Introduction to Mixed-Effects Models

2018-02-20

Preparation

We will use two datasets located in the *nbaLevel1.sav* file and the *nbaLevel2.sav* file. These data include player-level attributes for $n = 300$ NBA players, and team-level attributes for $N = 30$ different teams. The source of these data is: Woltman, Feldstein, MacKay, & Rocchi (2012). We will use these data to explore the question of whether how good a player is (`Shots_on_five`) predicts variation in life satisfaction.

The player-level attributes in the *nbaLevel1.sav* file include:

- `Team_ID`: The team ID number for each player
- `Shots_on_five`: A proxy for player quality/success. This indicates the number of successful shots (out of five taken). Higher values indicate a more succesful player.
- `Life_Satisfaction`: Score on a survey of life satisfaction. Scores range from 5 to 25, with higher scores indicating more life satisfaction.

The team-level attributes in the *nbaLevel2.sav* file include:

- `Team_ID`:The team ID number
- `Coach_Experience`: Years of coaching experience in the NBA

Importing SPSS Files into R

Both files have the extension *.sav*. This is the file saving extension SPSS appends to data files. To read a SAV file, we use the `read_sav()` function from the **haven** library. This function has similar syntax to the `read_csv()` we have been using. If you are using the `Import` button in RStudio, you can select `From SPSS...` Below we use this function to read in both datasets.

```
# Load libraries
library(AICcmodavg)
library(arm)
library(broom)
library(dplyr)
library(ggplot2)
library(haven) #for reading in the .sav files
library(lme4) #for fitting mixed-effects models
library(sm)

# Read in player-level data
nbaL1 = read_sav(file = "~/Dropbox/epsy-8252/data/nbaLevel1.sav")
head(nbaL1)
```

Team_ID	Shots_on_five	Life_Satisfaction
01	3	18.8
01	3	18.0
01	4	21.0
01	4	20.5
01	3	19.0
01	2	12.1

```
# Read in team-level data
nbaL2 = read_sav(file = "~/Dropbox/epsy-8252/data/nbaLevel2.sav")
head(nbaL2)
```

Team_ID	Coach_Experience
01	2
02	3
03	2
04	2
05	1
06	2

Merge the Player- and Team-Level Data

Before analyzing the data, we need to merge, or join, the two datasets together. To do this, we will use the `left_join()` function from the **dplyr** package. **dplyr** includes six different join functions. You can read about several different join functions here.

```
nba = left_join(nbaL1, nbaL2, by = "Team_ID")
head(nba)
```

Team_ID	Shots_on_five	Life_Satisfaction	Coach_Experience
01	3	18.8	2
01	3	18.0	2
01	4	21.0	2
01	4	20.5	2
01	3	19.0	2
01	2	12.1	2

Linear Regression: Fixed-Effects

To examine the research question of whether how good a player is predicts variation in life satisfaction, we might regress life satisfaction on the Shots-on-five variable (and any potential covariates) using the `lm()` function. The `lm()` function fits a *fixed-effects regression model*.

Fit Linear Models

```
lm.1 = lm(Life_Satisfaction ~ 1 + Shots_on_five, data = nba)
display(lm.1)
```

```
## lm(formula = Life_Satisfaction ~ 1 + Shots_on_five, data = nba)
##               coef.est coef.se
## (Intercept)   5.69     0.30
## Shots_on_five 3.66     0.11
## ---
## n = 300, k = 2
## residual sd = 2.44, R-Squared = 0.80
```

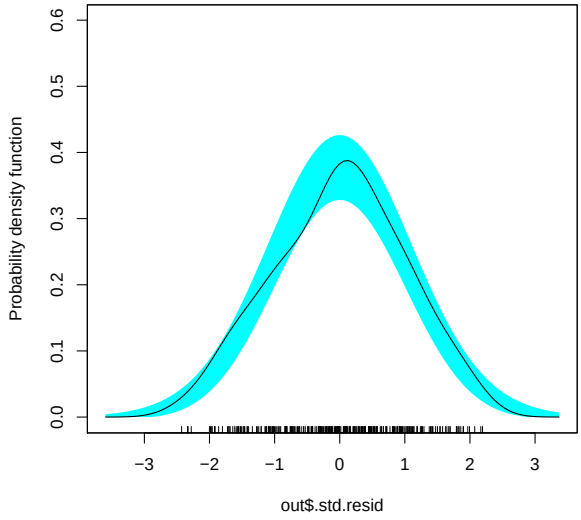
To have faith in the analytic results from this model, we need to evaluate whether the assumptions are satisfied.

Examine Residuals

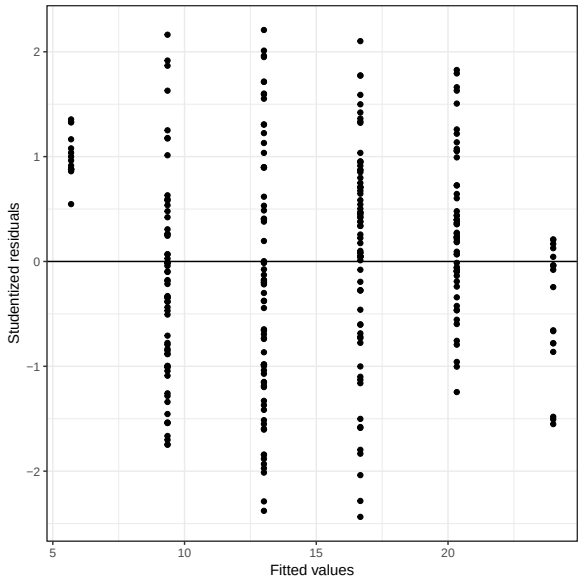
```
# Obtain the fortified data frame
out = augment(lm.1)
head(out)
```

Life_Satisfaction	Shots_on_five	.fitted	.se.fit	.resid	.hat	.sigma	.cooksd	.std.resid
18.8	3	16.7	0.151	2.131	0.004	2.44	0.001	0.873
18.0	3	16.7	0.151	1.327	0.004	2.45	0.001	0.544
21.0	4	20.3	0.215	0.667	0.008	2.45	0.000	0.274
20.5	4	20.3	0.215	0.167	0.008	2.45	0.000	0.069
19.0	3	16.7	0.151	2.327	0.004	2.44	0.002	0.954
12.1	2	13.0	0.151	-0.913	0.004	2.45	0.000	-0.374

```
# Normality
sm.density(out$.std.resid, model = "normal")
```




```
# All other assumptions
ggplot(data = out, aes(x = .fitted, y = .std.resid)) +
    geom_point() +
    geom_hline(yintercept = 0) +
    theme_bw() +
  xlab("Fitted values") +
  ylab("Studentized residuals")
```

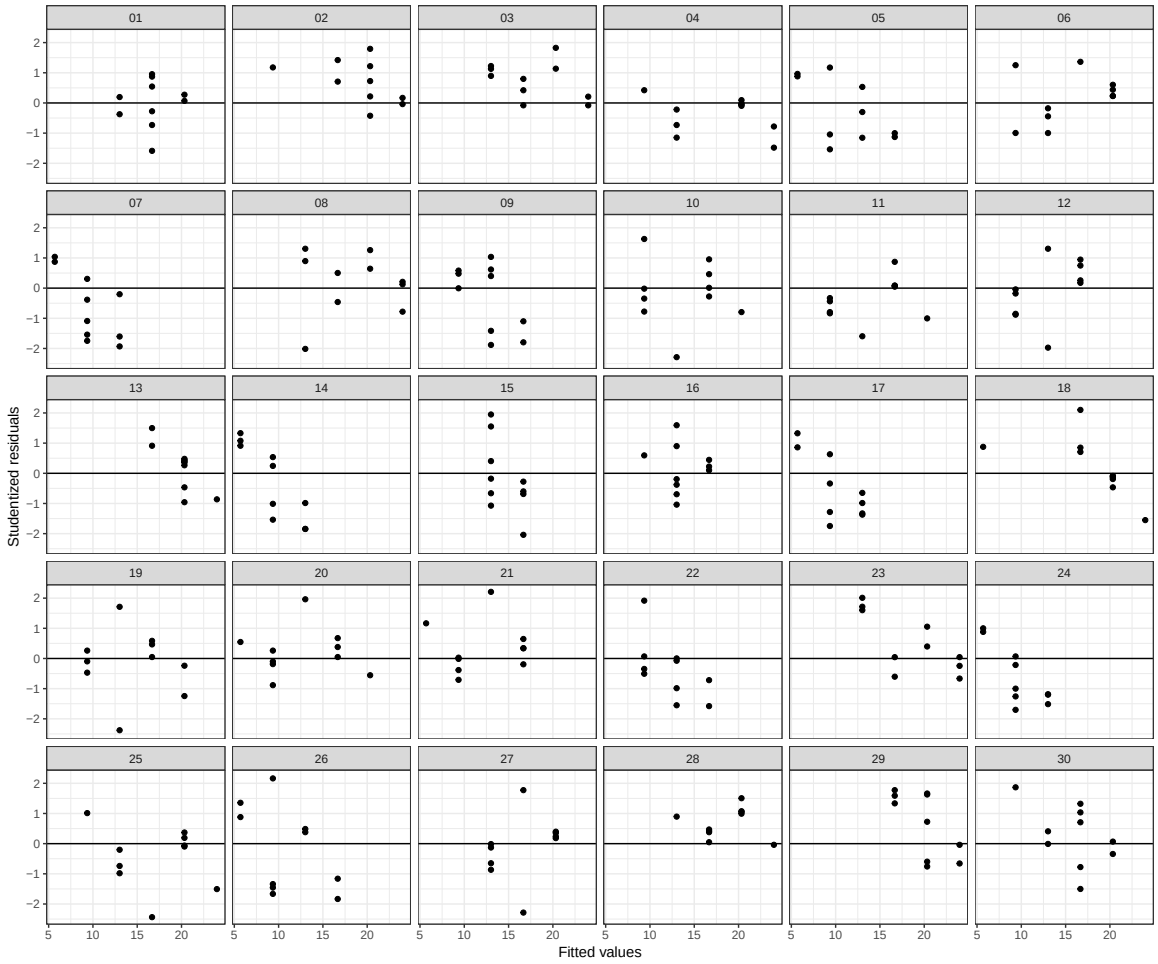


The assumptions of linearity, normality, and homoscedasticity seem reasonably satisfied. The assumption of independnece, however, is probably not tenable. The life satisfaction scores (and thus the residuals) are probably more correlated within teams than between teams—this is a violation of independence.

If we have a variable that identifies team, we can actually examine this by plotting the residuals separately for each team.

```
# Add Team_ID variable to fotified data
out$Team_ID = nba$Team_ID

### Show residuals by team
ggplot(data = out, aes(x = .fitted, y = .std.resid)) +
    geom_point() +
    geom_hline(yintercept = 0) +
    theme_bw() +
  xlab("Fitted values") +
  ylab("Studentized residuals") +
    facet_wrap(~Team_ID, nrow = 5)
```



The residuals are systematically over or under 0 within teams (e.g., Team 11, Team 28). This is a sign of non-independence of the residuals. To account for this we need to use a statistical model that accounts for the correlation among the residuals within teams. This is what *mixed-effects models* bring to the table. By correctly modeling the non-independence, we get more accurate standard errors and p-values.

Another benefit of using mixed-effects models is that we also get estimates of the variation accounted for at both the team- and player-levels. This disaggregating of the variation allows us to see which level is explaining more variation and to study predictors appropriate to explaining that variation. For example, suppose that in an educational study you disaggregated the variation in student achievement scores and found that:

- 96% of the variation in these scores was at the student-level, and
- 3% of the variation in these scores was at the classroom-level, and
- 1% of the variation in these scores was at the school-level.

By including school-level or classroom-level predictors in the model, you would only be "chipping away" at that 1% or 3%, respectively. You should focus your attention and resources on student-level predictors!

Conceptual Idea of Mixed-Effects Models

In this section we will outline the conceptual ideas behind mixed-effects models by linking the ideas behind these models to the conventional, fixed-effects regression model. *It is important to realize that this is just conceptual in nature. Its purpose is only to help you understand the output you get from a mixed-effects model analysis.*

To begin, we remind you of the fitted regression equation we obtained earlier, when we regressed life stais-faction on player success for the $n = 300$ players:

$$\hat{\text{Life Satisfaction}}_i = 5.70 + 3.66(\text{Player Success}_i)$$

Mixed-effects regression actually fits a global model (like the one above) AND a team-specific model for each team. Conceptually this is like fitting a regression model for each team separately. Below I will do this, but keep in mind that this is only to help you understand.

```
models = nba %>%
  group_by(Team_ID) %>%
  do( mod = lm(Life_Satisfaction  ~ Shots_on_five, data = .) ) %>%
  broom::tidy(mod)

models
```

```
## # A tibble: 60 x 6
## # Groups:   Team_ID [30]
##    Team_ID term          estimate std.error statistic   p.value
##    <chr>   <chr>            <dbl>     <dbl>     <dbl>     <dbl>
##  1 01      (Intercept)       4.73     3.04       1.55 0.159
##  2 01      Shots_on_five     3.98     0.992      4.01 0.00389
##  3 02      (Intercept)       9.96     1.80       5.52 0.000560
##  4 02      Shots_on_five     2.97     0.467      6.35 0.000220
##  5 03      (Intercept)       9.03     1.50       6.01 0.000321
##  6 03      Shots_on_five     3.20     0.432      7.40 0.0000760
##  7 04      (Intercept)       5.66     1.26       4.50 0.00200
##  8 04      Shots_on_five     3.38     0.353      9.57 0.0000118
##  9 05      (Intercept)       7.04     1.25       5.63 0.000493
## 10 05      Shots_on_five     2.33     0.688      3.39 0.00948
## # ... with 50 more rows
```

As an example, let's focus on the fitted model for Team 01.

$$\hat{\text{Life Satisfaction}}_i = 4.73 + 3.98(\text{Player Success}_i)$$

Comparing this team-specific model to the global model, we find that Team 01's intercept is lower than the intercept from the global model (by 0.97) and it's slope is higher than the slope from the global model (by 0.32). We can actually re-write the team-specific model using these ideas:

$$\hat{\text{Life Satisfaction}}_i = \Big[5.70 - 0.97\Big] + \Big[3.66 + 0.32\Big](\text{Player Success}_i)$$

In the language of mixed-effects modeling:

- The global intercept and slope are referred to as *fixed-effects*.
- The fixed-effect of intercept is 5.70; and
- The fixed effect of the slope is 3.66.
- The team-specific deviations from the fixed-effect values are referred to as *random-effects*.

- The random-effect of the intercept for Team 01 is -0.97; and
- The random-effect of the slope for Team 01 is $+0.32$.

Note, each team could potentially have a different random-effect for intercept and slope. For example, writing the team-specific fitted equation for Team 02 in this manner,

$$\begin{aligned}\widehat{\text{Life Satisfaction}}_i &= 9.96 + 2.97(\text{Player Success}_i) \\ &= \Big[5.70 + 4.26\Big] + \Big[3.66 + 0.69\Big](\text{Player Success}_i),\end{aligned}$$

we find that:

- The fixed-effects are the same (global) as they were for Team 01.
- The random-effect of intercept for Team 02 is 4.26.
- The random-effect of slope for Team 02 is 0.69.

Fitting the Mixed-Effects Regression Model in Practice

In practice, we use the `lmer()` function from the **lme4** library to fit mixed-effect regression models. This function will essentially do what we did in the previous section, but rather than independently fitting the team-specific models, it will fit all these models simultaneously and make use of the information in all the clusters (teams) to do this. This will result in better estimates for both the fixed- and random-effects.

The syntax looks similar to the syntax we use in `lm()` except now we split it into two parts. The first part of the syntax gives a model formula to specify the outcome and fixed-effects included in the model. This is identical to the syntax we used in the `lm()` function. In our example: `Life_Satisfaction ~ 1 + Shots_on_five` indicating that we want to fit a model that includes fixed-effects for both the intercept and the slope (the fixed-effect of player success).

We also have to declare that we want random-effects for these two coefficients. The second part of the syntax declares this: `(1 + Shots_on_five | Team_ID)`. This says fit random-effects for the intercept and slope and do so for each team. This is literally added (using `+`) to the fixed-effects formula.

```
# Fit mixed-effects regression model
lmer.1 = lmer(Life_Satisfaction ~ 1 + Shots_on_five + (1 + Shots_on_five | Team_ID), data = nba)
```

To view the fixed-effects, we use the `fixef()` function.

```
fixef(lmer.1)
```

```
##   (Intercept) Shots_on_five
##          6.43          3.29
```

To view the team-specific random-effects, we use the `ranef()` function (only the first 6 rows are shown).

```
ranef(lmer.1)
```

```
$Team_ID
   (Intercept) Shots_on_five
01      0.0779        0.0805
02      0.3611        0.3732
03      0.3844        0.3973
04     -0.0737       -0.0762
05     -0.2381       -0.2461
06      0.1734        0.1792
```

From these two sets of information, we can re-construct each team-specific fitted equation if we are so inclined. For example, to construct the team-specific fitted equation for Team 30, we first write the fitted equatin for the fixed-effects (global equation):

$$\widehat{\text{Life Satisfaction}}_i = 6.43 + 3.29(\text{Player Success}_i)$$

Then we add the estimated random-effects for Team 30 to the respective fixed-effects:

$$\begin{aligned}\widehat{\text{Life Satisfaction}}_i &= \Big[6.43 + 0.1463\Big] + \Big[3.29 + 0.1512\Big](\text{Player Success}_i) \\ &= 6.58 + 3.44(\text{Player Success}_i)\end{aligned}$$

Example 2: Beauty and Course Evaluations

In the second example, we will re-visit the `beauty` dataset to answer the question of whether there is a differential effect of beauty by gender on course evaluation scores. Recall that the data were collected from student evaluations of instructors' beauty and teaching quality for several courses at the University of Texas. The teaching evaluations were conducted at the end of the semester, and the beauty judgments were made later, by six students who had not attended the classes and were not aware of the course evaluations. The source of these data is Hamermesh & Parker (2005); made available by Gelman & Hill (2007).

The variables in the dataset are:

- `prof`: Professor ID number
- `avgeval`: Average course rating
- `btystdave`: Measure of the professor's beauty composed of the average score on six standardized beauty ratings
- `tenured`: 0 = non-tenured; 1 = tenured
- `nonenglish`: 0 = native English speaker; 1 = non-native English speaker
- `age`: Professor's age (in years)
- `female`: 0 = male; 1 = female
- `students`: Number of students enrolled in the course
- `percentevaluating`: Percentage of enrolled students who completed an evaluation

```
# Read in data
beauty = readr::read_csv(file = "~/Dropbox/epsy-8252/data/beauty.csv")
head(beauty)
```

prof	avgeval	btystdave	tenured	nonenglish	age	female	students	percentevaluating
1	4.3	0.202	0	0	36	1	43	55.8
2	4.5	-0.826	1	0	59	0	20	85.0
3	3.7	-0.660	1	0	51	0	55	100.0
4	4.3	-0.766	1	0	40	1	46	87.0
5	4.4	1.421	0	0	31	1	48	87.5
6	4.2	0.500	1	0	62	0	282	64.5

Fit a Fixed-Effects Model

These data, unlike the NBA data, are already merged; there are both student-level and professor-level variables in the same file. Below, we fit a fixed-effects regression model to predict variation in course evaluation scores based on three predictors: the professor's beauty, the professor's sex, and whether the professor is a native speaker of English.

```
lm.1 = lm(avgeval ~ 1 + btystdave + female + nonenglish, data = beauty)
```

For the results to be valid, the independence assumption needs to be satisfied. It is not unreasonable to think that course evaluation scores are correlated among students within the same course (professor). Since we have a variable in the data that indicates professor, we can examine the model's residuals grouped by professor to assess them for violation of the independence assumption.

There are 94 professors represented in the data, and rather than show all 94 plots (too much!) we randomly sample 25 and only show those plots.

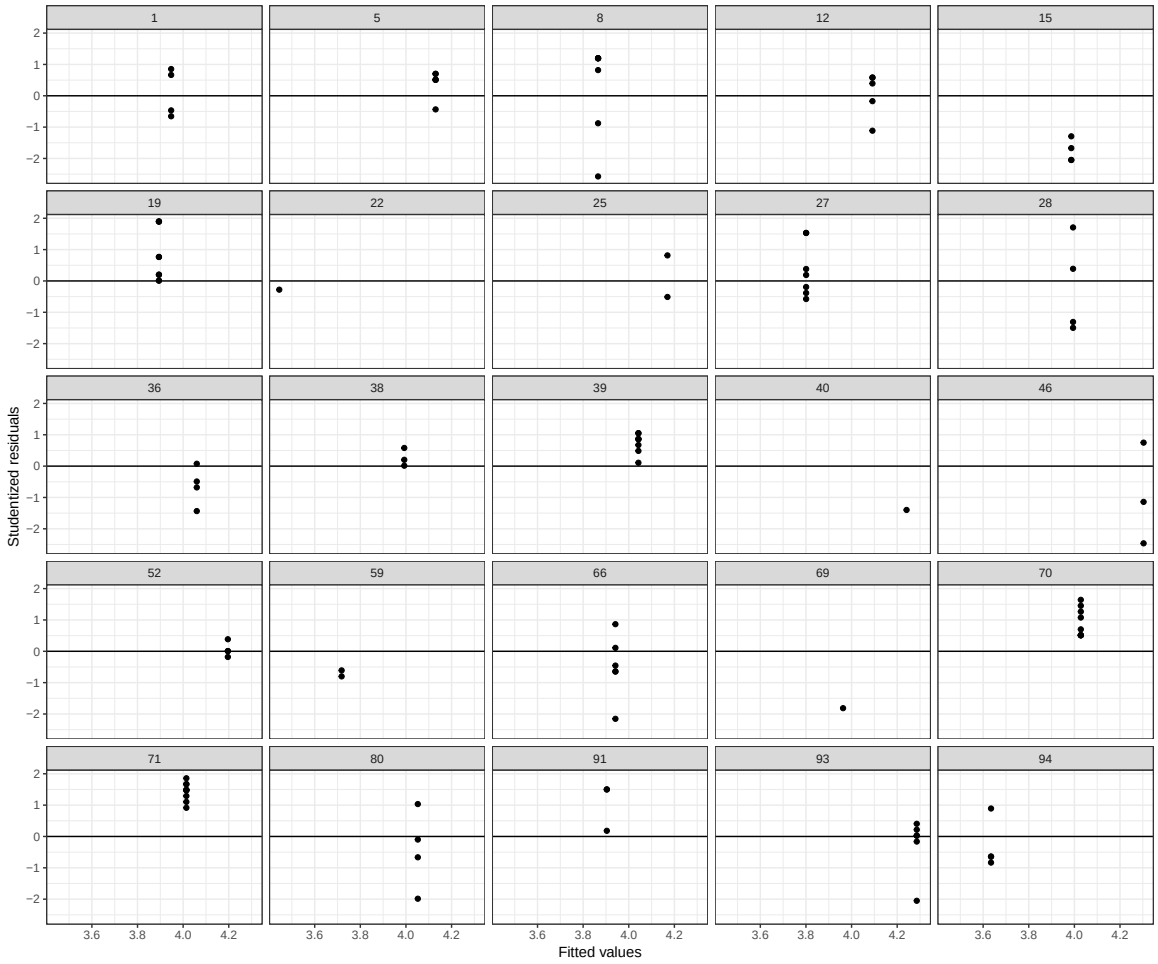
```
# Get model residuals, fitted values, etc.
out = augment(lm.1)

# Add professor ID to the data; Turn it into a factor for better plotting
out$prof = as.factor(beauty$prof)

# Randomly sample 25 professors
set.seed(1001)
my_profs = sample(unique(out$prof), size = 25, replace = FALSE)

# Filter the data to only select data from those professors
my_sample = out %>%
  filter(prof %in% my_profs)

# Show residuals by professor
ggplot(data = my_sample, aes(x = .fitted, y = .std.resid)) +
    geom_point() +
    geom_hline(yintercept = 0) +
    theme_bw() +
  xlab("Fitted values") +
  ylab("Studentized residuals") +
    facet_wrap(~prof, nrow = 5)
```



This plot shows some evidence of non-independence. To compensate for that, we should fit a mixed-effects regression model rather than a fixed-effects regression model.

Fit the Mixed-Effects Model

Below we fit a mixed-effects regression model that includes the same fixed-effects as in our fixed-effects model and a random-effect of intercept. (Note: To account for the dependence in the data we need to include a random-effect, but we do not need to include random-effects for all of our predictors. Here we chose to only include a random-effect for the intercept.)

```
# Fit model
lmer.1 = lmer(avgeval ~ 1 + btystdave + female + nonenglish + (1 | prof),
              data = beauty)

# Get fixed-effects
fixef(lmer.1)
```

```
## (Intercept)   btystdave      female  nonenglish
##       4.053       0.133      -0.205      -0.359
```

```r
# Get random-effects (only first 10 are shown)
ranef(lmer.1)
```

```
   (Intercept)
1      0.09349
2     -0.28198
3     -0.30658
4      0.22753
5      0.25545
6      0.27089
7      0.01611
8      0.18972
9      0.33576
10     0.39347
```

Combining the fixed- and random-effects, we can write the professor-specific models. For example, the professor-specific model for Professor 1 is:

$$\begin{aligned}\hat{\text{Eval}}_i &= \Big[4.053 + 0.0935\Big] + 0.133(\text{beauty rating}_i) - 0.205(\text{female}_i) - 0.359(\text{nonenglish}_i) \\ &= 4.15 + 0.133(\text{beauty rating}_i) - 0.205(\text{female}_i) - 0.359(\text{nonenglish}_i)\end{aligned}$$

If we write the professor-specific equation for Professor 2:

$$\begin{aligned}\hat{\text{Eval}}_i &= \Big[4.0534.053 - 0.2820\Big] + 0.133(\text{beauty rating}_i) - 0.205(\text{female}_i) - 0.359(\text{nonenglish}_i) \\ &= 3.77 + 0.133(\text{beauty rating}_i) - 0.205(\text{female}_i) - 0.359(\text{nonenglish}_i)\end{aligned}$$

Looking at these two equations gives us some insight into why the effects are referred to as fixed or random. The effects for each of the predictors in this model are fixed; we did not include a random-effect for any of them in the `lmer()` function. They have the exact same magnitude regardless of professor—they are fixed. We included a random-effect for intercept. In the two professor-specific equations the intercepts are different. They vary by professor—they are random.

References

Gelman, A., & Hill, J. (2007). *Data analysis using regression and multilevel/hierarchical models*. New York: Cambridge University Press.

Hamermesh, D. S., & Parker, A. M. (2005). Beauty in the classroom: Instructors' pulchritude and putative pedagogical productivity. *Economics of Education Review*, *24*, 369–376.

Woltman, H., Feldstein, A., MacKay, J. C., & Rocchi, M. (2012). An introduction to hierarchical linear modeling. *Tutorials in Quantitative Methods for Psychology*, *8*(1), 52–69.