Statistics 360: Advanced R for Data Science

MARS, part VI

Brad McNeney

Topics

- Recap of forward stepwise MARS algorithm (Algorithm 2)
- Pruning basis functions in the backward stepwise algorithm (Algorithm 3)

Recap of forward algorithm

- The MARS forward stepwise algorithm (Algorithm 2) builds a linear prediction equation that is linear in basis functions $B_1(x), \ldots, B_{M_{max}}(x)$.
  - Coefficients of the basis functions are by least squares.
  - Include an intercept term $B_0(x) = 1$
  - Basis functions are products of hinge functions:
- `fwd_stepwise()` function should return a list with elements
  1. the matrix `B`$= [B_0, B_1(x), \ldots, B_{M_{max}}(x)]$
  2. the list `Bfuncs`, whose $m + 1$st element is a data frame of variables, signs and knots that describes the mth basis function
  3. the response variable `y`
- Note 1 **Change** from last week: initialize `Bfuncs` to be of length `Mmax+1` and have its indices match those of `B`;

```
Bfuncs <- vector(mode="list",length=Mmax+1)
```

- Note 2 `y` is not an output of the `fwd_stepwise()` algorithm, but it will be needed in the `bwd_stepwise()` function, so it makes sense to bundle it with `B` and `Bfuncs`

Generalized cross-validation (GCV)

- The LOF measure $LOF(\hat{f}_M) = GCV(M)$ in Friedman's equations (30) and (32) is

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

  where M is the number of non-constant basis functions, $\tilde{C}(M) = C(M) + dM$, $C(M)$ is the sum of the hat-values from the fitted model and d is a smoothing parameter.
  - $C(M) = M + 1$ if there are no linear dependencies between basis functions, but summing the hat-values is safest.
  - Friedman suggests that $d = 3$ works well.
  - Denominator decreases, so GCV increases as M increases.

LOF suggestions

- You should write a `LOF()` function that takes a formula, data frame and control list as input.
  - Use `lm()` to fit the model.
  - Determine the number of non-constant basis functions from the number of non-intercept coefficients in the output of `lm()`
  - Calculate $\tilde{C}$ from Friedman's equation (32)

Backwards stepwise algorithm suggestions

- Use the notation from Algorithm 3 in your R code, but with `Mmax` defined as the number of basis functions other than the intercept.
  - Recall: Friedman's `Mmax` is the number *including* the intercept.
- Initialize `Jstar` to the indices of all non-intercept basis functions from the forward algorithm: `Jstar <- 2:(Mmax+1)`
- Initialize `Kstar <- Jstar`
  - As the algorithm loops over the number of basis functions (see below), `Kstar` contains the indices of the best set of basis functions from the previous iteration.
- Initialize best LOF so far, `lofstar`, to be the LOF for the model fit with all basis functions from the forward algorithm

Backwards stepwise algorithm suggestions, cont.

- Outer loop over model size M in Mmax+1 to 2:
  - Make a working copy of `Kstar`: `L <- Kstar` and initialize the best LOF for the inner loop to `b <- Inf`. Goal of inner loop is to minimize LOF.
  - Inner loop over m in L considers removing mth basis function from the working copy `L`, `K <- setdiff(L,m)`, fit model with basis functions in K and calculate LOF.
    - If LOF best seen so far in this iteration of the **inner** loop, update `Kstar`
    - If LOF is also best seen in all iterations so far of the **outer** loop, update `Jstar`
- Algorithm terminates with indices `Jstar`.
  - Add the index of the intercept to `Jstar`: `Jstar <- c(1,Jstar)`
  - Return `B[,Jstar]` and `Bfuncs[Jstar]`.

Back in main mars function

- Use lm to fit the model with the `B` returned by backward stepwise.
  - `B` already includes an intercept, so use the formula `y~.-1`.
  - This fit is a list of class "lm". Combine it with other objects from mars.
- Think about what you need to return:
  - function call?
  - Input formula and data?,
  - y and x extracted from formula and data?
  - Definitely B and Bfuncs
- Class of output should be "mars" with parent class "lm" so that we inherit all of the methods for lm objects.