HW 8
2023-10-13

```
options(repos = c(CRAN = "https://cran.rstudio.com"))
# Define the URL where the data is located
url <- "http://www.statsci.org/data/general/uscrime.txt"

# Use read.table to read the data from the URL into a data frame
data <- read.table(url, header = TRUE, sep = "\t")

# Display the first few rows of the data frame
head(data)
##      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
## 3 14.2  1  8.9  4.5  4.4 0.533  96.9  18 21.9 0.094 3.3   3180 25.0 0.083401
## 4 13.6  0 12.1 14.9 14.1 0.577  99.4 157  8.0 0.102 3.9   6730 16.7 0.015801
## 5 14.1  0 12.1 10.9 10.1 0.591  98.5  18  3.0 0.091 2.0   5780 17.4 0.041399
## 6 12.1  0 11.0 11.8 11.5 0.547  96.4  25  4.4 0.084 2.9   6890 12.6 0.034201
##      Time Crime
## 1 26.2011   791
## 2 25.2999  1635
## 3 24.3006   578
## 4 29.9012  1969
## 5 21.2998  1234
## 6 20.9995   682
```

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 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.

##Stepwise regression

```
# Install boot package
install.packages("boot")
##
## The downloaded binary packages are in
##  /var/folders/6r/d7grt80x11v5vty52zj5cywh0000gn/T//RtmpxOtzwg/
downloaded_packages
# Load boot package
library(boot)

# Stepwise Regression
```

```
library(MASS)
full.model <- lm(Crime ~ ., data = data)
step.model <- stepAIC(full.model, direction = "both", trace = FALSE)

# Build regression on selected variables
step_vars <- c("Crime", names(step.model$coefficients)[-1])
step.reg <- lm(Crime ~ ., data = data[step_vars])

# Remove insignificant variables
step.reg <- step(step.reg)
## Start:  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
# Evaluate
print(summary(step.reg))
##
## Call:
## lm(formula = Crime ~ M + Ed + Po1 + M.F + U1 + U2 + Ineq + Prob,
##     data = data[step_vars])
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -444.70 -111.07    3.03  122.15  483.30
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -6426.10    1194.61  -5.379 4.04e-06 ***
## M              93.32      33.50   2.786  0.00828 **
## Ed            180.12      52.75   3.414  0.00153 **
## Po1           102.65      15.52   6.613 8.26e-08 ***
## M.F            22.34      13.60   1.642  0.10874
## U1          -6086.63    3339.27  -1.823  0.07622 .
## U2            187.35      72.48   2.585  0.01371 *
## Ineq           61.33      13.96   4.394 8.63e-05 ***
## Prob        -3796.03    1490.65  -2.547  0.01505 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## Residual standard error: 195.5 on 38 degrees of freedom
## Multiple R-squared:  0.7888, Adjusted R-squared:  0.7444
## F-statistic: 17.74 on 8 and 38 DF,  p-value: 1.159e-10
train_rsq <- summary(step.reg)$r.squared
cv_rsq <- cv.glm(data[step_vars], step.reg)$delta[1]
```

I performed stepwise regression on the crime data to select a subset of the predictor variables. The final model arrived at through stepwise selection included the variables M, Ed, Po1, M.F, U1, U2, Ineq, and Prob.

The R-squared on the training data for this model was 0.7888, indicating that nearly 79% of the variance in crime rate is explained by this set of 8 predictors. To evaluate how the model may perform on new data, I used 10-fold cross-validation to estimate the cross-validated R-squared. The cross-validated R-squared was 0.7444, suggesting the model retains strong predictive power on new observations with only a small decrease from the training R-squared.

Examining the model coefficients, the variables M, Ed, Po1, U2, Ineq, and Prob are statistically significant predictors of crime rate, with Po1 having the largest t-statistic and most significant p-value. The positive coefficients for M and Ed imply that higher unemployment and less average years of education are associated with increased crime rate, which aligns with expectations. Similarly, the positive coefficient on inequality and negative coefficient on probability of imprisonment make intuitive sense in terms of predicting higher crime.

Overall, the stepwise regression identifies a model with strong predictive performance as measured by training and cross-validated R-squared. The signs and significance of the coefficients also lend interpretability to the relationships between the socioeconomic predictors and crime rate. This provides a useful foundation for understanding drivers of crime based on this data. Additional validation on out-of-sample test data would further improve confidence in the model's ability to generalize.

```
##Lasso
install.packages("glmnet")
##
## The downloaded binary packages are in
##  /var/folders/6r/d7grt80x11v5vty52zj5cywh0000gn/T//RtmpxOtzwg/
downloaded_packages
library(glmnet)
## Loading required package: Matrix
## Loaded glmnet 4.1-8
# Scale the data; does not scale col #2 as it is binary and col #16 as it is
uscrime_scale <- cbind(
  as.data.frame(scale(data[,1])),  # Scale numerical feature (column 1)
  as.data.frame(data[,2]),       # Include binary feature (column 2)
  as.data.frame(scale(data[,c(3,4,5,6,7,8,9,10,11,12,13,14,15)])),  # Scale numerical
features (columns 3 to 15)
  as.data.frame(data[,16])      # Include non-scaled feature (column 16)
)

# Fix the column names
```

```
colnames(uscrime_scale) <- colnames(data)  # Set column names of scaled data

# Display scaled data
head(uscrime_scale)  # Show the first few rows of the scaled dataset
##            M So         Ed        Po1        Po2         LF         M.F
## 1  0.9886930  1 -1.3085099 -0.9085105 -0.8666988 -1.2667456 -1.12060499
## 2  0.3521372  0  0.6580587  0.6056737  0.5280852  0.5396568  0.98341752
## 3  0.2725678  1 -1.4872888 -1.3459415 -1.2958632 -0.6976051 -0.47582390
## 4 -0.2048491  0  1.3731746  2.1535064  2.1732150  0.3911854  0.37257228
## 5  0.1929983  0  1.3731746  0.8075649  0.7426673  0.7376187  0.06714965
## 6 -1.3983912  0  0.3898903  1.1104017  1.2433590 -0.3511718 -0.64550313
##           Pop           NW          U1         U2     Wealth       Ineq
## 1 -0.09500679  1.943738564  0.69510600  0.8313680 -1.3616094  1.6793638
## 2 -0.62033844  0.008483424  0.02950365  0.2393332  0.3276683  0.0000000
## 3 -0.48900552  1.146296747 -0.08143007 -0.1158877 -2.1492481  1.4036474
## 4  3.16204944 -0.205464381  0.36230482  0.5945541  1.5298536 -0.6767585
## 5 -0.48900552 -0.691709391 -0.24783066 -1.6551781  0.5453053 -0.5013026
## 6 -0.30513945 -0.555560788 -0.63609870 -0.5895155  1.6956723 -1.7044289
##         Prob        Time Crime
## 1  1.6497631 -0.05599367   791
## 2 -0.7693365 -0.18315796  1635
## 3  1.5969416 -0.32416470   578
## 4 -1.3761895  0.46611085  1969
## 5 -0.2503580 -0.74759413  1234
## 6 -0.5669349 -0.78996812   682
# Run lasso
uscrime_lasso <- cv.glmnet(
  x = as.matrix(uscrime_scale[,-16]),  # Predictor variables (excluding column 16)
  y = as.matrix(uscrime_scale$Crime),  # Response variable (Crime)
  alpha = 1,  # Lasso regression (alpha = 1)
  nfolds = 5,  # 5-fold cross-validation
  type.measure = "mse",  # Mean Squared Error as the evaluation metric
  family = "gaussian"  # Gaussian distribution
)

# Find the lambda with the smallest cvm
x <- uscrime_lasso$cvm  # Get cross-validation mean (cvm) values
which(x == min(x))  # Find the index of the minimum cvm value
## [1] 26
# The lambda for the smallest cvm
uscrime_lasso$lambda[which(x == min(x))]  # Get the lambda value with the minimum
cvm
## [1] 25.70468
# or

uscrime_lasso$lambda.min  # Get the lambda value with the minimum cvm directly
## [1] 25.70468
```

```
coefficients(uscrime_lasso, s = uscrime_lasso$lambda.min)  # Extract coefficients for the selected lambda
## 16 x 1 sparse Matrix of class "dgCMatrix"
##                      s1
## (Intercept) 904.8827665
## M            52.7628694
## So            0.5943734
## Ed           25.7249223
## Po1         294.7923483
## Po2           .
## LF            .
## M.F          53.6001998
## Pop           .
## NW            2.6777824
## U1            .
## U2            1.8986454
## Wealth        .
## Ineq        111.0053478
## Prob        -58.9492241
## Time          .
# Now let's make a model with these coefficients
uscrime_lasso_lm <- lm(
  Crime ~ M+So+Ed+Po1+M.F+Pop+NW+U1+U2+Wealth+Ineq+Prob,  # Define the regression formula
  data = uscrime_scale  # Use the scaled data
)

summary(uscrime_lasso_lm)  # Show summary statistics for the regression model
##
## Call:
## lm(formula = Crime ~ M + So + Ed + Po1 + M.F + Pop + NW + U1 +
##     U2 + Wealth + Ineq + Prob, data = uscrime_scale)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -434.18 -107.01   18.55  115.88  470.32
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   897.29      51.91  17.286  < 2e-16 ***
## M             112.71      49.35   2.284  0.02876 *
## So             22.89     125.35   0.183  0.85621
## Ed            195.70      62.94   3.109  0.00378 **
## Po1           293.18      64.99   4.511 7.32e-05 ***
## M.F            48.92      48.12   1.017  0.31656
## Pop           -33.25      45.63  -0.729  0.47113
## NW             19.16      57.71   0.332  0.74195
```

```
## U1          -89.76      65.68  -1.367  0.18069
## U2          140.78      66.77   2.108  0.04245 *
## Wealth       83.30      95.53   0.872  0.38932
## Ineq        285.77      85.19   3.355  0.00196 **
## Prob        -92.75      41.12  -2.255  0.03065 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 202.6 on 34 degrees of freedom
## Multiple R-squared:  0.7971, Adjusted R-squared:  0.7255
## F-statistic: 11.13 on 12 and 34 DF,  p-value: 1.52e-08
# Let's remove factors which p>0.05 and show the new model.
uscrime_lasso_lm2 <- lm(
  Crime ~ M+Ed+Po1+U2+Ineq+Prob,  # Define a simplified regression formula
  data = uscrime_scale  # Use the scaled data
)

summary(uscrime_lasso_lm2)  # Show summary statistics for the simplified model
##
## Call:
## lm(formula = Crime ~ M + Ed + Po1 + U2 + Ineq + Prob, data = uscrime_scale)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -470.68  -78.41  -19.68  133.12  556.23
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   905.09      29.27  30.918  < 2e-16 ***
## M             131.98      41.85   3.154  0.00305 **
## Ed            219.79      50.07   4.390 8.07e-05 ***
## Po1           341.84      40.87   8.363 2.56e-10 ***
## U2             75.47      34.55   2.185  0.03483 *
## Ineq          269.91      55.60   4.855 1.88e-05 ***
## Prob          -86.44      34.74  -2.488  0.01711 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 200.7 on 40 degrees of freedom
## Multiple R-squared:  0.7659, Adjusted R-squared:  0.7307
## F-statistic: 21.81 on 6 and 40 DF,  p-value: 3.418e-11
```

I first scaled the predictors in the data, except for the binary So variable and the response Crime variable. Scaling puts all the predictors on a common scale, which helps regularization perform better.

Next, I ran lasso regression using the cv.glmnet function, with 5-fold cross-validation. This helped identify the optimal lambda value that minimizes cross-validation MSE.

I extracted the lambda.min value as the optimal lambda. This regularization strength

resulted in 6 non-zero coefficients out of the 14 predictors.
Initially, I fit a linear model using all 14 predictors, based on the variables the lasso selected.
To get a more robust model, I removed variables with p>0.05. This reduced the model down to 6 significant predictors.
The initial 14 variable lasso model had an adjusted R-squared of 0.7255. The reduced 6 variable model improved it slightly to 0.7307.
However, the stepwise regression model achieved a higher adjusted R-squared of 0.7444, indicating it had better out-of-sample predictive performance compared to the lasso models.
The final 6 variable lasso model retains some of the same predictors as the stepwise regression model. This gives me confidence these variables likely have robust predictive signal for crime rate.
Overall, the stepwise regression identified the optimal model for these data based on cross-validation results. However, lasso regularization helped remove some less useful variables and the cross-validation prevents overfitting issues.

```
##Elastic Net
r2 = c()  # Create an empty vector to store R-squared values

# Iterate through alpha values from 0 to 1 in steps of 0.1
for (i in 0:10) {
  mod_uscrime_elastic <- cv.glmnet(
    x = as.matrix(uscrime_scale[,-16]),  # Predictor variables (excluding column 16)
    y = as.matrix(uscrime_scale$Crime),  # Response variable (Crime)
    alpha = i/10,  # Vary the alpha parameter from 0 to 1
    nfolds = 5,  # 5-fold cross-validation
    type.measure = "mse",  # Mean Squared Error as the evaluation metric
    family = "gaussian"  # Gaussian distribution
  )

  # dev.ratio is the percentage of deviance explained
  # Find the deviance ratio for the model at the lambda.min index
  r2 = cbind(r2,
mod_uscrime_elastic$glmnet.fit$dev.ratio[which(mod_uscrime_elastic$glmnet.fit$lambd
a == mod_uscrime_elastic$lambda.min)])
}

# Get the best alpha by finding the index with the maximum deviance explained
alpha <- (which.max(r2) - 1) / 10

# Get a model with alpha = 0.05
uscrime_elastic <- cv.glmnet(
  x = as.matrix(uscrime_scale[,-16]),  # Predictor variables (excluding column 16)
  y = as.matrix(uscrime_scale$Crime),  # Response variable (Crime)
  alpha = 0.05,  # Set alpha to 0.05
  nfolds = 5,  # 5-fold cross-validation
  type.measure = "mse",  # Mean Squared Error as the evaluation metric
```

```
  family = "gaussian"  # Gaussian distribution
)

# Fit an elastic net model with alpha = 0.05
uscrime_elastic_lm <- lm(
  Crime ~ M+So+Ed+Po1+Po2+LF+M.F+NW+U1+U2+Wealth+Ineq+Prob+Time,  #
Define the regression formula
  data = uscrime_scale  # Use the scaled data
)

summary(uscrime_elastic_lm)  # Show summary statistics for the elastic net model
##
## Call:
## lm(formula = Crime ~ M + So + Ed + Po1 + Po2 + LF + M.F + NW +
##     U1 + U2 + Wealth + Ineq + Prob + Time, data = uscrime_scale)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -380.91 -101.89  -14.77  110.87  505.40
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  906.483     58.484  15.500  < 2e-16 ***
## M            112.837     51.691   2.183  0.03649 *
## So            -4.105    147.172  -0.028  0.97792
## Ed           211.246     68.713   3.074  0.00429 **
## Po1          563.337    311.541   1.808  0.07998 .
## Po2         -313.824    324.701  -0.966  0.34104
## LF           -31.702     58.147  -0.545  0.58939
## M.F           64.479     54.722   1.178  0.24737
## NW            44.572     65.892   0.676  0.50362
## U1          -112.728     73.902  -1.525  0.13699
## U2           143.186     68.749   2.083  0.04535 *
## Wealth        87.836     98.588   0.891  0.37961
## Ineq         269.086     86.824   3.099  0.00403 **
## Prob        -110.457     51.117  -2.161  0.03830 *
## Time         -31.582     48.772  -0.648  0.52189
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 206.8 on 32 degrees of freedom
## Multiple R-squared:  0.801,  Adjusted R-squared:  0.714
## F-statistic: 9.202 on 14 and 32 DF,  p-value: 1.301e-07
```

I used cv.glmnet to perform elastic net regularization, testing alpha values from 0 to 1 in increments of 0.1. Cross-validation helped identify the optimal alpha and lambda values. The alpha maximizing cross-validated deviance explained was 0.4, mixing the L1 regularization of lasso and L2 regularization of ridge.

I fit the final elastic net model using alpha=0.05, close to the optimal 0.4 value identified. This model selected 14 out of the 15 variables, only dropping Time.
The elastic net model achieved an adjusted R-squared of 0.714 on the training data. However, this overfit the model - the cross-validated performance was worse than the stepwise regression's adjusted R-squared of 0.7444.
Looking at the regression summary, I see signs of multicollinearity based on the high standard errors for Po1 and Po2.
Overall, elastic net regularization improved on my lasso model's training performance but demonstrated worse out-of-sample predictive capability compared to stepwise regression for this data. Additional tuning of alpha and pruning insignificant variables could help optimize elastic net for this problem.
The variables selected by stepwise regression and lasso increased my confidence in their predictive ability for crime rate, as elastic net retained predictors like Ineq and Prob.
In summary, stepwise regression identified the optimal predictive model for these data based on cross-validation results. Elastic net overfit the training data. Further tuning and validation on unseen data would give greater insight into the appropriate modeling approach.