Penalized regression

Statistical Learning, 2011

Niels Richard Hansen
September 19, 2011

Penalized regression is implemented in several different R packages. Ridge regression can, in principle, be carried out using `lm` or `lm.fit`, but it is also implemented in MASS as the `lm.ridge` function. A main package for penalized regression involving the 1-norm penalty is `glmnet`, which is an extremely fast and well implemented package. We illustrate the usages on the prostate cancer data set.

```
> require(glmnet)      ### elastic net (and lasso ...)
> require(MASS)        ### lm.ridge for ridge regression
> require(leaps)       ### Best subset selection
> require(ggplot2)
```

Below, we construct a data set, where we center all the variables and, in addition, scale the predictors to have unit variance. This is mostly done here explicitly for reasons of comparison. Some functions implement standardization as a default or an option before the parameters are estimated. This is to bring the coefficients on a comparable scale. The resulting estimated parameters are usually transformed back to the original scale before they are returned. This will ease future predictions. Below we are careful in standardizing the test data the same way as the training are standardized.

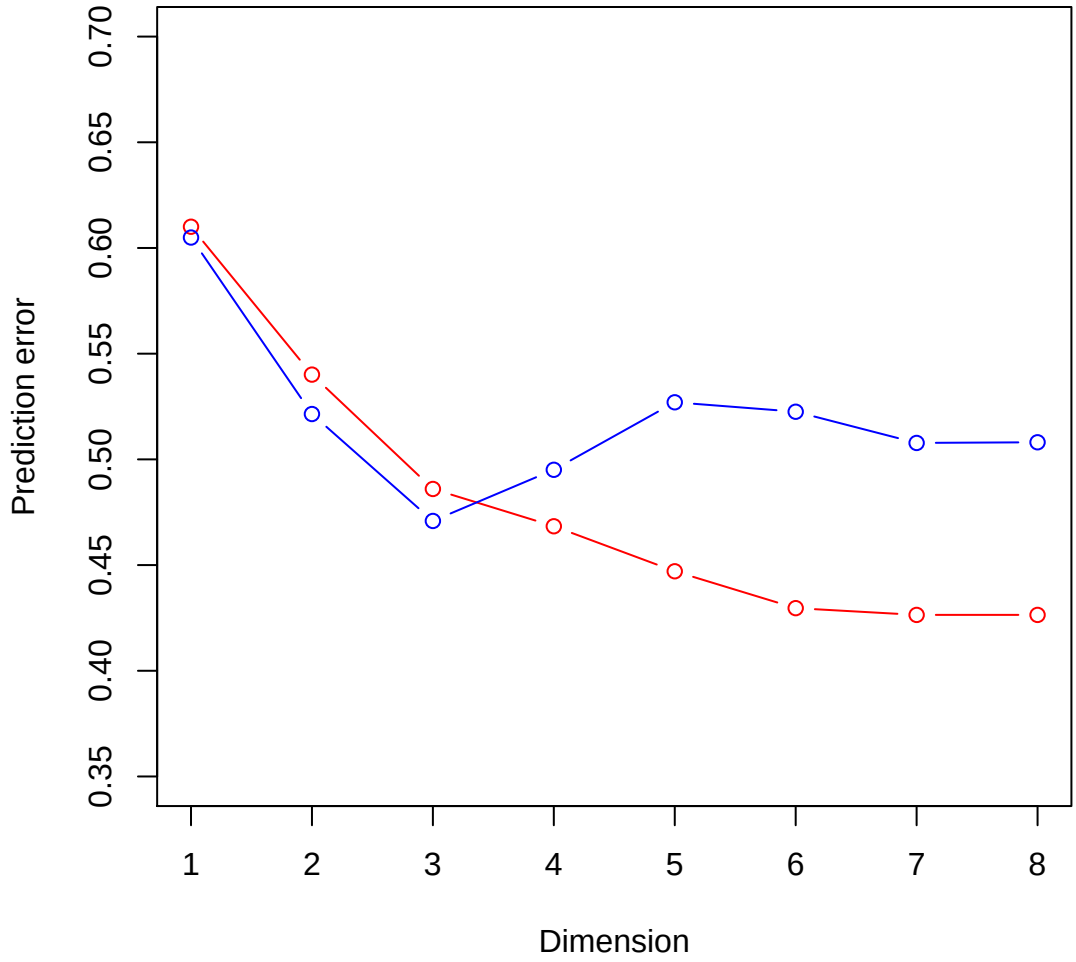
```
> prostate <- read.table("http://www-stat-
class.stanford.edu/%7Etibs/ElemStatLearn/datasets/prostate.data")

> ### Selection of the training data only
> ### Doubt about whether the original selection was random ...
> set.seed(1)
> trainIndex <- sample(prostate$train)
> prostateTest <- prostate[!trainIndex, 1:9]
> prostate <- prostate[trainIndex, 1:9]
> ### Centering and scaling
> xbar <- colMeans(prostate)
> vbar <- colSums(scale(prostate, center = xbar, scale =
FALSE)^2)/(dim(prostate)[1]-1)
> prostateScaled <- scale(prostate,
+                         center = xbar,
+                         scale = c(sqrt(vbar)[-9], 1))
> prostateTestScaled <- scale(prostateTest,
+                             center = xbar,
+                             scale = c(sqrt(vbar)[-9], 1))
> Ntrain <- dim(prostate)[1]
> Ntest <- dim(prostateTest)[1]
```

1 Best subset

First, we revisit best subset selection. We compute the best model for each dimension and predict on the test data set. We compute and plot the average prediction error on the test and training data, respectively. The best model is found, in this case, with 3 predictors (`lcavol`, `lweight` and `svi`).

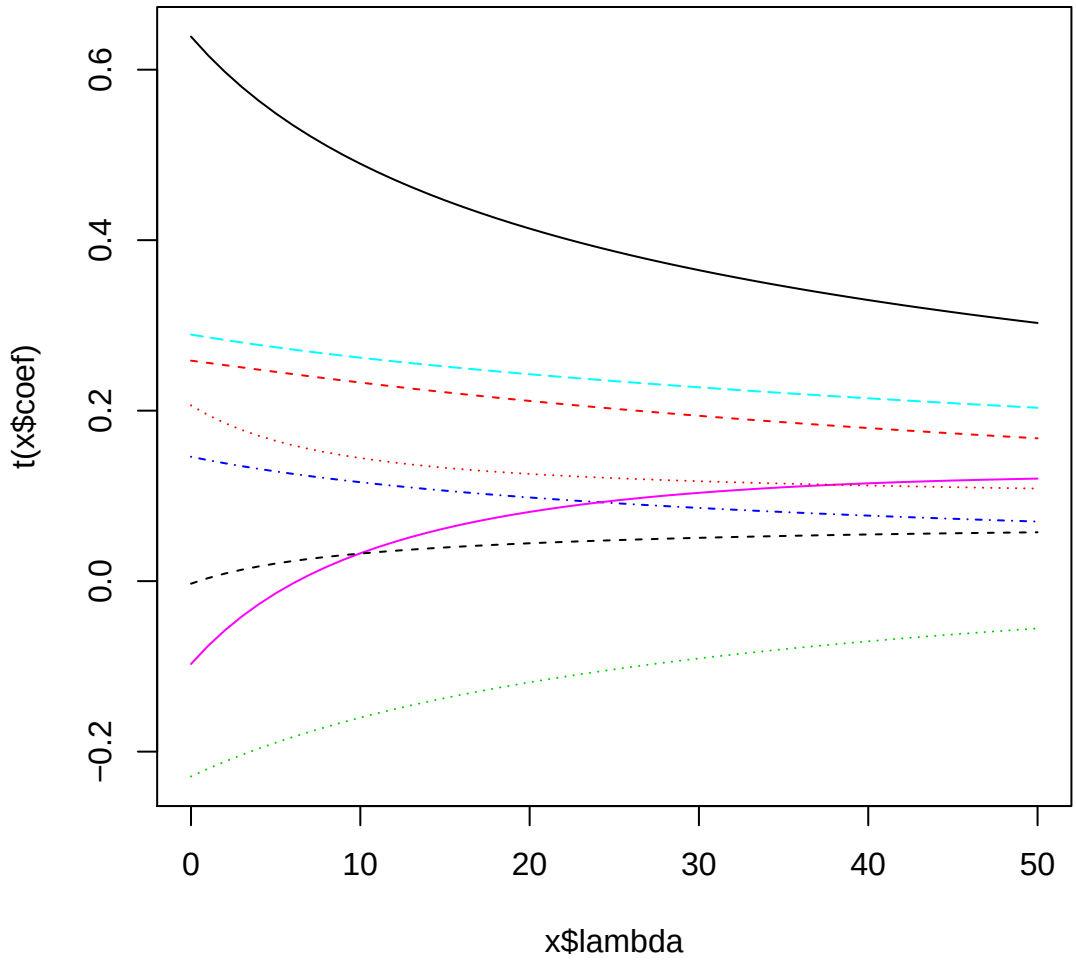
```
> prostateSubset <- regsubsets(x = prostateScaled[, -9],
+                              y = prostateScaled[, 9],
+                              intercept = FALSE,
+                              nbest = 1)
> bestCoef <- coefficients(prostateSubset, 1:8)
> rssPredict <- numeric()
> for(i in 1:8) {
+   X <- prostateTestScaled[, names(bestCoef[[i]]), drop = FALSE]
+   yHat <- X %*% bestCoef[[i]]
+   rssPredict[i] <- mean((prostateTestScaled[, 9] - yHat)^2)
+ }
> plot(1:8, prostateSubset$ress[, 1]/Ntrain, ylim = c(0.35, 0.7),
+      type = "b", col = "red", ylab = "Prediction error", xlab =
"Dimension")
> points(1:8, rssPredict, type = "b", col = "blue")
```



2 Ridge

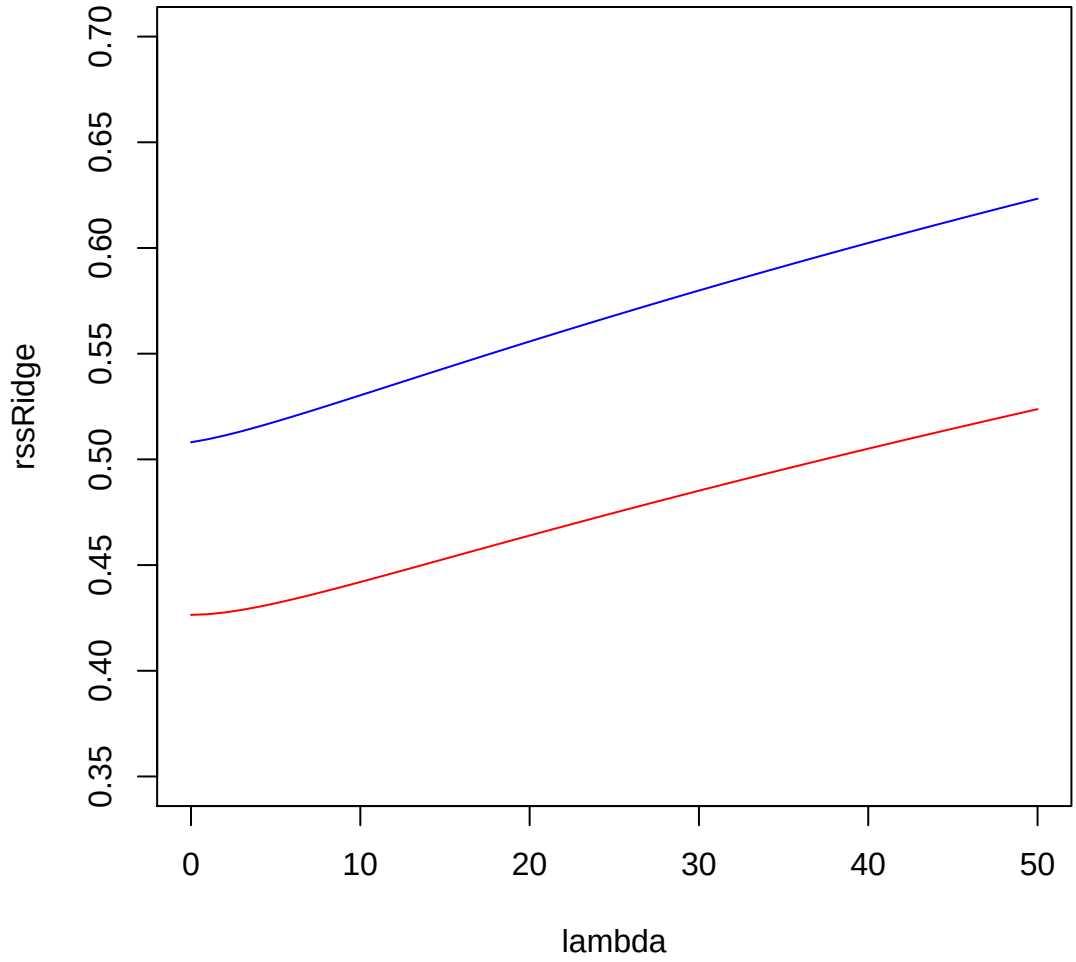
The ridge regression models are fitted by a call similar to an `lm` call except that we have to specify a λ sequence. From the result we can, for instance, produce a coefficient profile plot of the estimated parameters as a function of the penalty parameter λ. We clearly see the shrinkage effect. As a function of λ all the parameters shrink towards 0.

```
> lambda <- seq(0, 50, 1)
> prostateRidge <- lm.ridge(lpsa ~ . - 1,
+                           data = as.data.frame(prostateScaled),
+                           lambda = lambda)
> plot(prostateRidge)
```



As for best subset selection we can also plot the average prediction error for the test and the training data as a function of λ, which controls the model complexity in this case.

```
> rssRidge <-  colMeans((prostateScaled[, 9] -
+                         prostateScaled[, -9] %*%
+                         t(coefficients(prostateRidge)))^2
+                        )
> rssRidgePredict <- colMeans((prostateTestScaled[, 9] -
+                              prostateTestScaled[, -9] %*%
+                              t(coefficients(prostateRidge)))^2
+                             )
> plot(lambda, rssRidge, type = "l", col = "red", ylim = c(0.35, 0.7))
> lines(lambda, rssRidgePredict, col = "blue")
```

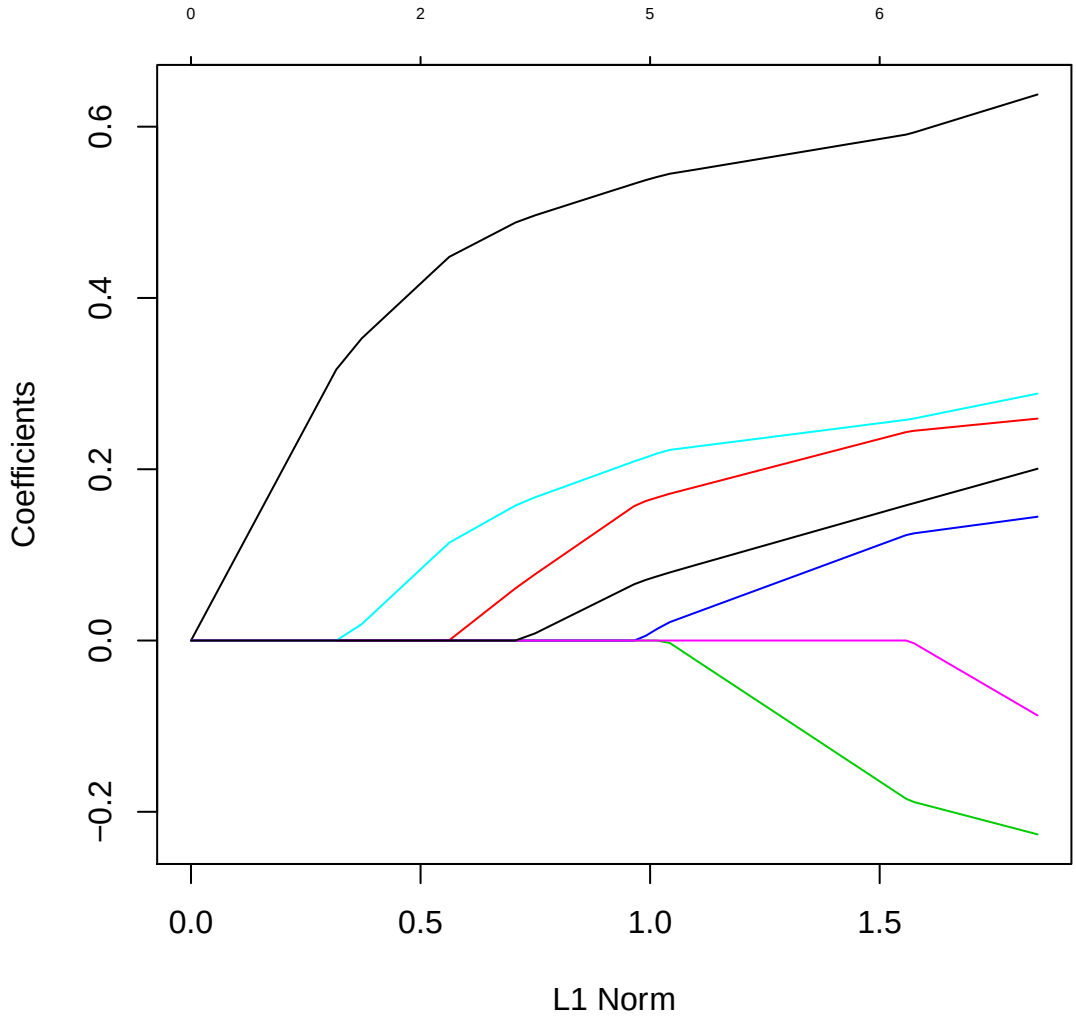


3 Lasso

For lasso we use `glmnet`, which has an interface more similar to `lm.fit` than to `lm`. You have to specify the matrix of predictors and the vector of response values. You don't need to specify a λ sequence, in fact, you should rely on the sequence of λ values computed automatically unless you know what you are doing. There is also an α argument to the function, which by default is 1 giving lasso. Values in $(0, 1)$ will produce the elastic net penalties, where the penalty function is a combination of the 1-norm as in pure lasso and the 2-norm as in ridge regression. Note that, though $\alpha = 0$ formally gives ridge regression, this choice of α is not possible with `glmnet`.

```
> prostateGlmnet <- glmnet(x = prostateScaled[, 1:8],
```

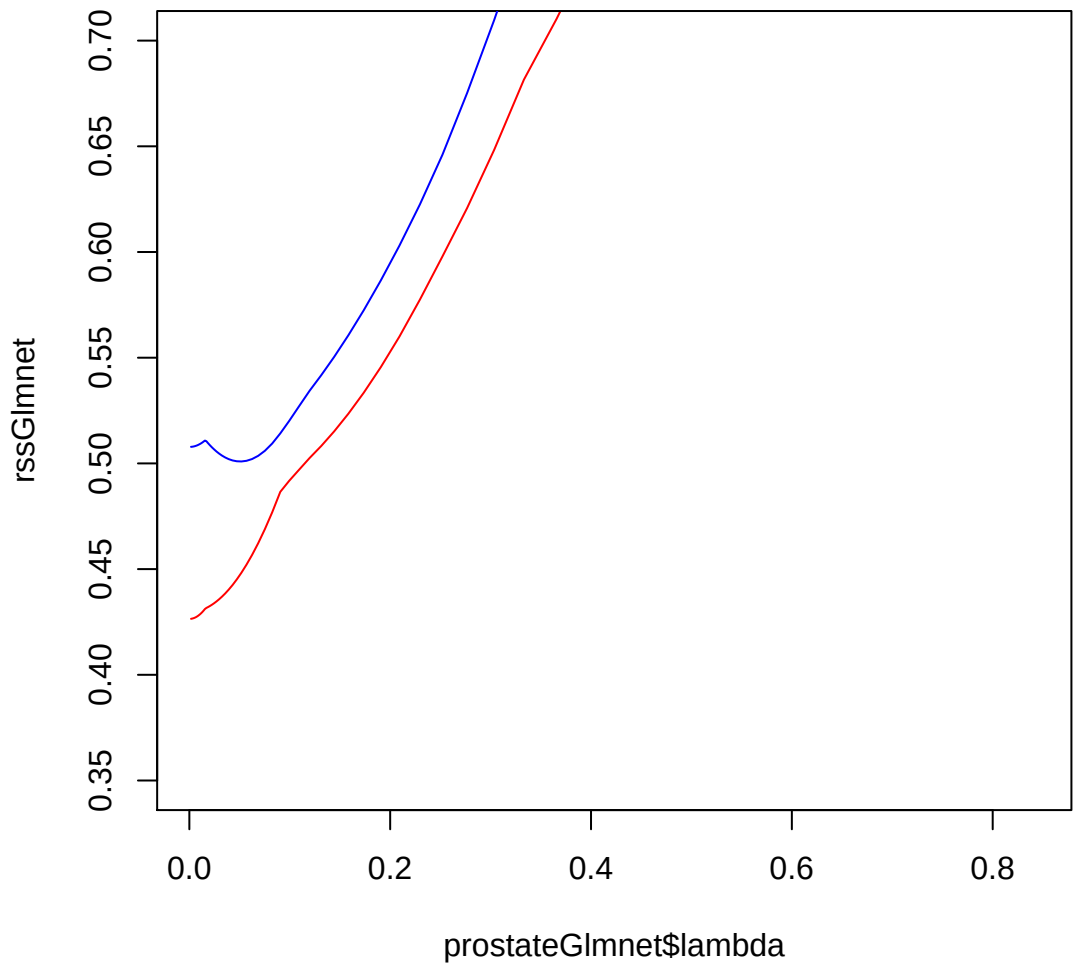
```
+                         y = prostateScaled[, 9],
+                         )
> plot(prostateGlmnet)
```



We compute and plot again the average prediction error for the test and the training data as a function of λ.

```
> rssGlmnet <- colMeans((prostateScaled[, 9] -
+                        predict(prostateGlmnet, prostateScaled[, -9]))^2
+                       )
> rssGlmnetPredict <- colMeans((prostateTestScaled[, 9] -
+                               predict(prostateGlmnet, prostateTestScaled[,
-9]))^2
+                              )
> plot(prostateGlmnet$lambda, rssGlmnet, type = "l", col = "red", ylim =
```

```
c(0.35, 0.7))
> lines(prostateGlmnet$lambda, rssGlmnetPredict, col = "blue")
```



We can, finally, compare the three resulting models and the average prediction errors estimated on the test data.

```
> compMat <- matrix(0, 8, 3)
> colnames(compMat) <- c("Best subset", "Ridge", "Lasso")
> rownames(compMat) <- names(prostate)[1:8]
> compMat[names(bestCoef[[3]]), "Best subset"] <- bestCoef[[3]]
> compMat[, "Ridge"] <-
coefficients(prostateRidge)[which.min(rssRidgePredict), ]
> compMat[, "Lasso"] <- coefficients(prostateGlmnet)[-1,
which.min(rssGlmnetPredict)]
> compMat
```

```
        Best subset        Ridge       Lasso
lcavol    0.6126496  0.643644414  0.56903878
lweight   0.2490049  0.260581059  0.20928299
age       0.0000000 -0.230861289 -0.09862584
lbph      0.0000000  0.147012639  0.07496092
svi       0.2921355  0.291408117  0.24104899
lcp       0.0000000 -0.097911853  0.00000000
gleason   0.0000000 -0.002976996  0.00000000
pgg45     0.0000000  0.207799316  0.12078733
```

```
> c(min(rssPredict), min(rssRidgePredict), min(rssGlmnetPredict))
```

```
[1] 0.4708751 0.5080935 0.5009177
```