**Regression with the Lasso**

Definition

The lasso is yet regression shrinkage method, like ridge regression. While ridge regression adds a shrinkage penalty proportional to the sum of the squared variable coefficients , the lasso uses a penalty proportional to the sum of the absolute values of the variable coefficients . There are two equivalent formulas for the lasso:

Uses

As with Ridge regression, the Lasso should be considered whenever there is a strong potential for variance and a small amount of bias may be acceptable. While variance is always present, it is of particular concern when the sample size is small or when the number of predictors is large compared to the sample size. The key benefit of the lasso is that the shrinkage penalty can **force variable coefficients to equal 0**, thus making the Lasso a variable selection method. This is extremely beneficial in cases where you have too many predictors, especially when your predictors outnumber your observations. The lasso also generates simpler models than ridge regression because ridge regression does not cause variable coefficients to exactly equal 0.

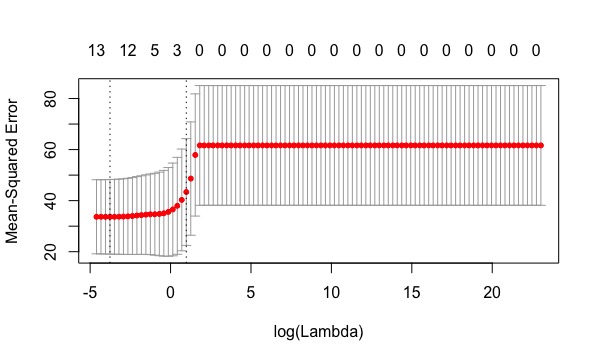
Application in R

**lasso.model = glmnet(*x*, *y*, alpha = 1, lambda = λ)**

* *x* = predictors – a matrix - must be constructed with the model.matrix() function, omitting the first column (the intercept)
* *y* = dependent variable – a vector
* alpha = 1 tells the glmnet() function to use the lasso (alpha = 0 is for ridge regression)
* lambda = your desired lambda values, which if specified must be a vector of decreasing multiple lambdas (we used grid <- 10 ^ seq(10, -2, length=100) and specified lambda=grid in the glmnet() function call).

**cv.lasso = cv.glmnet(x, y, alpha=1, lambda = λ)**

Performs cross validation on a lasso regression model. As with glmnet(), you can use your own vector of lambda values (should be vector of decreasing values – never a single value), or do not specify a lambda to allow the function to choose its own 100 default (decreasing) values for lambda. After executing this function, use **cv.lassot$lambda.min** to find the lambda that results in the lowest MSE. Here is what a plot of cv.lasso might look like:



To predict, use the predict() function as follows:

**lasso.pred <- predict(lasso.mod, s=bestlam, newx=x[test, ])**

* *s* is the lambda you want to use to make the prediction
* *newx* is the matrix of predictors to use

**Classification with the Lasso**

**Differences in R from Regression:**

* Need a vector (a factor) as the dependent variable
* Need to specify family="binomial" in the glmnet() and cv.glm() calls
* Need to specify type="class" in the predict() call to get class predictions. Type = “response” will return probabilities. You can use type=”coefficients” to get the coefficients for the specified value of s.

**Code Reference:**

See RidgeRegressionAndTheLasso.R