Adding Interaction terms
Multiple linear regression – Stat 230

In this class, you will extend the multiple linear regression model to include allow for the slope of one variable to depend on the value of another variable. You will consider how to do this, how to interpret the results, and how to determine whether the addition of an interaction term discernibly improves the model (i.e., explains discernibly more variability in the response).

In this activity we will begin by working with the `4_8Cyl` data set described in Sections 3.10. You can load the data with the following code

```
cars <- read.csv("https://aloy.rbind.io/kuiper_data/4_8Cyl.csv")
```

Fitting an interaction model

Task 1. Fit the following multiple linear regression model and report the fitted regression equation. Is this an example of the parallel line, different slope, or separate line model?

$$\mu(\texttt{Price}|\texttt{Mileage}, \texttt{Cyl}) = \beta_0 + \beta_1\texttt{Mileage} + \beta_2\texttt{Cyl}$$

Task 2. Fit the following multiple linear regression model and report the fitted regression equation. Is this an example of the parallel line, different slope, or separate line model?

$$\mu(\texttt{Price}|\texttt{Mileage}, \texttt{Cyl}) = \beta_0 + \beta_1\texttt{Mileage} + \beta_2\texttt{Cyl} + \beta_3\texttt{Mileage} \times \texttt{Cyl}$$

> **i** Interaction terms in R
>
> To add interaction terms in R you multiply the terms together in the model formula. For example, the above model formula would is `Price ~ Mileage + Cyl + Mileage*Cyl`. Additionally, R follows the conventional modeling advice to always include the main effects for all variables in an interaction term. Thus, `Price ~ Mileage*Cyl` implements the same model ad the previous formula.

Task 3. For both models, calculate the estimated price of a four-cylinder car with `Mileage` = 10000.

Task 4. Assuming `Mileage` = 10000, for both models explain how increasing from four to eight cylinders will impact the estimated price.

Plotting an interaction model

Task 5. Last class we used `plotModel()` to plot the fitted model with categorical variables, but we saw that it was rather restrictive and error prone. Today, we'll use an alternative approach.

- First, we'll create a function in R to represent the mean function of the regression model.
- Then, we'll use our standard plotting functions from {ggformula} to create the plot.

```
model1 <- makeFun(___) # fill in blank with Task 1 model
gf_point(Price ~ Mileage, color = ~factor(Cyl), data = cars) |>
  gf_function(model1, args = list(Cyl = 4), color = ~"4") |>
  gf_function(model1, args = list(Cyl = 5), color = ~"8")
```

Notice that we add two layers to our plot via `gf_function`: one for each line. In order to draw a fitted multiple linear regression model in two dimensions, you need to set the values of the predictor variables that are *not* on the x-axis. Here, we set `Cyl = 4` in the first layer by passing `args` a list of the variable name and value pairs. Further, we map the line color to the level of `Cyl`.

Use the above plotting code to plot the fitted models from Tasks 1 and 2.

Task 6. For each model, write the fitted model equation for each group. In other words, write down equations for the lines you just plotted in the previous task.

Do we need the interaction?

Task 7. Conduct an extra sums of squares F-test to determine if the interaction term discernibly improves the model.