STA 478/578 Homework 1 Solutions

Dr. Robert Buscaglia

September 03, 2025

I have prepared a quick solutions document that does not restate too much, but has been updated to use new tidyverse commands. Nothing here should be outdated.

Exercises

Exercise 1.

a

Load the data file `compstats1.RData` using the `load()` command.

```r
load('compstats1.RData')
```

b

Using the vector of ages create a factor that has levels Minor or Adult where any observation greater than or equal to 18 qualifies as an adult. Also, make sure that the order of the levels is Minor first and Adult second.

Solution.

The idea here is to use a logical statement to split the ages into 'Adult' and 'Minor' levels. We then need to ensure we label everything correctly.

```r
Minor.cutoff <- ages >= 18
factor(Minor.cutoff, labels=c('Minor', 'Adult'))
```

```
## [1] Minor Adult Minor Adult Adult Adult
## Levels: Minor Adult
```

If you had created the cutoffs differently (say you did `< 18`), you may have needed to `relevel` your factor to get `Minor` first.

Exercise 2.

The dataset ChickWeight tracks the weights of 48 baby chickens (chicks) feed four different diets.

a

Load the dataset using

```r
data(ChickWeight)
```

b

Remove all the observations except for the weights on day 10 and day 20.

Solution.

This is a great place for the `filter` command.

```r
ChickWeight.2 <- ChickWeight %>% filter(Time == 10 | Time == 20)
```

I use the 'or' command to do it in one line. Observe this new data.frame contains only Times 10 and 20.

```r
head(ChickWeight.2)
```

```
##   weight Time Chick Diet
## 1     93   10     1    1
## 2    199   20     1    1
## 3    103   10     2    1
## 4    209   20     2    1
## 5     99   10     3    1
## 6    198   20     3    1
```

c

Calculate the mean and standard deviation for each diet group on days 10 and 20.

Solution.

We use a little data.frame manipulation and a `summarize` command. I first need to group the diets and days, then find the means and standard deviations. The kable function just makes the output table.

```r
ChickWeight.2 %>% group_by(Diet, Time) %>%
   summarize(Mean = mean(weight), SD = sd(weight)) %>% kable(align='c')
```

Diet	Time	Mean	SD
1	10	93.05263	22.54249
1	20	170.41176	55.43584
2	10	108.50000	24.29563
2	20	205.60000	70.25224
3	10	117.10000	20.19048
3	20	258.90000	65.24390
4	10	126.00000	11.43095
4	20	233.88889	37.56809

If you would prefer to see the Times grouped, rather than the Diets grouped, you can switch the order of the `group_by` command.

Exercise 3.

A common task is to take a set of data that has multiple categorical variables and create a table of the number of cases for each combination. An introductory statistics textbook contains a dataset summarizing student surveys from several sections of an intro class. The two variables of interest for us are Gender and Year which are the students gender and year in college.

a

Download the dataset and correctly order the Year variable using the following:

```r
Survey <- read.csv("https://www.lock5stat.com/datasets3e/StudentSurvey.csv",
                   na.strings=c(""," ")) %>%
   mutate(Year = factor(Year, levels=c("FirstYear","Sophomore","Junior","Senior"))) %>%
   mutate(Sex = factor(Sex))
```

b

Using some combination of dplyr functions, produce a data set with eight rows that contains the number of responses for each gender:year combination. Notice there are two females that neglected to give their Year and you should remove them first. The function is.na(Year) will return logical values indicating if the Year value was missing and you can flip those values using the negation operator !. So you might consider using !is.na(Year) as the argument to a filter() command. Alternatively you sort on Year and remove the first two rows using slice(-2:-1). Next you'll want to summarize each Year/Gender group using the n() function which gives the number of rows in a data set.

Solution.

I tried to give you some hints in the question. The idea here is we are making a new data.frame that counts how many Males and Females are in each `Year`. This sounds daunting, but `dyplr` makes our life easy. Here is the solution in one shot. I added `dplyr::` package call here to select because the `select()` function is pervasive and shows up in many packages, and I want to ensure it uses the `dyplr` version.

```r
Survey.Counts <- Survey %>% dplyr::select(Year, Sex) %>%
   filter(!is.na(Year)) %>% group_by(Year, Sex) %>%
   summarize(Count = n())
Survey.Counts
```

```
## # A tibble: 8 x 3
## # Groups:   Year [4]
##   Year      Sex   Count
##   <fct>     <fct> <int>
## 1 FirstYear F        43
## 2 FirstYear M        51
## 3 Sophomore F        96
## 4 Sophomore M        99
## 5 Junior    F        18
## 6 Junior    M        17
## 7 Senior    F        10
## 8 Senior    M        26
```

There are other ways to accomplish this, but the `dyplr` flow makes sense. I grab the data of interest, remove the NAs, group my data in the way I want it, and then count!

c

Using *tidyr* commands, produce a table that summarizes the counts for each Gender / Year combination. The table should not have a row or column for 'NA'. This should be dealt with in part b).

Solution.

The output above would certainly look and read easier if it were Year on the rows, and Gender as columns, with the counts filled in. This is a 'pivot', and requires me to `spread` my data.frame so that I can make Counts the entries, and Gender the columns. *Spread* has become *pivot_wider* which is much easier to understand, we are taking a "long" table and making it "wide". Proper updated syntax below.

```r
Survey.Counts %>%
  pivot_wider(names_from=Year, values_from=Count) %>%
  kable(align=c('l','c','c'))
```

Sex	FirstYear	Sophomore	Junior	Senior
F	43	96	18	10
M	51	99	17	26

It is likely you will want to make some nice tables this semester, so I show an example using the `knitr` package and `kable` command. It is now easy to read how many `Females` are `Juniors`, or other similar questions. We will want to be good at `pivot_wider()` and `pivot_longer()` for data manipulation.

Exercise 4.

We'll next make some density plots that relate several factors towards the birthweight of a child.

a

Load the MASS library, which includes the data set `birthwt` which contains information about 189 babies and their mothers.

Solution.

I do what it says, and initialize the `birthwt` data.frame. As a note, it is the `MASS` package that is bringing in another `select()` function and masking the tidyverse version. There are ways to control this, but I mentioned above why we might use a `package::` syntax in certain cases.

```r
library(MASS)
```

```
##
## Attaching package: 'MASS'
## The following object is masked from 'package:dplyr':
##
##     select
```

```
data('birthwt')
```

b

Add better labels to the race and smoke variables using the following:

```
birthwt <- birthwt %>%
           mutate(race  = factor(race,  labels=c('White','Black','Other')),
           smoke = factor(smoke, labels=c('No Smoke', 'Smoke')))
```

Just some more factoring and labeling, these are good notes to keep for future assignments.

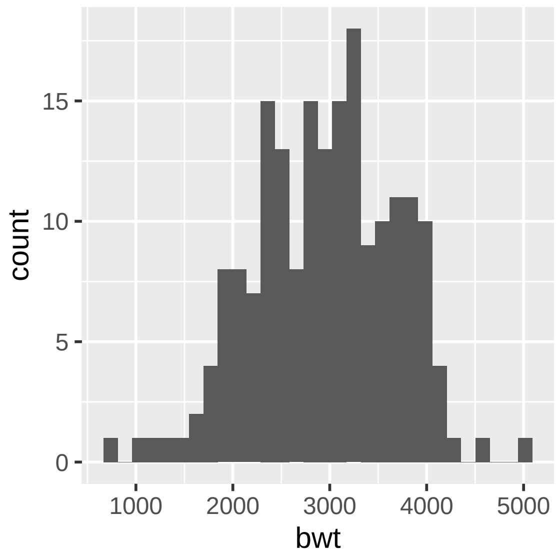
c

Graph a histogram of the birthweights `bwt` using `ggplot()`.

Solution.

Visualization is extremely important, be sure to refresh how we use `ggplot`. This requires the package `ggplot2`.

```
library(ggplot2)
ggplot(birthwt) + geom_histogram(aes(x = bwt))
```



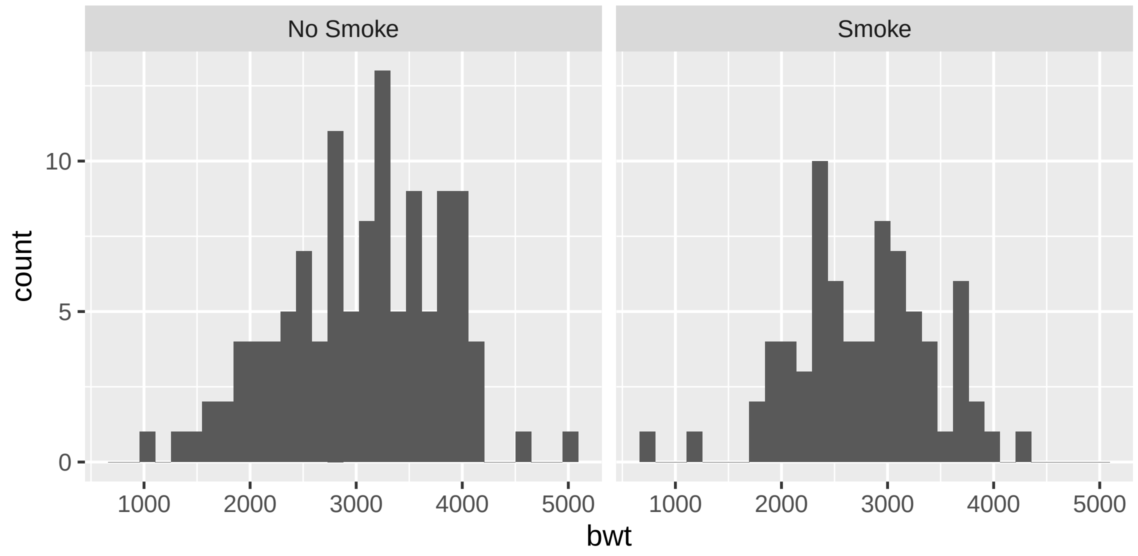
Simple enough and straightforward. R is one of the most versitile plotting languages, so lets make this fancier.

d

Make separate graphs that denote whether a mother smoked during pregnancy using the `facet_grid()` command.

Solution.

```
ggplot(birthwt) + geom_histogram(aes(x = bwt)) + facet_grid(.~smoke)
```



This is a bit more informative.

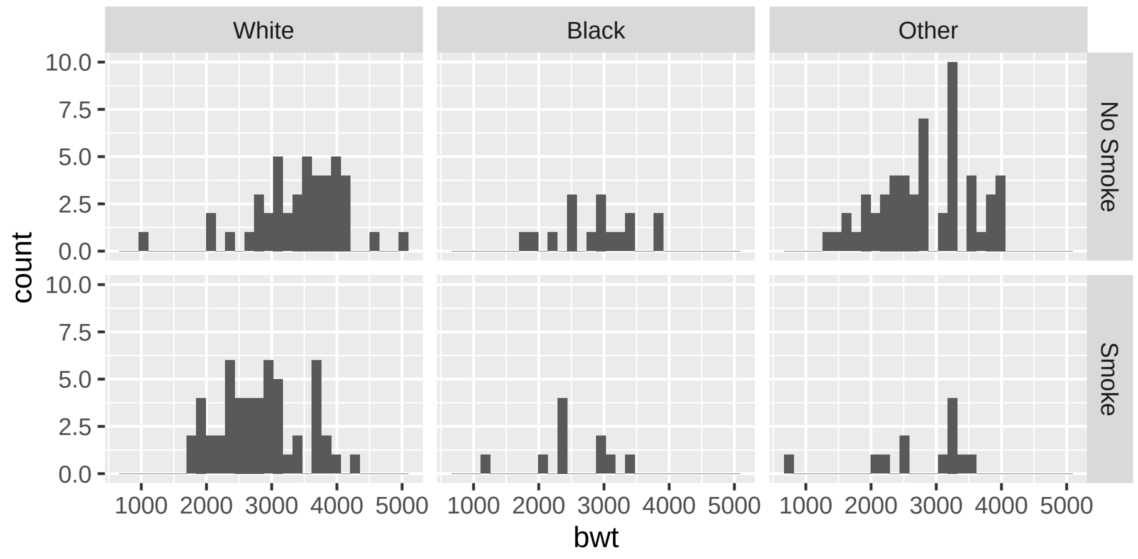
e

Perhaps race matters in relation to smoking. Make our grid of graphs vary with smoking status changing vertically, and race changing horizontally (that is the formula in `facet_grid()` should have smoking be the y variable and race as the x).

Solution.

Just keep improving upon our graphic. Remember in R it is always `y ~ x`, so notice I put `smoke` first.

```
ggplot(birthwt) + geom_histogram(aes(x = bwt)) + facet_grid(smoke~race)
```



We obtain a breakdown by smoking status and race. There are lots of color and style options as well. We could have also improved our axis labels.

f

Remove race from the facet grid, (so go back to the graph you had in part d). I'd like to next add an estimated density line to the graphs, but to do that, I need to first change the y-axis to be density (instead of counts), which we do by using `aes(y=..density..)` in the ggplot() aesthetics command.

Solution.

This question just informs you about the need to use density. No real work here to do. The syntax given is outdated and we must now use `y = after_stat(density)` to get the proper result without depreciation warnings.

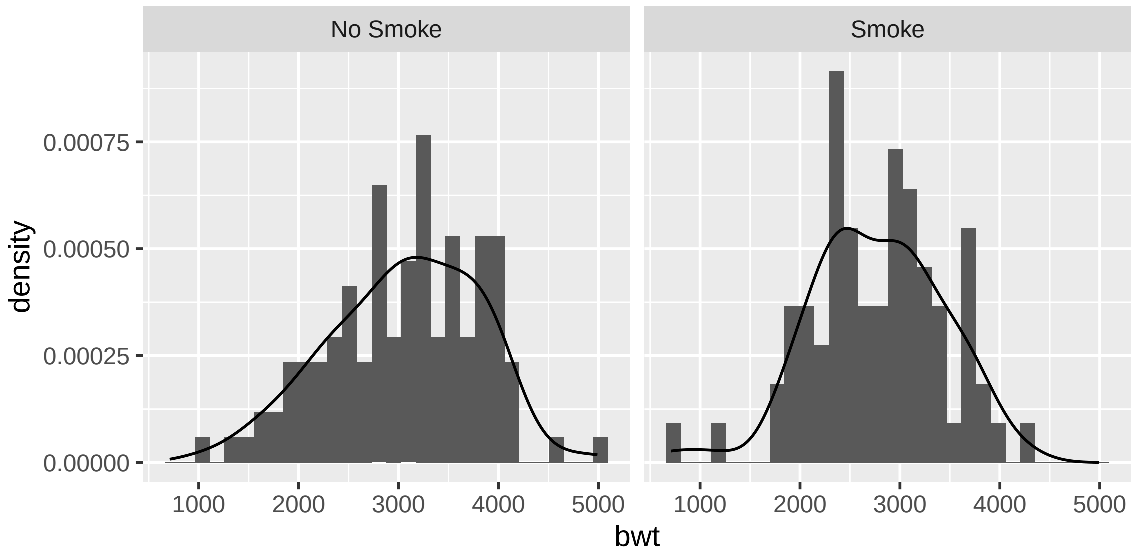
g

Add the estimated smooth density using the `geom_density()` command.

Solution.

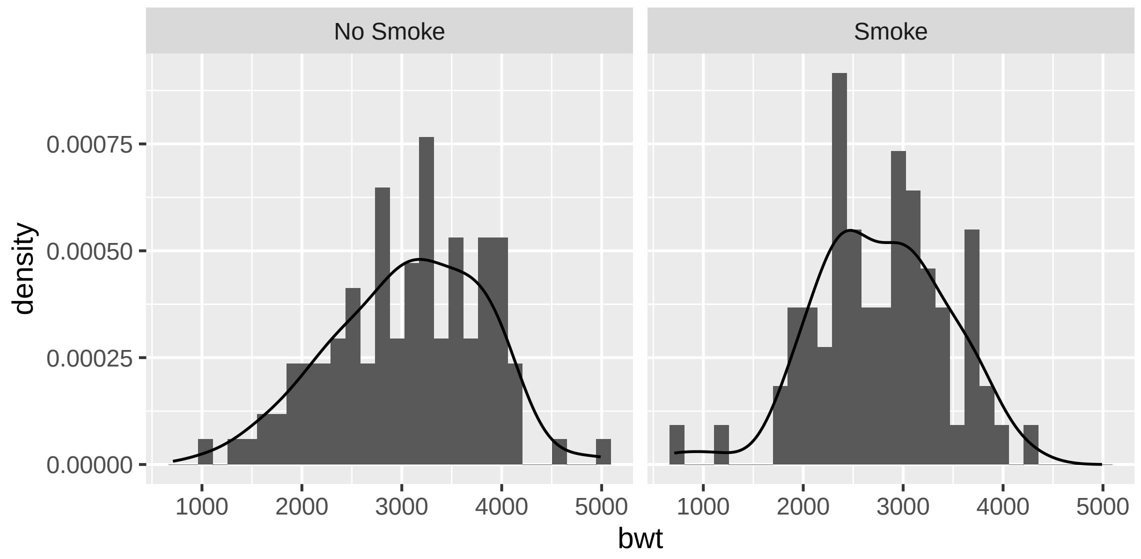
Let us make a histogram with the estimated density overlaid on the bins. Below I use the updated `after_stat()` syntax to make histograms on the density scale. If we do not convert both to density, the scales will be different, and we will not see the density curve.

```r
ggplot(birthwt, aes(x = bwt)) + geom_histogram(aes(y=after_stat(density))) +
   geom_density() + facet_grid(.~smoke)
```



If you are going to reuse aesthetics, it can be best to move them to the first `ggplot` call. Notice this is a little cleaner, but produces the same thing.

```r
ggplot(birthwt, aes(x=bwt, y=after_stat(density))) + geom_histogram() +
   geom_density() + facet_grid(.~smoke)
```



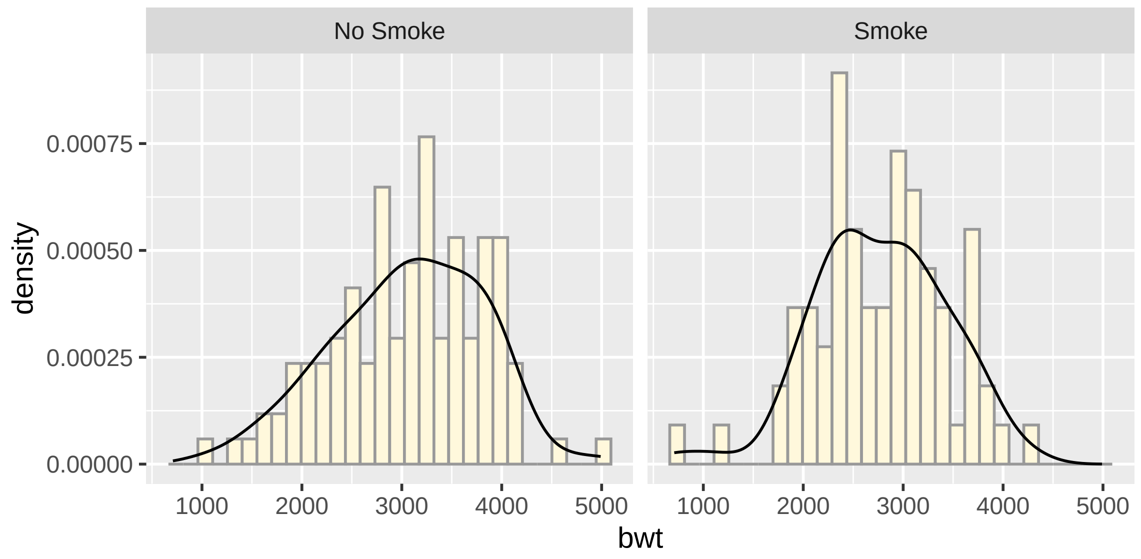
h

Change the fill color of the histograms to be something less dark, lets use `fill='cornsilk'` and `color='grey60'`.

Solution.

We are now just making the plots more beautiful. It is good practice to make the graphics easy to read, so changing the colors of the bars here does help, the smooth estimate becomes easier to see.

```r
ggplot(birthwt, aes(x=bwt, y=after_stat(density))) +
   geom_histogram(fill='cornsilk', color='grey60') +
   geom_density() + facet_grid(.~smoke)
```



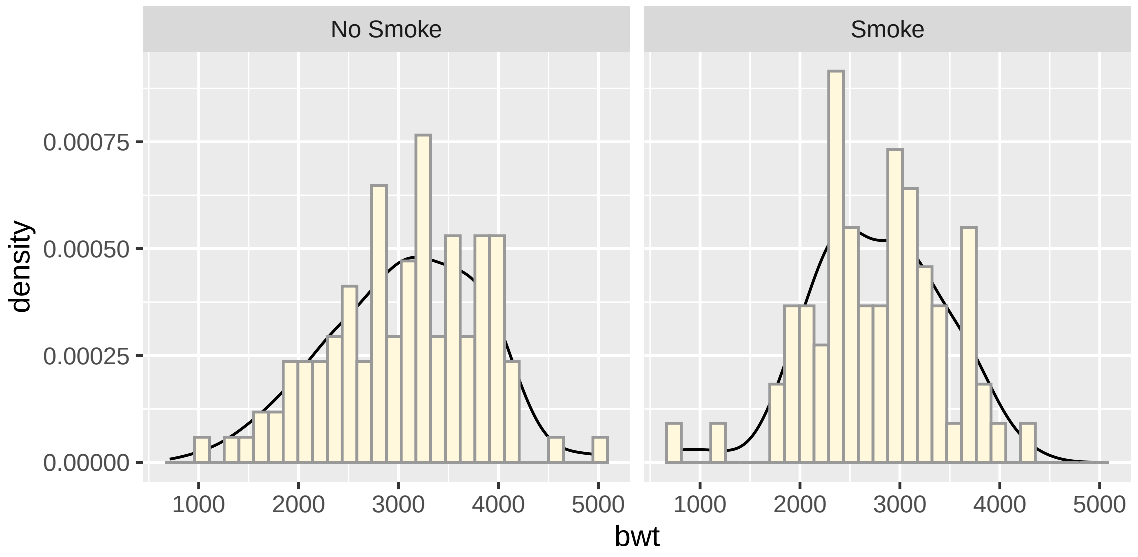
i

Change the order in which the histogram and the density line are added to the plot. Does the order matter? Which do you prefer?

Solution.

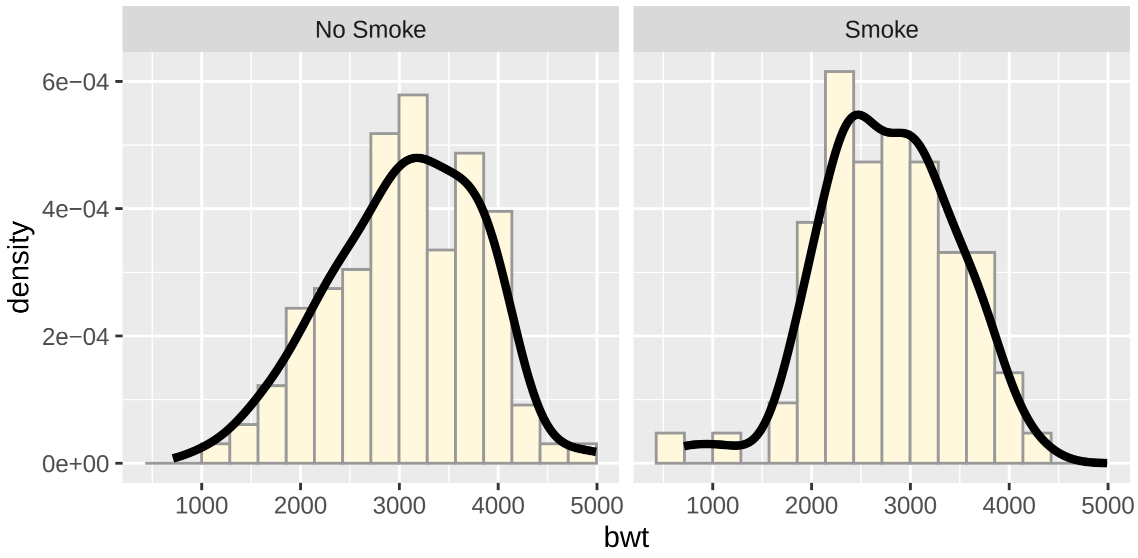
Order matters in R, notice my smooth line is moved to the back when I change the order.

```
ggplot(birthwt, aes(x=bwt, y=after_stat(density))) + geom_density() +
   geom_histogram(fill='cornsilk', color='grey60') + facet_grid(.~smoke)
```



I think it looks better with the smooth estimate in the front. Although I would still maybe change the bins and prefer the density in the front layer. My final graph might look a little more like this (although I'd change a lot of fonts and labels as well).

```
ggplot(birthwt, aes(x=bwt, y=after_stat(density))) +
  geom_histogram(fill='cornsilk', color='grey60', bins=16) +
  geom_density(linewidth = 1.5) +
  facet_grid(.~smoke)
```


No Smoke
Smoke
density
6e−04
4e−04
2e−04
0e+00
bwt