Coding Assignment 2

Due Monday, March 01

Implement Lasso using the Coordinate Descent (CD) algorithm and test your algorithm on the Boston housing data. Check [Rcode_W3_VarSel_RidgeLasso.html] on relevant background information on this dataset.

- First, load the transformed Boston Housing Data, `Coding2_myData.csv`.
- Next write your own function `MyLasso` to implement CD, which should output estimated Lasso coefficients similar to the ones returned by `R` with option "`standardized = TRUE`".

  In case you don't know where to start, you can follow the structure given on the next page to prepare your function. In our script, we run a fixed number of iterations, "`maxit = 100`," which seems enough for this assignment. You could set it to be a bigger number, or change it to a `while` loop to stop when some convergence criterion is satisfied.
- Test your algorithm with the following lambda sequence.

```
lam.seq = exp(seq(-1, -8, length.out = 80))
myout = MyLasso(X, y, lam.seq, maxit = 100)
```

- Produce a path plot for the 13 non-intercept coefficients along the lambda values in log scale.
- Check the accuracy of your algorithm against the output from `glmnet`. The maximum difference between the two coefficient matrices should be less than 0.005.

```
lasso.fit = glmnet(X, y, alpha = 1, lambda = lam.seq)
max(abs(coef(lasso.fit) - myout))
```

Students who use **Python** for this assignment can find the dataset `Coding2_myData.csv` and the target Lasso coefficients, `Coding2_lasso_coefs.csv` in the Coding Assignments folder at the Resources page on Piazza.

What you need to submit?

An R/Python Markdown file in HTML format, which should contain all code used to produce your results.

- You are allowed to use two packages: `MASS` (for the data) and `glmnet`.
- Name your file starting with `Assignment_2_xxxx_netID` where "`xxxx`" is the last 4-dig of your University ID.

```r
One_var_lasso = function(r, x, lam){
  ###############
  # YOUR CODE
  ###############
}

MyLasso = function(X, y, lam.seq, maxit = 50){
  # X: n-by-p design matrix without the intercept
  # y: n-by-1 response vector
  # lam.seq: sequence of lambda values
  # maxit: number of updates for each lambda

  n = length(y)
  p = dim(X)[2]
  nlam = length(lam.seq)

  ##############################
  # YOUR CODE
  # Center and scale X
  # Record the corresponding means and scales
  ##############################

  # Initialize coef vector b and residual vector r
  b = XXX
  r = XXX
  B = XXX
  # Triple nested loop
  for(m in 1:nlam){
    lam = XXX # assign lambda value
    for(step in 1:maxit){
      for(j in 1:p){
        r = r + (X[,j]*b[j])
        b[j] = One_var_lasso(r, X[, j], lam)
        r = r - X[, j] * b[j]
      }
    }
    B[m, -1] = b
  }

  ##############################
  # YOUR CODE
  # Scale back the coefficients and update the intercepts B[, 1]
  ##############################

  return(t(B))
}
```

Note: You need to write your own function `One_var_lasso` to solve the one-variable Lasso for β_j. Check hints given on the next page.

In the CD algorithm, at each iteration, we repeatedly solve a one-dimensional Lasso problem for β_j while holding the other $(p-1)$ coefficients at their current values:

$$\min_{\beta_j} \sum_{i=1}^{n} (y_i - \sum_{k \neq j} x_{ik} \hat{\beta}_k - x_{ij}\beta_j)^2 + \lambda \sum_{k \neq j} |\hat{\beta}_k| + \lambda |\beta_j|,$$

which is equivalent to solving

$$\min_{\beta_j} \sum_{i=1}^{n} (r_i - x_{ij}\beta_j)^2 + \lambda |\beta_j|. \tag{1}$$

where

$$r_i = y_i - \sum_{k \neq j} x_{ik} \hat{\beta}_k.$$

How to solve (1)? In class we have discussed how to find the minimizer of

$$f(x) = (x-a)^2 + \lambda |x|,$$

which is given by

$$x^* = \arg\min_x f(x) = \text{sign}(a)(|a| - \lambda/2)_+ = \begin{cases} a - \lambda/2, & \text{if } a > \lambda/2; \\ 0, & \text{if } |a| \leq \lambda/2; \\ a + \lambda/2, & \text{if } a < -\lambda/2. \end{cases} \tag{2}$$

We can rewrite (1) in the form of $f(x)$ and then use the solution given above.

Define two $n \times 1$ vectors: $\mathbf{r} = (r_1, \dots, r_n)^t$ with r_i being its i-element and $\mathbf{x}_j = (x_{1j}, \dots, x_{nj})^T$ with x_{ij} as its i-th element. Then

$$\begin{pmatrix} r_1 - x_{1j}\beta_j \\ r_2 - x_{2j}\beta_j \\ \cdots \\ r_n - x_{nj}\beta_j \end{pmatrix} = \begin{pmatrix} r_1 \\ r_2 \\ \cdots \\ r_n \end{pmatrix} - \begin{pmatrix} x_{1j} \\ x_{2j} \\ \cdots \\ x_{nj} \end{pmatrix} \beta_j = \mathbf{r} - \mathbf{x}_j \beta_j.$$

Then rewrite (1) as

$$\sum_{i=1}^{n} (r_i - x_{ij}\beta_j)^2 + \lambda |\beta_j| = \|\mathbf{r} - \mathbf{x}_j \beta_j\|^2 + \lambda |\beta_j|. \tag{3}$$

The first term above is like the RSS from a regression model with only one predictor (whose coefficient is β_j) without the intercept. The corresponding LS estimate is given by

$$\hat{\beta}_j = \mathbf{r}^t \mathbf{x}_j / \|\mathbf{x}_j\|^2.$$

Then we have

$$\begin{aligned}
\|\mathbf{r} - \mathbf{x}_j\beta_j\|^2 &= \|\mathbf{r} - \mathbf{x}_j\hat{\beta}_j + \mathbf{x}_j(\beta_j - \hat{\beta}_j)\|^2 \\
&= \|\mathbf{r} - \mathbf{x}_j\hat{\beta}_j\|^2 + \|\mathbf{x}_j(\beta_j - \hat{\beta}_j)\|^2 \\
&\quad +2 \times \text{inner-product-of-vectors } (\mathbf{r} - \mathbf{x}_j\hat{\beta}_j) \text{ and } \mathbf{x}_j(\beta_j - \hat{\beta}_j) \\
&= \|\mathbf{r} - \mathbf{x}_j\hat{\beta}_j\|^2 + \|\mathbf{x}_j(\beta_j - \hat{\beta}_j)\|^2, \qquad (4)
\end{aligned}$$

where the last equality is due to the fact that the inner product term is zero since vector $(\mathbf{r} - \mathbf{x}_j\hat{\beta}_j)$ is orthogonal to $\mathbf{x}_j$[1].

Note that the first term of (4) has nothing to do with β_j. So to minimize (1) or equivalently (3) with respect to β_j, we can ignore the first term and instead minimize

$$\begin{aligned}
\|\mathbf{x}_j(\beta_j - \hat{\beta}_j)\|^2 + \lambda|\beta_j| &= \|\mathbf{x}_j\|^2(\beta_j - \hat{\beta}_j)^2 + \lambda|\beta_j| \\
&= \|\mathbf{x}_j\|^2 \left((\beta_j - \hat{\beta}_j)^2 + \frac{\lambda}{\|\mathbf{x}_j\|^2}|\beta_j| \right) \\
&\propto (\beta_j - \hat{\beta}_j)^2 + \frac{\lambda}{\|\mathbf{x}_j\|^2}|\beta_j|.
\end{aligned}$$

Now we can use (2), the solution we derived for $f(x)$, with

$$a = \hat{\beta}_j = \mathbf{r}^t\mathbf{x}_j/\|\mathbf{x}_j\|^2, \quad \lambda = \lambda/\|\mathbf{x}_j\|^2.$$

[1]This is because $(\mathbf{r} - \mathbf{x}_j\hat{\beta}_j)$ represents the residual vector from a regression model with $\mathbf{x}_j$ being a column (actually the only column) of the design matrix, so it is orthogonal to $\mathbf{x}_j$.