StackExchange ▼ sign up log in tour help ▼ stack overflow careers

Is there a better alternative than string manipulation to programmatically build formulas?

Everyone else's functions seem to take formula objects and then do dark magic to them somewhere deep inside and I'm jealous.

I'm writing a function that fits multiple models. Parts of the formulas for these models remain the same and part change from one model to the next. The clumsy way would be to have the user input the formula parts as character strings, do some character manipulation on them, and then use `as.formula`.

But before I go that route, I just want to make sure that I'm not overlooking some cleaner way of doing it that would allow the function to accept formulas in the standard R format (e.g. extracted from other formula-using objects).

I want something like...

```
> LHS <- y~1; RHS <- ~a+b; c(LHS,RHS);
y ~ a + b
> RHS2 <- ~c;
> c(LHS, RHS, RHS2);
y ~ a + b + c
```

or...

```
> LHS + RHS;
y ~ a + b
> LHS + RHS + RHS2;
y ~ a + b + c
```

...but unfortunately neither syntax works. Does anybody know if there is something that does? Thanks.

r linear-regression

edited Oct 26 '12 at 19:48

Ben Bolker
73.2k ● 6 ● 99 ● 177

asked Oct 19 '12 at 5:17

f1r3br4nd
1,211 ● 1 ● 10 ● 30

Even though I ended up realizing that I didn't need quite that level of generality and instead made better use of the `update` function, mnel's answer below is a good and useful one, and may have done what I originally was attempting. In general, though, I upvote good answers but don't accept them until I actually try them and can vouch for them. In many cases I found better answers on my own and should really submit self-answers when I have time. Am I too stringent in my criteria for accepting answers? – f1r3br4nd Mar 11 '13 at 15:54

1 Answer

`reformulate` will do what you want.

```
reformulate(termlabels = c('x','z'), response = 'y')
## y ~ x + z
```

Or without an intercept

```
reformulate(termlabels = c('x','z'), response = 'y', intercept = FALSE)
## y ~ x + z - 1
```

Note that you cannot construct formulae with multiple `reponses` such as `x+y ~z+b`

```
reformulate(termlabels = c('x','y'), response = c('z','b'))
z ~ x + y
```

To extract the terms from an existing `formula` (given your example)

```
attr(terms(RHS), 'term.labels')
## [1] "a" "b"
```

To get the response is slightly different, a simple approach (for a single variable response).

```
as.character(LHS)[2]
## [1] 'y'


combine_formula <- function(LHS, RHS){
  .terms <- lapply(RHS, terms)
  new_terms <- unique(unlist(lapply(.terms, attr, which = 'term.labels')))
  response <- as.character(LHS)[2]

  reformulate(new_terms, response)


}


combine_formula(LHS, list(RHS, RHS2))

## y ~ a + b + c
## <environment: 0x577fb908>
```

I think it would be more sensible to specify the response as a character vector, something like

```
combine_formula2 <- function(response, RHS, intercept = TRUE){
  .terms <- lapply(RHS, terms)
  new_terms <- unique(unlist(lapply(.terms, attr, which = 'term.labels')))
  response <- as.character(LHS)[2]

  reformulate(new_terms, response, intercept)


}
combine_formula2('y', list(RHS, RHS2))
```

you could also define a `+` operator to work with formulae (update setting an new method for formula objects)

```
`+.formula` <- function(e1,e2){
  .terms <- lapply(c(e1,e2), terms)
  reformulate(unique(unlist(lapply(.terms, attr, which = 'term.labels'))))
}

RHS + RHS2
## ~a + b + c
```

You can also use `update.formula` using `.` judiciously

```
 update(~a+b, y ~ .)
 ##  y~a+b
```

edited Oct 27 '12 at 4:19

answered Oct 19 '12 at 5:22

mnel

59.2k ● 7 ● 117 ● 142

5 +1 for your great summary. But let's admit it: it just shows that using the approach via strings is the way to go if you want easy to read code. I doubt that, summed over lifetime, the gain in speed buys you the time for a coffee. – Dieter Menne Oct 19 '12 at 6:19

12 @DieterMenne The gain in speed is immaterial - the gain in safety is important. The first time someone tries to use your code with a non-syntactic variable name (i.e. "a b") it will break with a strange error that will take you hours and hours to find. – hadley Oct 19 '12 at 12:36

Is there any other way to use `.` than with `update.formula` ? I ended up using a `setdiff()` statement where I would have liked to use `~.-var1-var2` in `reformulate()` , and the help doesn't mention that use case. – Trevor Alexander Nov 29 '14 at 4:29