 'mForward' with the title 'Forward Residuals'. 5. Performs backward stepwise regression using the 'step' function, creating a model 'mBackward'. 6. Plots the residuals of 'mBackward' with the title 'Backward Residuals'. The code is numbered from 1 to 19. The bottom pane is currently empty.

```
1 data <- read.table("uscrime.txt", header=TRUE)
2
3 m <- lm(Crime ~ ., data=data)
4 m
5
6 mForward <- step(m, direction = "forward", scope = formula(~ .))
7 mForward
8
9 par(mfrow = c(1, 2))
10
11 # Forward stepwise regression residuals
12 plot(mForward$residuals, main = "Forward Residuals", ylab = "Residuals")
13
14 # Backward stepwise regression residuals
15 plot(mBackward$residuals, main = "Backward Residuals", ylab = "Residuals")
16
17 mBackward <- step(m, direction = "backward")
18 mBackward
19
```

**Figure 1.** This figure denotes the code used to run the stepwise regression models featured in the following figures. The variables names are labeled based on the direction of stepwise regression.

```

> data <- read.table("uscrime.txt", header=TRUE)
>
> m <- lm(Crime ~ ., data=data)
> m

Call:
lm(formula = Crime ~ ., data = data)

Coefficients:
(Intercept)      M      So      Ed      Po1      Po2      LF      M.F
-5.984e+03  8.783e+01 -3.803e+00  1.883e+02  1.928e+02 -1.094e+02 -6.638e+02  1.741e+01
      Pop      NW      U1      U2      wealth      Ineq      Prob      Time
-7.330e-01  4.204e+00 -5.827e+03  1.678e+02  9.617e-02  7.067e+01 -4.855e+03 -3.479e+00

>
> mForward <- step(m, direction = "forward", scope = formula(~.))
Start: AIC=514.65
Crime ~ M + So + Ed + Po1 + Po2 + LF + M.F + Pop + NW + U1 +
      U2 + Wealth + Ineq + Prob + Time

> mForward

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

Coefficients:
(Intercept)      M      So      Ed      Po1      Po2      LF      M.F
-5.984e+03  8.783e+01 -3.803e+00  1.883e+02  1.928e+02 -1.094e+02 -6.638e+02  1.741e+01
      Pop      NW      U1      U2      wealth      Ineq      Prob      Time
-7.330e-01  4.204e+00 -5.827e+03  1.678e+02  9.617e-02  7.067e+01 -4.855e+03 -3.479e+00

```

**Figure 2.** This figure is sharing the values generated from the forward stepwise regression and the base linear regression model that is ran off of the uscrime dataset.

```

> mBackward <- step(m, direction = "backward")
Start: AIC=514.65
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 512.65
- LF      1     8917 1363862 512.96
- Time    1    10304 1365250 513.00
- Pop     1    14122 1369068 513.14
- NW      1    18395 1373341 513.28
- M.F     1    31967 1386913 513.74
- Wealth  1    37613 1392558 513.94
- Po2     1    37919 1392865 513.95
<none>                 1354946 514.65
- U1      1    83722 1438668 515.47
- Po1     1   144306 1499252 517.41
- U2      1   181536 1536482 518.56
- M       1   193770 1548716 518.93
- Prob    1   199538 1554484 519.11
- Ed      1   402117 1757063 524.86
- Ineq    1   423031 1777977 525.42

Step: AIC=512.65
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 511.01
- LF      1    10878 1365852 511.03
- Pop     1    14127 1369101 511.14
- NW      1    21626 1376600 511.39
- M.F     1    32449 1387423 511.76
- Po2     1    37954 1392929 511.95
- Wealth  1    39223 1394197 511.99
<none>                 1354974 512.65
- U1      1    96420 1451395 513.88
- Po1     1   144302 1499277 515.41
- U2      1   189859 1544834 516.81
- M       1   195084 1550059 516.97
- Prob    1   204463 1559437 517.26
- Ed      1   403140 1758114 522.89
- Ineq    1   488834 1843808 525.13

Step: AIC=511.01
Crime ~ M + Ed + Po1 + Po2 + LF + M.F + Pop + NW + U1 + U2 +
      Wealth + Ineq + Prob

```

**Figure 3.** This figure is a small example of the process behind backwards stepwise regression and how it differs from forward regression. This process continued until the end result in Figure 4 was achieved.

```

Step: AIC=503.93
Crime ~ M + Ed + Po1 + M.F + U1 + U2 + Ineq + Prob

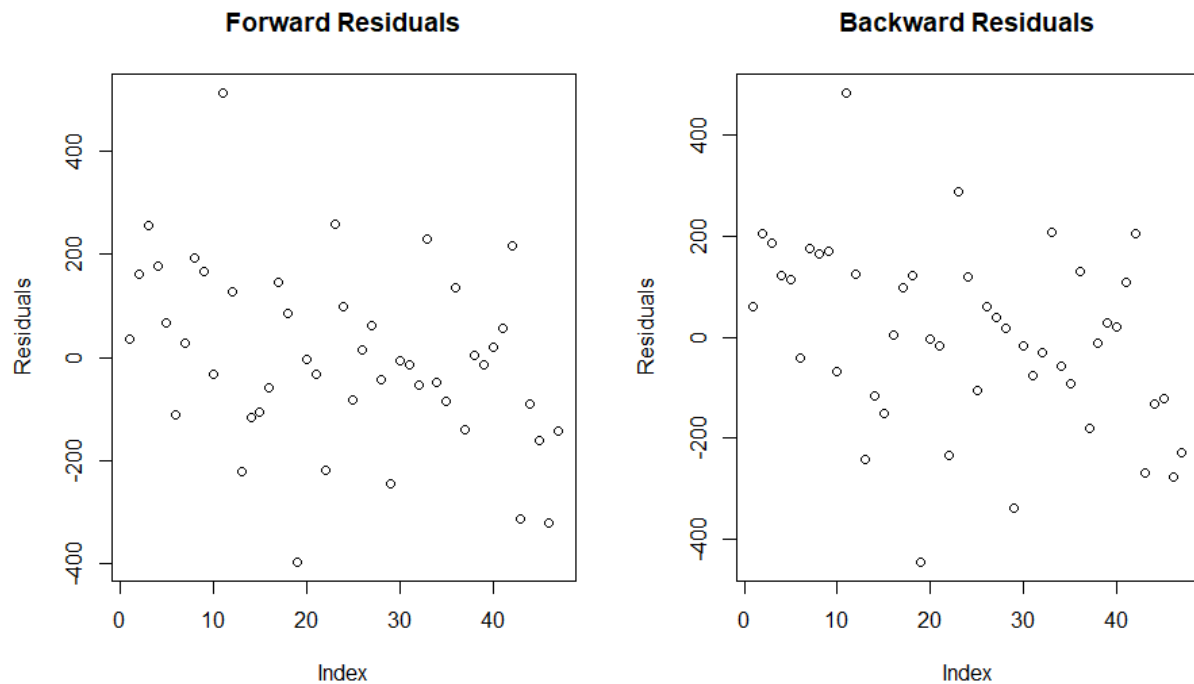
      Df Sum of Sq  RSS   AIC
<none>            1453068 503.93
- M.F    1    103159 1556227 505.16
- U1     1    127044 1580112 505.87
- Prob   1    247978 1701046 509.34
- U2     1    255443 1708511 509.55
- M      1    296790 1749858 510.67
- Ed     1    445788 1898855 514.51
- Ineq   1    738244 2191312 521.24
- Po1    1   1672038 3125105 537.93
> mBackward

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

Coefficients:
(Intercept)          M          Ed          Po1          M.F          U1          U2          Ineq          Prob
-6426.10         93.32        180.12        102.65         22.34       -6086.63        187.35         61.33       -3796.03

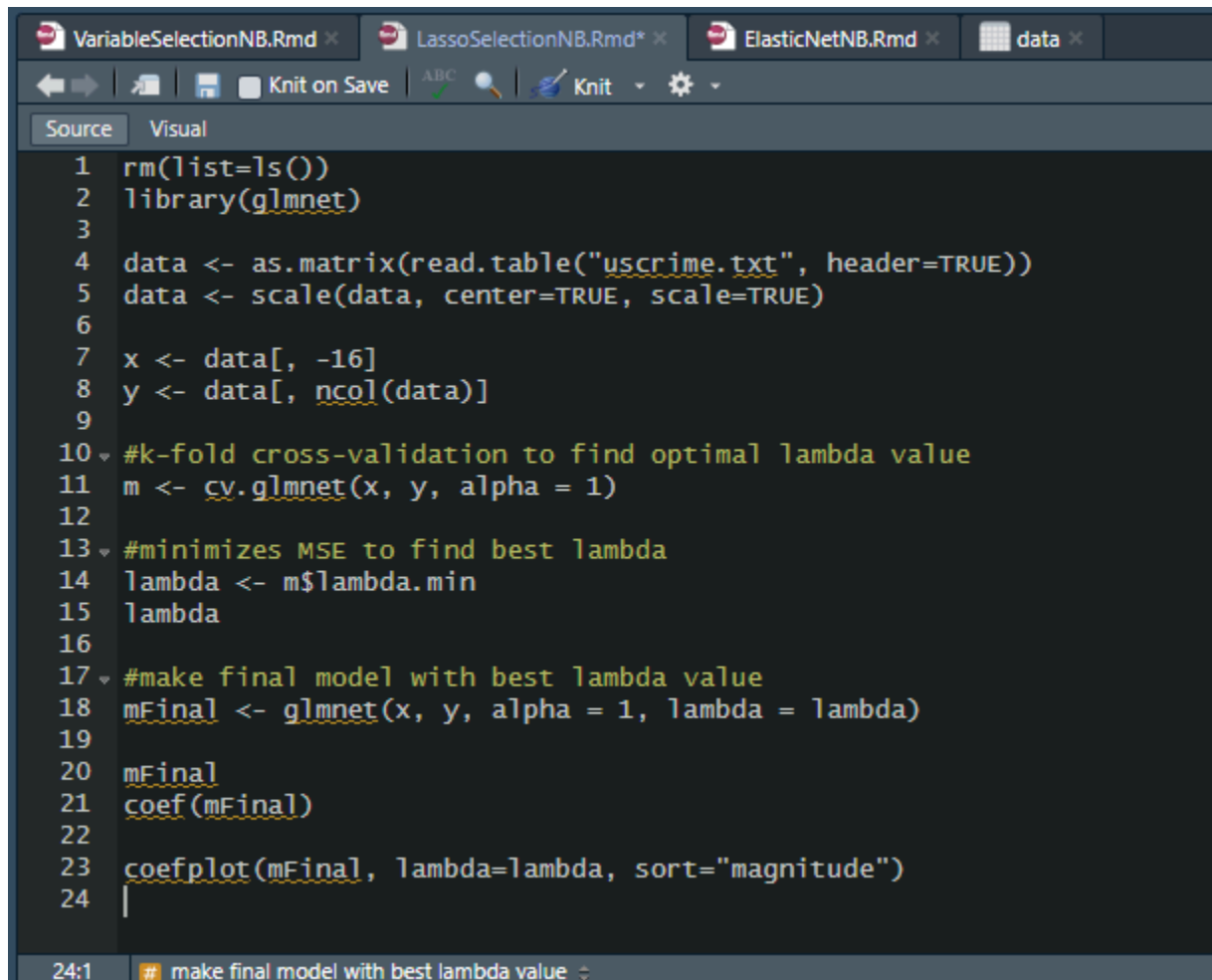
```

**Figure 4.** This figure is the model output from the backward stepwise regression. Noticeably it has lower AIC and fewer variables selected for the model.



**Figure 5.** This figure shows the side by side plots of the residuals from both the forward (left) and backward (right) stepwise regression models generated in the aforementioned figures. The

backward regression model is clustered much closer around the Residuals = 0 line, showing a better fit than the forward model.



```
1 rm(list=ls())
2 library(glmnet)
3
4 data <- as.matrix(read.table("uscrime.txt", header=TRUE))
5 data <- scale(data, center=TRUE, scale=TRUE)
6
7 x <- data[, -16]
8 y <- data[, ncol(data)]
9
10 #k-fold cross-validation to find optimal lambda value
11 m <- cv.glmnet(x, y, alpha = 1)
12
13 #minimizes MSE to find best lambda
14 lambda <- m$lambda.min
15 lambda
16
17 #make final model with best lambda value
18 mFinal <- glmnet(x, y, alpha = 1, lambda = lambda)
19
20 mFinal
21 coef(mFinal)
22
23 coefplot(mFinal, lambda=lambda, sort="magnitude")
24 |
```

24:1 # make final model with best lambda value

**Figure 6.** This figure denotes the code used to run the lasso variable selection model featured in the following figures. Notice that the alpha value is = 1, and the best lambda was chosen by minimizing the MSE as noted by Dr. Sokol in the lectures.

```

> #k-fold cross-validation to find optimal lambda value
> m <- cv.glmnet(x, y, alpha = 1)
>
> #minimizes MSE to find best lambda
> lambda <- m$lambda.min
> lambda
[1] 0.02388489
>
> #make final model with best lambda value
> mFinal <- glmnet(x, y, alpha = 1, lambda = lambda)
>
> mFinal

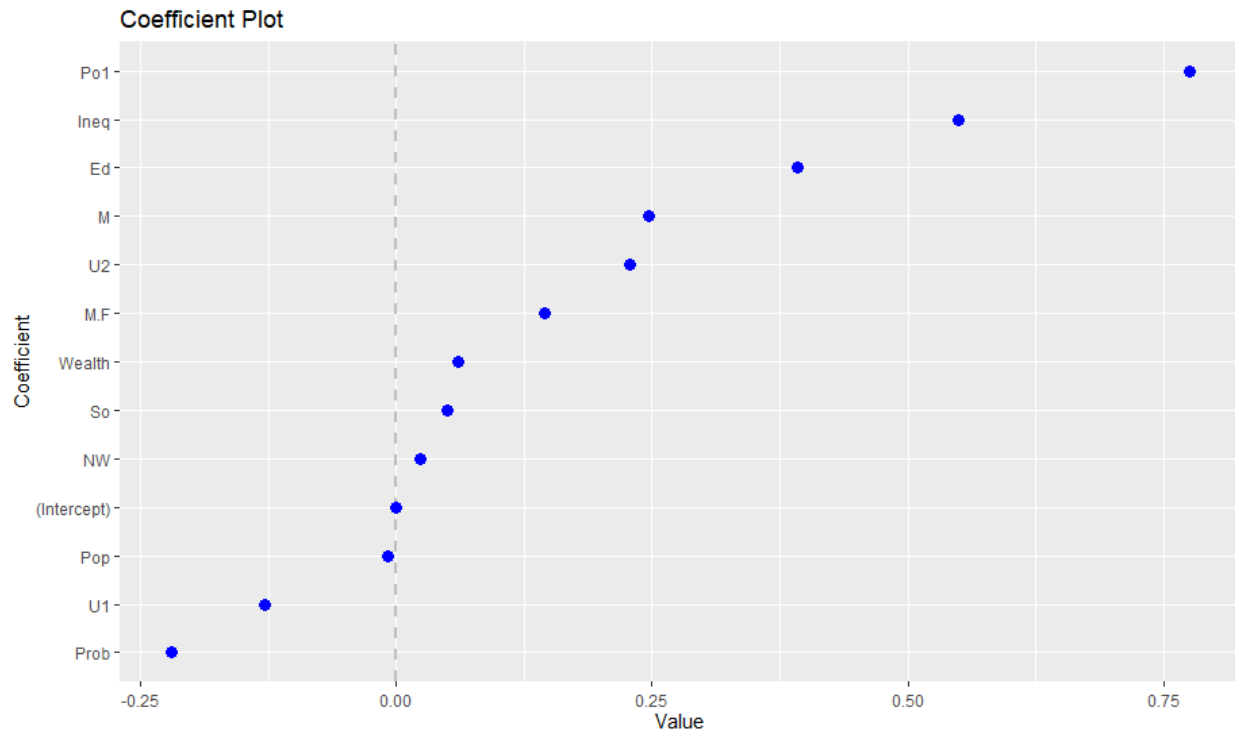
Call:  glmnet(x = x, y = y, alpha = 1, lambda = lambda)

      Df %Dev  Lambda
1 11 77.26 0.02388
> coef(mFinal)
16 x 1 sparse Matrix of class "dgCMatrix"
              s0
(Intercept) -3.248582e-16
M            2.306956e-01
So           5.432171e-02
Ed           3.562515e-01
Po1          7.888951e-01
Po2          .
LF           .
M.F          1.422196e-01
Pop          .
NW           1.623304e-02
U1           -9.277461e-02
U2           1.845836e-01
wealth       1.557059e-02
Ineq         4.985684e-01
Prob        -2.155591e-01
Time        .
> |

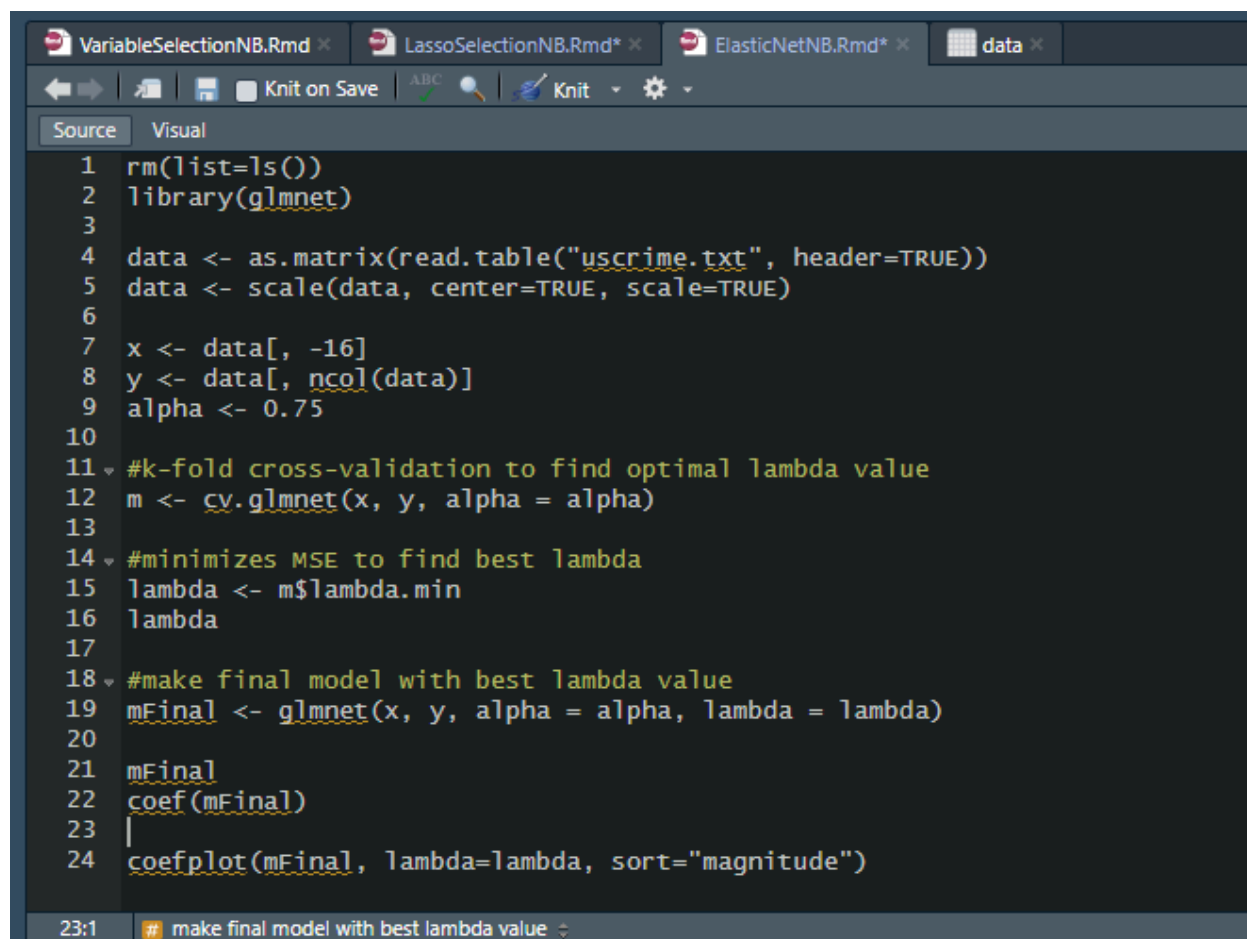
```

**Figure 7.** This figure is a screenshot of the glmnet model ran after variable selection was concluded by the lasso method. Many of the coefficients have no numerical value associated with them, this is because the item was not influential enough in the eyes of the model.





**Figure 7.1.** This figure shows the plot of the coefficients after running the lasso variable selection model. Notice how the variables that were assigned no numerical coefficients are left off.

The image shows a screenshot of the RStudio IDE. At the top, there are three tabs: 'VariableSelectionNB.Rmd', 'LassoSelectionNB.Rmd', and 'ElasticNetNB.Rmd'. The 'ElasticNetNB.Rmd' tab is active. Below the tabs is a toolbar with icons for navigation and execution, including 'Knit on Save', 'ABC', and 'Knit'. The main editor area is titled 'Source' and contains R code. The code is as follows:

```
1 rm(list=ls())
2 library(glmnet)
3
4 data <- as.matrix(read.table("uscrime.txt", header=TRUE))
5 data <- scale(data, center=TRUE, scale=TRUE)
6
7 x <- data[, -16]
8 y <- data[, ncol(data)]
9 alpha <- 0.75
10
11 #k-fold cross-validation to find optimal lambda value
12 m <- cv.glmnet(x, y, alpha = alpha)
13
14 #minimizes MSE to find best lambda
15 lambda <- m$lambda.min
16 lambda
17
18 #make final model with best lambda value
19 mFinal <- glmnet(x, y, alpha = alpha, lambda = lambda)
20
21 mFinal
22 coef(mFinal)
23 |
24 coefplot(mFinal, lambda=lambda, sort="magnitude")
```

At the bottom of the editor, there is a status bar showing '23:1' and a comment '# make final model with best lambda value'.

**Figure 8.** This figure denotes the code used to run the elastic net variable selection model featured in the following figures. Notice that the alpha value is = 0.5, and the best lambda was chosen by minimizing the MSE as noted by Dr. Sokol in the lectures. This code was run multiple times with differing alpha values, as specified in the following figure captions.

```

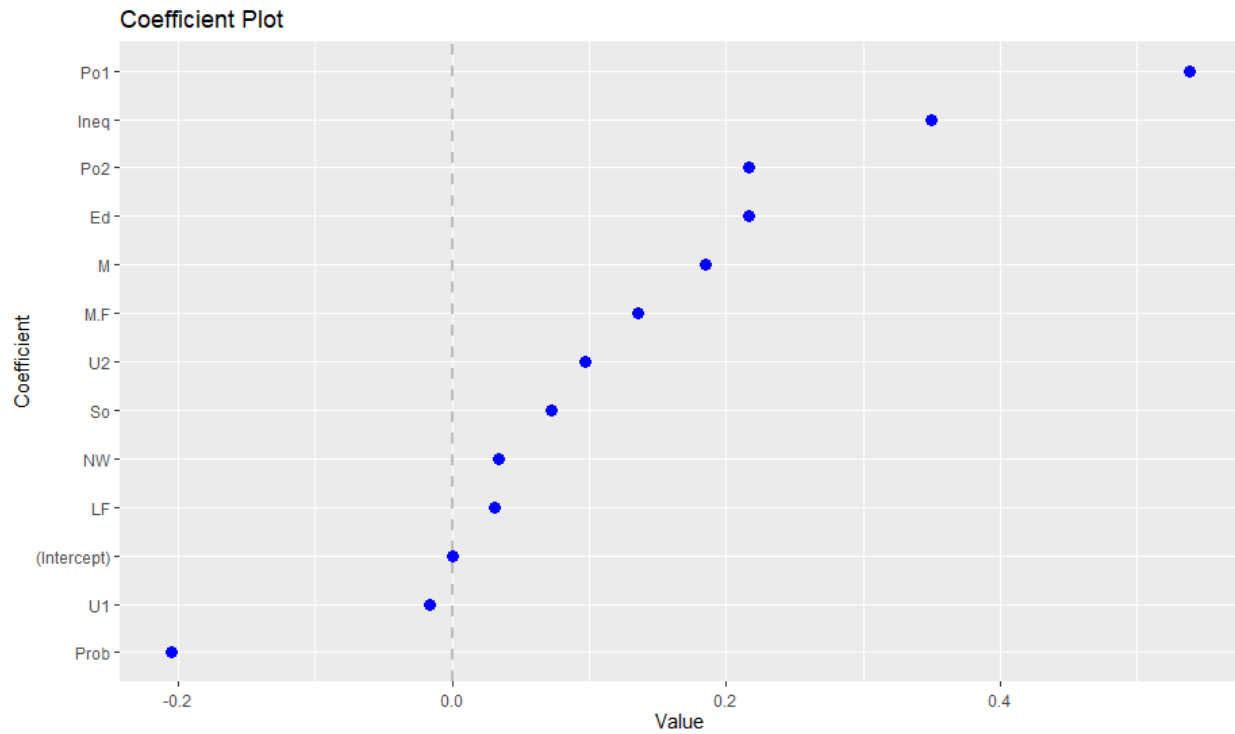
> lambda <- m$lambda.min
> lambda
[1] 0.0249072
>
> #make final model with best lambda value
> mFinal <- glmnet(x, y, alpha = 0.5, lambda = lambda)
>
> mFinal

call: glmnet(x = x, y = y, alpha = 0.5, lambda = lambda)

   Df %Dev Lambda
1 13 78.47 0.02491
> coef(mFinal)
16 x 1 sparse Matrix of class "dgCMatrix"
               s0
(Intercept) -3.526914e-16
M             2.453579e-01
So            5.518723e-02
Ed            3.936719e-01
Po1           6.861249e-01
Po2           5.186087e-02
LF            .
M.F           1.585780e-01
Pop          -1.359269e-02
NW            4.539580e-02
U1           -1.565928e-01
U2            2.550623e-01
wealth        9.632894e-02
Ineq          5.443236e-01
Prob         -2.284561e-01
Time          .
> |

```

**Figure 9.** This figure is a screenshot of the glmnet model ran after variable selection was concluded by the elastic net method at  $\alpha=0.5$ . Some of the coefficients have no numerical value associated with them, this is because the item was not influential enough in the eyes of the model. However, it differs from the lasso net model in that not there is more shrinkage of the coefficients rather than just setting them equal to 0.



**Figure 9.1.** This is the graph of the coefficients when the alpha value is equal to 0.5. They are fairly close to being bundled around Value = 0.0.

```

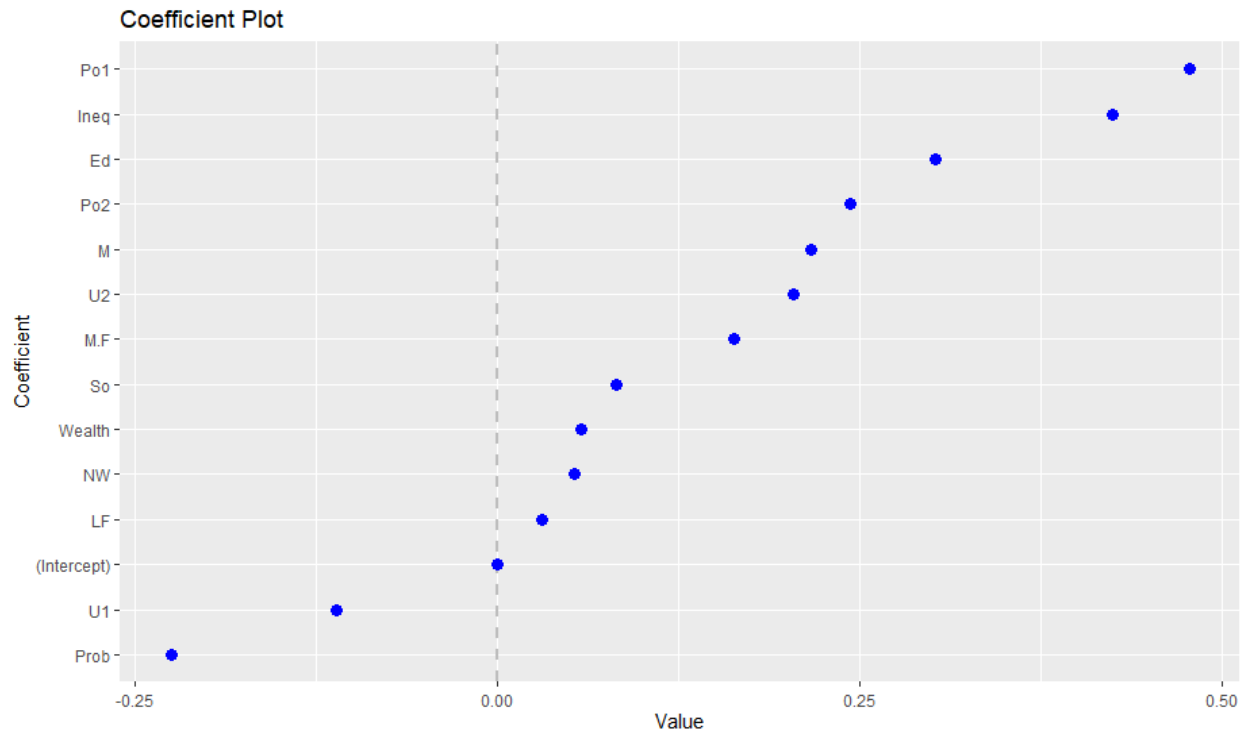
> lambda
[1] 0.07227217
>
> #make final model with best lambda value
> mFinal <- glmnet(x, y, alpha = alpha, lambda = lambda)
>
> mFinal

Call:  glmnet(x = x, y = y, alpha = alpha, lambda = lambda)

      Df %Dev  Lambda
1 13 75.7 0.07227
> coef(mFinal)
16 x 1 sparse Matrix of class "dgCMatrix"
              s0
(Intercept) -3.745426e-16
M            2.048714e-01
So           8.419568e-02
Ed           2.737007e-01
Po1          4.622282e-01
Po2          2.573977e-01
LF           3.634678e-02
M.F          1.589576e-01
Pop          .
NW           5.472262e-02
U1          -9.128023e-02
U2           1.814571e-01
wealth       4.244671e-02
Ineq         3.882248e-01
Prob        -2.213182e-01
Time        .
> |

```

**Figure 10.** This figure is a screenshot of the glmnet model ran after variable selection was concluded by the elastic net method at  $\alpha=0.25$ . Some of the coefficients have no numerical value associated with them, this is because the item was not influential enough in the eyes of the model. Noticeably, there are changes in the coefficients of the model compared to when  $\alpha=0.5$ .



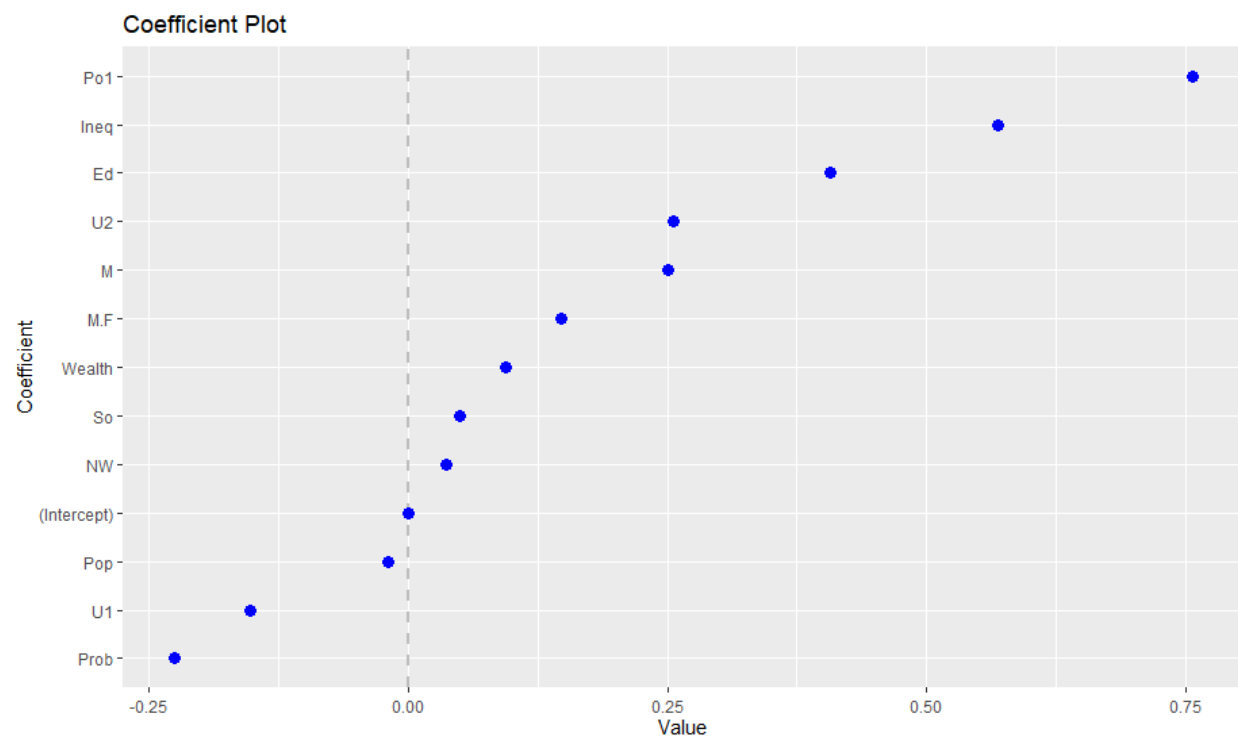
**Figure 10.1.** This is the graph of the coefficients when the alpha value is equal to 0.25.

```
Console Background Jobs x
R 4.4.1 · C:/Users/Toshan/Desktop/OmsaGT/ISYE6501/VariableSelection/ ↗
> lambda
[1] 0.02000054
>
> #make final model with best lambda value
> mFinal <- glmnet(x, y, alpha = alpha, lambda = lambda)
>
> mFinal

Call:  glmnet(x = x, y = y, alpha = alpha, lambda = lambda)

      Df %Dev Lambda
1 12 78.49  0.02
> coef(mFinal)
16 x 1 sparse Matrix of class "dgCMatrix"
              s0
(Intercept) -3.343330e-16
M            2.470476e-01
So           5.156054e-02
Ed           3.984165e-01
Po1          7.559043e-01
Po2          .
LF           .
M.F          1.493731e-01
Pop          -1.399054e-02
NW           3.543229e-02
U1           -1.450620e-01
U2           2.454619e-01
wealth       8.289418e-02
Ineq         5.538711e-01
Prob        -2.247154e-01
Time         .
> |
```

**Figure 11.** This figure is a screenshot of the glmnet model ran after variable selection was concluded by the elastic net method at  $\alpha=0.75$ . Some of the coefficients have no numerical value associated with them, this is because the item was not influential enough in the eyes of the model. Noticeably, there are changes in the coefficients of the model compared to when  $\alpha=0.25$  and  $0.5$ .



**Figure 11.1** This is the graph of the coefficients when the alpha value is equal to 0.75.