General Linear Statistical Models - Part III

Statistics 135

Autumn 2005

Interaction Models

Lets examine two models involving `Weight` and `Domestic` in the `cars93` dataset.

```
weight.domestic.lm <- lm(HighFuel ~ Weight + Domestic,
                                          data=cars93)

weight.domestic.int.lm <- lm(HighFuel ~ Weight * Domestic,
                                          data=cars93)
```

Remember that the second model is a shorthand for

```
HighFuel ~ Weight + Domestic + Weight : Domestic
```

```
> summary(weight.domestic.lm)

Residuals:
      Min        1Q    Median        3Q       Max
-0.781506 -0.244967  0.002068  0.180682  0.922104

Coefficients:
                   Estimate Std. Error t value Pr(>|t|)
(Intercept)       9.923e-01  1.853e-01   5.355  6.5e-07 ***
Weight            8.354e-04  6.065e-05  13.774  < 2e-16 ***
DomesticDomestic -3.449e-02  7.120e-02  -0.484    0.629
```

```
> summary(weight.domestic.int.lm)

Residuals:
     Min       1Q   Median       3Q      Max
-0.78647 -0.21346 -0.03952  0.17163  0.99145

Coefficients:
                          Estimate Std. Error t value Pr(>|t|)
(Intercept)              6.264e-01  2.504e-01   2.501   0.0142 *
Weight                   9.597e-04  8.347e-05  11.498   <2e-16 ***
DomesticDomestic         7.421e-01  3.721e-01   1.994   0.0492 *
Weight:DomesticDomestic -2.529e-04  1.190e-04  -2.125   0.0364 *
```

Both models fit give regression lines for `HighFuel` vs `Weight`. The first is the additive model, which is of the form

$$y_i = \beta_0 + \beta_1 w_i + \beta_2 d_i + \epsilon_i$$

The second model is of the form

$$y_i = \beta_0 + \beta_1 w_i + \beta_2 d_i + \beta_3 w_i d_i + \epsilon_i$$

where d_i is 1 for domestic cars and 0 for foreign cars

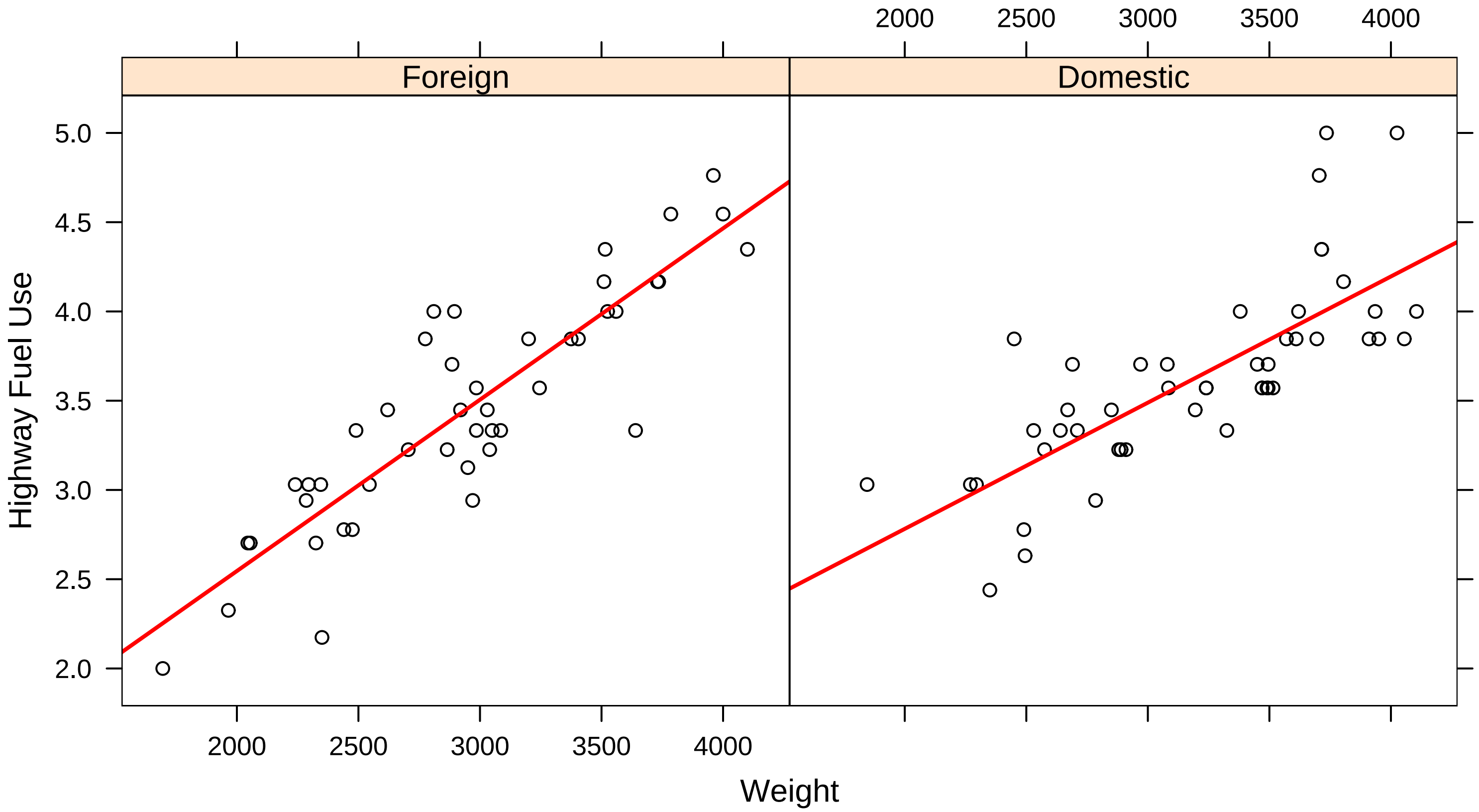
These can be written as

$$y_i = \begin{cases} \beta_0 + \beta_1 w_i + \epsilon_i & \text{Foreign car} \\ \underbrace{(\beta_0 + \beta_2)}_{\beta_0^*} + \beta_1 w_i + \epsilon_i & \text{Domestic car} \end{cases}$$

and

$$y_i = \begin{cases} \beta_0 + \beta_1 w_i + \epsilon_i & \text{Foreign car} \\ \underbrace{(\beta_0 + \beta_2)}_{\beta_0^*} + \underbrace{(\beta_1 + \beta_3)}_{\beta_1^*} w_i + \epsilon_i & \text{Domestic car} \end{cases}$$

The second model is an example of an interaction. In this case, the effect of weight depends on whether the car is domestically made or not. It is fitting the equivalent to what is displayed in the figure


Foreign
Domestic
Highway Fuel Use
Weight
5.0
4.5
4.0
3.5
3.0
2.5
2.0
2000
2500
3000
3500
4000

As mentioned last class to indicate interactions in **S**, you use either `*` or `:`. Usually you want to use `*`, as its shorter and it also leads to good statistical practice. Consider the models (where x and y are quantitative)

```
z ~ x*y                                    z ~ x:y
```

The first model is equivalent to `z ~ x + y + x:y`, a standard model of interest. The second model would fit

$$z_i = \beta_0 + \beta_1 x_i y_i$$

Usually you don't want to fit a model with the lower order terms missing. For example, with `HighFuel ~ Weight:Domestic`, **S** would be fitting

$$y_i = \begin{cases} \beta_0 + \beta_1 w_i + \epsilon_i & \text{Foreign car} \\ \beta_0 + \epsilon_i & \text{Domestic car} \end{cases}$$

Not a particularly reasonable model.

Note: this idea is a good rule of thumb. There may be situations where you might want to include higher order interactions but drop out lower order ones.

One possibility of this is

$$y_i = \beta_1 x_i + \beta_2 x_i d_i + \epsilon_i$$

Here the main effect for d_i is missing. This model is describing regression through the origin for x_i, with different slopes for different levels of d_i.

If terms get repeated in a model description, the repeats get dropped, so don't worry about them. So for example

```
y ~ A*B + B*C
```

is a fine way of describing the model

```
y ~ A + B + C + A:B + B:C
```

Polynomial Models

It is easy to fit polynomial models in **S**. However there is a slight trick to it (particularly in **R**). One approach you might consider is a call like

```
> weight2a.lm <- lm(HighFuel ~ Weight + Weight^2, data=cars93)
> summary(weight2a.lm)

Residuals:
      Min         1Q    Median        3Q       Max
-0.768280 -0.239028  0.005072  0.199289  0.909606

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 9.940e-01  1.845e-01   5.387 5.56e-07 ***
Weight      8.290e-04  5.898e-05  14.057  < 2e-16 ***
```

This only fits the linear term (the `Weight^2` term gets dropped). (Actually this only happens in **R**. It will do what you expect in **S-Plus**.)

Instead you need to use the `I()` operator as in the following

```
> weight2.lm <- lm(HighFuel ~ Weight + I(Weight^2), data=cars93)
> summary(weight2.lm)

Call:
lm(formula = HighFuel ~ Weight + I(Weight^2), data = cars93)

Residuals:
     Min       1Q   Median       3Q      Max
-0.76605 -0.23896  0.01345  0.19332  0.91241

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  8.278e-01  8.696e-01   0.952    0.344
Weight       9.430e-04  5.855e-04   1.611    0.111
I(Weight^2) -1.879e-08  9.607e-08  -0.196    0.845
```

```
Residual standard error: 0.3355 on 90 degrees of freedom
Multiple R-Squared: 0.6848,     Adjusted R-squared: 0.6778
F-statistic: 97.78 on 2 and 90 DF,  p-value: < 2.2e-16

> anova(weight2.lm)
Analysis of Variance Table

Response: HighFuel
            Df  Sum Sq Mean Sq  F value Pr(>F)
Weight       1 22.0027 22.0027 195.5175 <2e-16 ***
I(Weight^2)  1  0.0043  0.0043   0.0383 0.8454
Residuals   90 10.1282  0.1125
```

This gives you what you want.

Removing Terms from Models

It is also possible to remove terms from models. For example

```
y ~ A + B + C + A:B + A:B + B:C
```

could have been written

```
y ~ A*B*C - A:B:C
```

so it can be used as a shorthand to write more complicated models.

Another situation where it is useful to to compare two models. Consider in the crab example where we want to compare the models

```
RW ~ sex * sp
```

and

```
RW ~ sex + sp
```

One way of doing this is by

```
crab.int.lm <- lm(RW ~ sex * sp, data=crabs)
crab.add.lm <- update(crab.int.lm, . ~ . - sex:sp)
```

This could also be done by

```
crab.add2.lm <- lm(RW ~ sex + sp, data=crabs)
crab.int2.lm <- update(crab.add2.lm, . ~ . + sex:sp)
```

To see whether the interaction model gives a better fit, we can look at the command

```
> anova(crab.add.lm, crab.int.lm)
Analysis of Variance Table

Model 1: RW ~ sex + sp
Model 2: RW ~ sex * sp
  Res.Df    RSS  Df Sum of Sq      F    Pr(>F)
1    197 1074.4
2    196 1016.4   1      58.0 11.184 0.0009884 ***
```

Note that this isn't needed for this example as

```
> anova(crab.int.lm)
Analysis of Variance Table

Response: RW
           Df  Sum Sq Mean Sq F value    Pr(>F)
sex         1  112.05  112.05  21.608 6.133e-06 ***
sp          1  131.38  131.38  25.336 1.087e-06 ***
sex:sp      1   58.00   58.00  11.184 0.0009884 ***
Residuals 196 1016.36    5.19
```

gives the same information.

Another useful situation where removing a term may be useful is to get rid of the intercept. For example to fit a regression through the origin you can do

```
> weight.orig.lm <- lm(HighFuel ~ Weight - 1, data=cars93)
> summary(weight.orig.lm)

Residuals:
      Min        1Q    Median        3Q       Max
-0.820327 -0.227628 -0.009304  0.320788  1.050421

Coefficients:
        Estimate Std. Error t value Pr(>|t|)
Weight 1.141e-03  1.263e-05   90.33   <2e-16 ***

Residual standard error: 0.3811 on 92 degrees of freedom
Multiple R-Squared: 0.9888,     Adjusted R-squared: 0.9887
F-statistic:  8159 on 1 and 92 DF,  p-value: < 2.2e-16
```

There is some evidence for this model. First the physics suggests it. In addition

```
> weight.lm <- lm(HighFuel ~ Weight, data=cars93)
> summary(weight.lm)

Call:
lm(formula = HighFuel ~ Weight, data = cars93)

Residuals:
      Min        1Q    Median        3Q       Max
-0.768280 -0.239028  0.005072  0.199289  0.909606

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 9.940e-01  1.845e-01   5.387 5.56e-07 ***
Weight      8.290e-04  5.898e-05  14.057  < 2e-16 ***


Residual standard error: 0.3337 on 91 degrees of freedom
Multiple R-Squared: 0.6847,     Adjusted R-squared: 0.6812
F-statistic: 197.6 on 1 and 91 DF,  p-value: < 2.2e-16
```

```
> anova(weight.orig.lm, weight.lm)
Analysis of Variance Table

Model 1: HighFuel ~ Weight - 1
Model 2: HighFuel ~ Weight
  Res.Df     RSS Df Sum of Sq      F    Pr(>F)
1     92 13.3643
2     91 10.1325  1    3.2318 29.025 5.564e-07 ***
```

Removing the intercept is useful in some ANOVA models as it gives another parameterization. For example,

```
> type.lm

Call:
lm(formula = HighFuel ~ Type, data = cars93)

Coefficients:
(Intercept)    TypeLarge  TypeMidsize    TypeSmall   TypeSporty      TypeVan
     3.3768       0.3725       0.3965      -0.4979       0.1475       1.2098
```

```
> type.noint.lm

Call:
lm(formula = HighFuel ~ Type - 1, data = cars93)

Coefficients:
TypeCompact    TypeLarge  TypeMidsize    TypeSmall   TypeSporty      TypeVan
      3.377        3.749        3.773        2.879        3.524        4.587
```

In the second approach, the parameter estimate are the sample means for each Type.

Another example is

```
> weight.domestic.int.lm

Call: lm(formula = HighFuel ~ Weight * Domestic, data = cars93)

Coefficients:
            (Intercept)                   Weight         DomesticDomestic
              0.6263581                0.0009597                0.7420544
Weight:DomesticDomestic
             -0.0002529

> weight.domestic.int2.lm

Call: lm(formula = HighFuel ~ Domestic/Weight - 1, data = cars93)

Coefficients:
        DomesticForeign         DomesticDomestic   DomesticForeign:Weight
              0.6263581                1.3684125                0.0009597
DomesticDomestic:Weight
              0.0007069
```

The first gives the difference in the intercepts and slopes for domestic cars from foreign cars where the second gives the slopes and intercepts for both types.

The / is another way of describing interactions. The for is `a / x`, where `a` is a factor and `x` could be numeric, a factor, or a combination of things. This model says fit the model described by `x` for each level of `a`. The specification `a/x - 1` is equivalent to

```
a + a:x - 1
```

in terms of parameterization.

Prediction

It is easy to make predictions for new or hypothesized observations with the `predict` function. The form of the function is `predict(fit, newdata)`, where `fit` is the result of the `lm` command and `newdata` is a dataframe including all of the variables used in the fitting model

```
> newdata
  Weight Domestic
1   2000  Foreign
2   3000 Domestic
3   4000  Foreign
4   2000 Domestic
5   3000  Foreign
6   4000 Domestic
> predict(weight.domestic.int.lm,newdata)
       1        2        3        4        5        6
2.545835 3.489005 4.465312 2.782141 3.505573 4.195869
```

And to exhibit that different parametrizations give the same fitted values

```
> predict(weight.domestic.int.lm,newdata)
       1        2        3        4        5        6
2.545835 3.489005 4.465312 2.782141 3.505573 4.195869

> predict(weight.domestic.int2.lm,newdata)
       1        2        3        4        5        6
2.545835 3.489005 4.465312 2.782141 3.505573 4.195869
```