*Multivariate Behavioral Research* 1036965, 14-089.R2

**Using Lasso for Predictor Selection and to Assuage Overfitting: A Method Long Overlooked in Behavioral Sciences**

**Daniel M. McNeish**

*Department of Human Development and Quantitative Methodology,*

*University of Maryland, College Park*

**Appendix**

*Code to for Example Lasso Analysis in R and SAS*

**R Code**

***Lasso to select predictors and estimate regression coefficients***

ca<-read.csv(file.choose(), header=TRUE)

*### Read in the data from CSV. This command allows users to select the file from wherever it is located###*

IV<-(ca[,1:10])

*### Create an object of candidate predictors. The example had ten predictors so the value spans from 1 to 10. It is easiest to setup the data such that all candidate predictors are in sequential columns ###*

IV1=scale(IV,FALSE,FALSE)

*### Create a matrix like object from the candidate predictors. the "FALSE" arguments tell R not to center or standardize the variables ###*

require (glmnet)

*### The glmnet package must be downloaded before using the require command and the following functions ###*

attach(ca)

*### Attaching the data set does not require using data prefixes, this is just for convenience ###*

set.seed(1028)

### set a seed value so that a results are reproducible and do not change every time the program is run##

Lambda=cv.glmnet(IV1,testscr);Lambda

*### Estimate optimal values for the regularization parameter. This outputs the minimum and the value with one SE. testscr is the dependent variable in this example. Values will change slightly each time this code is run. ###*

coef(Lambda, s=Lambda$lambda.1se)

*### Obtain regression coefficients for candidate predictors. s = is the value for the regularization parameter ###*

RegCoef=glmnet(IV1,testscr,family = "gaussian",alpha = 1)

*### This function is used to plot the the coefficients in a later function ###*

plot(Lambda)

*### Plot the MSE for values of Lambda. This is what produces Figure 1 ###*

*### Code for Figure 2 - Run all three at once ###*

plot(RegCoef, xvar="lambda", lwd=1.8 ) abline(v=log(Lambda$lambda.1se))

abline(v=log(Lambda$lambda.min))

*### This plots the coefficients for different values of lambda without restricting the y-axis scale at all. The "abline" functions impose the minimum and within 1 SE values of Lambda onto the plot. ###*

*### Code for Figure 3 - Run all three at once ###*

plot(RegCoef, xvar="lambda",ylim=c(-1.5,1.5), lwd=1.8 )

abline(v=log(Lambda$lambda.1se))

abline(v=log(Lambda$lambda.min))

*### This plot shows the value of the regression coefficients for different values of lambda. The "abline" functions impose the minimum and within 1 SE values of Lambda onto the plot. The ylmin option restricts the range of the y axis to be only between -1.5 and 1.5 because most coefficients are small but one is much larger. Readers may need to change this depending on the scale of their variables ###*

***Outputting p-values for Lasso coefficients***

ca<-read.csv(file.choose(), header=TRUE)

*### Same as above, can be omitted is previously code is run first ###*

IV<-(ca[,1:10])

*### Same as above, can be omitted is previously code is run first ##*

df=nrow(IV)-1

*### the number of observations minus 1, used in computations below ###*

attach(ca)

*### Same as Above ###*

require (covTest)

*### Must download covTest package before using subsequent commands ###*

IV2=scale(IV,TRUE,TRUE)/sqrt(df)

*### this command does require that the values are centered and standardized ###*

LarsCoef=lars(IV2,testscr)

*### the object LarsCoef houses the steps of the Least Angle Regression algorithm for the IV2 candidate predictors and testscr dependent variable ###*

covTest(LarsCoef,IV2,testscr)

*### CovTest will output a p-value for each candidate predictor and the residual variance ###*

**SAS Code**

***Lasso to select predictors and estimate regression coefficients***

**proc** **glmselect** data= ca plots=criteria(unpack);

model testscr = enrltot teachers calwpct mealpct computer compstu expnstu str avginc elpct

/selection=lasso choose =cv cvmethod=random (**10**);**run**;

\*\*\*All candidate predictors appear after the equal sign in the model statement. By default SAS will use BIC (choose=SBC) and the local minimum to select lambda. choose=cv uses the cross-validation method as glmnet does in R and cvmethod=random (10) tells SAS to use 10 folds as is also done in R. The plots option in the proc glmselect statement compares different selection criteria for choosing how many predictors to retain in the model. The coefficient estimates are identical to the output in R.\*\*\*;

***Outputting regression coefficient plots similar to Figure 2***

**proc** **glmselect** data=ca plots(unpack)=(coefficients criteria);

model testscr= enrltot teachers calwpct mealpct computer compstu expnstu str avginc elpct

/selection=lasso stop=none choose=cv cvmethod=random (**10**);

**run**;

\*\*\*This code will produce a plot that looks similar to Figure 2 or Figure 3. The stop = none option will use global minima rather than local minimum to select predictors so the coefficient estimates output at the bottom of the output will differ from the coefficients output by the previous code (most often in the direction of selecting more predictors). It is recommended to use the first set of code to output the regression coefficients. SAS also does not offer the 1SE criteria for selecting choosing how harshly to penalize coefficients; the effect of this can be seen in the example since the selection criteria between 3 predictors and 5 predictors is extremely close \*\*\*;

\*\*\*The SAS plot uses different axes than the plot output by R although the idea is generally the same. Instead of log(lambda) on the x-axis, SAS uses either the L1-norm (stepaxis=normb) or more simply the step and the effect entering the model(stepaxis=effect). stepaxis=effect is the default. The y-axis also shows the standardized coefficients rather than the unstandardized coefficients. \*\*\*;