GTSummmary Presentation

Wyatt Tarter

## GTSummary

Gtsummary is a relatively newish table package with great breadth of features that works with the pipe operator and various other table packages (flextable, kableExtra, knitr, huxtable, specifically builds upon the gt package).

## Syntax

Several base functions that create default tables with minimal input

* tbl\_summary()
  + Great for quick, fast, nice looking table 1’s that can be edited to your hearts content
* tbl\_regression()
  + Creates quick, fast, nice looking tables to summarize regression models
  + Also supports survival models, and (admittedly limited support) Bayesian models from the rstanarm and brms packages
* tbl\_uvregression()
  + Creates quick, fast (noticing a theme here?) tables for univariate analysis

## Tbl\_summary Example

tbl\_summary(mtcars)

| Characteristic | N = 321 |
| --- | --- |
| mpg | 19.2 (15.4, 22.8) |
| cyl |  |
| 4 | 11 (34%) |
| 6 | 7 (22%) |
| 8 | 14 (44%) |
| disp | 196 (121, 326) |
| hp | 123 (96, 180) |
| drat | 3.70 (3.08, 3.92) |
| wt | 3.33 (2.58, 3.61) |
| qsec | 17.71 (16.89, 18.90) |
| vs | 14 (44%) |
| am | 13 (41%) |
| gear |  |
| 3 | 15 (47%) |
| 4 | 12 (38%) |
| 5 | 5 (16%) |
| carb |  |
| 1 | 7 (22%) |
| 2 | 10 (31%) |
| 3 | 3 (9.4%) |
| 4 | 10 (31%) |
| 6 | 1 (3.1%) |
| 8 | 1 (3.1%) |
| 1Median (IQR); n (%) | |

Can be output to html, pdf (allegedly), or word document interchangeably, with no extra lines of code needed.

## Tbl\_summary split example

To create a table one split by a variable, simply specify the by = option within the tbl\_summary() call:

tbl\_summary(mtcars, by = cyl)

| Characteristic | 4, N = 111 | 6, N = 71 | 8, N = 141 |
| --- | --- | --- | --- |
| mpg | 26.0 (22.8, 30.4) | 19.7 (18.6, 21.0) | 15.2 (14.4, 16.2) |
| disp | 108 (79, 121) | 168 (160, 196) | 350 (302, 390) |
| hp | 91 (66, 96) | 110 (110, 123) | 192 (176, 241) |
| drat | 4.08 (3.81, 4.16) | 3.90 (3.35, 3.91) | 3.12 (3.07, 3.22) |
| wt | 2.20 (1.89, 2.62) | 3.21 (2.82, 3.44) | 3.76 (3.53, 4.01) |
| qsec | 18.90 (18.56, 19.95) | 18.30 (16.74, 19.17) | 17.18 (16.10, 17.56) |
| vs | 10 (91%) | 4 (57%) | 0 (0%) |
| am | 8 (73%) | 3 (43%) | 2 (14%) |
| gear |  |  |  |
| 3 | 1 (9.1%) | 2 (29%) | 12 (86%) |
| 4 | 8 (73%) | 4 (57%) | 0 (0%) |
| 5 | 2 (18%) | 1 (14%) | 2 (14%) |
| carb |  |  |  |
| 1 | 5 (45%) | 2 (29%) | 0 (0%) |
| 2 | 6 (55%) | 0 (0%) | 4 (29%) |
| 3 | 0 (0%) | 0 (0%) | 3 (21%) |
| 4 | 0 (0%) | 4 (57%) | 6 (43%) |
| 6 | 0 (0%) | 1 (14%) | 0 (0%) |
| 8 | 0 (0%) | 0 (0%) | 1 (7.1%) |
| 1Median (IQR); n (%) | | | |

## Tbl\_summary example with all the stuff

However, if you want to modify your table, you can use the pipe operator to add in all sorts of handy calls!

theme\_gtsummary\_eda(set\_theme = T)  
tbl\_summary(trial,   
 by = trt,  
 missing = "ifany",  
 missing\_text = "Missing",  
 label = grade ~ "Tumor Grade",  
 statistic = list(age~"{median} ({p25},{p75})",  
 marker ~ "{mean} ({sd})",  
 all\_categorical() ~ "{n}/{N}")) %>%  
 add\_p(pvalue\_fun = ~ style\_pvalue(.x, digits = 2)) %>%  
 add\_overall() %>%  
 add\_q(method = "fdr") %>%  
 modify\_header(label = "\*\*Variables\*\*") %>%  
 bold\_labels() %>%  
 italicize\_levels() %>%  
 modify\_caption("\*\*Table 1.\*\*")

\*\*Table 1.\*\*

| Variables | Overall, N = 2001 | Drug A, N = 981 | Drug B, N = 1021 | p-value2 | q-value3 |
| --- | --- | --- | --- | --- | --- |
| **Age** |  |  |  | 0.72 | 0.87 |
| *Median (25%,75%)* | 47 (38,57) | 46 (37,59) | 48 (39,56) |  |  |
| *Missing* | 11 | 7 | 4 |  |  |
| **Marker Level (ng/mL)** |  |  |  | 0.085 | 0.51 |
| *Mean (SD)* | 0.92 (0.86) | 1.02 (0.89) | 0.82 (0.83) |  |  |
| *Missing* | 10 | 6 | 4 |  |  |
| **T Stage** |  |  |  | 0.87 | 0.87 |
| *T1* | 53/200 | 28/98 | 25/102 |  |  |
| *T2* | 54/200 | 25/98 | 29/102 |  |  |
| *T3* | 43/200 | 22/98 | 21/102 |  |  |
| *T4* | 50/200 | 23/98 | 27/102 |  |  |
| **Tumor Grade** |  |  |  | 0.87 | 0.87 |
| *I* | 68/200 | 35/98 | 33/102 |  |  |
| *II* | 68/200 | 32/98 | 36/102 |  |  |
| *III* | 64/200 | 31/98 | 33/102 |  |  |
| **Tumor Response** | 61/193 | 28/95 | 33/98 | 0.53 | 0.87 |
| *Missing* | 7 | 3 | 4 |  |  |
| **Patient Died** | 112/200 | 52/98 | 60/102 | 0.41 | 0.87 |
| **Months to Death/Censor** |  |  |  | 0.14 | 0.51 |
| *Median (IQR)* | 22.4 (16.0, 24.0) | 23.5 (17.4, 24.0) | 21.2 (14.6, 24.0) |  |  |
| *Mean (SD)* | 19.6 (5.3) | 20.2 (5.0) | 19.0 (5.5) |  |  |
| *Range* | 3.5, 24.0 | 3.5, 24.0 | 5.3, 24.0 |  |  |
| 1n/N | | | | | |
| 2Wilcoxon rank sum test; Pearson's Chi-squared test | | | | | |
| 3False discovery rate correction for multiple testing | | | | | |

In the above chunk, we specify whether to show missing values, change the text for the missing values, specify what statistics to calculate for each individual variable or an entire subtype of variables, add p-values and the number of digits to display, add a multiple comparison adjusted p-value column, an overall column, modify the header, make labels bold or italicized, add an overall caption. We can also set a theme prior to the table (this is the exploratory data analysis theme).

## Tbl\_strata example with double stratification

Sometimes you may want to to a double stratified table (maybe split by gender, and within gender split by treatment). I frequently struggle with such tables in other table packages, and usually have to construct them by “hand”.

#| message: false  
  
tbl\_strata(trial,  
 strata = trt,  
 .tbl\_fun = ~.x %>%   
 tbl\_summary(by=stage, missing = "ifany") %>%  
 add\_p() %>%  
 bold\_p()  
)

Table printed with {flextable}, not {gt}. Learn why at  
https://www.danieldsjoberg.com/gtsummary/articles/rmarkdown.html  
To suppress this message, include `message = FALSE` in the code chunk header.

|  | Drug A | | | | | Drug B | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Characteristic | T1, N = 281 | T2, N = 251 | T3, N = 221 | T4, N = 231 | p-value2 | T1, N = 251 | T2, N = 291 | T3, N = 211 | T4, N = 271 | p-value2 |
| Age |  |  |  |  | 0.5 |  |  |  |  | 0.6 |
| Median (IQR) | 43 (31, 53) | 48 (42, 62) | 48 (38, 60) | 46 (36, 60) |  | 47 (43, 57) | 49 (42, 53) | 53 (40, 59) | 45 (38, 54) |  |
| Mean (SD) | 44 (15) | 50 (13) | 49 (14) | 45 (17) |  | 50 (14) | 46 (12) | 50 (15) | 44 (15) |  |
| Range | 19, 76 | 31, 74 | 20, 78 | 6, 75 |  | 25, 83 | 9, 68 | 19, 76 | 10, 71 |  |
| Unknown | 2 | 1 | 2 | 2 |  | 0 | 0 | 0 | 4 |  |
| Marker Level (ng/mL) |  |  |  |  | 0.3 |  |  |  |  | 0.5 |
| Median (IQR) | 0.56 (0.22, 0.98) | 1.00 (0.25, 1.90) | 1.04 (0.29, 1.71) | 0.80 (0.27, 1.88) |  | 0.39 (0.17, 0.89) | 0.59 (0.09, 1.63) | 1.06 (0.31, 1.31) | 0.45 (0.22, 1.14) |  |
| Mean (SD) | 0.74 (0.79) | 1.22 (1.00) | 1.06 (0.81) | 1.08 (0.91) |  | 0.66 (0.74) | 1.01 (1.12) | 0.94 (0.68) | 0.67 (0.58) |  |
| Range | 0.00, 3.75 | 0.08, 3.87 | 0.06, 2.52 | 0.04, 2.77 |  | 0.02, 2.64 | 0.01, 3.64 | 0.06, 2.14 | 0.06, 2.21 |  |
| Unknown | 2 | 1 | 2 | 1 |  | 2 | 0 | 2 | 0 |  |
| Grade |  |  |  |  | 0.2 |  |  |  |  | >0.9 |
| I | 8 (28.6%) | 8 (32.0%) | 11 (50.0%) | 8 (34.8%) |  | 9 (36.0%) | 10 (34.5%) | 7 (33.3%) | 7 (25.9%) |  |
| II | 14 (50.0%) | 8 (32.0%) | 5 (22.7%) | 5 (21.7%) |  | 9 (36.0%) | 9 (31.0%) | 6 (28.6%) | 12 (44.4%) |  |
| III | 6 (21.4%) | 9 (36.0%) | 6 (27.3%) | 10 (43.5%) |  | 7 (28.0%) | 10 (34.5%) | 8 (38.1%) | 8 (29.6%) |  |
| Tumor Response | 7 (25.0%) | 6 (25.0%) | 8 (40.0%) | 7 (30.4%) | 0.7 | 11 (45.8%) | 7 (25.0%) | 7 (35.0%) | 8 (30.8%) | 0.4 |
| Unknown | 0 | 1 | 2 | 0 |  | 1 | 1 | 1 | 1 |  |
| Patient Died | 12 (42.9%) | 11 (44.0%) | 13 (59.1%) | 16 (69.6%) | 0.2 | 12 (48.0%) | 16 (55.2%) | 9 (42.9%) | 23 (85.2%) | **0.010** |
| Months to Death/Censor |  |  |  |  | 0.2 |  |  |  |  | **0.004** |
| Median (IQR) | 24.0 (18.3, 24.0) | 24.0 (20.1, 24.0) | 21.2 (16.1, 24.0) | 19.8 (16.3, 24.0) |  | 24.0 (18.2, 24.0) | 20.4 (15.2, 24.0) | 24.0 (17.5, 24.0) | 15.6 (10.1, 21.4) |  |
| Mean (SD) | 20.8 (4.9) | 21.1 (4.7) | 19.4 (5.5) | 19.4 (4.9) |  | 20.9 (4.6) | 19.3 (5.2) | 20.3 (5.5) | 16.1 (5.6) |  |
| Range | 7.4, 24.0 | 7.3, 24.0 | 3.5, 24.0 | 10.1, 24.0 |  | 12.2, 24.0 | 10.1, 24.0 | 5.3, 24.0 | 6.3, 24.0 |  |
| 1n (%) | | | | | | | | | | |
| 2Kruskal-Wallis rank sum test; Pearson's Chi-squared test | | | | | | | | | | |

## Tbl\_uvregression example

If you ever need to run and summarize a handful of univariate regressions, there is a helpful tbl\_uvregression function:

tbl\_uvregression(data = trial,  
 method = glm,  
 y = death,  
 method.args = list(family = binomial),  
 exponentiate = T) %>%  
 bold\_p() %>%  
 modify\_caption(caption = "Univariate regressions with Death as Outcome")

Univariate regressions with Death as Outcome

| Characteristic | N | OR1 | 95% CI1 | p-value |
| --- | --- | --- | --- | --- |
| Chemotherapy Treatment | 200 |  |  |  |
| Drug A |  | — | — |  |
| Drug B |  | 1.26 | 0.72, 2.22 | 0.4 |
| Age | 189 | 1.01 | 0.99, 1.03 | 0.3 |
| Marker Level (ng/mL) | 190 | 0.89 | 0.64, 1.25 | 0.5 |
| T Stage | 200 |  |  |  |
| T1 |  | — | — |  |
| T2 |  | 1.21 | 0.57, 2.59 | 0.6 |
| T3 |  | 1.27 | 0.57, 2.85 | 0.6 |
| T4 |  | 4.28 | 1.85, 10.5 | **<0.001** |
| Grade | 200 |  |  |  |
| I |  | — | — |  |
| II |  | 1.19 | 0.61, 2.35 | 0.6 |
| III |  | 2.17 | 1.08, 4.45 | **0.031** |
| Tumor Response | 193 | 0.38 | 0.20, 0.71 | **0.003** |
| Months to Death/Censor | 200 | 0.00 | 0.00, 0.00 | >0.9 |
| 1OR = Odds Ratio, CI = Confidence Interval | | | | |

## Tbl\_regression example

The regression summaries is probably my favorite part: you can create your model and either pipe it directly into the tbl\_regression summary or call it later. Syntax and helper functions are generally the same as tbl\_summary():

tbl\_a<-glm(mpg~factor(cyl)+wt+factor(vs)+factor(am)+hp,  
 data=mtcars,  
 family = "gaussian") %>%  
 tbl\_regression(label = list(`factor(cyl)`~"Cylinder #",  
 wt~"Weight (1000 lbs)",  
 `factor(vs)`~"Engine type",  
 `factor(am)`~"Transmission",  
 hp~"Horsepower")) %>%  
 bold\_p(t=0.05)  
  
tbl\_a

Table printed with {flextable}, not {gt}. Learn why at  
https://www.danieldsjoberg.com/gtsummary/articles/rmarkdown.html  
To suppress this message, include `message = FALSE` in the code chunk header.

| Characteristic | Beta | 95% CI1 | p-value |
| --- | --- | --- | --- |
| Cylinder # |  |  |  |
| 4 | — | — |  |
| 6 | -2.1 | -5.3, 1.1 | 0.2 |
| 8 | 0.29 | -5.9, 6.5 | >0.9 |
| Weight (1000 lbs) | -2.4 | -4.1, -0.63 | **0.013** |
| Engine type |  |  |  |
| 0 | — | — |  |
| 1 | 2.0 | -1.5, 5.4 | 0.3 |
| Transmission |  |  |  |
| 0 | — | — |  |
| 1 | 2.7 | -0.43, 5.8 | 0.10 |
| Horsepower | -0.03 | -0.06, -0.01 | **0.019** |
| 1CI = Confidence Interval | | | |

Another nice feature is including results in-line with the text, which can be done using the r inline\_text(model, variable) command, like so: -2.4 (95% CI -4.1, -0.63; p=0.013) is the estimate for the weight variable.

## Tbl\_regression combining multiple models example

One of my most frequent frustrations is creating multiple regressions, and either having to present them separately, or attempting to create a terrible franken-table that looks abjectly horrible. The tbl\_merge() function is great since it lets you merge multiple regression tables very easily (or just multiple gtsummary table objects, regardless of type).

Below are two logistic models: one that examines the association between grade, stage, and age, and another that adds in treatment.

tbl\_1<-glm(death~grade+stage+age,  
 family="binomial",  
 data=trial) %>%  
 tbl\_regression(exponentiate=T) %>%  
 bold\_p()   
  
  
tbl\_2<-glm(death~grade+stage+age+trt,  
 family="binomial",  
 data=trial) %>%  
 tbl\_regression(exponentiate=T) %>%  
 bold\_p()   
  
tbl\_merge(tbls=list(tbl\_1,  
 tbl\_2),  
 tab\_spanner = c("No treatment model",  
 "Treatment model"))

Table printed with {flextable}, not {gt}. Learn why at  
https://www.danieldsjoberg.com/gtsummary/articles/rmarkdown.html  
To suppress this message, include `message = FALSE` in the code chunk header.

|  | No treatment model | | | Treatment model | | |
| --- | --- | --- | --- | --- | --- | --- |
| Characteristic | OR1 | 95% CI1 | p-value | OR1 | 95% CI1 | p-value |
| Grade |  |  |  |  |  |  |
| I | — | — |  | — | — |  |
| II | 1.21 | 0.58, 2.52 | 0.6 | 1.20 | 0.58, 2.52 | 0.6 |
| III | 2.24 | 1.07, 4.79 | **0.035** | 2.21 | 1.05, 4.74 | **0.038** |
| T Stage |  |  |  |  |  |  |
| T1 | — | — |  | — | — |  |
| T2 | 1.30 | 0.59, 2.88 | 0.5 | 1.28 | 0.58, 2.85 | 0.5 |
| T3 | 1.20 | 0.51, 2.84 | 0.7 | 1.20 | 0.51, 2.84 | 0.7 |
| T4 | 4.54 | 1.85, 11.9 | **0.001** | 4.52 | 1.84, 11.8 | **0.001** |
| Age | 1.01 | 0.99, 1.04 | 0.2 | 1.01 | 0.99, 1.04 | 0.2 |
| Chemotherapy Treatment |  |  |  |  |  |  |
| Drug A |  |  |  | — | — |  |
| Drug B |  |  |  | 1.34 | 0.73, 2.46 | 0.4 |
| 1OR = Odds Ratio, CI = Confidence Interval | | | | | | |

You can add in more models (within reason), and it automatically matches up the variables in each row, leaving any that are not in both models blank for the respective model. This also works with a tbl\_summary() object in a similar fashion.

We can also stack regression models (or just general gtsummary tables) in a similar fashion:

tbl\_3 <-  
 glm(response ~ trt, trial, family = binomial) %>%  
 tbl\_regression(  
 exponentiate = TRUE,  
 label = list(trt ~ "Treatment (unadjusted)")  
 ) %>%  
 bold\_labels()  
  
tbl\_4 <-  
 glm(response ~ trt + grade + stage + marker, trial, family = binomial) %>%  
 tbl\_regression(  
 include = "trt",  
 exponentiate = TRUE,  
 label = list(trt ~ "Treatment (adjusted)")  
 ) %>%  
 bold\_labels()  
  
tbl\_stack(list(tbl\_3, tbl\_4))

Table printed with {flextable}, not {gt}. Learn why at  
https://www.danieldsjoberg.com/gtsummary/articles/rmarkdown.html  
To suppress this message, include `message = FALSE` in the code chunk header.

| Characteristic | OR1 | 95% CI1 | p-value |
| --- | --- | --- | --- |
| **Treatment (unadjusted)** |  |  |  |
| Drug A | — | — |  |
| Drug B | 1.21 | 0.66, 2.24 | 0.5 |
| **Treatment (adjusted)** |  |  |  |
| Drug A | — | — |  |
| Drug B | 1.48 | 0.78, 2.86 | 0.2 |
| 1OR = Odds Ratio, CI = Confidence Interval | | | |

## Survival models

The gtsummary can also summarize survival models with the same syntax as tbl\_regression, but has a nice helper function for survival probabilities from a survfit object to be displayed as well:

tbl\_survfit(  
 survfit(Surv(ttdeath, death) ~ trt, trial),  
 times = c(6, 12,18, 24),  
 label\_header = "\*\*{time} Month\*\*"  
) %>%  
 modify\_caption("\*\*Surival probabilities based on treatment\*\*")

Table printed with {flextable}, not {gt}. Learn why at  
https://www.danieldsjoberg.com/gtsummary/articles/rmarkdown.html  
To suppress this message, include `message = FALSE` in the code chunk header.

\*\*Surival probabilities based on treatment\*\*

| Characteristic | 6 Month | 12 Month | 18 Month | 24 Month |
| --- | --- | --- | --- | --- |
| Chemotherapy Treatment |  |  |  |  |
| Drug A | 99% (97%, 100%) | 91% (85%, 97%) | 70% (62%, 80%) | 47% (38%, 58%) |
| Drug B | 99% (97%, 100%) | 86% (80%, 93%) | 60% (51%, 70%) | 41% (33%, 52%) |

## Extra details

There are a few more functions that also seem helpful:

* as\_flextable()
* as\_kable()/as\_kableExtra()
* sort\_p()/filter\_p()
* add\_global\_p()/add\_vif()
* add\_difference()/add\_significance\_stars()

## Conclusion

* Gtsummary combines quick and easy table creation with the option for similarly easy modifications.
* Plays well with html, pdf, and word docs
* Also interfaces with other common table packages like kableExtra, flextable, knitr
* Relatively quick learning curve if you are familiar with the piping operator
* Has saved me time on implementation and on updating tables for collaborators, very nice and easy reproduceability
* Still need to play around with the tests used within add\_p() and some more of the finer details
* Issues with footnote spacing projecting into tables
* Can use as a base to create tables, modify as needed, and then convert to another table format for even more extra detail

## Bonus Fun: attempt at Bayesian output!

SAMPLING FOR MODEL '8ddbcc58cd404dcabb530cce7312f25f' NOW (CHAIN 1).  
Chain 1:   
Chain 1: Gradient evaluation took 0 seconds  
Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.  
Chain 1: Adjust your expectations accordingly!  
Chain 1:   
Chain 1:   
Chain 1: Iteration: 1 / 2000 [ 0%] (Warmup)  
Chain 1: Iteration: 200 / 2000 [ 10%] (Warmup)  
Chain 1: Iteration: 400 / 2000 [ 20%] (Warmup)  
Chain 1: Iteration: 600 / 2000 [ 30%] (Warmup)  
Chain 1: Iteration: 800 / 2000 [ 40%] (Warmup)  
Chain 1: Iteration: 1000 / 2000 [ 50%] (Warmup)  
Chain 1: Iteration: 1001 / 2000 [ 50%] (Sampling)  
Chain 1: Iteration: 1200 / 2000 [ 60%] (Sampling)  
Chain 1: Iteration: 1400 / 2000 [ 70%] (Sampling)  
Chain 1: Iteration: 1600 / 2000 [ 80%] (Sampling)  
Chain 1: Iteration: 1800 / 2000 [ 90%] (Sampling)  
Chain 1: Iteration: 2000 / 2000 [100%] (Sampling)  
Chain 1:   
Chain 1: Elapsed Time: 0.082 seconds (Warm-up)  
Chain 1: 0.079 seconds (Sampling)  
Chain 1: 0.161 seconds (Total)  
Chain 1:   
  
SAMPLING FOR MODEL '8ddbcc58cd404dcabb530cce7312f25f' NOW (CHAIN 2).  
Chain 2:   
Chain 2: Gradient evaluation took 0 seconds  
Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.  
Chain 2: Adjust your expectations accordingly!  
Chain 2:   
Chain 2:   
Chain 2: Iteration: 1 / 2000 [ 0%] (Warmup)  
Chain 2: Iteration: 200 / 2000 [ 10%] (Warmup)  
Chain 2: Iteration: 400 / 2000 [ 20%] (Warmup)  
Chain 2: Iteration: 600 / 2000 [ 30%] (Warmup)  
Chain 2: Iteration: 800 / 2000 [ 40%] (Warmup)  
Chain 2: Iteration: 1000 / 2000 [ 50%] (Warmup)  
Chain 2: Iteration: 1001 / 2000 [ 50%] (Sampling)  
Chain 2: Iteration: 1200 / 2000 [ 60%] (Sampling)  
Chain 2: Iteration: 1400 / 2000 [ 70%] (Sampling)  
Chain 2: Iteration: 1600 / 2000 [ 80%] (Sampling)  
Chain 2: Iteration: 1800 / 2000 [ 90%] (Sampling)  
Chain 2: Iteration: 2000 / 2000 [100%] (Sampling)  
Chain 2:   
Chain 2: Elapsed Time: 0.084 seconds (Warm-up)  
Chain 2: 0.072 seconds (Sampling)  
Chain 2: 0.156 seconds (Total)  
Chain 2:   
  
SAMPLING FOR MODEL '8ddbcc58cd404dcabb530cce7312f25f' NOW (CHAIN 3).  
Chain 3:   
Chain 3: Gradient evaluation took 0 seconds  
Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.  
Chain 3: Adjust your expectations accordingly!  
Chain 3:   
Chain 3:   
Chain 3: Iteration: 1 / 2000 [ 0%] (Warmup)  
Chain 3: Iteration: 200 / 2000 [ 10%] (Warmup)  
Chain 3: Iteration: 400 / 2000 [ 20%] (Warmup)  
Chain 3: Iteration: 600 / 2000 [ 30%] (Warmup)  
Chain 3: Iteration: 800 / 2000 [ 40%] (Warmup)  
Chain 3: Iteration: 1000 / 2000 [ 50%] (Warmup)  
Chain 3: Iteration: 1001 / 2000 [ 50%] (Sampling)  
Chain 3: Iteration: 1200 / 2000 [ 60%] (Sampling)  
Chain 3: Iteration: 1400 / 2000 [ 70%] (Sampling)  
Chain 3: Iteration: 1600 / 2000 [ 80%] (Sampling)  
Chain 3: Iteration: 1800 / 2000 [ 90%] (Sampling)  
Chain 3: Iteration: 2000 / 2000 [100%] (Sampling)  
Chain 3:   
Chain 3: Elapsed Time: 0.079 seconds (Warm-up)  
Chain 3: 0.072 seconds (Sampling)  
Chain 3: 0.151 seconds (Total)  
Chain 3:   
  
SAMPLING FOR MODEL '8ddbcc58cd404dcabb530cce7312f25f' NOW (CHAIN 4).  
Chain 4:   
Chain 4: Gradient evaluation took 0 seconds  
Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.  
Chain 4: Adjust your expectations accordingly!  
Chain 4:   
Chain 4:   
Chain 4: Iteration: 1 / 2000 [ 0%] (Warmup)  
Chain 4: Iteration: 200 / 2000 [ 10%] (Warmup)  
Chain 4: Iteration: 400 / 2000 [ 20%] (Warmup)  
Chain 4: Iteration: 600 / 2000 [ 30%] (Warmup)  
Chain 4: Iteration: 800 / 2000 [ 40%] (Warmup)  
Chain 4: Iteration: 1000 / 2000 [ 50%] (Warmup)  
Chain 4: Iteration: 1001 / 2000 [ 50%] (Sampling)  
Chain 4: Iteration: 1200 / 2000 [ 60%] (Sampling)  
Chain 4: Iteration: 1400 / 2000 [ 70%] (Sampling)  
Chain 4: Iteration: 1600 / 2000 [ 80%] (Sampling)  
Chain 4: Iteration: 1800 / 2000 [ 90%] (Sampling)  
Chain 4: Iteration: 2000 / 2000 [100%] (Sampling)  
Chain 4:   
Chain 4: Elapsed Time: 0.074 seconds (Warm-up)  
Chain 4: 0.077 seconds (Sampling)  
Chain 4: 0.151 seconds (Total)  
Chain 4:

mod\_bayes<-brm(response ~ trt + grade + stage + marker,  
 trial,   
 family = bernoulli(link = "logit"),  
 chains = 4,  
 iter = 2000)

mod\_bayes %>%  
tbl\_regression(exponentiate = T)

| Characteristic | Beta | 95% CI1 |
| --- | --- | --- |
| trt |  |  |
| trtDrugB | 0.41 | -0.23, 1.04 |
| grade |  |  |
| I | — | — |
| II | 0.16 | -0.66, 1.01 |
| III | 0.13 | -0.68, 0.94 |
| stage |  |  |
| T1 | — | — |
| T2 | -0.81 | -1.73, 0.09 |
| T3 | 0.03 | -0.93, 0.95 |
| T4 | -0.39 | -1.30, 0.49 |
| marker | 0.40 | 0.02, 0.79 |
| 1CI = Credible Interval | | |

Sadly the exponentiate = T does not really work for rstanarm/brms. For Bayesian models, I believe you need to specify the tidy\_fun = within the tbl\_regression() output and manually create a tidy helper to get more out of the model than the raw estimates and confidence intervals.