Question 11.1

Using the crime data set 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.

Response 11.1.1

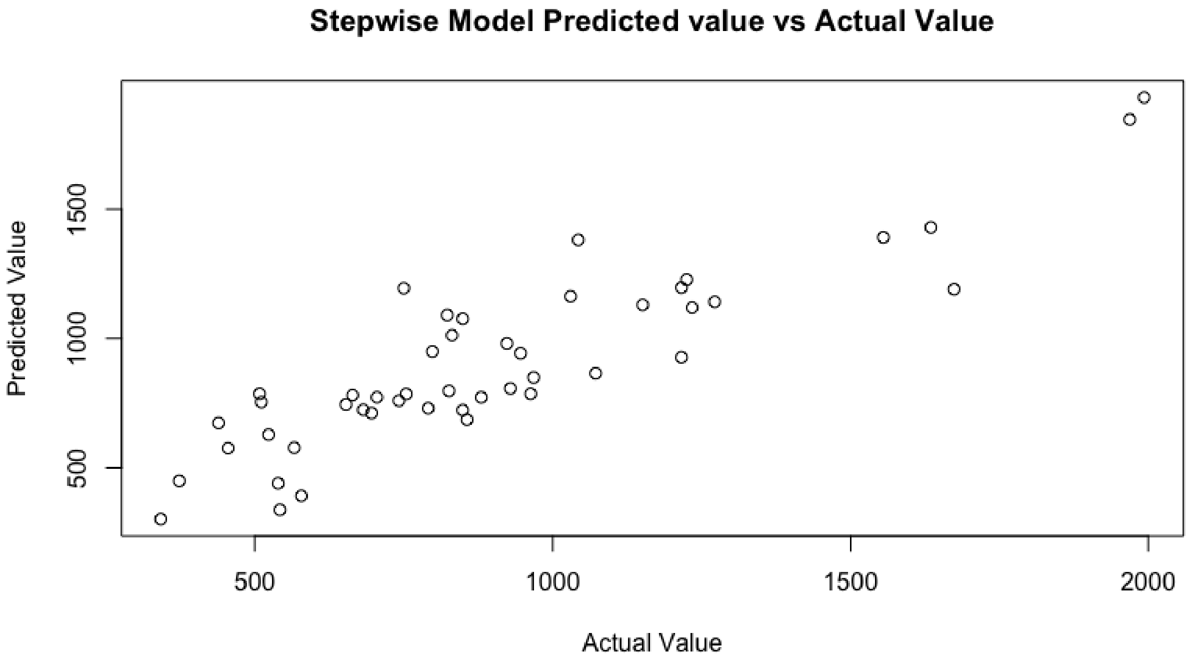
To begin solving the first question, I first read in the *crime data* data set, and saved it as *crime_data.* I next standardized the numeric predictor variables in the data set using R's *scale* function. The standardized variables were creating by using the formula $(x_i - \mu)/\sigma$. Variable standardization isn't critical in stepwise regression, but standardization or normalization is always a good idea when running a regression model, even if it isn't mandatory.

I next implemented stepwise regression using R's built-in *train* function, in order to determine which variables in the *crime_data* data set should be included in the creation of a linear model used to predict the response variable *Crime*. Stepwise regression is an iterative method used to perform variable selection in regression model. The specific brand of stepwise regression that I chose to implement is what is known as *backward elimination*. In backward elimination, a model is initially trained using all possible factors. After the model is trained with all variables, it is determined which variable is the best candidate for removal, and if a certain threshold is met, then the variable is removed. This process is repeated until it is determined that there are no more variables which should be removed. Since the variable selection is done in an iterative manner and only one variable is removed in each iteration, stepwise regression is an example of a *greedy algorithm.* There are various methods of determining which factor is the best candidate for removal (such as the R^2 value), but I chose to eliminate factors based upon the *Akaike Information Criterion* value (*AIC*), since the *AIC* value considers both the quality and complexity of the model. Within the *train* function, I specified that the variable selection, which takes place in each iteration, should be conducted using *5-fold cross validation*. This was done to ensure that the determination of which (if any) variable to remove was not tainted by overfitting resulting from training and validating using the same set of data.

After implementing *backward elimination*, it was determined that the variables *M, Ed, Po1, M.F, U1, U2, Ineq,* and *Prob* should be included in the linear model. After determining that these were the appropriate factors to include, I trained a linear model using R's built-in *lm* function. The linear model that was generated is defined as follows:

$$Crime = 905.3 + 117.3(M) + 201.5(Ed) + 305.1(Po1) + 65.8(M.F) - 109.7(U1) + \\ \ldots + 158.2(U2) + 244.7(Ineq) - 86.3(Prob)$$

The predictive results of this linear model are shown in the figure below:



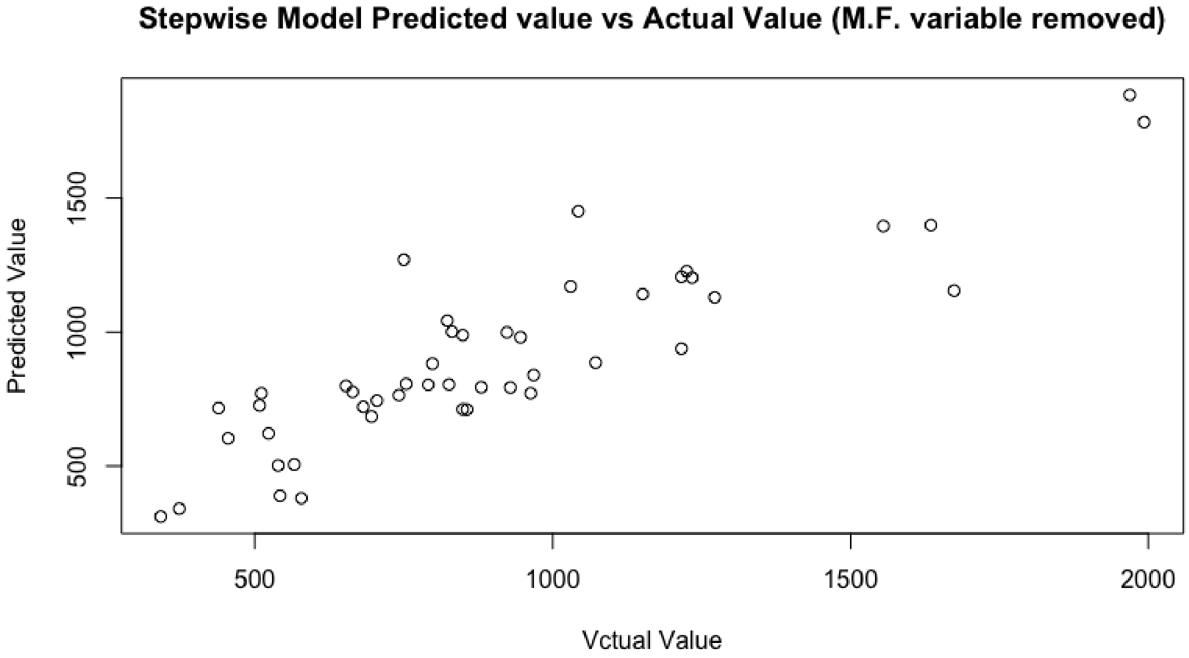
By simply observing the figure above, we can see that the linear model likely does a fairly good job of predicting the value of *Crime*, and indeed this is confirmed by its *R^2* value of *0.789* and the adjusted *R^2* value of *0.744* associated with the model.

While the above *R^2* and *adjusted R^2* values are indicative of a strong model, we must remember that if we train and validate a model using the same set of data, then we will likely find an overly optimistic estimation of the true performance we will see associated with the model when it is applied to other independent data sets. To find a more accurate estimation of the true performance of the model, I implemented *cross-validation*. Specifically, I implemented a special form of *k-fold cross validation* known as *leave-one out cross validation*. In *leave-one out cross validation, k-fold cross validation* is implemented, and all but one data points are used for training, while the remaining data point is used for validation. I implemented this form of *cross-validation* since the *crime_data* data set contains only *47* data points. After implementing cross-validation, I found a cross-validated *R^2* value of *0.668* and a cross-validated adjusted *R^2* value of *0.598* . This is a fairly substantial difference from the above non-cross validated *R^2* value, and therefore indicates that overfitting could be present.

To determine if overfitting was indeed occurring, I analyzed the *p-values* of each of the factors included in the model, and found that the factor, *F.M*, had a *p-value* greater than *0.1* This indicates that this variable may not be relevant in predicting *Crime.* To test this, I removed the variable *F.M* and retrained a new linear model. This linear model is defined below as follows:

$$\begin{aligned} &Crime = 905.1 + 134.2(M) + 244.4(Ed) + 314.9(Po1) - 63.6(U1) + 134.1(U2) + ... \\ &\quad ... + 264.7(Ineq) - 84.8(Prob) \end{aligned}$$

The predicted results of this model are shown in the following figure:



This plot again indicates that we have likely created an accurate model. This is confirmed by the model's R^2 value of *0.774* and *adjusted* R^2 value of *0.733*. For the reasons described above, I again implemented *leave one out* cross validation, in the same manner as described above, in order to estimate the true performance of the model. I found a cross-validated R^2 value of *0.662* and a cross-validated adjusted R^2 value of *0.608*. It should be noted that these performance values are quite similar to the values associated with the model above. This indicates that both models are likely good candidates for modeling the *crime_data* data set and will exhibit similar performance. So which model should be selected? In a real-world situation, both models will likely yield very similar results, but I would recommend implementing the second model, since if all other things are equal, the simpler model is generally the better choice. Thus, my conclusion is that the model given by

$$\begin{aligned} &Crime = 905.1 + 134.2(M) + 244.4(Ed) + 314.9(Po1) - 63.6(U1) + 134.1(U2) + ... \\ &\quad ... + 264.7(Ineq) - 84.8(Prob) \end{aligned}$$

is the ideal choice.

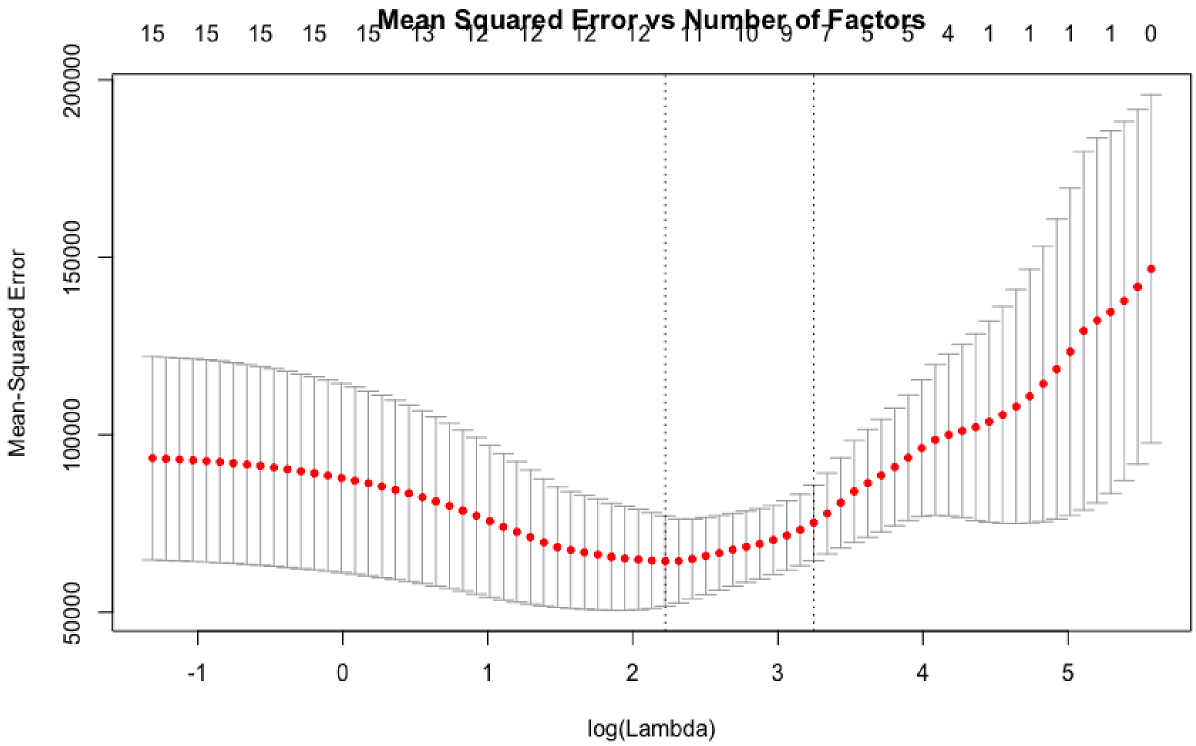
Response 11.1.2

To solve this question, I again began by reading in the data and saving it as *crime_data*. As in the question above, I standardized the numerical predictor variables before beginning my analysis, in the same manner described in the question above. It should be pointed out, however, that in the case of the *lasso* algorithm, variable standardization is critical in finding a meaningful result. This is because the *lasso* algorithm performs variable selection by minimizing the squared errors of the factors included in the regression model subject to placing a constraint on the sum of all coefficients. This means that if we do not perform variable standardization, the coefficients will have artificially different orders of magnitude, and therefore artificially different levels of importance.

After implementing variable standardization, I implemented the *lasso algorithm* using R's *cv.glmnet* function. The *lasso* algorithm is defined as follows:

$$\min_{\beta_0,\beta}\left\{\frac{1}{N}\|y-\beta_0-X\beta\|_2^2\right\} \text{ subject to } \|\beta\|_1 \le t.$$

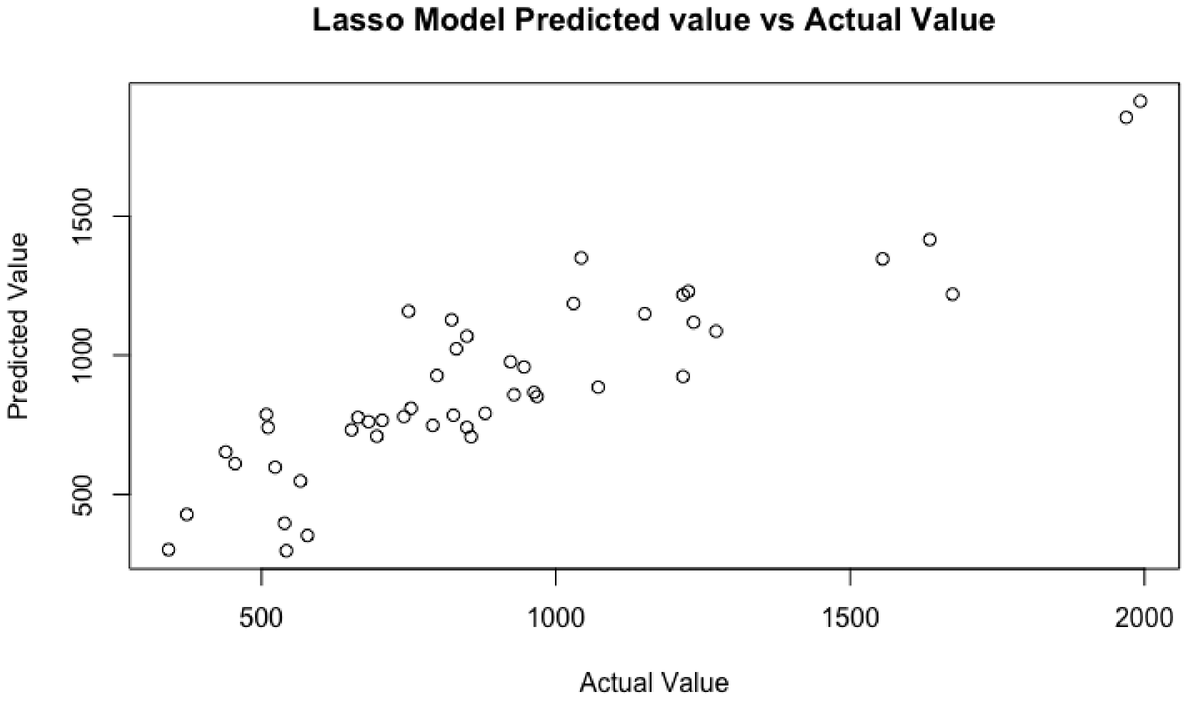
where *t* is a parameter that determines the amount of regularization. Within the *cv.glmnet* function, I specified that *5-fold cross validation* should be implemented in the process of variable selection. The results of the *lasso* algorithm are shown in the figure below:



In the figure above, we can see that the mean squared error is minimized when the number of factors is equal to *11*, as shown by the first vertical dotted line. The second vertical dotted line, it should be pointed out, represents the maximum value of lambda within 1 standard deviation of the minimum lambda value (the most highly regularized model). The *11* factors which the *lasso* function determined should be included in the linear regression model are as follows: *So, M, Ed, Po1, M.F, NW, U1, U2, Wealth, Ineq,* and *Prob.* I next created a linear regression to predict the value of *Crime* based upon these *11* factors. The resulting linear model is defined as follows:

$$Crime = 893.7 + 33.3(So) + 115(M) + 195.3(NW) + 275.7(Po1) + 64.5(M.F) + 15.9(NW) \\ - 94.6(U1) + 140.8(U2) + 73.6(Wealth) + 267(Ineq) - 87.6(Prob)$$

The predicted results of this model may be seen in the figure below:



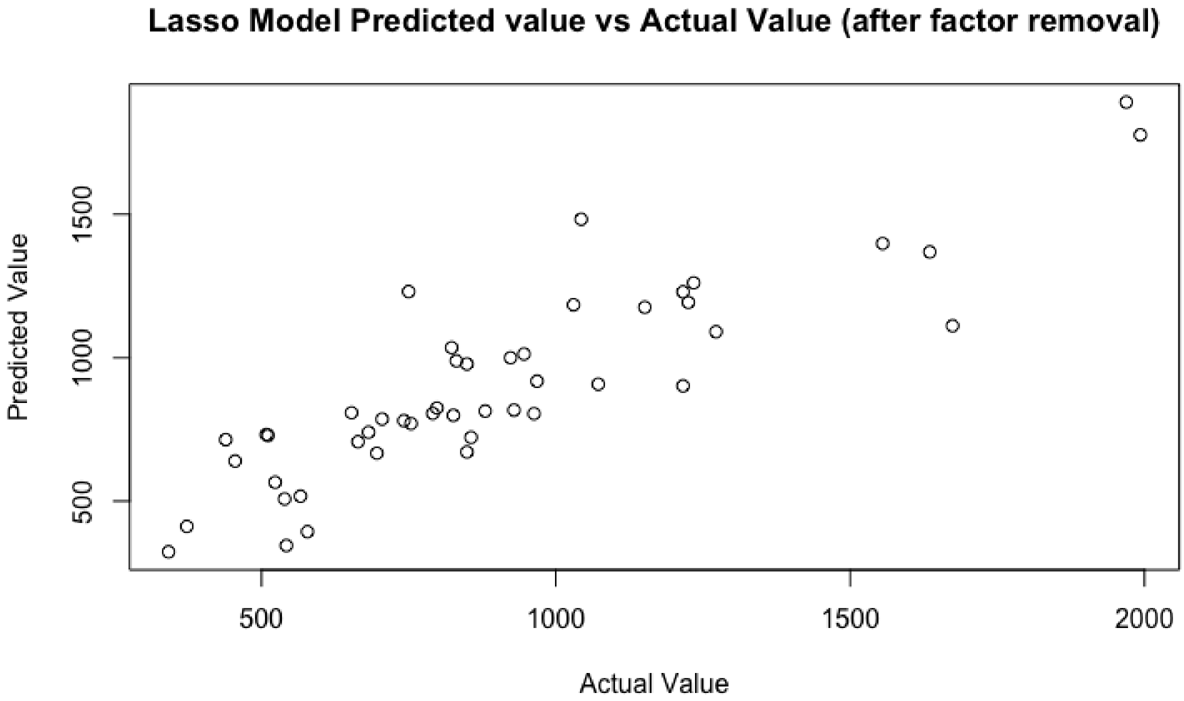
By viewing the figure above, we can again observe that this model appears to do a good job of forecasting the value of *Crime*. This is confirmed by the strong R^2 value of *0.794* and the strong *adjusted* R^2 value of *0.729*.

However, as described in the first question, cross-validation is necessary to determine how the model will likely perform when applied to independent data sets. Just as in the first question, I applied *leave-one-out k-fold cross validation.* I found a cross validated R^2 value of *0.605* and a cross validated *adjusted* R^2 value of *0.481* . The large discrepancy between the cross-validated and the non-cross-validated performance metrics indicates that overfitting is very likely present.

To further investigate the problem of overfitting, I analyzed the *p-values* associated with each of the predictive factors included in the above linear model. Upon analyzing the *p-values* I determined that the factors *So, M.F, NW, U1,* and *Wealth* all had *p-values* greater than *0.1*. I removed each of these values and I again trained a linear regression model using only the remaining values, which led to the following result:

$$Crime = 905.1 + 132(M) + 219.8(Ed) + 341.8(Po1) + 75.5(U2) + 269.9(Ineq)$$

The predicted values of this model are shown below:



The R^2 value associated with this model is equal to *0.766* and the adjusted R^2 value is equal to *0.731.* The cross-validated R^2 value associated with this model equal *0.666* and the cross validated adjusted R^2 value equals *0.534*. Thus, we can see that the removal of the variables *So, M.F, NW, U1,* and *Wealth* has led to a significant improvement over the original model. This is not surprising, since the original model suggested by the *lasso* algorithm included *11* factors – far too many factors for a data set which contains only *47* data points. Thus, the optimal model determined by the *lasso* method is given by

$$Crime = 905.1 + 132(M) + 219.8(Ed) + 341.8(Po1) + 75.5(U2) + 269.9(Ineq)$$

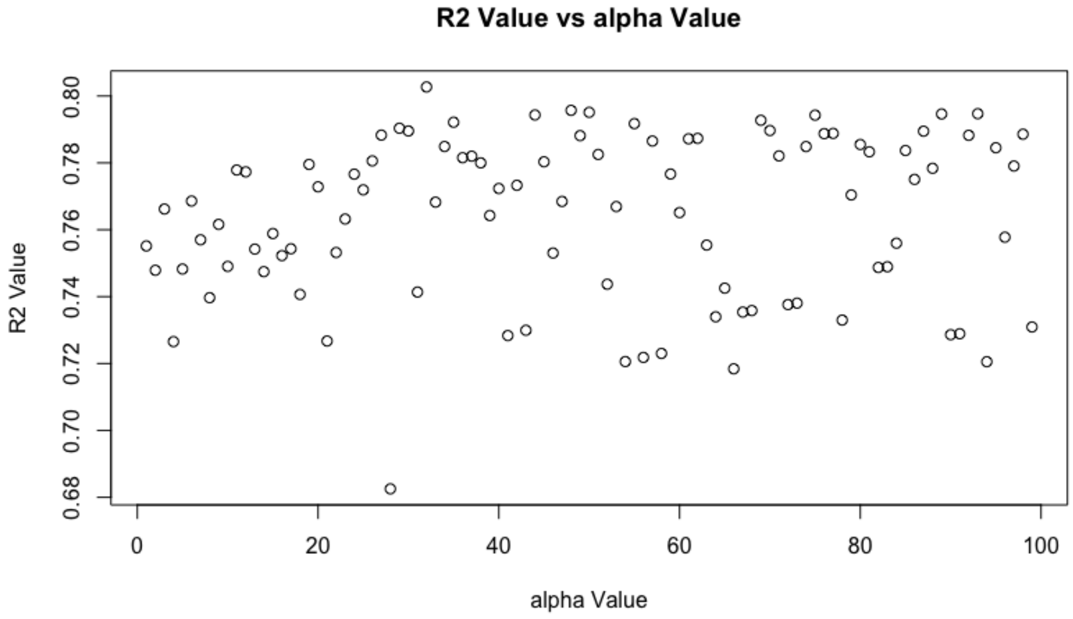
Response 11.1.3

To solve this final question, I once more began by reading in the data and saving it as *crime_data*. Before beginning the *elastic net* analysis, I once more standardized the numerical predictor variables in the same manner described in the questions above. It should again be pointed out that in the case of the *elastic net* algorithm, variable standardization is critical in finding a meaningful result. Just as in the *lasso* algorithm, variable selection is performed by minimizing the squared errors of the factors included in the regression model subject to placing a constraint on the sum of all coefficients. Therefore, if we do not perform variable standardization, the coefficients will have artificially different orders of magnitude, and, by extension, artificially different levels of importance.

After taking care of variable standardization, I used R's *cv.glmnet* algorithm to implement the *elastic net* algorithm. The formula that defines the *elastic net* algorithm is nearly identical to the formula that defines the *lasso* algorithm*,* except that it contains one additional squared penalty term. The *elastic net* formula is defined as follows:

$$\min_{\beta_0,\beta} \left\{ \|y - \beta_0 - X\beta\|_2^2 \right\} \text{ subject to } (1-\alpha)\|\beta\|_1 + \alpha\|\beta\|_2^2 \le t,$$

Within the *cv.glmnet* function, I once again specified that *5-fold cross validation* should be implemented in the process of variable selection. This was done so that over-fitting would not taint the *elastic net* algorithm's determination of the variables which should be included in the final linear regression model. I tested all values of *alpha* (as defined in the equation above) in the range *0.01, 0.02, … , 0.99* in order to determine the value of *alpha* which would produce the best model. The results of this testing are shown in the following figure:



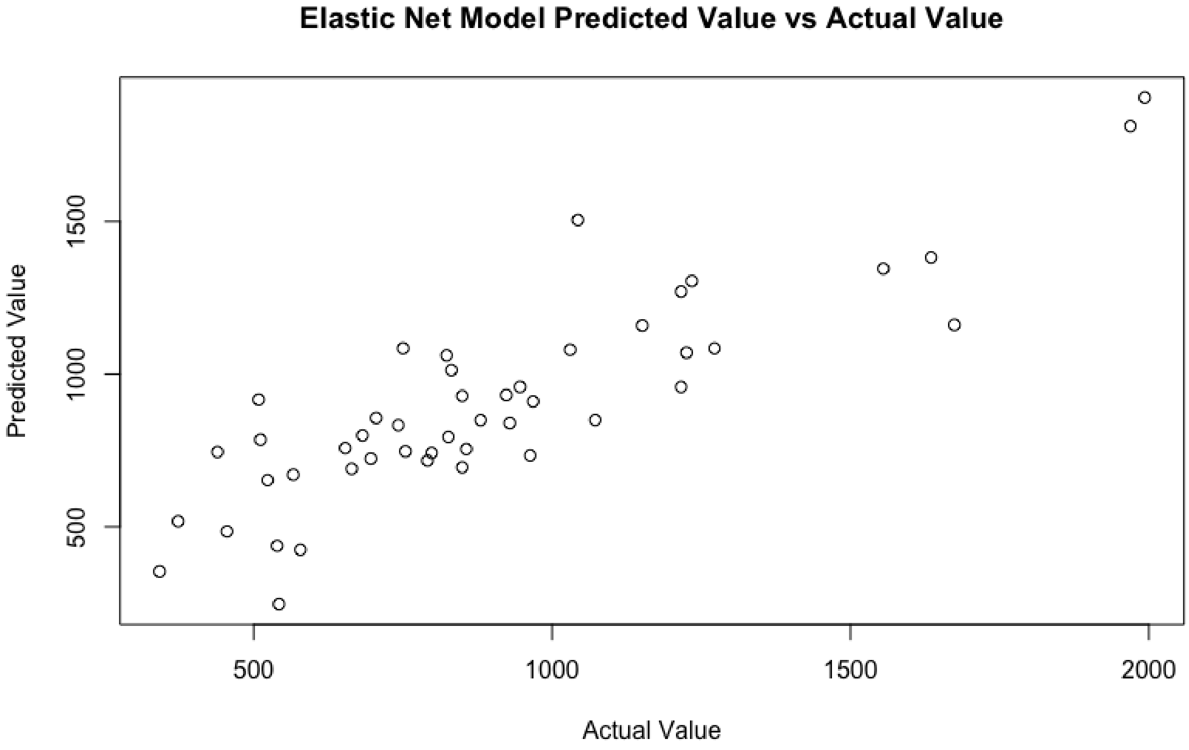
Although there is no clear trend present in the figure above, I determined that the most accurate model, as defined by the R^2 value, was produced when *alpha* was equal to *33*.

Thus, I set *alpha* to *33* in the *elastic net* model and I ran the algorithm once more using R's *cv.glmnet* function. This led to the determination that the following variables should be included in the linear regression model: *M, Ed, Po1, Po2, M.F, NW, Ineq,* and *Prob*.

I next created a linear regression to predict the value of *Crime* based upon these *8* factors. The resulting linear model is defined as follows:

Crime = 905.1 + 81.5(M) + 152.3(Ed) + 564.3(Po1) – 222.4(Po2) + 50.6(M.F) + ...
... + 34.5(NW) + 237.1(Ineq) – 96.1(Prob)

The predicted results of this model may be seen in the following figure:

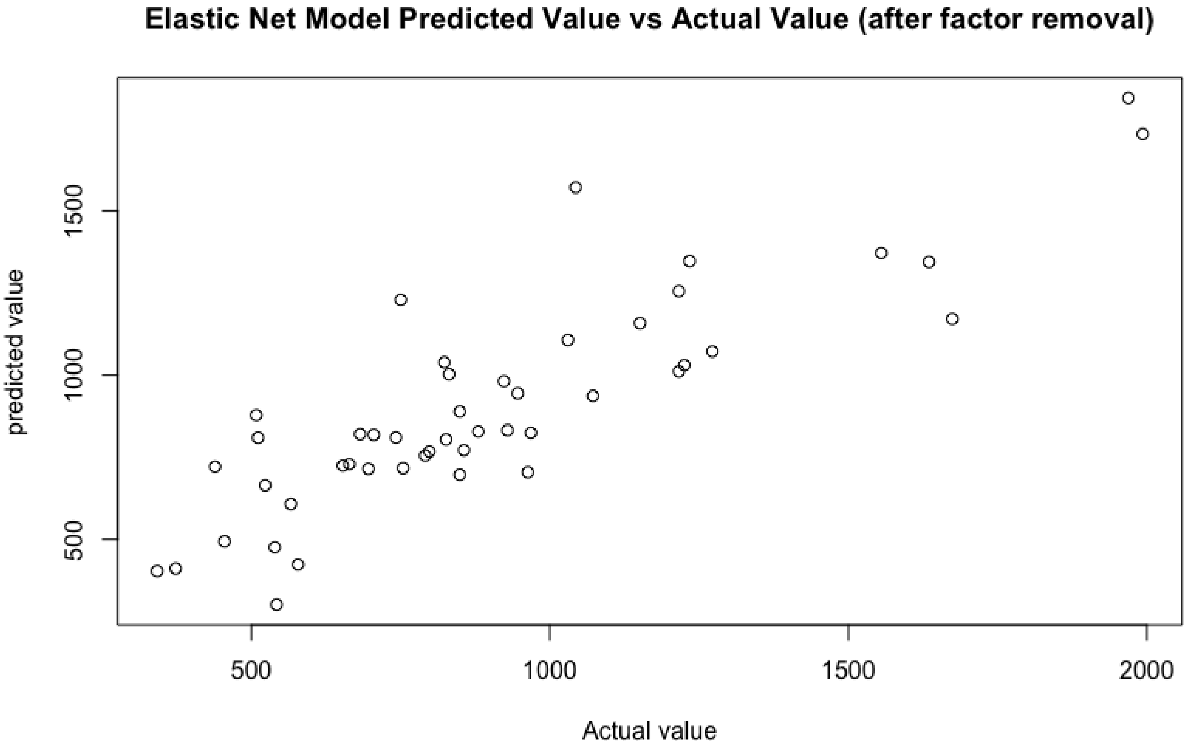


The R^2 value associated with this model is equal to *0.755* and the adjusted R^2 value associated with this model is equal to *0.704*. However, as mentioned in the above two questions, cross validation is needed to determine the true performance of this model. After performing *leave-one-out* cross validation, I found a cross-validated R^2 value of *0.611* and a cross-validated adjusted R^2 value of *0.529*. The large discrepancy between the cross-validated and the non-cross-validated performance metrics indicates that overfitting is very likely present.

To further investigate the problem of overfitting, I analyzed the *p-values* associated with each of the predictive factors included in the above linear model. After analyzing the *p-values,* I determined that the factors *Po2, M.F* and *NW* all had *p-values* greater than *0.1* . I removed each of these factors and once more trained a linear regression model using only the remaining values. This gave the following result:

$$Crime = 905.1 + 100.1(M) + 179.2(Ed) + 360.3(Po1) + 272.5(Ineq) – 87.9(Prob)$$

The predicted values of this model can be seen in the following figure:



The R^2 value associated with this model is equal to *0.738* and the adjusted R^2 value is equal to *0.706.* The cross-validated R^2 value associated with this model equal *0.629* and the cross validated adjusted R^2 value equals *0.553* . Thus, we can see that the removal of the variables *Po2, M.F* and *NW* has led to a modest improvement in model performance. This is not surprising, since the original linear model, as suggested by the *elastic net* algorithm, included *8* factors, which is likely too many for a data set which contains only *47* data points. Thus, the optimal model determined by the *elastic net* method is given by:

$$Crime = 905.1 + 132(M) + 219.8(Ed) + 341.8(Po1) + 75.5(U2) + 269.9(Ineq)$$

Conclusion

The above three methods of variable selection, *stepwise regression, lasso,* and *elastic net,* all have their own strengths and weaknesses. *Stepwise* variable selection is a fast algorithm, but it may tend to fit the data to random effects (although this was mitigated in the implementation above by applying cross-validation in the variable selection process). *Lasso* and *elastic net algorithms* are slower to implement, but are (generally) more accurate. Generally, this means the *lasso* and *elastic net* algorithms are the preferred methods of variable selection when speed is not a primary consideration.

So, which algorithm performed the best in determining the factors to include in the linear model used to predict the value of *Crime* based on the predictor variables in the *crime_data* data set? The final linear model generated via the *stepwise* algorithm, defined as:

$$Crime = 905.1 + 134.2(M) + 244.4(Ed) + 314.9(Po1) - 63.6(U1) + 134.1(U2) + \ldots$$
$$\ldots + 264.7(Ineq) - 84.8(Prob)$$

had a cross-validated R^2 value of *0.662* and a cross-validated adjusted R^2 value of *0.608*.

The final linear model generated via the *lasso* algorithm, was defined as:

$$Crime = 905.1 + 132(M) + 219.8(Ed) + 341.8(Po1) + 75.5(U2) + 269.9(Ineq)$$

and had a cross-validated R^2 value of *0.666* and a cross-validated adjusted R^2 value of *0.534*.

The final linear model generated via the *elastic net* algorithm, was defined as:

$$Crime = 905.1 + 132(M) + 219.8(Ed) + 341.8(Po1) + 75.5(U2) + 269.9(Ineq)$$

and had a cross-validated R^2 value of *0.629* and a cross-validated adjusted R^2 value of *0.553*.

So, we can see that based on the R^2 value, the *lasso* method had the best performance, followed by the *stepwise* algorithm, followed by the *elastic net* algorithm. However, based on cross-validated R^2 values, the *stepwise* algorithm had the best performance, followed by the *elastic net* algorithm, followed by the *lasso* algorithm. It should be pointed out though that all three models do in fact perform quite similarly with regard to both R^2 and adjusted R^2 values. Additionally, it should also be noted that the *stepwise* algorithm contains *7* predictor variables, while the *lasso* algorithm and the *elastic* net algorithm both contain only *5* predictor variables. Generally, if two models have comparable performance, the simpler model is considered preferential. Ultimately however, the above three approaches all produced such similar results that it is not possible to determine if one is definitively better than the rest.

It is also interesting to note, through observing the above figures, that all three approaches do a good job of predicting the value of Crime, except in the cases of the data points which had the largest values for *Crime*. In these cases, the models consistently under-predicted the correct value. This is likely due to the fact that these instances are simply anomalous, and thus, difficult to predict accurately. Overall, all three algorithms tested above are viable methods for tackling this problem.