Assignment #2

Janelle Morano

2/14/2022

This assignment fits and tests OLS, ridge, and lasso regression models to average annual runoff vs. many other characteristics for 211 catchments across the Northeast US.

A-D) Download and read in data. Standardize the variables.

```
# Read in data
Gages2 <- read.table("/Users/janellemorano/Git/Reference-R-scripts/Envtl-Multivariate-Stats/data/gages2

# Function to standardize
scale2 <- function(x) {
  x <- (x- mean(x))/sd(x)
  return(x)
}

# Standardize each variable in the dataset and save to Gages2.scale
# MARGIN = 2 applies the function scale2 to each column
Gages2.scale <- data.frame(apply(Gages2, MARGIN = 2, FUN=scale2))
# head(Gages2.scale)
```

E) Fit a standard linear regression for annual average runoff vs. the other covariates for the 211 catchments. Do not include an intercept in this regression because the data were standardized in the previous step by centering the runoff data around 0.

```
# '.' applies lm to all columns; +0 removes intercept
lm.Gages2.scale <- lm(runoff ~ . +0, data = Gages2.scale)
summary(lm.Gages2.scale)
```

```
##
## Call:
## lm(formula = runoff ~ . + 0, data = Gages2.scale)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.05998 -0.35788  0.00505  0.39857  1.55008
##
## Coefficients:
##                      Estimate Std. Error t value Pr(>|t|)
## PPTAVG_BASIN         0.247902   0.135405   1.831   0.0688 .
## DRAIN_SQKM.log       0.070113   0.107109   0.655   0.5136
## BAS_COMPACTNESS     -0.086435   0.083804  -1.031   0.3038
## T_AVG_BASIN         -0.029815   0.458921  -0.065   0.9483
## PET                 -0.434367   0.487480  -0.891   0.3741
## ARTIFPATH_PCT       -0.055182   0.072428  -0.762   0.4472
## BFI_AVE              0.110502   0.086445   1.278   0.2028
```

```
## PERDUN                 0.068328   0.082811   0.825   0.4104
## PERHOR                -0.112175   0.088308  -1.270   0.2057
## TOPWET                 0.154006   0.115716   1.331   0.1850
## DEVNLCD06             -0.043112   0.084609  -0.510   0.6110
## FORESTNLCD06           0.177977   0.219463   0.811   0.4185
## PLANTNLCD06           -0.079150   0.195206  -0.405   0.6856
## AWCAVE.log            -0.045114   0.140098  -0.322   0.7478
## PERMAVE.log           -0.005651   0.133691  -0.042   0.9663
## BDAVE                  0.056100   0.094705   0.592   0.5544
## OMAVE.log              0.027116   0.102819   0.264   0.7923
## WTDEPAVE               0.037886   0.120315   0.315   0.7532
## ROCKDEPAVE             0.219900   0.114812   1.915   0.0571 .
## NO4AVE                -0.073859   0.505194  -0.146   0.8839
## NO200AVE               0.110173   0.154788   0.712   0.4776
## NO10AVE                0.484342   0.533392   0.908   0.3651
## KFACT_UP              -0.136155   0.114211  -1.192   0.2348
## RFACT                  0.167609   0.153614   1.091   0.2767
## ELEV_MEAN_M_BASIN      0.170114   1.598947   0.106   0.9154
## ELEV_MAX_M_BASIN      -0.463264   0.322102  -1.438   0.1521
## ELEV_MIN_M_BASIN      -1.165825   1.022860  -1.140   0.2559
## ELEV_MEDIAN_M_BASIN    0.267400   1.323247   0.202   0.8401
## ELEV_STD_M_BASIN       0.056960   0.228043   0.250   0.8031
## ELEV_SITE_M            0.916682   0.934966   0.980   0.3282
## RRMEAN                 0.033227   0.254881   0.130   0.8964
## RRMEDIAN.log          -0.145778   0.274008  -0.532   0.5954
## SLOPE_PCT.log         -0.006511   0.147462  -0.044   0.9648
## ASPECT_DEGREES         0.075576   0.105432   0.717   0.4744
## ASPECT_NORTHNESS      -0.080235   0.054818  -1.464   0.1451
## ASPECT_EASTNESS       -0.069366   0.113888  -0.609   0.5433
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.6458 on 175 degrees of freedom
## Multiple R-squared:  0.6524, Adjusted R-squared:  0.5809
## F-statistic: 9.124 on 36 and 175 DF,  p-value: < 2.2e-16
```

```
# Remember that Betahat is the Estimate/StdError = t-value
```

Q: Which of the different predictors have a statistically significant relationship to flow based on a standard t-test framework? At what significance level?

Answer: Currently, none of the predictors have a statistically significant relationship to flow at <0.05, but PPTAVG_BASIN and ROCKDEPAVE do at <0.10.

Q: How would you interpret the magnitude of the regression coefficients for those covariates, i.e., how would you articulate how much runoff changes per change in the covariates? Think about the standardization you did in step c for your answer here.

Answer: The estimate for PPTAVG_BASIN is 0.247902 and ROCKDEPAVE is 0.219900, so the runoff would increase by a factor of 0.247902 and 0.219900 from the average runoff for every incremental increase of PPTAVG_BASIN and ROCKDEPAVE, respectively Y = (0.247902 * x_PPTAVG_BASIN) + (0.219900 * x_ROCKDEPAVE).

F) Create a vector of runoff predictions and calculate the root mean squared error (RMSE) of these predictions.

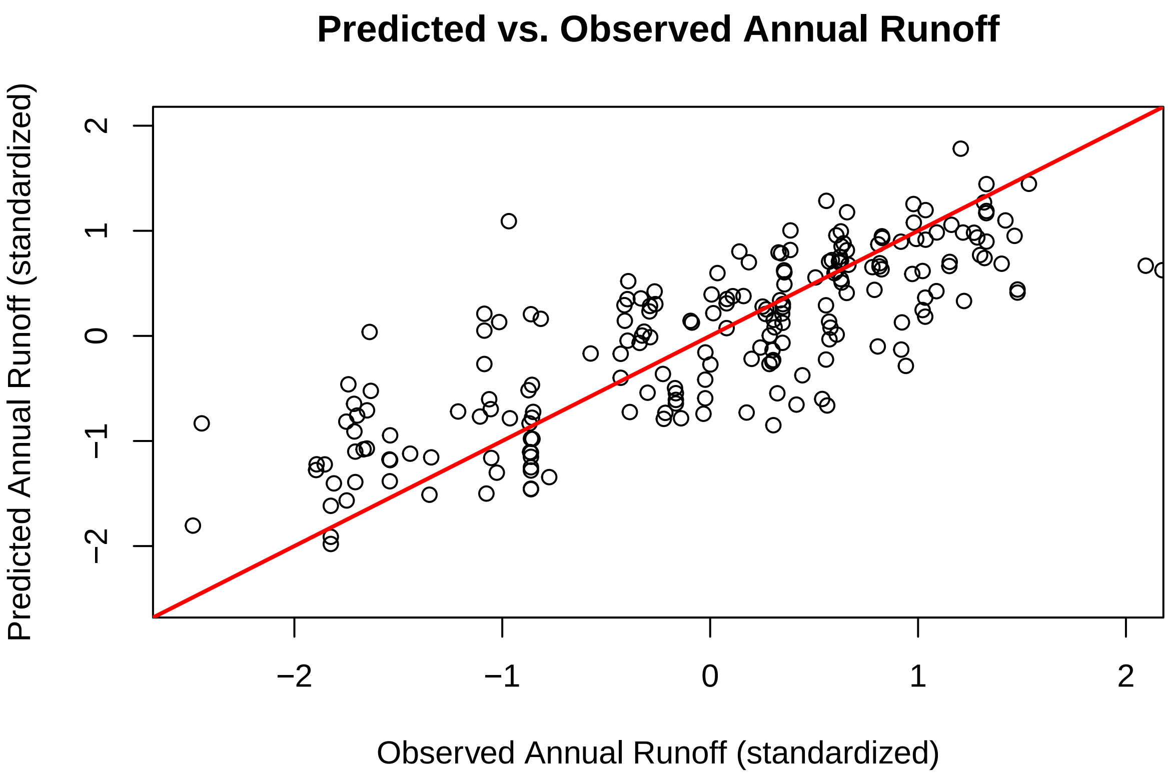
$$RMSE = \sqrt{\frac{\Sigma(Pred_i \breve{} Obs_i)^2}{n}}$$

```r
#Create a vector of runoff predictions based on the model. These are predictions
pred.runoff <- predict(lm.Gages2.scale)
#Calculate the root mean squared error (RMSE) of the predictions.
sqrt(mean((Gages2.scale$runoff - pred.runoff)^2))
```

```
## [1] 0.5881706
```

Plot your predicted values against the observed values. Be sure to include a 1:1 line and to label your axes appropriately (what exactly is being plotted on each axis?).

```r
plot(x = Gages2.scale$runoff,
     y = pred.runoff,
     xlim = c(-2.5, 2),
     ylim = c(-2.5, 2),
     xlab='Observed Annual Runoff (standardized)',
     ylab='Predicted Annual Runoff (standardized)',
     main='Predicted vs. Observed Annual Runoff')
# add line passing through the intercept and slope
abline(a = 0,
       b = 1,
       col = "red",
       lwd = 2)
```



G) Calculate the VIF for each covariate and sort the VIF values from smallest to largest.

```r
library(car)
```

```
## Loading required package: carData
```

```r
sort(vif(lm.Gages2.scale))
```

```
## Warning in vif.default(lm.Gages2.scale): No intercept: vifs may not be sensible.
```

```
##    ASPECT_NORTHNESS       ARTIFPATH_PCT              PERDUN     BAS_COMPACTNESS 
##            1.512900            2.641062            3.452593            3.535864 
##           DEVNLCD06             BFI_AVE              PERHOR               BDAVE 
##            3.604096            3.762243            3.926198            4.515552 
##           OMAVE.log      ASPECT_DEGREES       DRAIN_SQKM.log     ASPECT_EASTNESS 
##            5.322501            5.596471            5.775855            6.530206 
##            KFACT_UP          ROCKDEPAVE              TOPWET            WTDEPAVE 
##            6.567254            6.636530            6.741536            7.287941 
##         PERMAVE.log         PPTAVG_BASIN          AWCAVE.log       SLOPE_PCT.log 
##            8.998545            9.230811            9.881688           10.947796 
##               RFACT            NO200AVE         PLANTNLCD06        FORESTNLCD06 
##           11.880367           12.062673           19.184698           24.248889 
##     ELEV_STD_M_BASIN              RRMEAN        RRMEDIAN.log     ELEV_MAX_M_BASIN 
##           26.181840           32.707301           37.800295           52.234215 
##         T_AVG_BASIN                 PET              NO4AVE             NO10AVE 
##          106.033880          119.641542          128.494770          143.239126 
##          ELEV_SITE_M     ELEV_MIN_M_BASIN ELEV_MEDIAN_M_BASIN   ELEV_MEAN_M_BASIN 
##          440.109046          526.744829          881.556664         1287.172165 
```

The VIF values of many of the predictors are >10, therefore, several of these predictors are correlated with each other. This means that the variance for the predictors are inflated and the p-values are underestimating significance of predictors.

H) Fit ridge and lasso regressions in R using the glmnet package. Calculate the best lambda value for the regression based on a K-fold cross validation (CV).

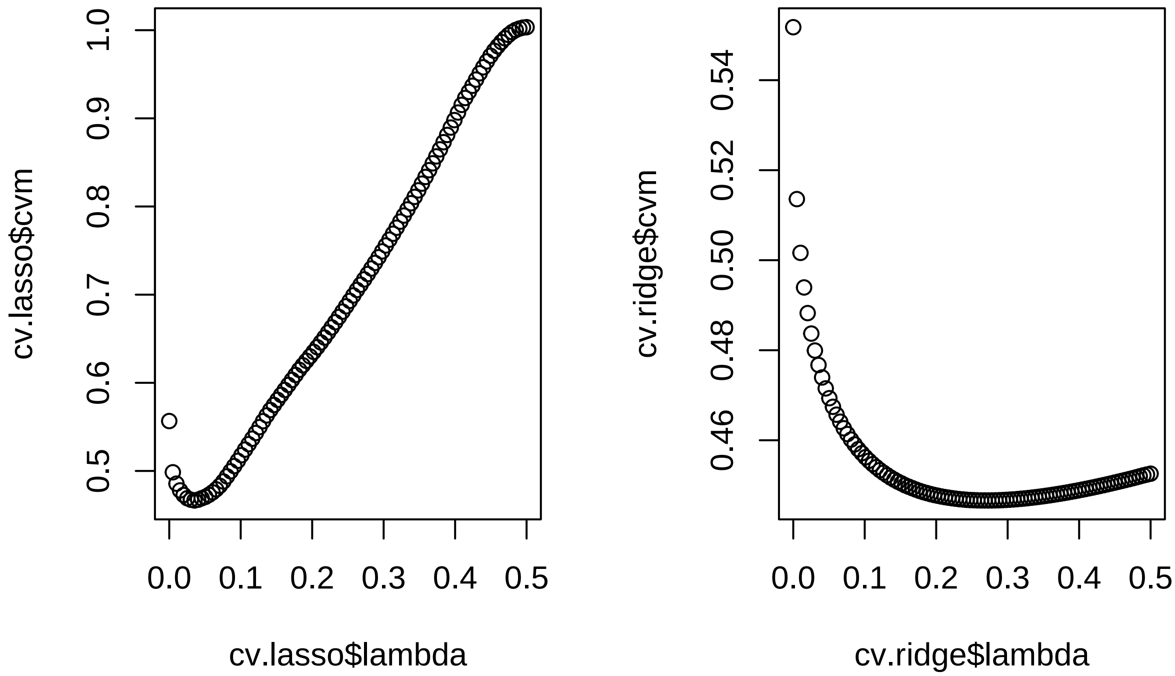
```r
library(glmnet)
```

```
## Loading required package: Matrix

## Loaded glmnet 4.1-3
```

```r
# Create 100 lambda values between 0 and 0.5 (use sequence() to make these lambda values.
lambda.seq <- seq(0, 0.5, length=100)
# Change the variable types from a data.frame to a data matrix to use in glmnet.
X <- data.matrix(Gages2.scale) # turns the data frame into a data matrix
x <- X[,2:ncol(X)] # pulls out the independent variables
y <- X[,1] # pulls out the dependent variable

# The cv.glmnet function will automatically select a 10-fold CV and average the cross-validated mean sq
# Lasso regression: alpha=1
cv.lasso <- cv.glmnet(x, y, lambda = lambda.seq, alpha = 1)
# Ridge regression: alpha = 0
cv.ridge <- cv.glmnet(x, y, lambda = lambda.seq, alpha = 0)
```

I) Plot the mean cross-validated error values vs. lambda. The lambda values and the cross-validated mean error values are stored in the objects returned by cv.glmnet. Select and report the lambda with the smallest mean cross-validated error for both ridge and lasso regression.

```r
# plot the mean cross-validated error values against the lambda values for both ridge and lasso cross-v
par(mfrow = c(1, 2))
plot(cv.lasso$lambda, cv.lasso$cvm)
plot(cv.ridge$lambda, cv.ridge$cvm)
```




```
# Select the lambda with the smallest mean cross-validated error
min(cv.lasso$lambda.min)
```

```
## [1] 0.03535354
```

```
min(cv.ridge$lambda.min)
```

```
## [1] 0.2727273
```

J) Fit a final lasso and ridge model using the glmnet() function and the selected lambda values.

```
# Lasso regression: alpha=1
lasso <- glmnet(x, y, lambda = cv.lasso$lambda.min, alpha = 1)

# Ridge regression: alpha = 0
ridge <- glmnet(x, y, lambda = cv.ridge$lambda.min, alpha = 0)
```

Report the fitted regression coefficients for the OLS, ridge, and lasso regressions and all covariates in a table.

```
# Grab the estimates for each of the models
beta.OLS <- as.matrix(summary(lm.Gages2.scale)$coefficients[,2])
beta.lasso <- lasso$beta
beta.ridge <- ridge$beta

# Put them together
table1 <- cbind(beta.OLS, beta.lasso, beta.ridge)
# print(table1)
# I can't figure out how to convert the sparse matrix into a data table or add a row to id the columns,
```

Q: Compare the coefficient values, focusing on variables with high VIF values from (g). What sign and magnitude of coefficients does OLS regression assign to these variables?

A: The estimates of the coefficients with high VIF values under OLS do not have any distinction from other estimates. All of the estimates have positive values and vary in magnitude.

Q:How do the lasso and ridge approaches differ in how they assign the coefficients compared to the OLS regression?

	Estimates		
Coefficient	OLS	Lasso	Ridge
PPTAVG_BASIN	0.13540533	0.30860799	0.180992467
DRAIN_SQKM.log	0.10710851	0.009324169	0.038193573
BAS_COMPACTNESS	0.08380379	-0.058969443	-0.053022563
T_AVG_BASIN	**0.45892125**	**-0.118143762**	**-0.110978783**
PET	**0.48748005**	**-0.061504135**	**-0.1059857**
ARTIFPATH_PCT	0.07242778	.	0.006221935
BFI_AVE	0.08644488	0.104383259	0.085865904
PERDUN	0.0828111	0.066957592	0.092138742
PERHOR	0.08830838	-0.025353064	-0.05646072
TOPWET	0.11571649	0.16284501	0.094914569
DEVNLCD06	0.08460851	-0.080182108	-0.07261981
FORESTNLCD06	**0.21946323**	**0.047263262**	**0.106476117**
PLANTNLCD06	**0.19520607**	**-0.176153134**	**-0.123388204**
AWCAVE.log	0.14009783	.	0.045576512
PERMAVE.log	0.13369095	0.032534501	0.080846274
BDAVE	0.09470464	.	0.026863513
OMAVE.log	0.10281908	.	0.017117419
WTDEPAVE	0.12031459	.	-0.006358601
ROCKDEPAVE	0.11481176	0.102052511	0.087135703
NO4AVE	**0.50519445**	**0.287112301**	**0.153737901**
NO200AVE	**0.15478812**	**.**	**0.019368301**
NO10AVE	**0.53339222**	**0.087747843**	**0.146863204**
KFACT_UP	0.11421095	-0.053033181	-0.048536545
RFACT	**0.15361399**	**0.011014292**	**0.102269712**
ELEV_MEAN_M_BASIN	**1.59894735**	**.**	**-0.002100631**
ELEV_MAX_M_BASIN	**0.32210197**	**.**	**-0.014515986**
ELEV_MIN_M_BASIN	**1.02285951**	**.**	**-0.025115908**
ELEV_MEDIAN_M_BASIN	**1.32324713**	**.**	**0.002091872**
ELEV_STD_M_BASIN	0.22804257	.	0.00557425
ELEV_SITE_M	0.93496638	.	-0.014858791
RRMEAN	**0.25488143**	**.**	**0.014815344**
RRMEDIAN.log	**0.27400812**	**.**	**-0.000735858**
SLOPE_PCT.log	0.14746168	.	-0.009755201
ASPECT_DEGREES	0.10543212	0.068495853	0.051229434
ASPECT_NORTHNESS	0.05481772	-0.082350009	-0.076045914
ASPECT_EASTNESS	0.11388835	.	-0.031895876

Figure 1: Estimates from OLS, Lasso, and Ridge regressions. Bolded coefficients and estimates had high (>10) VIF values.

A: I am not seeing a distinct rule, which makes me wonder if I did something wrong. But from what I see, under LASSO, estimates are missing from many of the coefficients with high VIF. Under LASSO and Ridge, many are negative, the opposite of the direction under OLS, and are smaller than the OLS estimates.

K) We will now test which of these regression approaches provide the best out-of-sample predictions. Randomly split the database into 2 equal size and mutually exclusive subsets (subset 1 and subset 2). You can use the "sample()" function to select half of the 211 catchments at random without replacement for subset 1, and put the remaining catchments into subset 2.

```
subset1 <-sample(1:nrow(X),
                 size=round(nrow(X)/2),
                 replace=F) #training data
subset2 <- (1:nrow(X))[-subset1] #validation set (new data)
# Take these randomly selected numbers and pull the rows of data that correspond to make subset dataset
Gages2.scale.subset1 <- Gages2.scale[subset1,]
Gages2.scale.subset2 <- Gages2.scale[subset2,]
```

L) Fit a linear regression via OLS to the data for subset 1, the training data.

```
subset1.OLS <- lm(runoff ~ 0+ ., data = Gages2.scale.subset1)
```

M) Predict annual runoff using the OLS-based model for subset 2, the validation data.

```
pred.subset1.OLS <- predict(subset1.OLS, newdata = Gages2.scale.subset2)
```

Calculate the root mean square error (RMSE) of these predictions.

```
# sqrt of the mean of observed values (from observation data) minus the estimated (prediction) data
sqrt(mean((Gages2.scale.subset2$runoff - pred.subset1.OLS)^2))
```

```
## [1] 0.7892588
```

N) Select lambda values for both lasso and ridge regressions based on the cv.glmnet function and the data in subset 1.

```
# Repeat lasso and ridge procedure in (h) above, but using subset1 data
# Create 100 lambda values between 0 and 0.5 (use sequence() to make these lambda values.
lambda.seq2 <- seq(0, 0.5, length=100)
# Change the variable types from a data.frame to a data matrix to use in glmnet.
subset1.matrix <- data.matrix(Gages2.scale.subset1) # turns the data frame into a data matrix
x2 <- subset1.matrix[,2:ncol(subset1.matrix)] # pulls out the independent variables
y2 <- subset1.matrix[,1] # pulls out the dependent variable

# The cv.glmnet function will automatically select a 10-fold CV and average the cross-validated mean sq
# Lasso regression: alpha=1
subset1.lasso <- cv.glmnet(x2, y2, lambda = lambda.seq2, alpha = 1)
# Ridge regression: alpha = 0
subset1.ridge <- cv.glmnet(x2, y2, lambda = lambda.seq2, alpha = 0)
```

O) Fit lasso and ridge regression models to the subset1 data using the optimized lambda values.

```
# Lasso regression: alpha=1
subset1.lasso <- glmnet(x, y, lambda = subset1.lasso$lambda.min, alpha = 1)

# Ridge regression: alpha = 0
subset1.ridge <- glmnet(x, y, lambda = subset1.ridge$lambda.min, alpha = 0)
```

P) Predict annual runoff using the fitted ridge and lasso models for the data in subset 2. Calculate the mean square error of these predictions.

```r
# make testing data a matrix now
subset2.matrix <- data.matrix(Gages2.scale.subset2)

subset1.lasso.pred <- predict(subset1.lasso, newx = subset2.matrix[,2:ncol(subset2.matrix)])

subset1.ridge.pred <- predict(subset1.ridge, newx = subset2.matrix[,2:ncol(subset2.matrix)])

# RMSE
#...observed values (from test/validation data) minus the estimated (prediction) data
sqrt(mean((subset2.matrix[,1] - subset1.lasso.pred)^2))
```

```
## [1] 0.6341788
```

```r
sqrt(mean((subset2.matrix[,1] - subset1.ridge.pred)^2))
```

```
## [1] 0.6410159
```

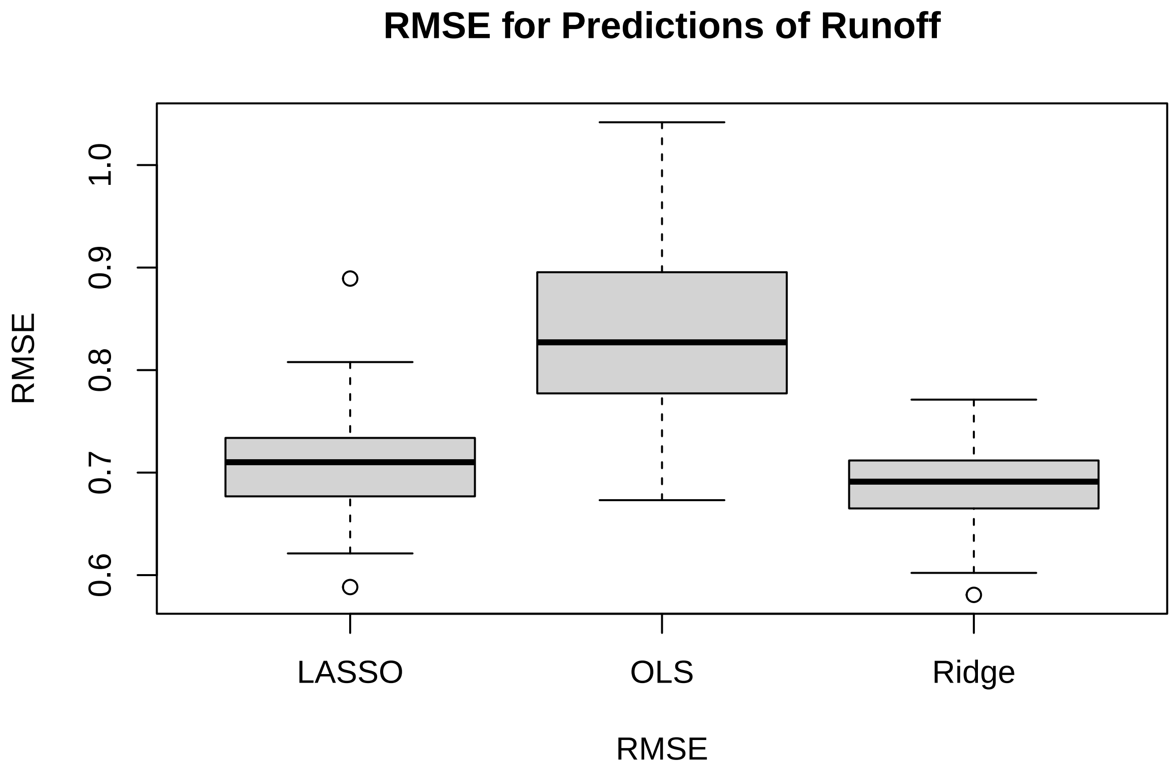
Q) Repeat process 100 times.

```r
rmse.OLS <- c()
rmse.lasso <- c()
rmse.ridge <- c()

for (i in 1:100) {
#Randomly split database Gages2.scale into 2 equal sizes (train and validate)
  train <-sample(1:nrow(X),
                 size=round(nrow(X)/2),
                 replace=F) #training data
  test <- (1:nrow(X))[-train] #validation/testing set (new data)
  train <- Gages2.scale[train,]
  test <- Gages2.scale[test,]

#Fit OLS on train
  fit.OLS <- lm(runoff ~ 0+ ., data = train)

#Predict OLS on test
  predict.OLS <- predict(fit.OLS, newdata = test)

#Calculate RMSE and add to results
  a <- c(sqrt(mean((test$runoff - predict.OLS)^2)))
  rmse.OLS <- rbind(rmse.OLS, a)

# Select lambda values
  lambda.seql <- seq(0, 0.5, length = 100)
  train.matrix <- data.matrix(train)
  x <- train.matrix[,2:ncol(train.matrix)]
  y <- train.matrix[,1]
  train.lasso <- cv.glmnet(x,y, lambda = lambda.seql, alpha = 1)
  train.ridge <- cv.glmnet(x,y, lambda = lambda.seql, alpha = 0)

# Fit lasso and ridge with optimized lambda values on set1
  fit.lasso <- glmnet(x,y, lambda = train.lasso$lambda.min, alpha = 1)
  fit.ridge <- glmnet(x,y, lambda = train.ridge$lambda.min, alpha = 0)

# Predict with validation set
```

```
  test.matrix <- data.matrix(test)
  pred.lasso <- predict(fit.lasso, newx = test.matrix[,2:ncol(test.matrix)])
  pred.ridge <- predict(fit.ridge, newx = test.matrix[,2:ncol(test.matrix)])

# Calculate RMSE and report
# ...observed values (from test/validation data) minus the estimated (prediction) data
  b <- sqrt(mean((test.matrix[,1] - pred.lasso)^2))
  rmse.lasso <- rbind(rmse.lasso, b)
  c <- sqrt(mean((test.matrix[,1] - pred.ridge)^2))
  rmse.ridge <- rbind(rmse.ridge, c)
  }

# Convert each matrix of rmse to dataframe, rename, and add column to ID
rmse.OLS <- as.data.frame(rmse.OLS)
rmse.OLS$RMSE <- rmse.OLS$V1
rmse.OLS$regtype <- "OLS"

rmse.lasso <- as.data.frame(rmse.lasso)
rmse.lasso$RMSE <- rmse.lasso$V1
rmse.lasso$regtype <- "LASSO"

rmse.ridge <- as.data.frame(rmse.ridge)
rmse.ridge$RMSE <- rmse.ridge$V1
rmse.ridge$regtype <- "Ridge"

results <- rbind(rmse.OLS, rmse.lasso, rmse.ridge)
```

R) Boxplot of RMSE

```
boxplot(RMSE ~ regtype,
        data = results,
        main ="RMSE for Predictions of Runoff",
        xlab ="RMSE")
```



The prediction errors with the penalized regression methods, LASSO and Ridge, had significantly lower errors in the predictions than with OLS. This is because of collinearity between multiple predictors in the model, which inflates the variance of the predictors. Under LASSO and Ridge regression, the variances are constrained, which is reflected in the above graph.

S) Dormann et al. 2013 How do lasso and ridge regression approaches compared to other methods for their out-of-sample prediction skill? Many methods performed adequately at low to moderate levels of collinearity, but LASSO and ridge regression were among the few methods that predicted well under high levels of collinearity.