# Regression Tables with huxreg

*David Hugh-Jones*

*2019-03-26*

## Regression tables with `huxreg`

From version 0.2, huxtable includes the function `huxreg` to build a table of regressions.

`huxreg` can be called with a list of models. These models can be of any class which has a `tidy` method defined in the broom package. The method should return a list of regression coefficients with names `term`, `estimate`, `std.error` and `p.value`. That covers most standard regression packages.

Let's start by running some regressions to predict a diamond's price.

```
data(diamonds, package = 'ggplot2')

lm1 <- lm(price ~ carat + depth, diamonds)
lm2 <- lm(price ~ depth + factor(color, ordered = FALSE), diamonds)
lm3 <- lm(log(price) ~ carat + depth, diamonds)
```

Now, we call `huxreg` to display the regression output side by side.

```
huxreg(lm1, lm2, lm3)
```

| | (1) | (2) | (3) |
|---|---|---|---|
| (Intercept) | 4045.333 *** | 6491.466 *** | 7.313 *** |
| | (286.205) | (730.537) | (0.074) |
| carat | 7765.141 *** | | 1.971 *** |
| | (14.009) | | (0.004) |
| depth | -102.165 *** | -53.835 *** | -0.018 *** |
| | (4.635) | (11.815) | (0.001) |
| factor(color, ordered = FALSE)E | | -95.142 | |
| | | (62.037) | |
| factor(color, ordered = FALSE)F | | 554.742 *** | |
| | | (62.374) | |
| factor(color, ordered = FALSE)G | | 832.357 *** | |
| | | (60.338) | |
| factor(color, ordered = FALSE)H | | 1324.183 *** | |
| | | (64.296) | |
| factor(color, ordered = FALSE)I | | 1929.902 *** | |
| | | (71.561) | |
| factor(color, ordered = FALSE)J | | 2164.044 *** | |
| | | (88.144) | |
| N | 53940 | 53940 | 53940 |
| R2 | 0.851 | 0.032 | 0.847 |
| logLik | -472488.441 | -522908.139 | -26617.649 |
| AIC | 944984.882 | 1045834.277 | 53243.298 |

*** p < 0.001; ** p < 0.01; * p < 0.05.

The basic output includes estimates, standard errors and summary statistics.

Some of those variable names are hard to read. We can change them by specifying a named list of variables in the `coefs` argument, like this:

```
color_names <- paste0('factor(color, ordered = FALSE)', LETTERS[5:10])
names(color_names) <- paste('Color:', LETTERS[5:10])

huxreg(lm1, lm2, lm3, coefs = c('Carat' = 'carat', 'Depth' = 'depth', color_names))
```

| | (1) | (2) | (3) |
|---|---|---|---|
| Carat | 7765.141 *** | | 1.971 *** |
| | (14.009) | | (0.004) |
| Depth | -102.165 *** | -53.835 *** | -0.018 *** |
| | (4.635) | (11.815) | (0.001) |
| Color: E | | -95.142 | |
| | | (62.037) | |
| Color: F | | 554.742 *** | |
| | | (62.374) | |
| Color: G | | 832.357 *** | |
| | | (60.338) | |
| Color: H | | 1324.183 *** | |
| | | (64.296) | |
| Color: I | | 1929.902 *** | |
| | | (71.561) | |
| Color: J | | 2164.044 *** | |
| | | (88.144) | |
| N | 53940 | 53940 | 53940 |
| R2 | 0.851 | 0.032 | 0.847 |
| logLik | -472488.441 | -522908.139 | -26617.649 |
| AIC | 944984.882 | 1045834.277 | 53243.298 |

*** p < 0.001; ** p < 0.01; * p < 0.05.

Alternatively, since the output from `huxreg` is just a huxtable, we could just edit its contents directly before we print it:

```
diamond_regs <- huxreg(lm1, lm2, lm3)
diamond_regs[seq(8, 18, 2), 1] <- paste('Color:', LETTERS[5:10])
diamond_regs
```

| | (1) | (2) | (3) |
|---|---|---|---|
| (Intercept) | 4045.333 *** | 6491.466 *** | 7.313 *** |
| | (286.205) | (730.537) | (0.074) |
| carat | 7765.141 *** | | 1.971 *** |
| | (14.009) | | (0.004) |
| depth | -102.165 *** | -53.835 *** | -0.018 *** |
| | (4.635) | (11.815) | (0.001) |
| Color: E | | -95.142 | |
| | | (62.037) | |
| Color: F | | 554.742 *** | |
| | | (62.374) | |
| Color: G | | 832.357 *** | |
| | | (60.338) | |
| Color: H | | 1324.183 *** | |
| | | (64.296) | |
| Color: I | | 1929.902 *** | |
| | | (71.561) | |
| Color: J | | 2164.044 *** | |
| | | (88.144) | |
| N | 53940 | 53940 | 53940 |
| R2 | 0.851 | 0.032 | 0.847 |
| logLik | -472488.441 | -522908.139 | -26617.649 |
| AIC | 944984.882 | 1045834.277 | 53243.298 |

*** p < 0.001; ** p < 0.01; * p < 0.05.

Of course, we aren't limited to just changing names. We can also make our table prettier. Let's add the "article" theme, and a vertical stripe for background colour, tweak a few details like font size, and add a caption. All of these are just standard huxtable commands.

```
suppressPackageStartupMessages(library(dplyr))
diamond_regs                                                    %>%
      theme_article                                             %>%
      set_background_color(1:nrow(diamond_regs), evens, grey(.95)) %>%
      set_font_size(final(), 1, 9)                              %>%
      set_bold(final(), 1, FALSE)                               %>%
      set_top_border(final(), 1, 1)                             %>%
      set_caption('Linear regressions of diamond prices')
```

Table 1: Linear regressions of diamond prices

| | (1) | (2) | (3) |
|---|---|---|---|
| **(Intercept)** | 4045.333 *** | 6491.466 *** | 7.313 *** |
| | (286.205) | (730.537) | (0.074) |
| **carat** | 7765.141 *** | | 1.971 *** |
| | (14.009) | | (0.004) |
| **depth** | -102.165 *** | -53.835 *** | -0.018 *** |
| | (4.635) | (11.815) | (0.001) |
| **Color: E** | | -95.142 | |
| | | (62.037) | |
| **Color: F** | | 554.742 *** | |
| | | (62.374) | |
| **Color: G** | | 832.357 *** | |
| | | (60.338) | |
| **Color: H** | | 1324.183 *** | |
| | | (64.296) | |
| **Color: I** | | 1929.902 *** | |
| | | (71.561) | |
| **Color: J** | | 2164.044 *** | |
| | | (88.144) | |
| **N** | 53940 | 53940 | 53940 |
| **R2** | 0.851 | 0.032 | 0.847 |
| **logLik** | -472488.441 | -522908.139 | -26617.649 |
| **AIC** | 944984.882 | 1045834.277 | 53243.298 |

*** p < 0.001; ** p < 0.01; * p < 0.05.

By default, standard errors are shown below coefficient estimates. To display them in a column to the right, use `error_pos = 'right'`:

```
huxreg(lm1, lm3, error_pos = 'right')
```

| | (1) | | (2) | |
|---|---|---|---|---|
| (Intercept) | 4045.333 *** | (286.205) | 7.313 *** | (0.074) |
| carat | 7765.141 *** | (14.009) | 1.971 *** | (0.004) |
| depth | -102.165 *** | (4.635) | -0.018 *** | (0.001) |
| N | 53940 | | 53940 | |
| R2 | 0.851 | | 0.847 | |
| logLik | -472488.441 | | -26617.649 | |
| AIC | 944984.882 | | 53243.298 | |

*** p < 0.001; ** p < 0.01; * p < 0.05.

This will give column headings a column span of 2.

To display standard errors in the same cell as estimates, use `error_pos = 'same'`:

```
huxreg(lm1, lm3, error_pos = 'same')
```

| | (1) | (2) |
|---|---|---|
| (Intercept) | 4045.333 *** (286.205) | 7.313 *** (0.074) |
| carat | 7765.141 *** (14.009) | 1.971 *** (0.004) |
| depth | -102.165 *** (4.635) | -0.018 *** (0.001) |
| N | 53940 | 53940 |
| R2 | 0.851 | 0.847 |
| logLik | -472488.441 | -26617.649 |
| AIC | 944984.882 | 53243.298 |

*** p < 0.001; ** p < 0.01; * p < 0.05.

You can change the default column headings by giving names to your models:

```
huxreg('Price' = lm1, 'Log price' = lm3)
```

| | Price | Log price |
|---|---|---|
| (Intercept) | 4045.333 *** | 7.313 *** |
| | (286.205) | (0.074) |
| carat | 7765.141 *** | 1.971 *** |
| | (14.009) | (0.004) |
| depth | -102.165 *** | -0.018 *** |
| | (4.635) | (0.001) |
| N | 53940 | 53940 |
| R2 | 0.851 | 0.847 |
| logLik | -472488.441 | -26617.649 |
| AIC | 944984.882 | 53243.298 |

*** p < 0.001; ** p < 0.01; * p < 0.05.

To display a particular row of summary statistics, use the `statistics` parameter. This should be a character vector. Valid values are anything returned from your models by `broom::glance`. Another valid value is `"nobs"`, which returns the number of observations from the regression. If the `statistics` vector has names, these will be used for row headings:

```
broom::glance(lm1)
```

```
## # A tibble: 1 x 11
##   r.squared adj.r.squared sigma statistic p.value    df  logLik    AIC
##       <dbl>         <dbl> <dbl>     <dbl>   <dbl> <int>   <dbl>  <dbl>
## 1     0.851         0.851 1542.   153635.       0     3 -4.72e5 9.45e5
## # ... with 3 more variables: BIC <dbl>, deviance <dbl>, df.residual <int>
```

```
huxreg(lm1, lm3, statistics = c('# observations' = 'nobs', 'R squared' = 'r.squared', 'F statistic' = '
  'P value' = 'p.value'))
```

| | (1) | (2) |
|---|---|---|
| (Intercept) | 4045.333 *** | 7.313 *** |
| | (286.205) | (0.074) |
| carat | 7765.141 *** | 1.971 *** |
| | (14.009) | (0.004) |
| depth | -102.165 *** | -0.018 *** |
| | (4.635) | (0.001) |
| # observations | 53940 | 53940 |
| R squared | 0.851 | 0.847 |
| F statistic | 153634.765 | 149771.327 |
| P value | 0.000 | 0.000 |

*** p < 0.001; ** p < 0.01; * p < 0.05.

By default, `huxreg` displays significance stars. You can alter the symbols used and significance levels with the `stars` parameter, or set `stars = NULL` to turn off significance stars completely.

```
huxreg(lm1, lm3, stars = c(`*` = 0.1, `**` = 0.05, `***` = 0.01)) # a little boastful?
```

| | (1) | (2) |
|---|---|---|
| (Intercept) | 4045.333 *** | 7.313 *** |
| | (286.205) | (0.074) |
| carat | 7765.141 *** | 1.971 *** |
| | (14.009) | (0.004) |
| depth | -102.165 *** | -0.018 *** |
| | (4.635) | (0.001) |
| N | 53940 | 53940 |
| R2 | 0.851 | 0.847 |
| logLik | -472488.441 | -26617.649 |
| AIC | 944984.882 | 53243.298 |

*** p < 0.01; ** p < 0.05; * p < 0.1.

```
huxreg(lm1, lm3, stars = NULL)
```

| | (1) | (2) |
|---|---|---|
| (Intercept) | 4045.333 | 7.313 |
| | (286.205) | (0.074) |
| carat | 7765.141 | 1.971 |
| | (14.009) | (0.004) |
| depth | -102.165 | -0.018 |
| | (4.635) | (0.001) |
| N | 53940 | 53940 |
| R2 | 0.851 | 0.847 |
| logLik | -472488.441 | -26617.649 |
| AIC | 944984.882 | 53243.298 |

You aren't limited to displaying standard errors of the estimates. If you prefer, you can display t statistics or p values, using the `error_format` option. Any column from `tidy` can be used by putting it in curly brackets:

```
huxreg(lm1, lm3, error_format = '({statistic})')
```

| | (1) | (2) |
|---|---|---|
| (Intercept) | 4045.333 *** | 7.313 *** |
| | (14.134) | (99.383) |
| carat | 7765.141 *** | 1.971 *** |
| | (554.282) | (547.305) |
| depth | -102.165 *** | -0.018 *** |
| | (-22.041) | (-14.936) |
| N | 53940 | 53940 |
| R2 | 0.851 | 0.847 |
| logLik | -472488.441 | -26617.649 |
| AIC | 944984.882 | 53243.298 |

*** p < 0.001; ** p < 0.01; * p < 0.05.

```
huxreg(lm1, lm3, error_format = '({p.value})')
```

| | (1) | (2) |
|---|---|---|
| (Intercept) | 4045.333 *** | 7.313 *** |
| | (0.000) | (0.000) |
| carat | 7765.141 *** | 1.971 *** |
| | (0.000) | (0.000) |
| depth | -102.165 *** | -0.018 *** |
| | (0.000) | (0.000) |
| N | 53940 | 53940 |
| R2 | 0.851 | 0.847 |
| logLik | -472488.441 | -26617.649 |
| AIC | 944984.882 | 53243.298 |

*** p < 0.001; ** p < 0.01; * p < 0.05.

Or you can display confidence intervals. Use `ci_level` to set the confidence level for the interval, then use `{conf.low}` and `{conf.high}` in `error_format`:

```
huxreg(lm1, lm3, ci_level = .99, error_format = '{conf.low} to {conf.high}')
```

| | (1) | (2) |
|---|---|---|
| (Intercept) | 4045.333 *** | 7.313 *** |
| | 3308.091 to 4782.576 | 7.123 to 7.502 |
| carat | 7765.141 *** | 1.971 *** |
| | 7729.054 to 7801.228 | 1.962 to 1.981 |
| depth | -102.165 *** | -0.018 *** |
| | -114.105 to -90.225 | -0.021 to -0.015 |
| N | 53940 | 53940 |
| R2 | 0.851 | 0.847 |
| logLik | -472488.441 | -26617.649 |
| AIC | 944984.882 | 53243.298 |

*** p < 0.001; ** p < 0.01; * p < 0.05.

To change the footnote, use `note`. If `note` contains the string `"{stars}"` it will be replaced by a description of the significance stars used. If you don't want a footnote, just set `note = NULL`.

```
huxreg(lm1, lm3, note = 'Linear regressions on diamond price. {stars}.')
```

| | (1) | (2) |
|---|---|---|
| (Intercept) | 4045.333 *** | 7.313 *** |
| | (286.205) | (0.074) |
| carat | 7765.141 *** | 1.971 *** |
| | (14.009) | (0.004) |
| depth | -102.165 *** | -0.018 *** |
| | (4.635) | (0.001) |
| N | 53940 | 53940 |
| R2 | 0.851 | 0.847 |
| logLik | -472488.441 | -26617.649 |
| AIC | 944984.882 | 53243.298 |

Linear regressions on diamond price. *** p < 0.001; ** p < 0.01; * p < 0.05.

To change number formatting, set the `number_format` parameter. This works the same as the `number_format` property for a huxtable - if it is numeric, numbers will be rounded to that many decimal places; if it is character, it will be taken as a format to the base R `sprintf` function. `huxreg` tries to be smart and to format summary statistics like `nobs` as integers.

```
huxreg(lm1, lm3, number_format = 2)
```

| | (1) | (2) |
|---|---|---|
| (Intercept) | 4045.33 *** | 7.31 *** |
| | (286.21) | (0.07) |
| carat | 7765.14 *** | 1.97 *** |
| | (14.01) | (0.00) |
| depth | -102.17 *** | -0.02 *** |
| | (4.64) | (0.00) |
| N | 53940 | 53940 |
| R2 | 0.85 | 0.85 |
| logLik | -472488.44 | -26617.65 |
| AIC | 944984.88 | 53243.30 |

*** p < 0.001; ** p < 0.01; * p < 0.05.

Lastly, if you want to bold all significant coefficients, set the parameter `bold_signif` to a maximum significance level:

```
huxreg(lm1, lm3, bold_signif = 0.05)
```

| | (1) | (2) |
|---|---|---|
| (Intercept) | **4045.333 *** ** | **7.313 *** ** |
| | **(286.205)** | **(0.074)** |
| carat | **7765.141 *** ** | **1.971 *** ** |
| | **(14.009)** | **(0.004)** |
| depth | **-102.165 *** ** | **-0.018 *** ** |
| | **(4.635)** | **(0.001)** |
| N | 53940 | 53940 |
| R2 | 0.851 | 0.847 |
| logLik | -472488.441 | -26617.649 |
| AIC | 944984.882 | 53243.298 |

*** p < 0.001; ** p < 0.01; * p < 0.05.

## Altering data

Sometimes, you want to report different statistics for a model. For example, you might want to use robust standard errors.

One way to do this is to pass a `tidy`-able test object into `huxreg`. The function `coeftest` in the "lmtest" package has `tidy` methods defined:

```
library(lmtest)
library(sandwich)
lm_robust <- coeftest(lm1, vcov = vcovHC)
huxreg("Normal SEs" = lm1, "Robust SEs" = lm_robust)
```

```
## Warning in FUN(X[[i]], ...): No `glance` method for model of class coeftest
```

| | Normal SEs | Robust SEs |
|---|---|---|
| (Intercept) | 4045.333 *** | 4045.333 *** |
| | (286.205) | (369.327) |
| carat | 7765.141 *** | 7765.141 *** |
| | (14.009) | (25.114) |
| depth | -102.165 *** | -102.165 *** |
| | (4.635) | (5.948) |
| N | 53940 | |
| R2 | 0.851 | |
| logLik | -472488.441 | |
| AIC | 944984.882 | |

*** p < 0.001; ** p < 0.01; * p < 0.05.

If that is not possible, you can compute statistics yourself and add them to your model using the `tidy_override` function:

```
lm_fixed <- tidy_override(lm1, p.value = c(0.5, 0.2, 0.06))
huxreg("Normal p values" = lm1, "Supplied p values" = lm_fixed)
```

| | Normal p values | Supplied p values |
|---|---|---|
| (Intercept) | 4045.333 *** | 4045.333 |
| | (286.205) | (286.205) |
| carat | 7765.141 *** | 7765.141 |
| | (14.009) | (14.009) |
| depth | -102.165 *** | -102.165 |
| | (4.635) | (4.635) |
| N | 53940 | 53940 |
| R2 | 0.851 | 0.851 |
| logLik | -472488.441 | -472488.441 |
| AIC | 944984.882 | 944984.882 |

*** p < 0.001; ** p < 0.01; * p < 0.05.

You can override any statistics returned by `tidy` or `glance`.