Lab 2, week 3

Pulindu Ratnasekera

Add to shell implementation of MARS algorithm

In this lab we will investigate the possibility of having the `fwd_stepwise()` function and `bwd_stepwise()` functions from lab 1 use `lm()` and `step()`.

1. Simulate some test data as follows:

```
set.seed(1)
x1 <- 1:100
x2 <- rnorm(100); x3 <- rnorm(100)
y <- x1+rnorm(100)
data <- data.frame(y,x1,x2,x3)
```

2. Write a couple of lines of R code to (i) fit a model with `lm()` with response variable y and predictors x1, x2 and x3 and (ii) use `step()` on the fitted model to do stepwise model reductions. In your call to lm, use the `.` in your formula and pass it the data frame constructed in the code chunk.

3. Now try to do the same thing using your `mars()` function from lab 1 with the following modifications. Recall that your `mars()` function took an R formula, `data` and `control` as inputs and called functions `fwd_stepwise()` and `bwd_stepwise()`.

    a. Pass the formula and data arguments to `fwd_stepwise()`. Have `fwd_stepwise()` call the `lm()` function on the formula and data and return the resulting `lm` object.
    b. As in lab 1, your `bwd_stepwise()` function takes the output of `fwd_stepwise()`. Now have `bwd_stepwise()` call the `step()` function on its input, and return the output of `step()`.
    c. As in lab 1, your `mars()` function returns the output of `bwd_stepwise()`.
    d. Test your implementation with the call `mars(y~.,data=data)`.

4. Change the name of the data frame `data` in your workspace to `testdata` are re-run mars. What happens? Use `traceback()` to identify the function call that threw an error. What does this tell you about where this function found the data in part 3?