CompBio HW3: Regression

Yifeng Tao

January 17, 2016

1 Multivariable Regression

1.1 Maximum Likelihood & MSE

We know:

$$y_i = \beta^T x_i + \epsilon_i$$

and

$$\epsilon_i \sim N(0, \sigma^2)$$

Consider MSE, its goal:

$$\hat{\beta} = \arg\min_{\beta} \sum_{i=1}^{N} (y_i - \beta^T x_i)^2 = \arg\min_{\beta} M(\beta)$$

Consider Maximum likelihood,

$$H(\beta) = \ln \prod_{i=1}^{N} \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(y_i - \beta^T x_i)^2}{2\sigma^2}} = -\frac{1}{2\sigma^2} \sum_{i=1}^{N} (y_i - \beta^T x_i)^2 - N \ln(\sqrt{2\pi}\sigma) = -\frac{1}{2\sigma^2} M(\beta) - N \ln(\sqrt{2\pi}\sigma)$$

Its goal is:

$$\hat{\beta}^* = \arg\max_{\beta} H(\beta)$$

Note σ is a constant. Obviously,

$$\hat{\beta} = \arg\max_{\beta} H(\beta) \Leftrightarrow \hat{\beta} = -\frac{1}{2\sigma^2} \arg\min_{\beta} M(\beta) - N \ln(\sqrt{2\pi}\sigma) \Leftrightarrow \hat{\beta} = \arg\min_{\beta} M(\beta)$$

And,

$$\hat{\beta}^* = \hat{\beta}$$

1.2 $\beta_i^* \& \beta_i$

If we denote that:

$$X^T = [x^{(1)}, ..., x^{(N)}]$$
$$Y^T = [y^{(1)}, ..., y^{(N)}]$$
$$\beta^T = [\beta_1, ..., \beta_p]$$

Then, in the multivariable regression,

$$\beta = (X^T X)^{-1} X^T y$$

and β_i is the ith element in β

Similarly, we can also derive regression coefficient of single variable x_i:

$$\beta_i^* = \frac{[x_i^{(1)}, ..., x_i^{(N)}]}{[x_i^{(1)}, ..., x_i^{(N)}][x_i^{(1)}, ..., x_i^{(N)}]^T} Y$$

We found that, if we regress only to a single variable, the coefficient is only related to the observations of this variable itself. However, if we regress to all the variable, the coefficient is related to the covariance between different variables.

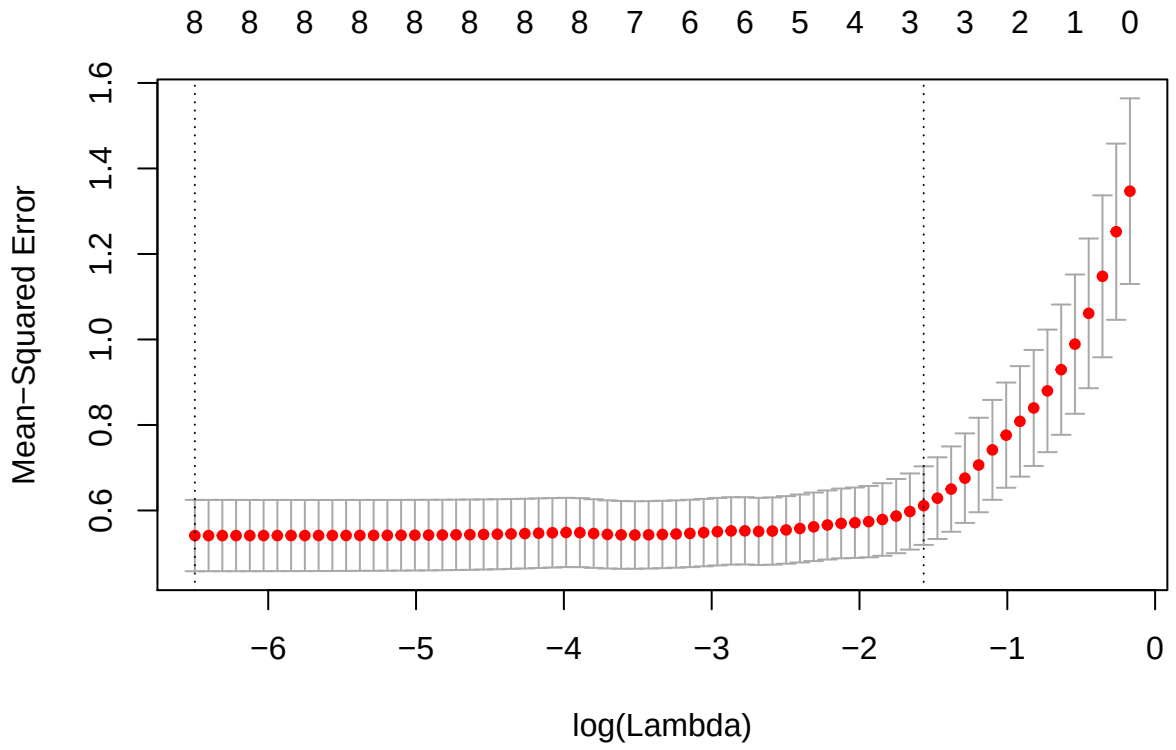
2 LASSO Algorithm

Read in the data:

```
data = read.table("Prostate.txt", header = TRUE, sep = "\t", row.names = 1)
x = data[,1:8]
y = data[,9]
x = as.matrix(x)
y = as.matrix(y)
```

Apply Glmnet to realize LASSO:

```
# Glmnet for realizing LASSO.
cvob = cv.glmnet(x, y, nfolds = 100)
plot(cvob)
```



```r
paraG = coef(cvob$glmnet.fit, s = cvob$lambda.1se)
```

Check the result:

```r
# Output Glmnet algorithm.
print(paraG)
```

```
## 9 x 1 sparse Matrix of class "dgCMatrix"
##                      1
## (Intercept) 0.7820637
## lcavol      0.4485189
## lweight     0.2804033
## age         .
## lbph        .
## svi         0.3383490
## lcp         .
## gleason     .
## pgg45       .
```

Use LARS to realize LASSO:

```r
# My Algorithm for realizing LASSO.
x = scale(data[,1:8])
sigma = attr(x, "scaled:scale")
y = data[,9]
y = y - mean(y)

lambda = cvob$lambda.1se
I = diag(8)

# Initialization.
para = solve((t(x) %*% x +lambda*I)) %*% t(x) %*% y

# Iteration.
while(TRUE) {
  temp=para
  for(i in 1:8) {
    c = (t(x[,i]) %*% (y-x %*% para+para[i]*x[,i]))/length(data[,1])
    para[i] = sign(c) * max((abs(c)-lambda),0)
  }
  if(max(abs(para-temp)) < 1e-16)
    break
}
# Normalization.
paraM = para/sigma
```

Output the result of our algorithm:

```r
# Output my algorithm.
print(paraM)
```

```
##              [,1]
## lcavol  0.4424871
## lweight 0.2791078
## age     0.0000000
## lbph    0.0000000
## svi     0.3416167
## lcp     0.0000000
## gleason 0.0000000
## pgg45   0.0000000
```

Note that the result of our algorithm is similar to that of Glmnet.