Cross Validated

Why do Lars and Glmnet give different solutions for the Lasso problem?

Asked 9 years, 1 month ago Active 3 years, 2 months ago Viewed 8k times

23

11

I want to better understand the R packages `Lars` and `Glmnet`, which are used to solve the Lasso problem:

$$min_{(\beta_0\beta)\in R^{p+1}}\left[\frac{1}{2N}\sum_{i=1}^{N}(y_i-\beta_0-x_i^T\beta)^2+\lambda||\beta||_{l_1}\right]$$

(for p Variables and N samples, see www.stanford.edu/~hastie/Papers/glmnet.pdf on page 3)

Therefore, I applied them both on the same toy dataset. Unfortunately, the two methods do not give the same solutions for the same data input. Does anybody have an idea where the difference comes from?

I obtained the results as follows: After generating some data (8 samples, 12 features, Toeplitz design, everything centered), I computed the whole Lasso path using Lars. Then, I ran Glmnet using the sequence of lambdas computed by Lars (multiplied by 0.5) and hoped to obtain the same solution, but I did not.

One can see that the solutions are similar. But how can I explain the differences? Please find my code below. There is a related question here: GLMNET or LARS for computing LASSO solutions? , but it does not contain the answer to my question.

Setup:

```
# Load packages.
library(lars)
library(glmnet)
library(MASS)

# Set parameters.
nb.features <- 12
nb.samples <- 8
nb.relevant.indices <- 3
snr <- 1
nb.lambdas <- 10

# Create data, not really important.
sigma <- matrix(0, nb.features, nb.features)
for (i in (1:nb.features)) {
  for (j in (1:nb.features)) {
    sigma[i, j] <- 0.99 ^ (abs(i - j))
  }
}

x <- mvrnorm(n=nb.samples, rep(0, nb.features), sigma, tol=1e-6, empirical=FALSE)
relevant.indices <- sample(1:nb.features, nb.relevant.indices, replace=FALSE)
x <- scale(x)
```

```
beta <- rep(0, times=nb.features)
beta[relevant.indices] <- runif(nb.relevant.indices, 0, 1)
epsilon <- matrix(rnorm(nb.samples),nb.samples, 1)
simulated.snr <-(norm(x %*% beta, type="F")) / (norm(epsilon, type="F"))

epsilon <- epsilon * (simulated.snr / snr)
y <- x %*% beta + epsilon
y <- scale(y)
```

lars:

```
la <- lars(x, y, intercept=TRUE, max.steps=1000, use.Gram=FALSE)
co.lars <- as.matrix(coef(la, mode="lambda"))
print(round(co.lars, 4))

#          [,1] [,2] [,3]   [,4]   [,5]   [,6]    [,7]   [,8]    [,9]   [,10]
#  [1,]  0.0000    0    0 0.0000 0.0000 0.0000  0.0000 0.0000  0.0000  0.0000
#  [2,]  0.0000    0    0 0.0000 0.0000 0.1735  0.0000 0.0000  0.0000  0.0000
#  [3,]  0.0000    0    0 0.2503 0.0000 0.4238  0.0000 0.0000  0.0000  0.0000
#  [4,]  0.0000    0    0 0.1383 0.0000 0.7578  0.0000 0.0000  0.0000  0.0000
#  [5,] -0.1175    0    0 0.2532 0.0000 0.8506  0.0000 0.0000  0.0000  0.0000
#  [6,] -0.3502    0    0 0.2676 0.3068 0.9935  0.0000 0.0000  0.0000  0.0000
#  [7,] -0.4579    0    0 0.6270 0.0000 0.9436  0.0000 0.0000  0.0000  0.0000
#  [8,] -0.7848    0    0 0.9970 0.0000 0.9856  0.0000 0.0000  0.0000  0.0000
#  [9,] -0.3175    0    0 0.0000 0.0000 3.4488  0.0000 0.0000 -2.1714  0.0000
# [10,] -0.4842    0    0 0.0000 0.0000 4.7731  0.0000 0.0000 -3.4102  0.0000
# [11,] -0.4685    0    0 0.0000 0.0000 4.7958  0.0000 0.1191 -3.6243  0.0000
# [12,] -0.4364    0    0 0.0000 0.0000 5.0424  0.0000 0.3007 -4.0694 -0.4903
# [13,] -0.4373    0    0 0.0000 0.0000 5.0535  0.0000 0.3213 -4.1012 -0.4996
# [14,] -0.4525    0    0 0.0000 0.0000 5.6876 -1.5467 1.5095 -4.7207  0.0000
# [15,] -0.4593    0    0 0.0000 0.0000 5.7355 -1.6242 1.5684 -4.7440  0.0000
# [16,] -0.4490    0    0 0.0000 0.0000 5.8601 -1.8485 1.7767 -4.9291  0.0000
#         [,11]  [,12]
#  [1,]  0.0000 0.0000
#  [2,]  0.0000 0.0000
#  [3,]  0.0000 0.0000
#  [4,] -0.2279 0.0000
#  [5,] -0.3266 0.0000
#  [6,] -0.5791 0.0000
#  [7,] -0.6724 0.2001
#  [8,] -1.0207 0.4462
#  [9,] -0.4912 0.1635
# [10,] -0.5562 0.2958
# [11,] -0.5267 0.3274
# [12,]  0.0000 0.2858
# [13,]  0.0000 0.2964
# [14,]  0.0000 0.1570
# [15,]  0.0000 0.1571
```

glmnet with lambda=(lambda_lars / 2):

```
glm2 <- glmnet(x, y, family="gaussian", lambda=(0.5 * la$lambda), thresh=1e-16)
co.glm2 <- as.matrix(t(coef(glm2, mode="lambda")))
print(round(co.glm2, 4))

#     (Intercept)      V1 V2 V3     V4     V5     V6      V7     V8      V9
# s0            0  0.0000  0  0 0.0000 0.0000 0.0000  0.0000 0.0000  0.0000
# s1            0  0.0000  0  0 0.0000 0.0000 0.0000  0.0000 0.0000  0.0000
# s2            0  0.0000  0  0 0.2385 0.0000 0.4120  0.0000 0.0000  0.0000
# s3            0  0.0000  0  0 0.2441 0.0000 0.4176  0.0000 0.0000  0.0000
# s4            0  0.0000  0  0 0.2466 0.0000 0.4200  0.0000 0.0000  0.0000
# s5            0  0.0000  0  0 0.2275 0.0000 0.4919  0.0000 0.0000  0.0000
# s6            0  0.0000  0  0 0.1868 0.0000 0.6132  0.0000 0.0000  0.0000
# s7            0 -0.2651  0  0 0.2623 0.1946 0.9413  0.0000 0.0000  0.0000
# s8            0 -0.6609  0  0 0.7328 0.0000 1.6384  0.0000 0.0000 -0.5755
# s9            0 -0.4633  0  0 0.0000 0.0000 4.6068  0.0000 0.0000 -3.3547
# s9            0 -0.4633  0   0 0.0000 0.0000 4.6069  0.0000 0.0000 -3.2547
# s10           0 -0.4819  0   0 0.0000 0.0000 4.7546  0.0000 0.0000 -3.3929
# s11           0 -0.4767  0   0 0.0000 0.0000 4.7839  0.0000 0.0567 -3.5122
# s12           0 -0.4715  0   0 0.0000 0.0000 4.7915  0.0000 0.0965 -3.5836
# s13           0 -0.4510  0   0 0.0000 0.0000 5.6237 -1.3909 1.3898 -4.6583
# s14           0 -0.4552  0   0 0.0000 0.0000 5.7064 -1.5771 1.5326 -4.7298
#         V10     V11     V12
# s0    0.0000  0.0000 0.0000
# s1    0.0000  0.0000 0.0000
# s2    0.0000  0.0000 0.0000
# s3    0.0000  0.0000 0.0000
# s4    0.0000  0.0000 0.0000
# s5    0.0000 -0.0464 0.0000
# s6    0.0000 -0.1293 0.0000
# s7    0.0000 -0.4868 0.0000
# s8    0.0000 -0.8803 0.3712
# s9    0.0000 -0.5481 0.2792
# s10   0.0000 -0.5553 0.2939
# s11   0.0000 -0.5422 0.3108
# s12   0.0000 -0.5323 0.3214
# s13 -0.0503  0.0000 0.1711
# s14   0.0000  0.0000 0.1571
```

r regression machine-learning lasso regularization

Share Cite Edit Follow Flag

edited Apr 13 '17 at 12:44
Community Bot
1

asked Aug 4 '12 at 15:17
Andre
561 3 8

3 Answers

Active Oldest Votes

22

Finally we were able to produce the same solution with both methods! First issue is that glmnet solves the lasso problem as stated in the question, but lars has a slightly different normalization in the objective function, it replaces $\frac{1}{2N}$ by $\frac{1}{2}$. Second, both methods normalize the data differently, so the normalization must be swiched off when calling the methods.

To reproduce that, and see that the same solutions for the lasso problem can be computed using lars and glmnet, the following lines in the code above must be changed:

```
la <- lars(X,Y,intercept=TRUE, max.steps=1000, use.Gram=FALSE)
```

to

```
la <- lars(X,Y,intercept=TRUE, normalize=FALSE, max.steps=1000, use.Gram=FALSE)
```

and

```
glm2 <- glmnet(X,Y,family="gaussian",lambda=0.5*la$lambda,thresh=1e-16)
```

to

```
glm2 <-
glmnet(X,Y,family="gaussian",lambda=1/nbSamples*la$lambda,standardize=FALSE,thresh=1e-

16)
```

Share Cite Edit Follow Flag

answered Aug 9 '12 at 8:48
Andre
561 3 8

2 I'm glad you figured this out. Any thoughts on which normalization method makes more sense? I've actually gotten worse results using normalization in glmnet (for lasso) and I'm still not sure why. – Ben Ogorek Nov 24 '13 at 22:58

I actually normalize the data offhand and apply these methods and compare if they are similar. Variables with smaller effects are usually seen to have different coefficients – KarthikS May 31 '18 at 18:15

0

Obviously if the methods use different models you will get different answers. Subtracting off the intercept terms does not lead to the model without the intercept because the best fitting coefficients will change and you do not change them the way you are approaching it. You need to fit the same model with both methods if you want the same or nearly the same answers.

Share Cite Edit Follow Flag

answered Aug 4 '12 at 16:01
Michael R. Chernick
39.2k 28 72 141

1 Yes, you are right, the methods use slightly different models, I was not aware of that. Thanks for the hint. (I will explain the differences more detailedly in a separate answer) – Andre Aug 9 '12 at 8:36

-2

Results have to be the same. lars package uses by default type="lar", change this value to type="lasso". Just lower the parameter 'thresh=1e-16' for glmnet since coordinate descent is based on convergence.

Share Cite Edit Follow Flag

answered Jul 12 '18 at 0:41
Marcool Lopez Cruz
1

2 Thank you for your answer. Maybe I'm misreading it, but it seems at odds with the resolution posted in Andre's answer six years ago. Please consider elaborating your post to include a fuller explanation of what you're trying to say and showing why we should believe it is correct and the other is not. – whuber ♦ Jul 12 '18 at 2:15