1 Useful Packages & Dataset

Packages: {parameters}, {marginaleffects}, {performance}, {see}

Dataset used here: https://vincentarelbundock.github.io/Rdatasets/csv/openintro/evals.csv

2 Fitting Linear Regression

Simple linear regression	`lm(y ~ x)`
Multiple linear regression	`lm(y ~ x + w + z)`
Regression with interaction	`lm(y ~ x * w)`
Regression with interaction (alt.)	`lm(y ~ x + w + x:w)`
Polynomial regression	`lm(y ~ poly(x, 2))`
Polynomial regression (alt.)	`lm(y ~ x + x^2)`

Data Source: If your variables are nested inside a dataframe, you need to also provide `data` argument. Example: `lm(y ~ x, data = data)`.

Storing Output: You can assign a name to your model to store it for later. Example: `model <- lm(y ~ x, data = data)`.

3 Categorical Predictors

Categorical predictors have to be encoded into numerical variables before fitting a regression model. The most common type of encoding is *dummy encoding*.

Teacher position
Teaching
Nontenured
Tenured
Tenured

→

Teaching	Nontenured	Tenured
1	0	0
0	1	0
0	0	1
0	0	1

Dummy variable trap: Adding all dummy variables + the intercept into a model results in one more parameter than needed. Either drop one of the dummy categories or the intercept.

Dummy encoding in R: R will encode factor/character variables into dummy variables and drop the unnecessary parameter for you, you don't have to worry about anything.

4 Regression Coefficients

Data from study on relationship between teacher attractiveness, gender and course evaluations by students. Fit the model first:

```
m1 <- lm(score ~ bty_avg + gender)
```

Extract model coefficients:

Base R version	`coef(m1)`
Base R prettier table	`summary(m1)`
Pretty table & advanced options	`parameters::parameters(m1)`

Results:

Parameter	Value
Intercept	3.75
bty_avg	0.07
gender [male]	0.17

Interpretation: The intercept represents expected value of the dependent variable when all predictors are set to zero. Example: The expected course rating of a female teacher with attractiveness score of zero is 3.75 points.

Other parameters represent expected change in the dependent variable associated with the change in predictor variable (while controlling for other predictors). Example: teachers who have one unit higher attractiveness score (bty_avg) are expected to have 0.07 points higher course rating (controlling for gender). Male teachers are expected to have 0.17 higher course rating than female teachers (controlling for attractiveness score).

Controlling for variables means comparing expected values of dependent variable for different values of one predictor, while fixing other predictors at a specific value (usually zero). Example: Male teachers are expected to have 0.17 higher course rating than female teachers, *when comparing teachers with the same (zero) attractiveness score*.

5 Interactions

Interaction allows relationship between outcome and a predictor to change depending on the value of third variable. Mathematically, it's the product of two predictors:

```
m2 <- lm(score ~ bty_avg * gender)
```

Results:

Parameter	Value
Intercept	3.95
bty_avg	0.03
gender [male]	-0.18
bty_avg × gender [male]	0.08

Interpretation: The relationship between attractiveness score and course rating for reference gender group (female) is 0.03; female teachers is one unit higher attractiveness score are expected to have 0.03 points higher course ratings.

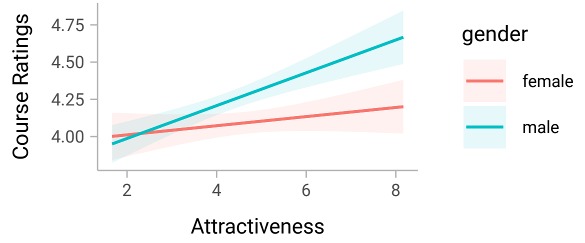
For male teachers, relationship strength between attractiveness and course rating is stronger. The relationship is 0.11 (= 0.03 + 0.08), i.e. one unit increase in attractiveness is associated with 0.11 point increase in course rating.

The difference in relationship strength between groups is 0.08, as represented by the regression coefficient.

Interaction ambiguity: Models don't understand which interaction variable is the "moderator". To our model, stating *"relationship between attractiveness and course ratings depends on gender"* is the same as stating *"relationship between gender and course ratings depends on attractiveness"*. Which interpretation is preferable depends on context.

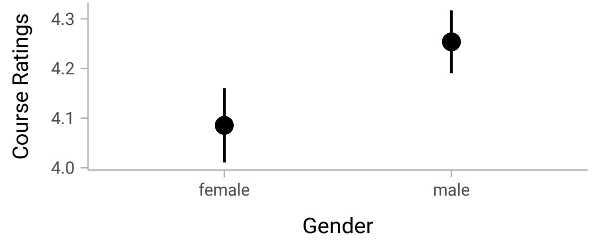
6 Model Visualization

```
marginaleffects::plot_predictions(m2, condition = c("bty_avg", "gender"))
```



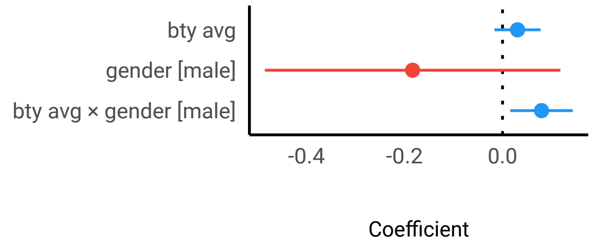
You can plot up to three predictors at once, the rest is controlled for. Example: Expected means for course ratings based on gender, *while controlling for attractiveness*:

```
plot_predictions(m2, condition = "gender")
```



You can also plot model coefficients:

```
parameters::parameters(m2) |> see::plot()
```



{ggplot2} all the way down: Both `{marginaleffects}` and `{parameters}` use `{ggplot2}` for plotting. You can modify the plots the way you are used to. Example:

```
plot_predictions(m1, condition = "gender") + theme_grey()
```

7 Expected (Predicted) Values

Expected (predicted) values for specific predictor, while controlling for the rest, can be obtained using `{marginaleffects}`:

```
avg_predictions(m1, variables = "gender")
```

gender	Estimate	SE	z	Pr(>|z|)	S	2.5%	97.5%
female	4.09	0.038	107	<0.001	Inf	4.01	4.16
male	4.25	0.0323	131	<0.001	Inf	4.19	4.32

The expected course rating for female teachers is 4.09 points. For male teachers, the expected rating is 4.25 points. Both when controlling (averaging over) attractiveness.

You can get predicted values for representative values of numerical predictors as well:

```
avg_predictions(m1, variables = "bty_avg")
```

You can also get predicted values for specific predictor values. First line below returns expected course ratings for respondents with attractiveness scores of −1, 0 and 1, *averaged over both genders*. Second line computes expected ratings for specified attractiveness scores for all unique gender categories (male and female):

```
predictions(m1, newdata = datagrid(bty_avg = c(-1,0,1)))

predictions(m1, newdata = datagrid(bty_avg = c(-1,0,1),
                                   gender  = unique))
```

8 Marginal effects

Average marginal effects (AMEs): Average relationship between variables. Example:

```
marginaleffects::avg_slopes(m2)

Term       Estimate
bty_avg    0.0767
gender     0.1682
```

On average, teachers with one unit higher attractiveness scores have 0.077 higher course ratings (average over men and women). On average, male teachers have 0.17 higher course ratings (averaged over all observed attractiveness levels). Compare with plot in section 6.

Conditional average marginal effects (CATEs): Average relationship between variables by subpopulations:

```
marginaleffects::avg_slopes(m2, , variables = "bty_avg",
by = "gender")

Term       gender   Estimate
bty_avg    male     0.1103
gender     female   0.0306
```

On average, male teachers with one unit higher attractiveness scores have 0.11 points higher course ratings. On average, female teachers with one unit higher attractiveness scores have 0.03 points higher course ratings.

9 Model Fit

The basic measure of fit is *Coefficient of determination* (R^2).

```
performance::r2(m1, ci = 0.95)

       R2: 0.059 [0.022, 0.105]
  adj. R2: 0.055 [0.020, 0.099]
```

Interpretation: R^2 represents the proportion of dependent variable's variance predicted by our model. Example: Our model succesfully predicts 5.9% of course scores' variance.

Adjusted R^2: Simple R^2 increases whenever we add a new predictor. To compare models with different number of predictors, use *adjusted* R^2.

Predictive performance: (adj.) R^2 measures predictive power. You can't use it to find the best model if your goal is to describe relationship between variables (inference).

Comparing multiple models: You can compare multiple models using `performance::compare_performance(m1, m2)`.

10 Model Assumptions

Checking model assumptions can be done by using diagnostic plots.

Base R version	`plot(m1)`
Better Plots	`performance::check_model(m1)`

Linear regression assumptions in order of general importance:

1. Validity & Reliability: All variables are measured using valid and reliable instruments. Violation leads to biased coefficients.

2. Representativity: Sample is representative of population, either by design or model adjustment. Violation leads to biased coefficients.

3. Linearity & additivity: Relationship between dependent and independent variables can be (and are) modeled using a linear combination of predictors. Check by looking at residual vs fitted plot. Violation means coefficients and predicted values will be biased.

4. Independence: Residuals are independent, meaning there is no unaccounted relationship between observations. Can't be easily check, think about research design. Violation leads to biased standard errors.

5. Homogeneity: Variance of residuals is constant for all values of dependent variable. Check by looking residual vs fitted plot. Violation leads to biased standard errors.

6. Normality: Residuals are normally distributed. Check by QQ plot/ histogram. Violation leads to biased standard errors (in small samples) and biased prediction intervals.

11 Nonlinear Relationships

Can be used when linearity assumption is violated.

Simple polynomials: Can account for simple nonlinear relationships. Example: `lm(score ~ poly(bty_avg, 2))`.

Linear splines: Models relationships using multiple joined lines. Example: `lm(score ~ lspline::lspline(bty_avg, knots = c(4, 6))`.

Natural splines: Most flexible way to model nonlinear relationships. Example: `lm(score ~ splines::ns(bty_avg, df = 3))`.

Use plots and marginal effects to interpret models with nonlinear relationships.

12 Heteroscedasticity & Dependence

Robust standard errors: Can be used when homogeneity assumption is violated. Estimate residual variance for every level of dependent variable instead of assuming it's constant.

Coefficients	`parameters(m1, vcov = "HC3")`
Marginal effects	`avg_slopes(m1, vcov = "HC3")`
Predictions	`avg_predictions(m1, vcov = "HC3")`

Robust Errors come in multiple flavors (HC0, HC1, HC2,...). Anything except for HC0 is ok.

Clustered errors: Can be used when independence assumptions is violated. Clustered errors also account for homoscedasticity.Example: Account for the fact that scores for courses taught by the same teacher are correlated:

```
avg_slopes(m1, variables = "bty_avg", vcov = ~prof_id)
```

13 Non-normality

Bootstrapping can be used to account for non-normal residuals (and heteroscedasticity):

```
parameters(model2, bootstrap = TRUE, iterations = 1000)

avg_slopes(model2) %>%
  inferences(method = "boot", R = 1000)
```

Bootstrapping is simulation based technique. The more simulations you do, the more reliable the results will be, but the longer you'll need to wait for results! Use at least 1000.

Sampling seed: You'll get slightly different results each time you do boostrapping. To fix results, use `set.seed()`.