Models to Compare Two Population Means

In this class, you will work with your group to explore how to fit and interpret models to compare two population means. While you work through this activity, make sure that all group members are engage and contribute ideas and follow the code.

The R Manual has useful R code for today's activities.

Part 1: Understanding the study design

Before you begin analyzing the games data discussed in Chapter 2, you need to review the study design and consider how the design impacts the conclusions that you can draw.

Task 1: Review the study design on page 31 and complete activities 1-4.

Activity 4 asks you to create an "individual values plot." This is just a scatterplot with the response variable on the y-axis and the explanatory variable on the x-axis. To do this, load the data

```
games1 <- read.csv("https://aloy.rbind.io/kuiper_data/Games1.csv")
```

and then fill in the necessary blanks of the below code snippet:

```
gf_point(___ ~ ___, data = ___, xlab = "Game type", ylab= "Seconds", width = 0.1)
```

Part 2: Fitting a regression model to compare means

As you learned in preparation for this class, a simple linear regression model can be used in place of the two-sample procedure you just considered. The regression model will be of the form

$$y_i = \beta_0 + \beta_1 x_i + \varepsilon_i, \quad \text{for } i = 1, 2, \ldots, n \quad \text{where } \varepsilon_i \sim \mathcal{N}(0, \sigma^2)$$

Recall further that x_i will need to be an **indicator variable**.

Task 2: Complete activities 11-13. The R Manual has useful code.

To define an indicator variable in R, you can use the `==` logical operator. Below, `dummy1` is an indicator for the color distractor group.

```
dummy1 <- games1$Type == "Color"
```

`dummy1` is a vector of TRUEs and FALSEs. R interprets TRUE as 1 and FALSE as 0, but if you want to force the above logical vector into an integer vector, then you can do so

```
dummy1 <- as.integer(dummy1)
```

To fit a regression model with `dummy1` as the predictor, use the below snippet:

```
games_lm1 <- lm(Time ~ dummy1, data = games1)
```

Once you have a fitted model stored in R, you can use the `summary()` function to see the table of coefficients (needed for a hypothesis test) and `confint()` to construct confidence intervals.

Part 3: Check the model

Before you can trust your inferential results, you should verify that the necessary assumptions/conditions are met. That is you need to check that

1. the error terms (residuals) are i.i.d.
2. the error terms follow a normal distribution
3. the error terms have mean 0
4. the come from a single population with variance σ^2 (i.e., the variance doesn't depend on the value of x_i)

We typically rely on graphical tools to check these assumptions.

Data plot

You can assess some aspects of the error term distribution using a plot of the data (i.e., an individual value plot).

- Is the spread the same between groups? Is it drastically different?
- Are there extreme outliers (unusual observations)?

Residual plots

Most of the assumptions can be checked by estimating the error terms (i.e., calculating the residuals) and creating sensible plots.

- Histogram: could the residuals follow a normal distribution (are they bell shaped)?
- Histogram by group: how does the spread compare between groups? Are both groups centered around 0?
- Boxplots by group: is the spread (estimated by the IQR) about the same between groups?
- Residuals vs. time: is there any pattern in a plot of the residuals over time?

R instructions

To obtain the residuals from a regression model, we use the `resid()` function. Below we calculate the residuals from model `games_lm1`:

```
lm1_resid <- resid(games_lm1)
```

Now, you can calculate a histogram of the residuals

```
gf_histogram(~lm1_resid, data = games_lm1, bins = 15, xlab = "Residual")
```

and a scatterplot of the residuals vs. the explanatory variable (with a horizontal line at 0)

```
gf_point(lm1_resid ~ dummy1, xlab = "Color", ylab = "Residual") |>
  gf_hline(yintercept = ~0, color = "blue")
```

as in Figure 2.3. You can create a plot of the residuals vs. time order if you have a variable in the data set that records this time (or by assuming the observations were recorded in order). Here, we'll assume that `studentID` gives the order:

```
gf_point(lm1_resid ~ studentID, data = games1, xlab = "Order", ylab = "Residual") |>
  gf_line() |>
  gf_hline(yintercept = ~0, color = "blue")
```

Task 3: Complete activity 14.

Part 4: Communicate the results

If your model seems adequate, then you should clearly communicate your findings. This can be in the form of a graphic, an equation, or a couple sentences.

Task 4: Complete activity 15.

You can use the below code snippet to add the fitted regression line to the scatterplot. Notice that

```
gf_jitter(Time ~ dummy1, data = games1, xlab = "Color indicator", width = 0.05) |>
  gf_lm(Time ~ dummy1)
```

Part 5: Comparing the results to the two-sample t-test

As you saw before class, the underlying model for a two-sample t-test is given by

$$y_{ij} = \mu_i + \varepsilon_{ij}, \quad \text{for } i = 1, 2, \quad j = 1, \ldots, n_i, \quad \text{where } \varepsilon_{ij} \sim \mathcal{N}(0, \sigma^2)$$

Let group 1 be the color distractor group and group 2 be the standard game group.

Task 5. The sample means for the two groups are given below. Show how you can use these two sample means to calculate your estimate of the slope, $\hat{\beta}_1$ from the regression model in activity 11.

```
mean(Time ~ Type, data = games1)
```

```
Color Standard
38.10    35.55
```