Methods of Data Analysis

ST 412/512

2025-01-15

Lab 2: Multiple Linear Regression in R

Goals for Today's Lab

1. Understand how factors work in R and how to specify indicator variables.
2. Learn to define and interpret multiple regression models.
3. Practice adjusting factor levels and adding interaction terms.

Introduction to Factors and Indicator Variables

In many analyses, we need to work with categorical data. R uses **factors** to handle categorical variables, and creating **indicator variables** (also known as dummy variables) is often necessary to incorporate these into regression models. Today's lab will:

- Demonstrate converting numeric or character variables to factors.
- Show how to create and use indicator variables.
- Guide you through fitting multiple linear regression models using these factors and interactions.

Guidance Understanding factors is crucial because regression models treat factors differently than numeric variables. Incorrect handling of factors can lead to misinterpretation of model coefficients. We will learn to specify baseline levels and interpret coefficients accordingly.

Working with the `case0901` Dataset

We'll work with the `case0901` dataset from the `Sleuth3` package. This dataset involves flower counts under different light intensities and start times (morning/evening).

Inspecting the Data

Start by loading the data and exploring its structure:

```r
library(Sleuth3)
str(case0901)
```

```
'data.frame':   24 obs. of  3 variables:
 $ Flowers  : num  62.3 77.4 55.3 54.2 49.6 61.9 39.4 45.7 31.3 44.9 ...
 $ Time     : int  1 1 1 1 1 1 1 1 1 1 ...
 $ Intensity: int  150 150 300 300 450 450 600 600 750 750 ...
```

Guidance

The `str()` function provides a concise summary of the dataset, including variable types and the first few observations. Notice the types of variables. For example, `Time` might be encoded as integers but represents categorical information (morning vs. evening), which we will handle as a factor.

Creating Indicator Variables

The `Time` variable encodes two different scenarios:

- 1 for one condition (e.g., "Late start")
- 2 for another condition (e.g., "Early start")

We'll create an indicator variable `Day24` that flags the "Early start" scenario.

```r
case0901$Day24 <- ifelse(case0901$Time == 2, 1, 0)
head(case0901$Day24)
```

```
[1] 0 0 0 0 0 0
```

Guidance

The `ifelse()` function is useful for recoding variables. Here, it checks if `Time` equals 2, assigns `1` if true, otherwise `0`. This creates a binary variable that can be directly used in regression models.

Fitting a Model with an Indicator Variable

Now, fit a linear model predicting the number of flowers based on light intensity and `Day24`.

```r
summary(lm(Flowers ~ Intensity + Day24, data = case0901))
```

```

Call:
lm(formula = Flowers ~ Intensity + Day24, data = case0901)

Residuals:
   Min     1Q Median     3Q    Max 
-9.652 -4.139 -1.558  5.632 12.165 

Coefficients:
             Estimate Std. Error t value Pr(>|t|)    
(Intercept) 71.305833   3.273772  21.781 6.77e-16 ***
Intensity   -0.040471   0.005132  -7.886 1.04e-07 ***
Day24       12.158333   2.629557   4.624 0.000146 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 6.441 on 21 degrees of freedom
Multiple R-squared:  0.7992,	Adjusted R-squared:   0.78 
F-statistic: 41.78 on 2 and 21 DF,  p-value: 4.786e-08
```

Guiding Questions:

- How do the coefficients for `Intensity` and `Day24` relate to the average number of flowers?
- What does the coefficient for `Day24` tell us about the difference between early and late start times?

Guidance

The coefficient for `Day24` represents the average change in flower count when moving from a late start (`Day24 = 0`) to an early start (`Day24 = 1`), holding intensity constant. This value quantifies the effect of an early start on flower count. If the coefficient is significant, it suggests that start time has a meaningful impact on the number of flowers, beyond the effect of intensity.

Using Factors Directly

Instead of manually creating indicator variables, you can tell R to treat a variable as a categorical factor directly.

```r
summary(lm(Flowers ~ Intensity + factor(Time), data = case0901))
```

```

Call:
lm(formula = Flowers ~ Intensity + factor(Time), data = case0901)

Residuals:
   Min     1Q Median     3Q    Max 
-9.652 -4.139 -1.558  5.632 12.165 

Coefficients:
               Estimate Std. Error t value Pr(>|t|)    
(Intercept)   71.305833   3.273772  21.781 6.77e-16 ***
Intensity     -0.040471   0.005132  -7.886 1.04e-07 ***
factor(Time)2 12.158333   2.629557   4.624 0.000146 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 6.441 on 21 degrees of freedom
Multiple R-squared:  0.7992,	Adjusted R-squared:   0.78 
F-statistic: 41.78 on 2 and 21 DF,  p-value: 4.786e-08
```

Guidance

By using `factor(Time)`, R automatically creates dummy variables for each level of `Time`. The output will include coefficients comparing each level to a baseline (reference) level, which by default is the first level.

Creating and Reordering Factors

Sometimes, you need to reorder factor levels to change the baseline category. For example, we'll convert `Intensity` to a factor and reorder its levels.

```
# Convert Intensity to factor
case0901$intensity_f <- factor(case0901$Intensity)
str(case0901$intensity_f)
```

```
 Factor w/ 6 levels "150","300","450",..: 1 1 2 2 3 3 4 4 5 5 ...
```

```
levels(case0901$intensity_f)
```

```
[1] "150" "300" "450" "600" "750" "900"
```

```
# Reordering factor levels
case0901$intensity_f2 <- factor(case0901$Intensity,
  levels = c("750", "300", "450", "150", "600", "900"))
levels(case0901$intensity_f2)
```

```
[1] "750" "300" "450" "150" "600" "900"
```

Guidance

Reordering levels changes the baseline when factors are used in models. This can be important when comparing groups or when a particular level should serve as the reference.

You can also change the reference level using `relevel()`:

```
case0901$intensity_f3 <- relevel(case0901$intensity_f, ref = "300")
levels(case0901$intensity_f3)
```

```
[1] "300" "150" "450" "600" "750" "900"
```

Guiding Questions:

- Why might you want to change the reference level of a factor?
- How do the model coefficients change when the reference level changes?

Guidance

Changing the reference level changes the interpretation of coefficients. With a different baseline, each coefficient shows the difference between that level and the new baseline. This may simplify interpretation or focus on specific comparisons of interest in your analysis.

Fitting Multiple Linear Regression Models

Let's fit several regression models using the different ways to handle factors and see how the results change.

```r
# Basic model using numeric variables
summary(lm(Flowers ~ Intensity + Time, data = case0901))
```

```

Call:
lm(formula = Flowers ~ Intensity + Time, data = case0901)

Residuals:
   Min     1Q Median     3Q    Max 
-9.652 -4.139 -1.558  5.632 12.165 

Coefficients:
             Estimate Std. Error t value Pr(>|t|)    
(Intercept) 59.147500   4.954465  11.938 8.01e-11 ***
Intensity   -0.040471   0.005132  -7.886 1.04e-07 ***
Time        12.158333   2.629557   4.624 0.000146 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 6.441 on 21 degrees of freedom
Multiple R-squared:  0.7992,	Adjusted R-squared:   0.78 
F-statistic: 41.78 on 2 and 21 DF,  p-value: 4.786e-08
```

```r
# Model with factor conversion on the fly
fit_1 <- lm(Flowers ~ factor(Intensity) + Time, data = case0901)
summary(fit_1)
```

```
Call:
lm(formula = Flowers ~ factor(Intensity) + Time, data = case0901)

Residuals:
   Min     1Q Median     3Q    Max 
-8.979 -4.308 -1.342  5.204 10.204 

Coefficients:
                     Estimate Std. Error t value Pr(>|t|)    
(Intercept)            55.038      5.312  10.361 9.18e-09 ***
factor(Intensity)300   -9.125      4.751  -1.921 0.071715 .  
factor(Intensity)450  -13.375      4.751  -2.815 0.011919 *  
factor(Intensity)600  -23.225      4.751  -4.888 0.000138 ***
factor(Intensity)750  -27.750      4.751  -5.841 1.97e-05 ***
factor(Intensity)900  -29.350      4.751  -6.178 1.01e-05 ***
Time                   12.158      2.743   4.432 0.000365 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 6.719 on 17 degrees of freedom
Multiple R-squared:  0.8231,	Adjusted R-squared:  0.7606 
F-statistic: 13.18 on 6 and 17 DF,  p-value: 1.427e-05
```

Guidance

By converting `Intensity` to a factor inside the `lm()` function, R treats each level of `Intensity` separately rather than assuming a linear relationship. This model will have more parameters and can capture non-linear effects of intensity on flowers.

Changing the Baseline for a Factor

```r
case0901$Intensity_new <- relevel(factor(case0901$Intensity), ref = "600")
fit_3 <- lm(Flowers ~ Intensity_new + Time, data = case0901)
summary(fit_3)
```

```
Call:
lm(formula = Flowers ~ Intensity_new + Time, data = case0901)

Residuals:
```

```
   Min     1Q Median     3Q    Max 
-8.979 -4.308 -1.342  5.204 10.204 

Coefficients:
                 Estimate Std. Error t value Pr(>|t|)    
(Intercept)        31.813      5.312   5.989 1.47e-05 ***
Intensity_new150   23.225      4.751   4.888 0.000138 ***
Intensity_new300   14.100      4.751   2.968 0.008627 ** 
Intensity_new450    9.850      4.751   2.073 0.053665 .  
Intensity_new750   -4.525      4.751  -0.952 0.354232    
Intensity_new900   -6.125      4.751  -1.289 0.214601    
Time               12.158      2.743   4.432 0.000365 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 6.719 on 17 degrees of freedom
Multiple R-squared:  0.8231,    Adjusted R-squared:  0.7606 
F-statistic: 13.18 on 6 and 17 DF,  p-value: 1.427e-05
```

Guidance

Using `relevel()` as shown above, we change the reference category to "600". This shifts the baseline and alters the interpretation of the coefficients corresponding to `Intensity_new`. Now, coefficients represent differences from the "600" intensity level.

Adding Interaction Terms

Interactions help model situations where the effect of one predictor depends on the level of another. For instance, the effect of light intensity on flower count might differ between start times.

```r
fit_int <- lm(Flowers ~ Intensity + Time + Time:Intensity, data = case0901)
summary(fit_int)
```

```
Call:
lm(formula = Flowers ~ Intensity + Time + Time:Intensity, data = case0901)

Residuals:
```

```
   Min     1Q Median     3Q    Max
-9.516 -4.276 -1.422  5.473 11.938

Coefficients:
               Estimate Std. Error t value Pr(>|t|)
(Intercept)    60.10000    9.71192   6.188  4.8e-06 ***
Intensity      -0.04229    0.01663  -2.543   0.0193 *
Time           11.52333    6.14236   1.876   0.0753 .
Intensity:Time  0.00121    0.01052   0.115   0.9096
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 6.598 on 20 degrees of freedom
Multiple R-squared:  0.7993,	Adjusted R-squared:  0.7692
F-statistic: 26.55 on 3 and 20 DF,  p-value: 3.549e-07
```

Guiding Questions: - What does the interaction term `Time:Intensity` represent in this context? - How would you interpret a significant interaction coefficient?

Guidance

The interaction term `Time:Intensity` allows the effect of intensity on flower count to change depending on the start time. A significant interaction coefficient indicates that the slope relating intensity to flower count is different for early vs. late start times. This suggests that the relationship between intensity and flowers is not consistent across times.

Working with Quadratic Terms

Sometimes a relationship between variables is non-linear. We can add squared terms to the model to capture curvature.

Example with Corn Yield Data

We'll use the `ex0915` dataset, which relates rainfall to corn yield.

```r
head(ex0915)
```

```
  Year Yield Rainfall
1 1890  24.5      9.6
2 1891  33.7     12.9
3 1892  27.9      9.9
4 1893  27.5      8.7
5 1894  21.7      6.8
6 1895  31.9     12.5
```

Add a squared term for rainfall and fit a quadratic model:

```r
ex0915$rainfall_sq <- ex0915$Rainfall^2
summary(lm(Yield ~ Rainfall + rainfall_sq, data = ex0915))
```

```

Call:
lm(formula = Yield ~ Rainfall + rainfall_sq, data = ex0915)

Residuals:
    Min      1Q  Median      3Q     Max 
-8.4642 -2.3236 -0.1265  3.5151  7.1597 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)   
(Intercept) -5.01467   11.44158  -0.438  0.66387   
Rainfall     6.00428    2.03895   2.945  0.00571 **
rainfall_sq -0.22936    0.08864  -2.588  0.01397 * 
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 3.763 on 35 degrees of freedom
Multiple R-squared:  0.2967,	Adjusted R-squared:  0.2565 
F-statistic: 7.382 on 2 and 35 DF,  p-value: 0.002115
```

Guiding Questions:

- Why include a squared term in the model?
- How do you interpret the coefficients on both `Rainfall` and `rainfall_sq`?

Guidance

Including a squared term allows the model to capture a curved relationship between rainfall and yield, rather than assuming a straight-line relationship. The coefficient on `Rainfall` indicates the linear effect, while the coefficient on `rainfall_sq` reveals how that effect changes as rainfall increases, indicating curvature such as diminishing returns or acceleration.

Exercise: Applying These Concepts

Using the `ex0923` dataset, practice fitting multiple regression models with factors and interactions. Start by exploring the data:

```r
head(ex0923)
```

```
  Subject Gender   AFQT Educ Income2005
1       2 female  6.841   12       5500
2       6   male 99.393   16      65000
3       7   male 47.412   12      19000
4       8 female 44.022   14      36000
5       9   male 59.683   14      65000
6      13   male 72.313   16       8000
```

Now, fit various models:

```r
lm(log(Income2005) ~ Educ + Gender, data = ex0923)
```

```

Call:
lm(formula = log(Income2005) ~ Educ + Gender, data = ex0923)

Coefficients:
(Intercept)         Educ   Gendermale
     8.5002       0.1161       0.6427
```

```r
lm(log(Income2005) ~ Educ + relevel(Gender, ref = "male"), data = ex0923)
```

```

Call:
lm(formula = log(Income2005) ~ Educ + relevel(Gender, ref = "male"),
    data = ex0923)
```

```
Coefficients:
                 (Intercept)                              Educ
                      9.1429                            0.1161
relevel(Gender, ref = "male")female
                     -0.6427
```

```r
lm(log(Income2005) ~ factor(Educ) + Gender, data = ex0923)
```

```
Call:
lm(formula = log(Income2005) ~ factor(Educ) + Gender, data = ex0923)

Coefficients:
   (Intercept)   factor(Educ)7   factor(Educ)8   factor(Educ)9  factor(Educ)10
        9.3778          0.4133          0.3109         -0.1848          0.1236
factor(Educ)11  factor(Educ)12  factor(Educ)13  factor(Educ)14  factor(Educ)15
        0.1717          0.5073          0.7583          0.7054          0.7318
factor(Educ)16  factor(Educ)17  factor(Educ)18  factor(Educ)19  factor(Educ)20
        1.0790          1.0877          1.3093          1.0066          1.3048
    Gendermale
        0.6487
```

```r
lm(log(Income2005) ~ Educ + Gender + Educ:Gender, data = ex0923)
```

```
Call:
lm(formula = log(Income2005) ~ Educ + Gender + Educ:Gender, data = ex0923)

Coefficients:
    (Intercept)             Educ       Gendermale  Educ:Gendermale
        8.53145          0.11386          0.58519          0.00414
```

```r
lm(log(Income2005) ~ factor(Educ) + Gender + factor(Educ):Gender, data = ex0923)
```

```
Call:
lm(formula = log(Income2005) ~ factor(Educ) + Gender + factor(Educ):Gender,
    data = ex0923)

Coefficients:
```

```
              (Intercept)             factor(Educ)7
                   8.6995                    1.1649
            factor(Educ)8             factor(Educ)9
                   0.9486                    0.1596
           factor(Educ)10            factor(Educ)11
                   0.9468                    0.8302
           factor(Educ)12            factor(Educ)13
                   1.2036                    1.3987
           factor(Educ)14            factor(Educ)15
                   1.4245                    1.4062
           factor(Educ)16            factor(Educ)17
                   1.6881                    1.7676
           factor(Educ)18            factor(Educ)19
                   1.9157                    1.8868
           factor(Educ)20                Gendermale
                   2.0836                    1.6661
 factor(Educ)7:Gendermale  factor(Educ)8:Gendermale
                  -1.3841                   -0.9632
 factor(Educ)9:Gendermale factor(Educ)10:Gendermale
                  -0.5404                   -1.3244
factor(Educ)11:Gendermale factor(Educ)12:Gendermale
                  -0.9914                   -1.0517
factor(Educ)13:Gendermale factor(Educ)14:Gendermale
                  -0.9326                   -1.1094
factor(Educ)15:Gendermale factor(Educ)16:Gendermale
                  -1.0079                   -0.8856
factor(Educ)17:Gendermale factor(Educ)18:Gendermale
                  -1.0212                   -0.8595
factor(Educ)19:Gendermale factor(Educ)20:Gendermale
                  -1.4789                   -1.1661
```

Guidance & Questions:

- Compare the outputs of these models. How does treating `Educ` as a factor change the coefficients compared to using it as a numeric variable?
- What does the interaction between `Educ` and `Gender` tell you about income differences across education levels and genders?
- Why might you transform `Income2005` with a logarithm before modeling?

Guidance

- Treating `Educ` as a factor allows the model to estimate separate effects for each education level rather than assuming a constant change per unit increase in education. This often provides a more accurate picture if the relationship is not strictly linear.

- An interaction between `Educ` and `Gender` suggests that the effect of education on income may vary by gender. It highlights that income gaps between genders could change at different education levels.
- Log-transforming `Income2005` can stabilize variance and make the relationship between predictors and income more linear, which often leads to a better-fitting model and easier interpretation of percentage changes.

Take time to experiment with the code, change factor levels, add different interaction terms, and interpret the outputs. Understanding these concepts will greatly enhance your ability to build and interpret multiple regression models in R.

Summary

- **Factors & Indicators:** Converting variables to factors and creating indicator variables are key for including categorical predictors in regression models.
- **Multiple Regression:** Models can include multiple predictors and interaction terms to capture complex relationships.
- **Interpretation:** Changing factor baselines and adding interaction terms affects how you interpret coefficients.