Lasso and Ridge, Model and Coefficient stability

Walter K. Kremers, Mayo Clinic, Rochester MN

22 March 2024

The matter

It is well recognized that repeat terms in a predictor set will not impact a lasso model. The lasso model will split the coefficient weight over the multiple repeat predictors without changing either the L1 penalty or the model prediction. It is also well recognized that the ridge model will distribute the weight for a coefficient equally among repeat terms of the predictor to minimize the loss function for any given lambda. For the lasso then the models are invariant to adding a repeat predictor even if the individual coefficients may change or be "unstable". For the ridge model the coefficients are generally uniquely determined for any particular lambda penalty, but the addition of a repeat predictor can change all the non-zero coefficients in a model.

An example dataset

Set up a simple dataset with multiple predictors. For the moment there will be no repeat predictors.

```
# Simulate a simple example data set
set.seed(1)
nobs=100
beta = c(1,0.9,0.8,0.7,0.6,0.5,0.4,0.3,0.2,0.1,0.1,0.1,0,0,0)
xs1 = matrix(runif(nobs*length(beta)),nrow = nobs, ncol = length(beta))
## the first few rows of the matrix
round( xs1[1:5,] , digits=4 )
```

```
##        [,1]   [,2]   [,3]   [,4]   [,5]   [,6]   [,7]   [,8]   [,9]  [,10]
## [1,] 0.2655 0.6547 0.2675 0.6737 0.6589 0.5542 0.8143 0.9297 0.8587 0.8319
## [2,] 0.3721 0.3532 0.2186 0.0949 0.1851 0.6883 0.9288 0.9009 0.0344 0.7668
## [3,] 0.5729 0.2703 0.5168 0.4926 0.9544 0.6581 0.1475 0.7509 0.9710 0.2728
## [4,] 0.9082 0.9927 0.2690 0.4616 0.8978 0.6633 0.7498 0.6766 0.7451 0.1882
## [5,] 0.2017 0.6335 0.1812 0.3752 0.9437 0.4722 0.9757 0.6480 0.2733 0.2258
##       [,11]  [,12]  [,13]  [,14]  [,15]
## [1,] 0.5308 0.6904 0.3665 0.5243 0.2396
## [2,] 0.6849 0.2538 0.7414 0.1604 0.6478
## [3,] 0.3833 0.4110 0.9335 0.2051 0.9757
## [4,] 0.9550 0.1229 0.6732 0.9752 0.3780
## [5,] 0.1184 0.1320 0.7014 0.9664 0.4641
```

```
#y_ = xs1 %*% beta + 0.4*rnorm(nobs)
y_ = xs1 %*% beta + 0.5*rnorm(nobs)
dim(xs1)
```

```
## [1] 100  15
```

””

```
set.seed(2)
fold_n = 10
foldid = sample( rep( 1:fold_n , ceiling(nobs/fold_n) )[ 1:nobs ] , nobs )
```

```
lasso.fit1 = cv.glmnet(xs1,y_, family="gaussian", foldid=foldid)
beta.lasso.1 = as.matrix(coef(lasso.fit1))
lasso.lambda = lasso.fit1$lambda
```

```
ridge.fit1 = cv.glmnet(xs1,y_, family="gaussian", alpha=0, foldid=foldid)
beta.ridge.1 = as.matrix(coef(ridge.fit1))
ridge.lambda = ridge.fit1$lambda
```

Here we see that the models have very similar reductions in mean square error, though the ridge model has a slightly larger value.

```
## create a repeat of the first predictor
 xs2 = cbind( xs1, xs1[,1] )
# xs2 = cbind( xs1, xs1[,5] )

## fit the lasso model to the updated predictor set
lasso.fit2 = cv.glmnet(xs2,y_, family="gaussian", foldid=foldid, lambda=lasso.lambda)
beta.lasso.2 = as.matrix(coef(lasso.fit2))
```

```
## fit the ridge model to the updated predictor set
ridge.fit2 = cv.glmnet(xs2,y_, family="gaussian", alpha=0, foldid=foldid, lambda=ridge.lambda )
beta.ridge.2 = as.matrix(coef(ridge.fit2))
```

```
## put the betas from the 4 models into a common matrix
betas = cbind(rbind(beta.lasso.1,0), beta.lasso.2, rbind(beta.ridge.1,0), beta.ridge.2)
colnames(betas) = c("lasso 1", "lasso 2", "ridge 1", "ridge 2")
rownames(betas)[17] = "V1.2"

## betas for the 2 lasso and 2 ridge models
betas
```

```
##                lasso 1    lasso 2     ridge 1     ridge 2
## (Intercept) 1.47901155 1.47899647  1.45170094  1.52813904
## V1          0.86958738 0.81903337  0.56092086  0.34507829
## V2          0.52934281 0.52933904  0.41440465  0.35616655
## V3          0.48006198 0.48006188  0.40854852  0.34919842
## V4          0.61569576 0.61570293  0.43089265  0.38196236
## V5          0.00000000 0.00000000  0.09119542  0.07982309
## V6          0.08071029 0.08070591  0.22475712  0.19453154
## V7          0.00000000 0.00000000  0.15775255  0.13606545
## V8          0.08853796 0.08853647  0.20400382  0.16774467
## V9          0.06847093 0.06846833  0.21498740  0.18594321
## V10         0.00000000 0.00000000 -0.03939882 -0.02413824
## V11         0.02921647 0.02922120  0.14377377  0.13659904
## V12         0.00000000 0.00000000 -0.04680542 -0.03086014
## V13         0.00000000 0.00000000 -0.02078948 -0.02306384
## V14         0.00000000 0.00000000  0.03632404  0.02648512
## V15         0.00000000 0.00000000  0.03067040  0.02423857
## V1.2        0.00000000 0.05058390  0.00000000  0.34506906
```

```
## betas terms for the 2 repeat predictors
betas[c(2,17),]
```

```
##        lasso 1   lasso 2   ridge 1   ridge 2
## V1   0.8695874 0.8190334 0.5609209 0.3450783
## V1.2 0.0000000 0.0505839 0.0000000 0.3450691
```

```
## sum of betas terms for the 2 repeat predictors, i.e. the collective
## weight of the repeated term
colSums(betas[c(2,17),])
```

```
##   lasso 1   lasso 2   ridge 1   ridge 2
## 0.8695874 0.8696173 0.5609209 0.6901473
```

Here we see how for lasso model that the total weight for the coefficients for repeat terms have changed only in the 4'th decimal place, presumably because of the numerical algorithm. The weights for all other terms too are vary similar, differing only after a few decimal places. For the ridge model, the total weight for the coefficients for repeat terms have substantially increased form about 0.68 to 0.82, a change in the first decimal palce. Further, the weights for many cofficients have changed in either the first or 2nd decimal place. We can see this by calculating the change sin teh coefficients

Impact of repeats in the predicteds

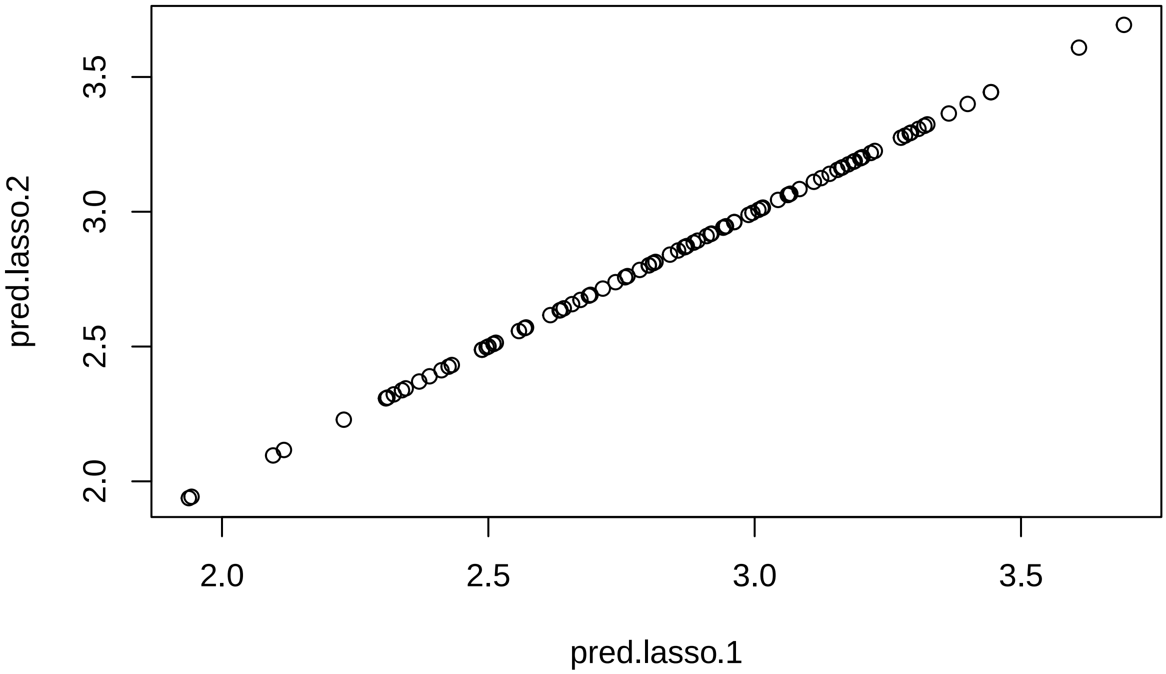
To understand the impact of these coefficient changes we look at the predicteds, i.e. 'X*betas', in the data used to fit the model. For the lasso model we see

```
pred.lasso.1 = predict( lasso.fit1, xs1)
pred.lasso.2 = predict( lasso.fit2, xs2)
cor(pred.lasso.1, pred.lasso.2)
```

```
##            lambda.1se
## lambda.1se          1
```

that the predicteds have a correlation of (almost) 1, and plotting one against teh other it is difficlut to see any differneces between the two.

```
plot(pred.lasso.1, pred.lasso.2)
```



For tthat the predicteds have a correlation of (almost) 1, and plotting one against teh other it is difficlut to see any differneces between the two.

Repearint thee calculaitons fo rhte reidge predicteds we see

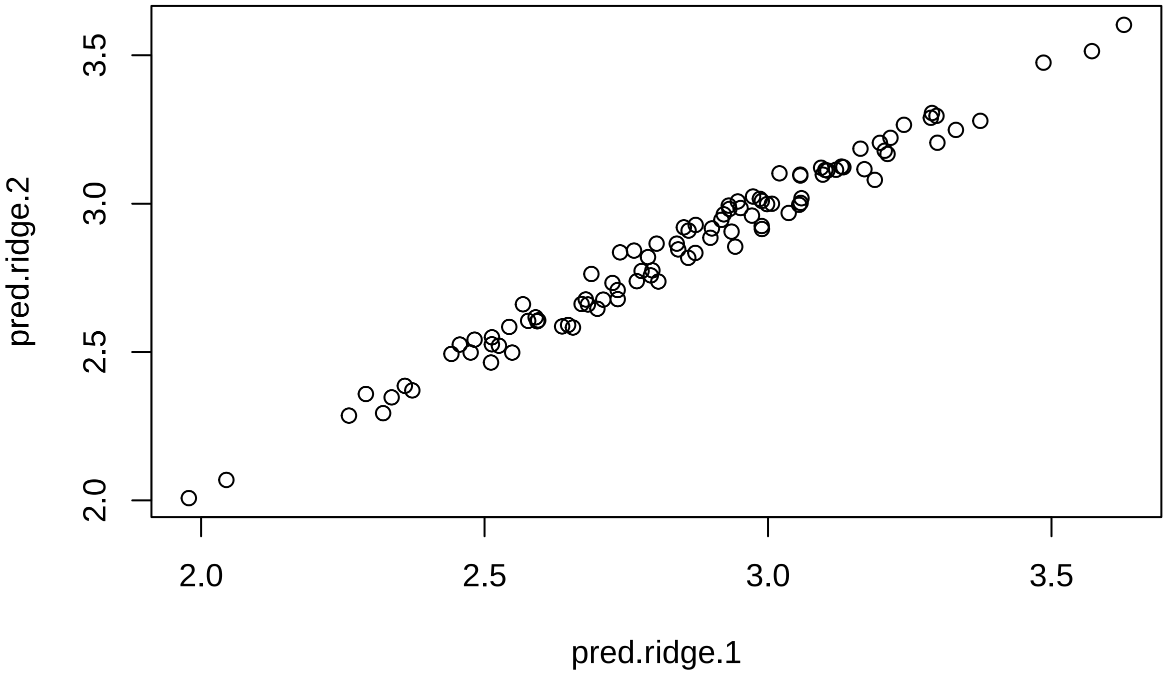
```
pred.ridge.1 = predict( ridge.fit1, xs1)
pred.ridge.2 = predict( ridge.fit2, xs2)
cor(pred.ridge.1, pred.ridge.2)
```

```
##              lambda.1se
## lambda.1se   0.989375
```

```
#ridge.compare = glm(pred.ridge.2 ~ pred.ridge.1, family="gaussian")
#summary(ridge.compare)
#names( ridge.compare )
#1 - ridge.compare$deviance/ridge.compare $null.deviance
```

a very large correlation of 0.989 but a plot of one against the other shows clear differences between the two.

```
plot(pred.ridge.1, pred.ridge.2)
```



The inclusion of a repeat term in the design matrix did not meaningful change the reduction in MSE for the lasso model but did have an ever so small impact on the MSE for the ridge model, as decribe by

```
devratio
```

```
## lasso 1 lasso 2 ridge 1 ridge 2
##  0.5424  0.5424  0.5477  0.5459
```

We do not compare here the deviance ratios between the lasso and ridge models as these are biased when we calculate them on the same data as used to derive the models. This can be done with nested cross validation, but this is better done with real data.