STAT361 Laboratory for Advanced R for Data Science

Lab 8

Sidi Wu

March 16/17, 2023

Department of Statistics and Actuarial Science
Simon Fraser University

Implementing and Testing ‘LOF()’ & ‘bwd_stepwise()’

Objective

- Understanding the importance of GCV in LOF()
  - Implement/modify LOF function
  - Test the output of LOF()
- Implement backward stepwise function
  - Write the backward selection algorithm of MARS
  - Test the output of bwd_stepwise()

Test Data

- Pull Stat360 class repository and look for 'ProjectTestfiles' directory
- Copy the R data files in 'ProjectTestfiles' to your mars project in GitHub (tests/testthat/)

Importance of GCV

- To understand the importance of GCV in LOF, do the following tests.
  - Load 'testfwd_stepwise.Rdata' to your R session and run the following lines of code,

```
dat <- data.frame(y=testfwd$y,testfwd$B)
ff <- lm(y~.,dat)
```

  - Print the coefficients of the model with **coefficients(ff)** - you should see 'NA' for some of the coefficients. The NAs mean that there are collinearities in the model; i.e., some of the terms in the model are linear combinations of the others.
  - One obvious collinearity is that B_0 is an intercept, and lm() also adds an intercept when passed a formula with y~.
    - Stop R from adding an intercept with the formula y~.-1
    - Re-fit using **lm(y~.-1,dat)** and re-print the coefficients
    - Will still see evidence of collinearity (NAs)

- Including GCV criterion in LOF may overcome the problem, thus we need to modify LOF() in such a way that it returns the GCV criterion (refer to mars3.pdf).

Modify LOF()

- **Inputs**: formula, data, mars.control object (which includes 'Mmax', 'trace', and 'd')
- **Fit the linear model and obtain the residual sum of squares (RSS)**
- **Calculate number of rows and columns of the basis matrix**
  - Number of rows can be obtained from the data argument (N)
  - Number of columns of the basis matrix can be obtained by the fitted model (M) – Note: Make sure to deduct 1 from the number of coefficients
  - C(M) is the sum of the hat-values from the fitted model
  - d is the smoothing parameter

$$\frac{1}{N} \frac{\sum_{i=1}^{N} (y_i - \hat{f}_M(x_i))^2}{(1 - \tilde{C}(M)/N)^2} = RSS \times \frac{N}{(N - \tilde{C}(M))^2}$$

$$\tilde{C}(M) = C(M) + dM$$

- **Output**: Value of the GCV criterion

Test LOF()

- Load 'testLOF.Rdata' to your R session and run the following,

```
lof <- LOF(y~.-1,dat,testmc)
all.equal(lof,testLOF)
```

- If the output is 'TRUE', it suggests that implementation of LOF() is correct

Implement bwd_stepwise() - mars6.pdf

- Inputs:
  - Output of fwd_stepwise()
  - mars.control object

Algorithm 3 (MARS—backwards stepwise)

$J^* = \{1, 2, \ldots, M_{\max}\}$; $K^* \leftarrow J^*$
lof* $\leftarrow \min_{\{a_j | j \in J^*\}} \mathrm{LOF}(\sum_{j \in J^*} a_j B_j(\mathbf{x}))$
For $M = M_{\max}$ to 2 do: $b \leftarrow \infty$; $L \leftarrow K^*$
  For $m = 2$ to M do: $K \leftarrow L - \{m\}$
    lof $\leftarrow \min_{\{a_k | k \in K\}} \mathrm{LOF}(\sum_{k \in K} a_k B_k(\mathbf{x}))$
    if lof $< b$, then $b \leftarrow$ lof; $K^* \leftarrow K$ end if
    if lof $<$ lof*, then lof* $\leftarrow$ lof; $J^* \leftarrow K$ end if
  end for
end for
end algorithm

- Some hints...
  - Initialize J^*: need Mmax – you may use fwd object obtain Mmax
  - Initialize K^*
  - Create a data frame with response variable and basis matrix
  - Compute LOF, which will be your 'lof*'
  - Implement M and m loop
  - Calculate LOF within LOF for subset of data (consider using setdiff function for subsetting)
  - Update LOF accordingly
  - Return y, B and Bfuncs accordingly as a list at the end

Test bwd_stepwise()

- Load ‘testbwd_stepwise.RData’ to your R session and run the following,

```
bwd <- bwd_stepwise(testfwd,testmc)
all.equal(bwd,testbwd)
```

- If the output is ‘TRUE’, it suggests that implementation of backward stepwise algorithm is correct