ISAYE 6501 HW 8

Ryan Cherry

2024-02-29

In the previous homework, there were problems with multicollinearity in the models that were created. Multicollinearity occurs when several of the predictors are heavily correlated with each other, which can decrease the reliability of the prediction. This may be a sign that there are several redundant variables that may need to be removed. Also, a simpler model may be more appropriate for explaining crime rate.

In this assignment, three variable selection procedures will be conducted to see if a stronger model can be created with fewer variables. The methods that will be completed are stepwise regression, LASSO regression, and Elastic Net regression.

```
crime_variables <- as.data.frame(read.table("C:/Users/ryanc/Downloads/uscrime.txt", header = TRUE))
head(crime_variables, 2)
```

```
##      M So   Ed  Po1 Po2    LF   M.F Pop   NW    U1  U2 Wealth Ineq     Prob
## 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
## 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
##      Time Crime
## 1 26.2011   791
## 2 25.2999  1635
```

Question 11.1

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

1. Stepwise regression

Stepwise regression is a variable selection technique that combines both backward selection and forward selection. Just as in backward selection, a model with each of the predictors is created as a starting point and from there, variables are removed or added back in based on a quality metric, usually AIC, BIC, MSE, or R-squared.

Before completing stepwise regression, the crime data set was divided into training and test sets, 80% and 20% respectively.

```
set.seed(123)
#sample 80% of the observations to be used as a training set
train_sample <- sample(1:nrow(crime_variables), as.integer(0.8 * nrow(crime_variables)), replace = FALSE)

#split into training and test sets, with the training using 80% of the data and the remaining 20% going
train <- crime_variables[train_sample,]
test <- crime_variables[-train_sample,]
```

A full model needed to be created to serve as a starting point for the stepwise selection procedure. The model with each predictor overfits the data, as demonstrated by the large difference in r-squared and adjusted r-squared values (0.837 vs 0.721) and the large number of predicotrs insignificant at a level of 0.05.

```
#create a full model with each predictor from the crime dataset to serve as a base model for stepwise s
full_model <- lm(train$Crime ~ ., train[1:15])
summary(full_model)
```

```
##
## Call:
## lm(formula = train$Crime ~ ., data = train[1:15])
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -312.69 -115.78  -13.32  121.88  493.30
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -8.187e+03  2.063e+03  -3.968 0.000701 ***
## M            9.611e+01  4.469e+01   2.150 0.043322 *
## So           1.665e+02  1.936e+02   0.860 0.399615
## Ed           2.068e+02  7.140e+01   2.896 0.008643 **
## Po1          1.629e+02  1.253e+02   1.300 0.207524
## Po2         -9.652e+01  1.410e+02  -0.684 0.501156
## LF          -1.395e+03  1.818e+03  -0.767 0.451576
## M.F          4.824e+01  2.511e+01   1.921 0.068459 .
## Pop          7.300e-01  1.628e+00   0.448 0.658522
## NW           2.490e+00  7.386e+00   0.337 0.739359
## U1          -8.315e+03  4.895e+03  -1.699 0.104165
## U2           1.754e+02  1.017e+02   1.724 0.099352 .
## Wealth       6.753e-02  1.237e-01   0.546 0.590899
## Ineq         4.348e+01  2.813e+01   1.545 0.137166
## Prob        -3.693e+03  2.504e+03  -1.475 0.155118
## Time         2.541e+00  8.199e+00   0.310 0.759681
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 215 on 21 degrees of freedom
## Multiple R-squared:  0.8372, Adjusted R-squared:  0.7209
## F-statistic: 7.199 on 15 and 21 DF,  p-value: 2.976e-05
```

To improve the model, the stepwise selection procedure was conducted, using AIC as the metric for removing or adding predictors to the model. The stepwise model reduced the number of predictors from 15 to 9 and resulted in a much smaller difference between r-squared and adjusted r-squared (0.825 vs 0.767). In addition, 7 of the 9 predictors are significant at a level of 0.05.

```
stepwise_model <- stepAIC(full_model, direction = "both", trace = FALSE, k = 2)
summary(stepwise_model)
```

```
##
## Call:
## lm(formula = train$Crime ~ M + So + Ed + Po1 + M.F + U1 + U2 +
##     Ineq + Prob, data = train[1:15])
```

```
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -424.1 -108.7  -10.5  116.3  538.9
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -7335.78    1443.80  -5.081 2.45e-05 ***
## M              96.51      37.53   2.572  0.01595 *
## So            198.48     132.50   1.498  0.14573
## Ed            186.36      58.27   3.198  0.00352 **
## Po1            91.69      16.45   5.575 6.53e-06 ***
## M.F            36.00      15.34   2.346  0.02655 *
## U1          -6978.64    3899.48  -1.790  0.08474 .
## U2            194.94      89.87   2.169  0.03904 *
## Ineq           38.40      17.27   2.223  0.03477 *
## Prob        -4171.05    1742.11  -2.394  0.02386 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 196.5 on 27 degrees of freedom
## Multiple R-squared:  0.8251, Adjusted R-squared:  0.7668
## F-statistic: 14.15 on 9 and 27 DF,  p-value: 4.263e-08
```

To get a better measure of model quality, the model needed to be evaluated on the testing data. The key metrics I chose to consider were RMSE, r-squared, and adjusted r-squared. From the output below, the stepwise model resulted in an r-squared value of 0.825, adjusted r-squared of 0.767, and root mean squared error (RMSE) of 167.84.

When modeled on the test data, the RMSE was quite higher, reaching 234.87, higher than the original full model. However, the r-squared and adjusted r-squared values did not change, indicating that this model does not seem to overfit the data like the original model.

However, the best model cannot be determined until I conduct LASSO and Elastic Net regression, further down.

```
#evaluate stepwise model on both training and test data using r-squared
training_pred = predict(stepwise_model, newdata = train)
test_pred = predict(stepwise_model, newdata = test)

#create a function to provide the r-squared and rmse metrics on both the training and test data

stepwise_evaluation = function(model, data, data_predictions, response_variable) {
r2 = as.character(round(summary(model)$r.squared, 3))
adj = as.character(round(summary(model)$adj.r.squared, 3))
residual = (data[, response_variable] - data_predictions)^2
rmse = as.character(round(sqrt(sum(residual)/length(data_predictions)), 2))
print(paste("R-squared:", r2))
print(paste("Adjusted r-squared:", adj))
print(paste("RMSE:", rmse))
}

#obtain RMSE and r-squared values for the stepwise model's performance on training data
stepwise_evaluation(stepwise_model, train, training_pred, response_variable = 'Crime')
```

```
## [1] "R-squared: 0.825"
## [1] "Adjusted r-squared: 0.767"
## [1] "RMSE: 167.84"
```

```
#obtain RMSE and r-squared values for the stepwise model's performance on test data
stepwise_evaluation(stepwise_model, test, test_pred, "Crime")
```

```
## [1] "R-squared: 0.825"
## [1] "Adjusted r-squared: 0.767"
## [1] "RMSE: 234.56"
```

In LASSO regression, we try to minimize mean squared error by shrinking each predictor's parameter value toward a central mean. Lambda, which represents the amount we shrink each coefficient, is a parameter that can be tuned to achieve the best balance between bias and variance. This technique encourages simpler models with fewer predictors a s aresult.

2. Lasso

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.

In order to conduct LASSO regression, the data needed to be scaled beforehand, which the code below accomplishes.

```
#scale data set before use in both LASSO regression and elastic net regression
x_train<- scale(as.matrix(train)[,-16], scale = TRUE)
y_train<- scale(as.matrix(train)[,16], scale = TRUE)
x_test<- scale(as.matrix(test)[,-16],  scale = TRUE)
y_test<- scale(as.matrix(test)[,16],scale = TRUE)
```

In order to determine the best value of lambda that finds the appropriate balance between bias and variance, cross-validation was performed below using mean-squared error as the key metric of quality. In the output below, the lower value of lambda is the one that will be used to build a LASSO regression model. That value is 0.01320. In the same row as the minimum lambda value, the nonzero column indicates the number of predictors the algorithm chose to include in the model. Compared the stepwise regression, LASSO only reduced the number of predictors by 3, down from 15 to 12.

```
#perform cross validation to determine the optimal value for lambda when performing LASSO regression
lasso_initial <-  cv.glmnet(x_train, y_train, family = "gaussian", alpha = 1, type.measure = "mse")

lasso_initial
```

```
##
## Call:  cv.glmnet(x = x_train, y = y_train, type.measure = "mse", family = "gaussian",      alpha = 1)
##
## Measure: Mean-Squared Error
##
##     Lambda Index Measure     SE Nonzero
## min 0.0159    42  0.5173 0.1404      12
## 1se 0.1231    20  0.6561 0.1671       6
```

```r
#store optimal lambda value in an object
optimal_lambda_lasso <- lasso_initial$lambda.min
```

The LASSO regression model was built and then prediction metrics, r-squared and RMSE, were determined. From the output below, when predicted using the training data, the model achieved an adjusted r-squared value of 0.8265, slightly lower than the stepwise model, and an RMSE of 0.4133. Since the data is scaled in order to run both LASSO and Elastic Net regression, the RMSE values cannot be compared to those achieved from stepwise regression. As a result, adjusted r-squared will be the comparison metric used in the rest of this assignment.

```r
#fit a LASSO regression model using the training data and the optimal lambda value from above
lasso_model <- glmnet(x = x_train, y = y_train, family = "gaussian", alpha = 1, lambda = optimal_lambda_

#determine performance on training data
predict_lasso_train <- lasso_model %>% predict(x_train)

#obtain adjusted_r_squared and RMSE values from the LASSO model's performance on training data
data.frame( R2 = R2(predict_lasso_train, y_train),
            RMSE = RMSE(predict_lasso_train, y_train))
```

```
##          s0      RMSE
## 1 0.8244347 0.4167912
```

The LASSO model was evaluated using testing data to get a better feel for its performance. The resulting adjusted r-squared value was 0.2857, significantly lower than the performance on the training data. This indicates that the LASSO regression model may have too many predictors and the model underfits to new data. As of now, the stepwise model woul dbe the one I would choose to preeict crime rate.

```r
#determine performance on test data
predict_lasso_test <- lasso_model %>% predict(x_test)

#obtain adjusted_r_squared and RMSE values from the LASSO model's performance on test data
data.frame( R2 = R2(predict_lasso_test, y_test),
            RMSE = RMSE(predict_lasso_test, y_test))
```

```
##          s0      RMSE
## 1 0.2847797 0.8382247
```

Elastic Net regression also requires the data to be scaled. This technique is a combination of ridge regression and LASSO regression in that both penalty parameters from these techniques are included. We choose values of alpha and lambda that minimizes the squared errors. This technique deals with highly correlated predictors better than LASSO so it should produce a better model than 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.

The optimal values for alpha and lambda were determined by using a repeated cross-validation method where 10 different combinations of alpha and lambda values are tested, and the best combination is selected. From the output below, the optimal parameter combination features a lambda of 0.0187 and an alpha of 0.5514.

```r
set.seed(123)

#set parameters for training
train_control <- trainControl(method = "repeatedcv", number = 10, repeats = 8, search = "random")

#Train the model using the glmnet method
elastic_train <- train(Crime ~ .,data = as.matrix(scale(train)),method = "glmnet",preProcess = c("cente
tuneLength = 10, trControl = train_control)
```

```
## Warning in nominalTrainWorkflow(x = x, y = y, wts = weights, info = trainInfo, :
## There were missing values in resampled performance measures.
```

```r
elastic_train$bestTune
```

```
##      alpha     lambda
## 6 0.551435 0.01874915
```

An elastic net model was fitted on the training data using the optimal values of alpha and lambda chosen above. Then, the adjusted r-squared and RMSE values were obtained. We cannot use RMSE to compare with the stepwise model since the data is scaled, but we can compare it to the model produced with the LASSO regression.

On the training data, the Elastic Net model produced an adjusted r-squared of 0.8270, identical to the r-squared value produced by the LASSO model initially. The RMSE was 0.4129, nearly identical to the RMSE from the LASSO regression on the training data.

```r
#fitting an elastic net model using the optimal alpha and lambda values determined above
elastic_net_model <- glmnet(x_train, y_train, alpha = 0.551435, lambda = 0.01874915, family = "gaussian

#determine performance on training data
predict_elasticnet_train <- elastic_net_model %>% predict(x_train)

#obtain adjusted_r_squared and RMSE values from the elastic net model's performance on training data
data.frame(R2 = R2(predict_elasticnet_train, y_train),
           RMSE = RMSE(predict_elasticnet_train, y_train))
```

```
##          s0      RMSE
## 1 0.8270326 0.4128849
```

As before, the Elastic NEt model's performance was evaluated by running it on test data. The adjusted r-squared value was much lower on the test data just like the LASSO model, producing a value of 0.2759. The resulting RMSE value was noticeably higher as well. Compared to the LASSO regression model, the adjusted r-squared value is slightly lower and the RMSE is slightly higher. However,neither the LASSO and Elastic Net models did not produce a better model than stepwise selection.

```r
#determine performance on test data
predict_elasticnet_test <- elastic_net_model %>% predict(x_test)

#obtain adjusted_r_squared and RMSE values from the elastic net model's performance on test data
data.frame(R2 = R2(predict_elasticnet_test, y_test),
           RMSE = RMSE(predict_elasticnet_test, y_test))
```

```
##          s0      RMSE
## 1 0.2759048 0.8451056
```

In conclusion, I would choose the model produced by stepwise regression. This model had the fewest predictors with 9, did not overfit the data, and produced the highest adjusted r-squared on the test data (0.767). Since the stepwise model was the only one that did not do noticeably worse on the test data, I would use the stepwise model to predict crime rate.