{gtsummary}

What is `{gtsummary}`?

- Create tables that are publication-ready
- Highly customizable
- Descriptive tables, regression tables, etc.

gtsummary::tbl_summary()

```
1  library(gtsummary)
2
3  tbl_summary(
4    nlsy,
5    by = sex_cat,
6    include = c(sex_cat, race_et
7                eyesight_cat, gl
```

```r
tbl_summary(
  nlsy,
  by = sex_cat,
  include = c(sex_cat, race_eth_cat, region_c
              eyesight_cat, glasses, age_bir)
  label = list(
    race_eth_cat ~ "Race/ethnicity",
    region_cat ~ "Region",
    eyesight_cat ~ "Eyesight",
    glasses ~ "Wears glasses",
    age_bir ~ "Age at first birth"
  ),
  missing_text = "Missing")
```

```r
tbl_summary(
  nlsy,
  by = sex_cat,
  include = c(sex_cat, race_eth_cat,
              eyesight_cat, glasses, age_bir)
  label = list(
    race_eth_cat ~ "Race/ethnicity",
    eyesight_cat ~ "Eyesight",
    glasses ~ "Wears glasses",
    age_bir ~ "Age at first birth"
  ),
  missing_text = "Missing") |>
  add_p(test = list(all_continuous() ~ "t.tes
                    all_categorical() ~ "chis
  add_overall(col_label = "**Total**") |>
  bold_labels() |>
  modify_footnote(update = everything() ~ NA)
  modify_header(label = "**Variable**", p.val
```

`tbl_summary()`

- Incredibly customizeable
- Really helpful with Table 1
- I often just view in the web browser and copy and paste into a Word document
- Can also be used within quarto/R Markdown[1]
- If output is Word, I use `as_flex_table()`
- Make even more customizeable with `as_gt()`
  - Then can output to Word with `gt::as_word()`

1. This is where the bug comes in; soon to be fixed!

Univariate regressions

Fit a series of univariate regressions of income on other variables.

```
tbl_uvregression(
  nlsy,
  y = income,
  include = c(sex_cat, race_et
              eyesight_cat, in
  method = lm)
```

Can also do logistic regression

```
tbl_uvregression(
  nlsy,
  y = glasses,
  include = c(sex_cat, race_et
              eyesight_cat, gl
  method = glm,
  method.args = list(family =
  exponentiate = TRUE)
```

Customizable just like `tbl_summary()`

Some regressions

```r
linear_model <- lm(income ~ sex_cat + age_bir + race_eth_cat,
                   data = nlsy)
```

```r
linear_model_int <- lm(income ~ sex_cat*age_bir + race_eth_cat,
                   data = nlsy)
```

```r
logistic_model <- glm(glasses ~ eyesight_cat + sex_cat + income,
                      data = nlsy, family = binomial())
```

gtsummary::tbl_regression()

```
tbl_regression(
  linear_model,
  intercept = TRUE,
  label = list(
    sex_cat ~ "Sex",
    race_eth_cat ~ "Race/ethnicity",
    age_bir ~ "Age at first birth"
  ))
```

```
tbl_regression(
  logistic_model,
  exponentiate = TRUE,
  label = list(
    sex_cat ~ "Sex",
    eyesight_cat ~ "Eyesight",
    income ~ "Income"
  ))
```

You could put several together

```
tbl_no_int <- tbl_regression(
  linear_model,
  intercept = TRUE,
  label = list(
    sex_cat ~ "Sex",
    race_eth_cat ~ "Race/ethnicity",
    age_bir ~ "Age at first birth"
  ))

tbl_int <- tbl_regression(
  linear_model_int,
  intercept = TRUE,
  label = list(
    sex_cat ~ "Sex"
```

You could put several together

```
1 tbl_merge(list(tbl_no_int, tbl_int),
2           tab_spanner = c("**Model 1**", "**Model 2**"))
```

Inline text

bonus: broom::tidy()

```
bind_rows(
  broom::tidy(linear_model, conf.int = TRUE),
  broom::tidy(linear_model_int, conf.int = TRUE),
  .id = "model"
) |>
  mutate(model = factor(model,
                        labels = c("main terms", "sex-age interact
  ggplot(aes(x = term, y = estimate,
             ymin = conf.low, ymax = conf.high,
             color = model)) +
  geom_point(position = position_dodge(width = .5)) +
  geom_errorbar(position = position_dodge(width = .5))
```