Categorical predictors and interactions in GLM

Advanced statistical methods and models in experimental design

Bartosz Maćkiewicz

Categorical predictors: how to think about them

Until now, our focus has been primarily on numerical variables and their relationships. We will now broaden our scope to include linear models that incorporate categorical predictors. Examples of such predictors include *Gender*, *Education Level*, *Political Preferences*, *Religion*, and *Experimental Condition* (e.g., treatment group).

We will use the Salaries dataset from the carData library.

```
library(carData)
head(Salaries)
```

```
##        rank discipline yrs.since.phd yrs.service  sex salary
## 1      Prof          B            19           18 Male 139750
## 2      Prof          B            20           16 Male 173200
## 3  AsstProf          B             4            3 Male  79750
## 4      Prof          B            45           39 Male 115000
## 5      Prof          B            40           41 Male 141500
## 6 AssocProf          B             6            6 Male  97000
```

Two levels of categorical predictor

Suppose we want to determine if there is a difference in earnings between men and women. We can include sex as a predictor in our linear regression model:

```
fit <- lm(salary ~ sex, data = Salaries)
summary(fit)
```

```
##
## Call:
## lm(formula = salary ~ sex, data = Salaries)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -57290 -23502  -6828  19710 116455
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   101002       4809  21.001  < 2e-16 ***
## sexMale        14088       5065   2.782  0.00567 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 30030 on 395 degrees of freedom
## Multiple R-squared:  0.01921,    Adjusted R-squared:  0.01673
```

Two levels of categorical predictor

You might wonder why we do not simply perform a t-test for two independent samples. It is possible to conduct such a test, and it would yield results equivalent to those from fitting a linear model:

```
pander::pander(t.test(salary ~ sex, data = Salaries, var.equal = T))
```

Table 1: Two Sample t-test: `salary` by `sex` (continued below)

Test statistic	df	P value	Alternative hypothesis
-2.782	395	0.005667 * *	two.sided

mean in group Female	mean in group Male
101002	115090

Simple (treatment) contrasts (*dummy coding*)

By default, R employs *dummy coding* for categorical variables in regression analysis. This method involves the following steps:

- **Contrast Selection**: Select a baseline category.
- **Binary Variables**: Create a new binary variable for each level of the categorical variable, except the baseline category.

```r
Salaries$isMale <- ifelse(Salaries$sex == "Male", 1, 0)
fit <- lm(salary ~ isMale, data = Salaries)
summary(fit)
```

```
##
## Call:
## lm(formula = salary ~ isMale, data = Salaries)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -57290 -23502  -6828  19710 116455
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   101002       4809  21.001  < 2e-16 ***
## isMale         14088       5065   2.782  0.00567 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 30030 on 395 degrees of freedom
```

Simple (treatment) contrast (*dummy coding*)

Understanding Dummy Variables. Generally, you need $k-1$ dummy variables to code a categorical factor with k levels. Consider the `rank` variable in the `Salaries` dataset. This variable has three levels: `Prof` (Professor), `AsstProf` (Assistant Professor), and `AssocProf` (Associate Professor).

```
fit <- lm(salary ~ rank, data = Salaries)
broom::tidy(summary(fit)) %>%
  mutate(p.value = scales::pvalue(p.value)) %>% knitr::kable()
```

term	estimate	std.error	statistic	p.value
(Intercept)	80775.99	2887.313	27.976183	<0.001
rankAssocProf	13100.45	4130.850	3.171369	0.002
rankProf	45996.12	3230.540	14.237906	<0.001

```
contrasts(Salaries$rank) %>% knitr::kable()
```

	AssocProf	Prof
AsstProf	0	0
AssocProf	1	0
Prof	0	1

Regression equation for models with categorical variables

Formulating the Equation. The general form of the regression equation for our model is:

$$\hat{Y} = B_1 C_1 + B_2 C_2 + B_0$$

We have three categories: `AsstProf`, `AssocProf`, and `Prof`. To code them we created two dummy variables: C_1 and C_2.

`AsstProf` is a **baseline group** or a **reference level**. It is coded as $(C_1 = 0, C_2 = 0)$. `AssocProf` is coded as $(C_1 = 1, C_2 = 0)$ and `Prof` as $(C_1 = 0, C_2 = 1)$.

For `AsstProf` we can thus reduce the equation to:

$$\hat{Y} = B_0$$

for `AssocProf` to:

$$\hat{Y} = B_1 C_1 + B_0$$

and for `Prof` to:

$$\hat{Y} = B_2 C_2 + B_0$$

Regression equation for models with categorical variables

Interpretation

- The intercept (B_0) represents the mean salary for the baseline group (AsstProf).
- The coefficients B_1 and B_2 represent the salary differences relative to `AsstProf` for `AssocProf` and `Prof`, respectively.

Problems

- How to choose the right reference group?
- When it is useful to use such a coding?

Deviation (sum to zero) contrasts

Deviation contrasts, also known as *sum to zero* contrasts, offer an alternative method for coding categorical predictors. This approach modifies the interpretation of both the regression coefficients and the intercept.

```
contr.sum(levels(Salaries$rank))
```

```
##           [,1] [,2]
## AsstProf     1    0
## AssocProf    0    1
## Prof        -1   -1
```

Deviation (sum to zero) contrasts

```
fit <- lm(salary ~ rank,
          data = Salaries,
          contrasts = list(rank = standardize::named_contr_sum(Salaries
summary(fit)
```

```
##
## Call:
## lm(formula = salary ~ rank, data = Salaries, contrasts = list(rank =
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -68972 -16376  -1580  11755 104773
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)     100475       1459  68.855  < 2e-16 ***
## rankAssocProf   -19699       2215  -8.892  < 2e-16 ***
## rankAsstProf     -6598       2245  -2.940  0.00348 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 23630 on 394 degrees of freedom
## Multiple R-squared:  0.3943, Adjusted R-squared:  0.3912
```

Deviation (sum to zero) contrasts

Interpretation

- `Intercept` (B_0) is a *grand mean* (mean of means)

```
mean(tapply(Salaries$salary, Salaries$rank, mean))
## [1] 100474.8
```

- Regression coefficients (B_n) represent deviations from the grand mean

```
tapply(Salaries$salary, Salaries$rank, mean) - mean(tapply(
##   AsstProf  AssocProf       Prof
## -19698.859  -6598.406  26297.265
```

- Null hypotheses for regression coefficients state that the deviations are equal to 0

Other contrasts

Helmert contrasts

```
contr.helmert(levels(Salaries$rank))
fit <- lm(salary ~ rank, data = Salaries, contrasts =
            list(rank = contr.helmert))
summary(fit)
```

Successive difference contrasts

```
contr.sum(levels(Salaries$rank))
fit <- lm(salary ~ rank, data = Salaries, contrasts =
            list(rank = MASS::contr.sdif))
summary(fit)
```

The concept of interaction

By interactions, we mean an interplay among predictors that produces an effect on the outcome Y that is different from the sum of the effects of the individual predictors.

No interaction

Consider as an example how ability (X) and motivation (Z) impact achievement in graduate school (F). One possibility is that their effects are additive. The combined impact of ability and motivation on achievement equals the sum of their separate effects; there is no interaction between X and Z. We might say that the whole equals the sum of the parts.

The concept of interaction

Synergy

A second alternative is that ability and motivation may interact *synergistically*, such that graduate students with both high ability and high motivation achieve much more in graduate school than would be expected from the simple sum of the separate effects of ability and motivation. Graduate students with both high ability and high motivation become "superstars"; we would say that the whole is greater than the sum of the parts.

The concept of interaction

Compensation

A third alternative is that ability and motivation compensate for one another. For those students who are extremely high in ability, motivation is less important to achievement, whereas for students highest in motivation, sheer native ability has less impact. Here we would say that the whole is less than the sum of the parts; there is some partial trade-off between ability and motivation in the prediction of achievement.

Interactions between two continuous predictors

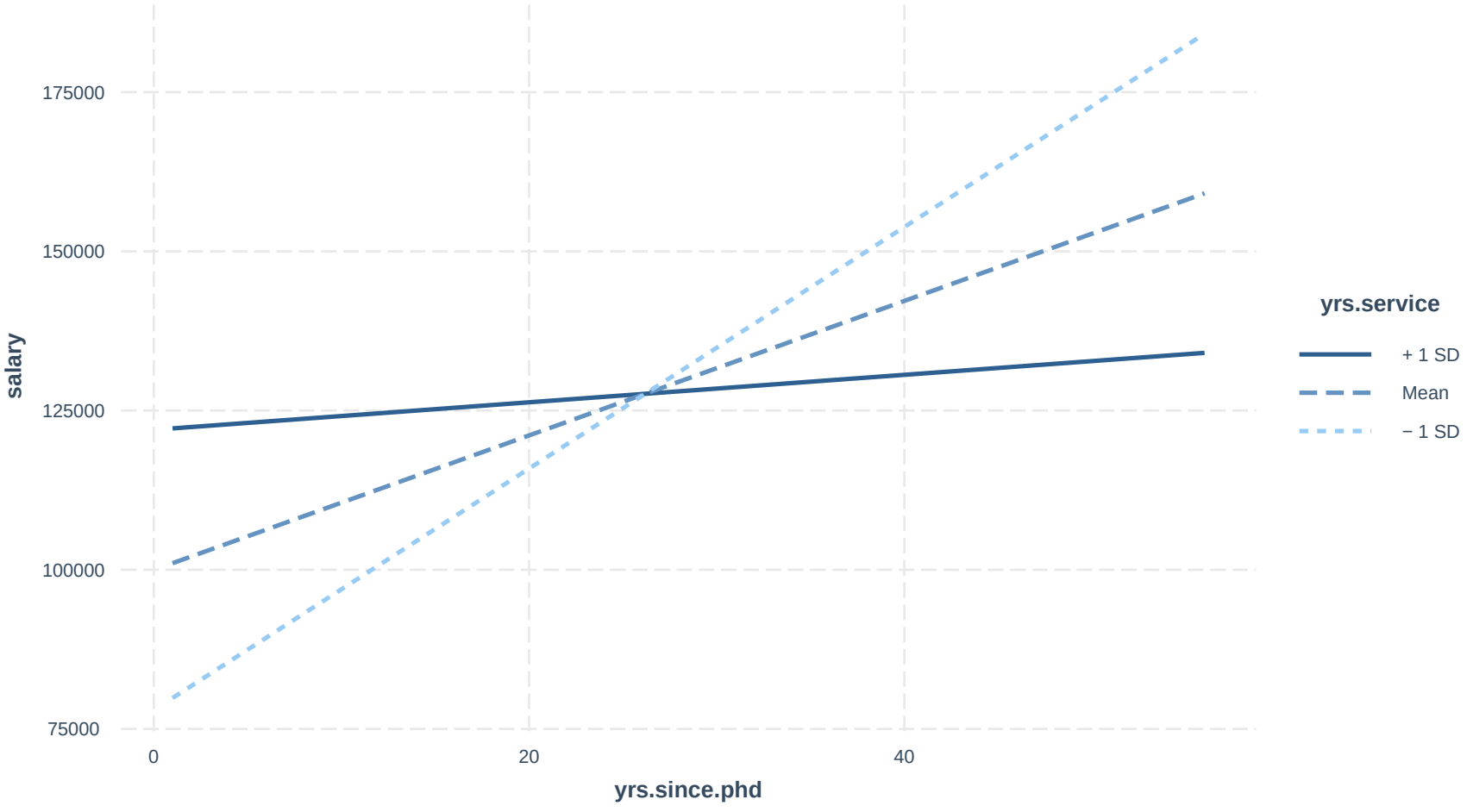
```r
# Model with no interactions
fit <- lm(salary ~ yrs.service + yrs.since.phd, data = Salaries)

# Model with an interaction
fit_i <- lm(salary ~ yrs.service * yrs.since.phd, data = Salaries)

#summary(fit) # run this code
#summary(fit_i) # run this code
```

Interactions between two continuous predictors

```
interactions::interact_plot(fit_i,
                            pred = "yrs.since.phd",
                            modx = "yrs.service"
                            )
```



Interactions between categorical and continuous predictors

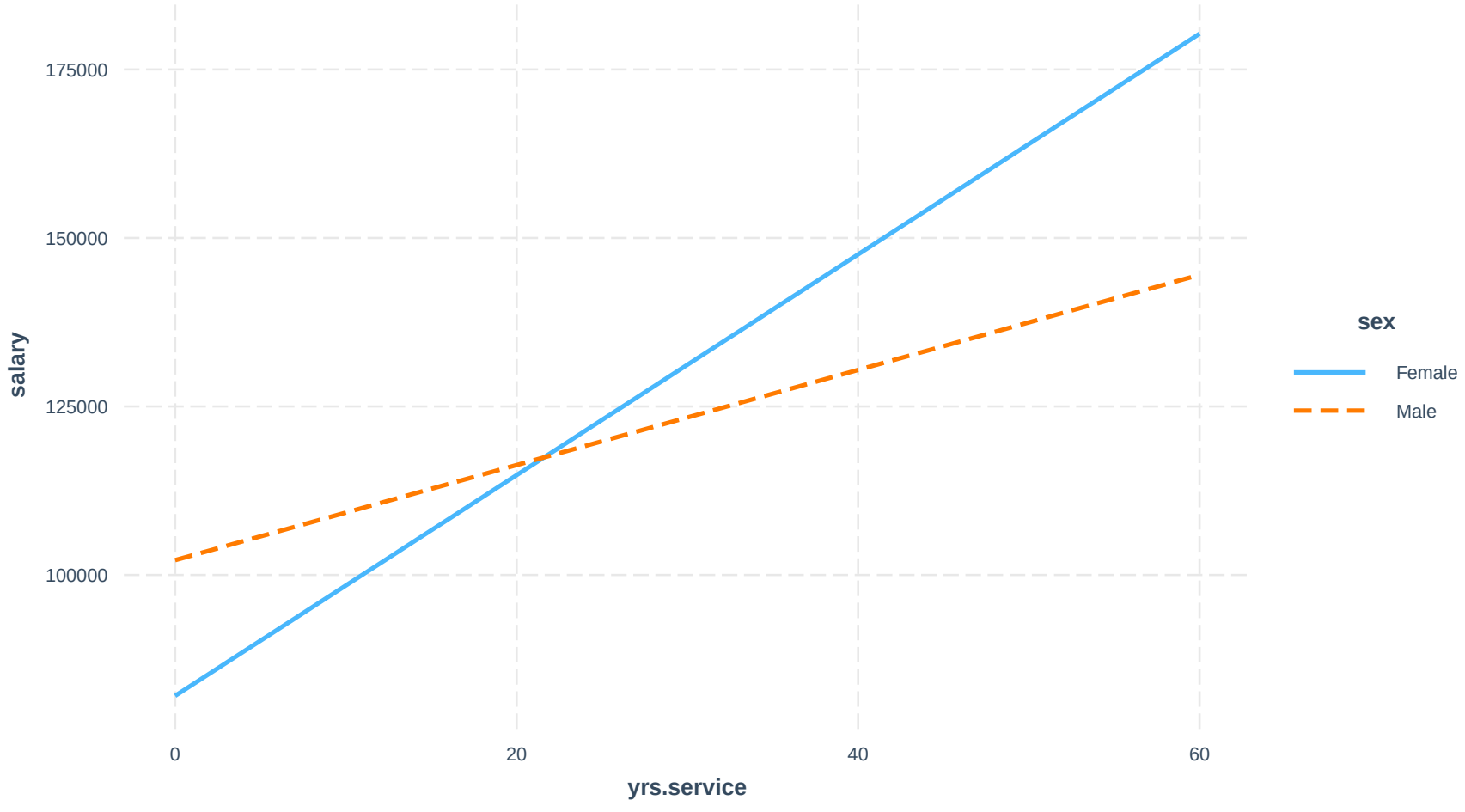
```r
# Model with no interactions
fit <- lm(salary ~ yrs.service + sex, data = Salaries)

# Model with an interaction
fit_i <- lm(salary ~ yrs.service * sex, data = Salaries)

#summary(fit) # run this code
#summary(fit_i) # run this code
```

Interactions between categorical and continuous predictors

```
interactions::interact_plot(fit_i,
                            pred = "yrs.service",
                            modx = "sex"
                            )
```



Interactions between two categorical predictors

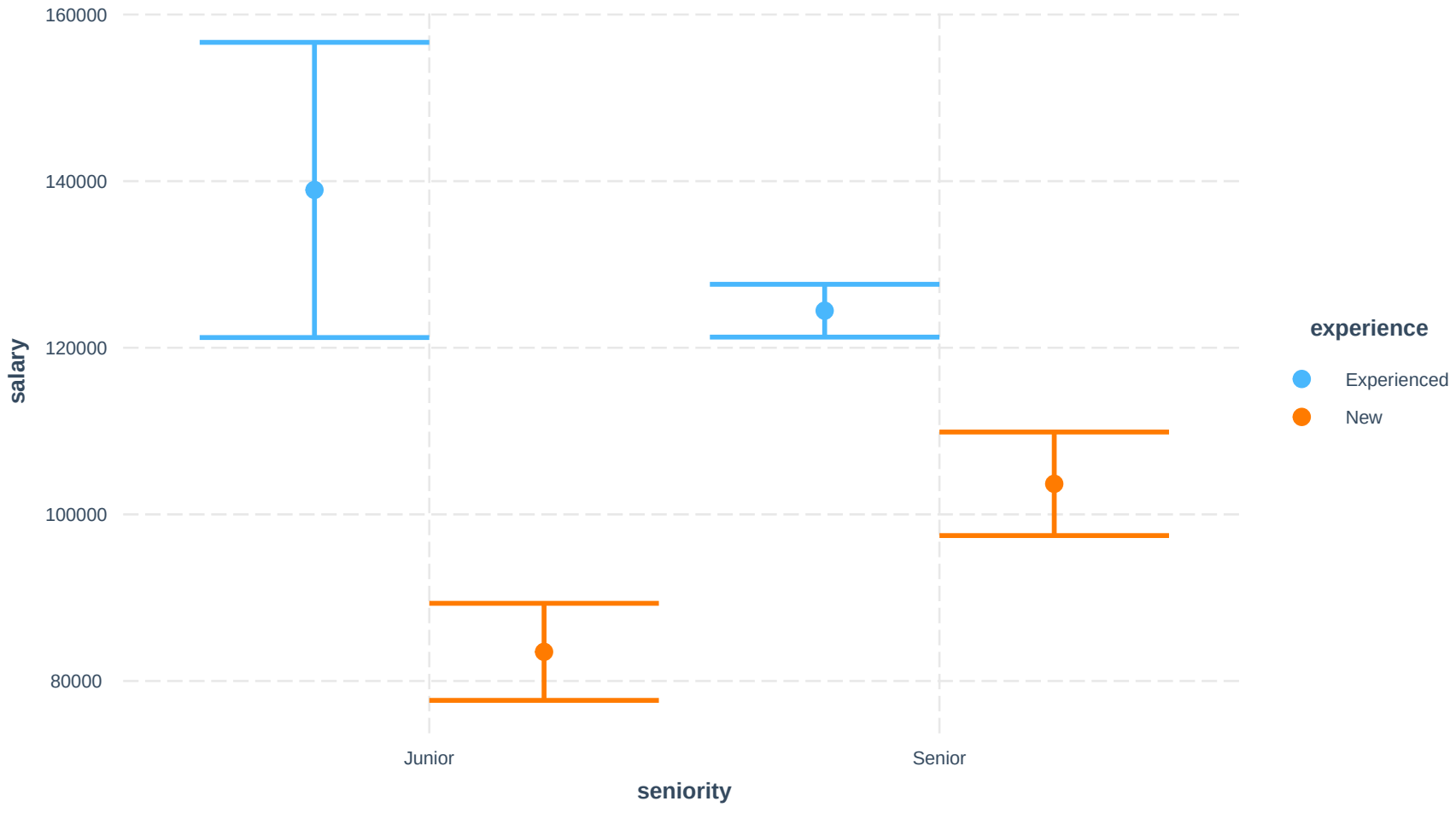
```r
# Fake variables...
Salaries$seniority <- ifelse(Salaries$yrs.service > 5,
                             "Senior",
                             "Junior")
Salaries$experience <- ifelse(Salaries$yrs.since.phd > 15,
                              "Experienced",
                              "New")
# Model with no interactions
fit <- lm(salary ~ seniority + experience, data = Salaries)

# Model with interactions
fit_i <- lm(salary ~ seniority * experience, data = Salaries)

#summary(fit) # run this code
#summary(fit_i) # run this code
```

Interactions between two categorical predictors

```
interactions::cat_plot(fit_i,
                       pred = "seniority",
                       modx = "experience"
                       )
```



Interactions in multiple linear regression: example

Wagner, Compas, and Howell (1988) hypothesized that individuals who experience more stress, as assessed by a measure of daily hassles, will exhibit higher levels of symptoms than those who experience little stress. That is what, in analysis of variance terms, would be the main effect of hassles. However, they also expected that if a person had a high level of social support to help deal with his or her stress, symptoms would increase only slowly with increases in hassles. For those who had relatively little social support, symptoms were expected to rise more quickly as hassles increased.

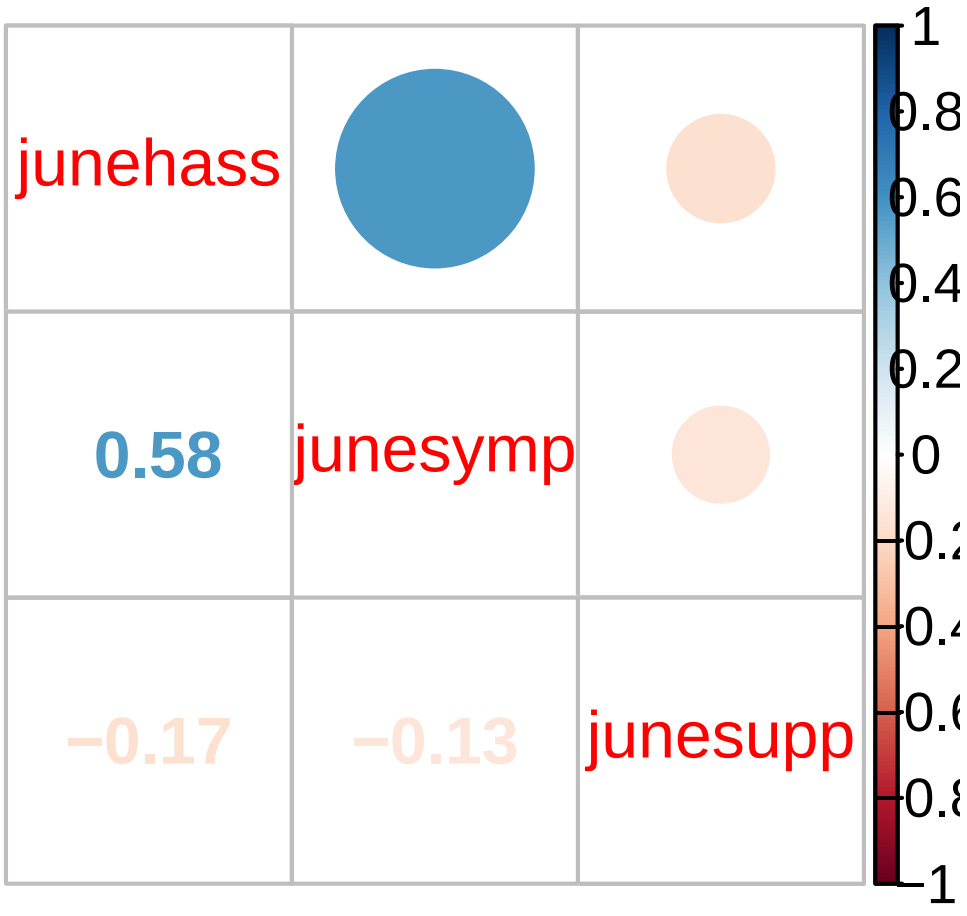
```
library(dplyr); hassles <- read.csv("hassles.csv")
hassles %>%
  select(id, junehass, junesymp, junesupp) %>%
  slice_head(n = 5)
```

```
##   id junehass junesymp junesupp
## 1  5      176       73       10
## 2 26      379       88       50
## 3 58      126      118       45
## 4 41      193       79       40
## 5 63      229      127       40
```

Interactions in multiple linear regression: example

First we can look at the correlation matrix between all three variables.

```
corrplot::corrplot.mixed(
  cor(hassles %>% select(junehass, junesymp, junesupp)))
```



Interactions in multiple linear regression: example

If we just multiply Hassles and Support together, there will be two problems with what results.

In the first place, either Hassles or Support or both will be highly correlated with their product, which will make for multicollinearity in the data.

Second, any effect of Hassles or Support in the regression analysis will be evaluated at a value of 0 for the other variable. In other words the test on Hassles will be a test on whether Hassles is related to Symptoms if a participant had exactly no social support. Similarly the test on Support would be evaluated for those participants who have exactly no hassles.

To circumvent these two problems we are going to **center** our data

```
hassles$c_hassles <- hassles$junehass - mean(hassles$junehass)
hassles$c_supp <- hassles$junesupp - mean(hassles$junesupp)
```

Interactions in multiple linear regression: example

We can compare two models – with interaction term and without it.

```
fit_no_interact <- lm(junesymp ~ c_hassles + c_supp,
                      data = hassles)
fit_interact <- lm(junesymp ~ c_hassles * c_supp,
                   data = hassles)
anova(fit_no_interact, fit_interact)
```

```
## Analysis of Variance Table
##
## Model 1: junesymp ~ c_hassles + c_supp
## Model 2: junesymp ~ c_hassles * c_supp
##   Res.Df   RSS Df Sum of Sq      F  Pr(>F)
## 1     53 16151
## 2     52 14840  1    1311.3 4.5948 0.03677 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' '
```

Interactions in multiple linear regression: example

```
summary(fit_interact)
```

```
##
## Call:
## lm(formula = junesymp ~ c_hassles * c_supp, data = hassles)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -38.734 -13.379  -0.444   8.689  35.523
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)      89.584940   2.291503  39.094  < 2e-16 ***
## c_hassles         0.085942   0.019213   4.473 4.22e-05 ***
## c_supp            0.146358   0.305244   0.479   0.6336
## c_hassles:c_supp -0.005065   0.002363  -2.144   0.0368 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 16.89 on 52 degrees of freedom
## Multiple R-squared:  0.3885, Adjusted R-squared:  0.3532
## F-statistic: 11.01 on 3 and 52 DF,  p-value: 1.046e-05
```

Interactions in multiple linear regression: example

```
interactions::interact_plot(fit_interact,
                c_hassles, # predictor variable
                c_supp) # moderator variable
```