UPPSALA
UNIVERSITET

Uppsala Child & Baby Lab

Statistics for Psychology 2

R for Psychology Research

Marcus Lindskog, Docent
2019-10-14

Overview

1. A note on the formula (~) notation.
2. Linear regression
3. Logistic regression
4. ANOVA
5. Repeated measures ANOVA

A note on the formula (~) notation

Formula notation

- Many packages in R make use of formulas.
- Formulas aRe a general purpose tool that allows you to capture i) an unevaluated expression, and ii) the context in which the expression was created.
- The majority of modeling functions in R use a standard conversion from formulas to functions.
- E.g.: `y~x` is translated to $y = a_1 + a_2 * x$

A few examples

```
library(modelr)
df <- tibble(y= c(4,5), x1 = c(2,1), x2 =  c(5,6))
```

```
model_matrix(df, y~x1)
```

```
## # A tibble: 2 x 2
##   `(Intercept)`    x1
##           <dbl> <dbl>
## 1             1     2
## 2             1     1
```

```
model_matrix(df, y~x1+x2)
```

```
## # A tibble: 2 x 3
##   `(Intercept)`    x1    x2
##           <dbl> <dbl> <dbl>
## 1             1     2     5
## 2             1     1     6
```

A few more examples

```
model_matrix(df, y~x1 - 1)
```

```
## # A tibble: 2 x 1
##      x1
##   <dbl>
## 1     2
## 2     1
```

```
model_matrix(df, y~x1*x2)
```

```
## # A tibble: 2 x 4
##   `(Intercept)`    x1    x2 `x1:x2`
##           <dbl> <dbl> <dbl>   <dbl>
## 1             1     2     5      10
## 2             1     1     6       6
```

A final example

```
model_matrix(df, y~x1 + x2 + x1:x2)
```

```
## # A tibble: 2 x 4
##   `(Intercept)`    x1    x2 `x1:x2`
##           <dbl> <dbl> <dbl>   <dbl>
## 1             1     2     5      10
## 2             1     1     6       6
```

Linear Regression

Simple linear regression

```
lm(formula, data, subset, weights, na.action,
   method = "qr", model = TRUE, x = FALSE, y = FALSE, qr = TRUE,
   singular.ok = TRUE, contrasts = NULL, offset, ...)
```

- Data

```
lm_data <- cars
```

- Fit the model

```
simple_lm <- lm(speed~dist, data = lm_data)
```

Get summary 1

```
simple_lm
```

```
##
## Call:
## lm(formula = speed ~ dist, data = lm_data)
##
## Coefficients:
## (Intercept)         dist
##      8.2839       0.1656
```

Get summary 2

```
summary(simple_lm)
```

```
##
## Call:
## lm(formula = speed ~ dist, data = lm_data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.5293 -2.1550  0.3615  2.4377  6.4179
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  8.28391    0.87438   9.474 1.44e-12 ***
## dist         0.16557    0.01749   9.464 1.49e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.156 on 48 degrees of freedom
## Multiple R-squared:  0.6511,    Adjusted R-squared:  0.6438
## F-statistic: 89.57 on 1 and 48 DF,  p-value: 1.49e-12
```

Multiple regression

- Data

```
multiple_lm_data <- diamonds
```

- Fit the model

```
multiple_lm_model <- lm(price~carat + depth + table,
                        data = multiple_lm_data)
```

Get summary

```
summary(multiple_lm_model)
```

```
##
## Call:
## lm(formula = price ~ carat + depth + table, data = multiple_lm_data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -18288.0   -785.9    -33.2    527.2  12486.7
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 13003.441    390.918   33.26   <2e-16 ***
## carat        7858.771     14.151  555.36   <2e-16 ***
## depth        -151.236      4.820  -31.38   <2e-16 ***
## table        -104.473      3.141  -33.26   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1526 on 53936 degrees of freedom
## Multiple R-squared:  0.8537,    Adjusted R-squared:  0.8537
## F-statistic: 1.049e+05 on 3 and 53936 DF,  p-value: < 2.2e-16
```

Get `tidy` summary

```
tidy(multiple_lm_model)
```

```
## # A tibble: 4 x 5
##   term        estimate std.error statistic   p.value
##   <chr>          <dbl>     <dbl>     <dbl>     <dbl>
## 1 (Intercept)   13003.    391.        33.3 3.51e-240
## 2 carat          7859.     14.2      555.  0.
## 3 depth          -151.      4.82     -31.4 3.50e-214
## 4 table          -104.      3.14     -33.3 4.18e-240
```

Hierarchical Linear Regression

```
# Model 1
hier_lm_model_one <- lm(price~carat,
                        data = multiple_lm_data)
# Model 2

hier_lm_model_two <- update(hier_lm_model_one, .~. + depth)

# Model 3
hier_lm_model_three <- update(hier_lm_model_two, .~. + table)
```

Compare models

```
# Model 1
anova(hier_lm_model_one, hier_lm_model_two, hier_lm_model_three)
```

```
## Analysis of Variance Table
##
## Model 1: price ~ carat
## Model 2: price ~ carat + depth
## Model 3: price ~ carat + depth + table
##   Res.Df        RSS Df  Sum of Sq       F    Pr(>F)
## 1  53938 1.2935e+11
## 2  53937 1.2819e+11  1 1154586899  495.75 < 2.2e-16 ***
## 3  53936 1.2561e+11  1 2576133006 1106.13 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Logistic Regression

More complex models

- It is, of course, possible to fit more complex models with R.
- For example, if our data requires something more than a linear model, we can choose to model it with a *generalized linear model*.
- In base R, this can be done using `glm()`

```
lm(formula,
   family = gaussian,
   data, weights, subset,    na.action, start = NULL,
   etastart, mustart, offset,
    control = list(...), model = TRUE, method = "glm.fit",
    x = FALSE, y = TRUE, singular.ok = TRUE, contrasts = NULL, ...)
```

Example 1

```r
lm_data <- cars

simple_lm <- glm(speed~dist, data = lm_data,
                 family = "gaussian")
```

Example 1

```
summary(simple_lm)
```

```
##
## Call:
## glm(formula = speed ~ dist, family = "gaussian", data = lm_data)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -7.5293  -2.1550   0.3615   2.4377   6.4179
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  8.28391    0.87438   9.474 1.44e-12 ***
## dist         0.16557    0.01749   9.464 1.49e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 9.958776)
##
##     Null deviance: 1370.00  on 49  degrees of freedom
## Residual deviance:  478.02  on 48  degrees of freedom
## AIC: 260.78
##
## Number of Fisher Scoring iterations: 2
```

Example 2

```
logit_lm_data <- mtcars

simple_logit <- glm(vs~mpg+wt, data = logit_lm_data,
                    family = "binomial")
```

Example 2

```
summary(simple_logit)
```

```
##
## Call:
## glm(formula = vs ~ mpg + wt, family = "binomial", data = logit_lm_data)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.2020  -0.5835  -0.2311   0.5376   1.7142
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -12.5412     8.4660  -1.481   0.1385
## mpg           0.5241     0.2604   2.012   0.0442 *
## wt            0.5829     1.1845   0.492   0.6227
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 43.860  on 31  degrees of freedom
## Residual deviance: 25.298  on 29  degrees of freedom
## AIC: 31.298
##
## Number of Fisher Scoring iterations: 6
```

More information about models

Confidence intervals

```
confint(multiple_lm_model)
```

```
##                   2.5 %      97.5 %
## (Intercept) 12237.2375 13769.64352
## carat        7831.0347  7886.50634
## depth        -160.6834  -141.78932
## table        -110.6296   -98.31594
```

Standardized coefficients (beta weights)

```r
install.packages("sjstats")
library(sjstats)
```

```r
std_beta(multiple_lm_model)
```

```
##    term std.estimate   std.error    conf.low   conf.high
## 1 carat   0.93375156 0.001681357  0.93045616  0.93704696
## 2 depth  -0.05430948 0.001730839 -0.05770187 -0.05091710
## 3 table  -0.05851534 0.001759410 -0.06196372 -0.05506697
```

ANOVA

One-Way ANOVA

Some data

```r
DV <- rnorm(120, 100, 10)
IV <- factor(rep(c("A", "B", "C"), each = 40))

one_way_data <- tibble(IV, DV)
```

The one-way ANOVA

```
aov(formula, data = NULL, projections = FALSE, qr = TRUE,
    contrasts = NULL, ...)
```

- Fit the model

```
one_way_aov <- aov(DV~IV, data = one_way_data)
```

Get the summary

```
one_way_aov
```

```
## Call:
##    aov(formula = DV ~ IV, data = one_way_data)
##
## Terms:
##                        IV Residuals
## Sum of Squares    232.297 10936.536
## Deg. of Freedom         2       117
##
## Residual standard error: 9.66823
## Estimated effects may be unbalanced
```

Get the F-table

```
summary(one_way_aov)
```

```
##              Df Sum Sq Mean Sq F value Pr(>F)
## IV            2    232  116.15   1.243  0.292
## Residuals   117  10937   93.47
```

Make the summary `tidy`

```
tidy_one_way_aov <- tidy(one_way_aov) #From the `broom` package
tidy_one_way_aov
```

```
## # A tibble: 2 x 6
##   term         df   sumsq meansq statistic p.value
##   <chr>     <dbl>   <dbl>  <dbl>     <dbl>   <dbl>
## 1 IV            2    232.  116.       1.24   0.292
## 2 Residuals   117 10937.    93.5     NA     NA
```

```
tidy_one_way_aov$p.value[1]
```

```
## [1] 0.2924233
```

Post-hoc tests - Option 1

```
TukeyHSD(one_way_aov)
```

```
##   Tukey multiple comparisons of means
##     95% family-wise confidence level
##
## Fit: aov(formula = DV ~ IV, data = one_way_data)
##
## $IV
##           diff       lwr      upr     p adj
## B-A  1.894379 -3.237737 7.026495 0.6563078
## C-A -1.506302 -6.638418 3.625814 0.7658515
## C-B -3.400682 -8.532798 1.731434 0.2613676
```

Post-hoc tests - Option 2

```r
library(DescTools)
PostHocTest(x, which = NULL,
            method = c("hsd", "bonferroni", "lsd",
                       "scheffe", "newmankeuls",  "duncan"),
            conf.level = 0.95, ordered = FALSE, ...)
```

```r
PostHocTest(one_way_aov, method = "lsd")
```

```
##
##   Posthoc multiple comparisons of means : Fisher LSD
##     95% family-wise confidence level
##
## $IV
##           diff    lwr.ci   upr.ci   pval
## B-A  1.894379 -2.387114 6.175873 0.3827
## C-A -1.506302 -5.787796 2.775191 0.4873
## C-B -3.400682 -7.682176 0.880812 0.1184
##
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Non-parametric one-way ANOVA

```
kruskal.test(formula, data, subset, na.action, ...)
```

```
kruskal_test <- kruskal.test(DV~IV, data = one_way_data)
```

Get summary

```
kruskal_test
```

```
## 
## 	Kruskal-Wallis rank sum test
## 
## data:  DV by IV
## Kruskal-Wallis chi-squared = 2.0363, df = 2, p-value = 0.3613
```

```
summary(kruskal_test)
```

```
##           Length Class  Mode     
## statistic 1      -none- numeric  
## parameter 1      -none- numeric  
## p.value   1      -none- numeric  
## method    1      -none- character
## data.name 1      -none- character
```

Two-Way ANOVA

A new data set

```
DV <- rnorm(120, 100, 10)
IV1 <- factor(rep(c("A", "B"), each = 60))
IV2 <- factor(rep(rep(c("C", "D"), each = 30), 2))

two_way_data <- tibble(DV, IV1, IV2)
```

The two-way ANOVA

```
aov(formula, data = NULL, projections = FALSE, qr = TRUE,
    contrasts = NULL, ...)
```

- Fit the model

```
two_way_aov_v1 <- aov(DV~IV1*IV2, data = two_way_data)

two_way_aov_v2 <- aov(DV~IV1 + IV2 + IV1:IV2, data = two_way_data)
```

Get the summary V1

```
two_way_aov_v1
```

```
## Call:
##    aov(formula = DV ~ IV1 * IV2, data = two_way_data)
##
## Terms:
##                       IV1       IV2   IV1:IV2 Residuals
## Sum of Squares     31.401   228.841   140.393 10980.245
## Deg. of Freedom         1         1         1       116
##
## Residual standard error: 9.729197
## Estimated effects may be unbalanced
```

Get the summary V2

```
two_way_aov_v2
```

```
## Call:
##    aov(formula = DV ~ IV1 + IV2 + IV1:IV2, data = two_way_data)
##
## Terms:
##                      IV1       IV2   IV1:IV2 Residuals
## Sum of Squares    31.401   228.841   140.393 10980.245
## Deg. of Freedom        1         1         1       116
##
## Residual standard error: 9.729197
## Estimated effects may be unbalanced
```

Get the F-table

```
summary(two_way_aov_v1) #DV~IV1*IV2
```

```
##              Df Sum Sq Mean Sq F value Pr(>F)
## IV1           1     31   31.40   0.332  0.566
## IV2           1    229  228.84   2.418  0.123
## IV1:IV2       1    140  140.39   1.483  0.226
## Residuals   116  10980   94.66
```

```
summary(two_way_aov_v2) #DV~IV1 + IV2 + IV1:IV2
```

```
##              Df Sum Sq Mean Sq F value Pr(>F)
## IV1           1     31   31.40   0.332  0.566
## IV2           1    229  228.84   2.418  0.123
## IV1:IV2       1    140  140.39   1.483  0.226
## Residuals   116  10980   94.66
```

Just the main effects

```
two_way_aov_main <- aov(DV~IV1 + IV2, data = two_way_data)

summary(two_way_aov_main)
```

```
##              Df Sum Sq Mean Sq F value Pr(>F)
## IV1           1     31   31.40   0.330  0.567
## IV2           1    229  228.84   2.408  0.123
## Residuals   117  11121   95.05
```

Post hoc test

```
PostHocTest(two_way_aov_v1, which = "IV2", method = "scheffe")
```

```
##
##   Posthoc multiple comparisons of means : Scheffe Test
##     95% family-wise confidence level
##
## $IV2
##         diff    lwr.ci   upr.ci   pval
## D-C 2.761886 -2.277433 7.801206 0.4931
##
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Repeated measures ANOVA

The biggest difference

- We need to help R specify how to adjust the error term.
- And we need to tell R what how to figure out which observations belong together.

```
DV <- rnorm(120, 100, 10)
IV <- factor(rep(c("A", "B", "C"), each = 40))
ID <- factor(rep(1:40, 3))

one_way_rep_data <- tibble(ID, DV, IV)
```

The one-way repeated measures ANOVA

```
aov(formula, data = NULL, projections = FALSE, qr = TRUE,
    contrasts = NULL, ...)
```

- Fit the model

```
one_way_rep_aov <- aov(DV~IV + Error(ID/IV),
                       data = one_way_rep_data)
```

Get the F-table

```
summary(one_way_rep_aov)
```

```
##
## Error: ID
##           Df Sum Sq Mean Sq F value Pr(>F)
## Residuals 39   4384   112.4
##
## Error: ID:IV
##           Df Sum Sq Mean Sq F value Pr(>F)
## IV         2    273   136.3   1.358  0.263
## Residuals 78   7827   100.3
```

Two-Way repeated measures ANOVA

```
DV <- rnorm(120, 100, 10)
IV1 <- factor(rep(c("A", "B"), each = 60))
IV2 <- factor(rep(rep(c("C", "D"), each = 30), 2))
ID <- factor(rep(1:60, 2))

two_way_rep_data <- tibble(ID, DV, IV1, IV2)
```

Fit the model

```
two_way_rep_aov <- aov(DV~IV1*IV2 + Error(ID/IV1),
                       data = two_way_rep_data)
```

Get the F-table

```
summary(two_way_rep_aov)
```

```
##
## Error: ID
##           Df Sum Sq Mean Sq F value Pr(>F)
## IV2        1    145  144.89   1.839   0.18
## Residuals 58   4571   78.81
##
## Error: ID:IV1
##           Df Sum Sq Mean Sq F value Pr(>F)
## IV1        1      2    1.82   0.015  0.904
## IV1:IV2    1      7    7.05   0.057  0.812
## Residuals 58   7160  123.45
```

Two-Way repeated measures ANOVA

```
DV <- rnorm(120, 100, 10)
IV1 <- factor(rep(c("A", "B"), each = 60))
IV2 <- factor(rep(rep(c("C", "D"), each = 30), 2))
ID <- factor(rep(1:30, 4))

two_way_rep_data_2 <- tibble(ID, DV, IV1, IV2)
```

Fit the model

```
two_way_rep_aov_2 <- aov(DV~IV1*IV2 + Error(ID/(IV1*IV2)),
                         data = two_way_rep_data_2)
```

Get the F-table

```
summary(two_way_rep_aov_2)
```

```
##
## Error: ID
##           Df Sum Sq Mean Sq F value Pr(>F)
## Residuals 29   4058   139.9
##
## Error: ID:IV1
##           Df Sum Sq Mean Sq F value Pr(>F)
## IV1        1  331.4   331.4   3.293 0.0799 .
## Residuals 29 2917.9   100.6
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Error: ID:IV2
##           Df Sum Sq Mean Sq F value Pr(>F)
## IV2        1      4    4.04   0.049  0.827
## Residuals 29   2417   83.34
##
## Error: ID:IV1:IV2
##           Df Sum Sq Mean Sq F value Pr(>F)
## IV1:IV2    1     25   24.98   0.315  0.579
## Residuals 29   2299   79.26
```

That's all folks!