431 Class 03

Thomas E. Love

2019-09-03

Today's Agenda

Using R, RStudio and R Markdown and the 431 RStudio Cloud

Contact us at 431-help@case.edu

Our web site: https://github.com/THOMASELOVE/2019-431

A Worked Day 1 Survey Analysis

We have data on the site in a file called `surveyday1_2019.csv`. I built a R Markdown file, and then knitted it into an a PDF and an HTML then posted links to the Class 04 README.

- A "floating" table of contents
- Key verbs in the tidyverse for data wrangling
  - select, filter, count, arrange, mutate, group_by, summarize
- Building Histograms to describe a single quantitative variable
- Comparing a distribution of a quantity within groups
  - Faceted histogram
  - Comparison boxplot
- Obtaining numerical summaries
- Scatterplots with `ggplot`

All of this is also in the Course Notes.

Analyzing the Day 1 Survey

Load the R Packages we need

```
library(magrittr); library(tidyverse)
## always need tidyverse, can include other packages too
```

Analyzing the Day 1 Survey

Load the Data

We will read in the .csv file of the data, and then pipe that result into the `tbl_df` function, which turns the data frame into a nicely organized *tibble*.

- Since we've stored the data file in the same directory as our R Project, we can read it in directly.

```
day1 <- read.csv("surveyday1_2019.csv") %>% tbl_df
```

Print your tibble by typing its name

```
day1
```

```
# A tibble: 315 x 21
   student sex   glasses english statsofar ageguess
     <int> <fct> <fct>   <fct>       <int>    <int>
 1  201901 <NA>  y       y               6       42
 2  201902 <NA>  y       y               7       53
 3  201903 <NA>  y       y               4       45
 4  201904 <NA>  y       y               7       45
 5  201905 <NA>  y       y               6       42
 6  201906 <NA>  y       y               7       50
 7  201907 <NA>  y       y               5       56
 8  201908 <NA>  n       n               6       50
 9  201909 <NA>  n       y               6       52
10  201910 <NA>  n       y               4       42
# ... with 305 more rows, and 15 more variables:
#   smoke <int>, h.left <int>, h.right <int>,
```

Use select to pick columns / variables from your tibble

```
day1 %>%
    select(favcolor, haircut)
```

```
# A tibble: 315 x 2
   favcolor haircut
   <fct>       <dbl>
 1 teal        120
 2 blue         20
 3 purple       20
 4 blue          0
 5 blue          6.99
 6 <NA>        NA
 7 green        25
 8 red          80
 9 green        16
10 blue         12.5
```

Use filter to pick rows / subjects from your tibble

```
day1 %>%
    filter(year == 2019)
```

```
# A tibble: 61 x 21
   student sex   glasses english statsofar ageguess
     <int> <fct> <fct>   <fct>        <int>    <int>
 1  201901 <NA>  y       y                6       42
 2  201902 <NA>  y       y                7       53
 3  201903 <NA>  y       y                4       45
 4  201904 <NA>  y       y                7       45
 5  201905 <NA>  y       y                6       42
 6  201906 <NA>  y       y                7       50
 7  201907 <NA>  y       y                5       56
 8  201908 <NA>  n       n                6       50
 9  201909 <NA>  n       y                6       52
10  201910 <NA>  n       y                4       42
# ... with 51 more rows, and 15 more variables:
```

Use count to count the number of observations meeting a criterion

```
day1 %>%
    count(favcolor == "red")
```

```
# A tibble: 3 x 2
  `favcolor == "red"`     n
  <lgl>               <int>
1 FALSE                 280
2 TRUE                   30
3 NA                      5
```

Or to provide a cross-classification:

```
day1 %>%
    count(favcolor == "blue", factor(english))
```

```
Warning: Factor `factor(english)` contains implicit NA,
consider using `forcats::fct_explicit_na`
```

```
# A tibble: 7 x 3
  `favcolor == "blue"` `factor(english)`     n
  <lgl>                <fct>               <int>
1 FALSE                n                      33
2 FALSE                y                     154
3 FALSE                <NA>                    2
4 TRUE                 n                      24
5 TRUE                 y                      97
6 NA                   n                       2
7 NA                   y                       3
```

Use arrange to arrange the rows of a tibble

```
day1 %>%
    count(statsofar) %>%
    arrange(desc(n))
```

```
# A tibble: 7 x 2
  statsofar     n
      <int> <int>
1         5   122
2         6    69
3         7    45
4         4    37
5         3    31
6         2     7
7         1     4
```

Add new variables with mutate

```
day1 %>%
    mutate(guess_error = ageguess - lovetrueage) %>%
    select(ageguess, lovetrueage, guess_error) %>%
    summary()
```

```
    ageguess      lovetrueage     guess_error      
 Min.   :21.0   Min.   :47.50   Min.   :-31.500  
 1st Qu.:45.0   1st Qu.:48.50   1st Qu.: -6.500  
 Median :48.0   Median :50.50   Median : -2.500  
 Mean   :47.3   Mean   :50.13   Mean   : -2.837  
 3rd Qu.:52.0   3rd Qu.:51.50   3rd Qu.:  0.500  
 Max.   :70.0   Max.   :52.50   Max.   : 20.500  
 NA's   :6                      NA's   :6        
```

Get grouped summaries with group_by and summarize

```
day1 %>%
    group_by(year) %>%
    summarize(n = n(), average_guess = mean(ageguess),
              min_error = min(ageguess),
              max_error = max(ageguess),
              actual = mean(lovetrueage))
```

```
# A tibble: 6 x 6
   year     n average_guess min_error max_error actual
  <int> <int>         <dbl>     <dbl>     <dbl>  <dbl>
1  2014    42          NA         NA        NA   47.5
2  2015    49          47.1       36        57   48.5
3  2016    64          NA         NA        NA   49.5
4  2017    48          46.5       29        58   50.5
5  2018    51          NA         NA        NA   51.5
6  2019    61          NA         NA        NA   52.5
```

Dealing with Missingness

We could filter our data to only include the subjects who provided a guess. . .

```
day1 %>%
    filter(complete.cases(ageguess)) %>%
    group_by(year) %>%
    summarize(n = n(), average_guess = mean(ageguess),
              min_error = min(ageguess),
              max_error = max(ageguess),
              actual = mean(lovetrueage))
```

```
# A tibble: 6 x 6
   year     n average_guess min_error max_error actual
  <int> <int>         <dbl>     <int>     <int>  <dbl>
1  2014    41          47.3        38        58   47.5
2  2015    49          47.1        36        57   48.5
3  2016    61          46.0        24        70   49.5
4  2017    48          46.5        29        58   50.5
```

Looking at Errors, instead

What if, instead, we wanted to look at the errors made, by subtracting off my true age from everyone's guess?

```
day1 %>%
    filter(complete.cases(ageguess)) %>%
    mutate(guess_error = ageguess - lovetrueage) %>%
    group_by(year) %>%
    summarize(n = n(), average_error = mean(guess_error),
              min_error = min(guess_error),
              max_error = max(guess_error))
```

```
# A tibble: 6 x 5
   year     n average_error min_error max_error
  <int> <int>         <dbl>     <dbl>     <dbl>
1  2014    41        -0.159      -9.5      10.5
2  2015    49        -1.38      -12.5       8.5
3  2016    61        -3.53      -25.5      20.5
4  2017    48        -3.96      -21.5       7.5
```

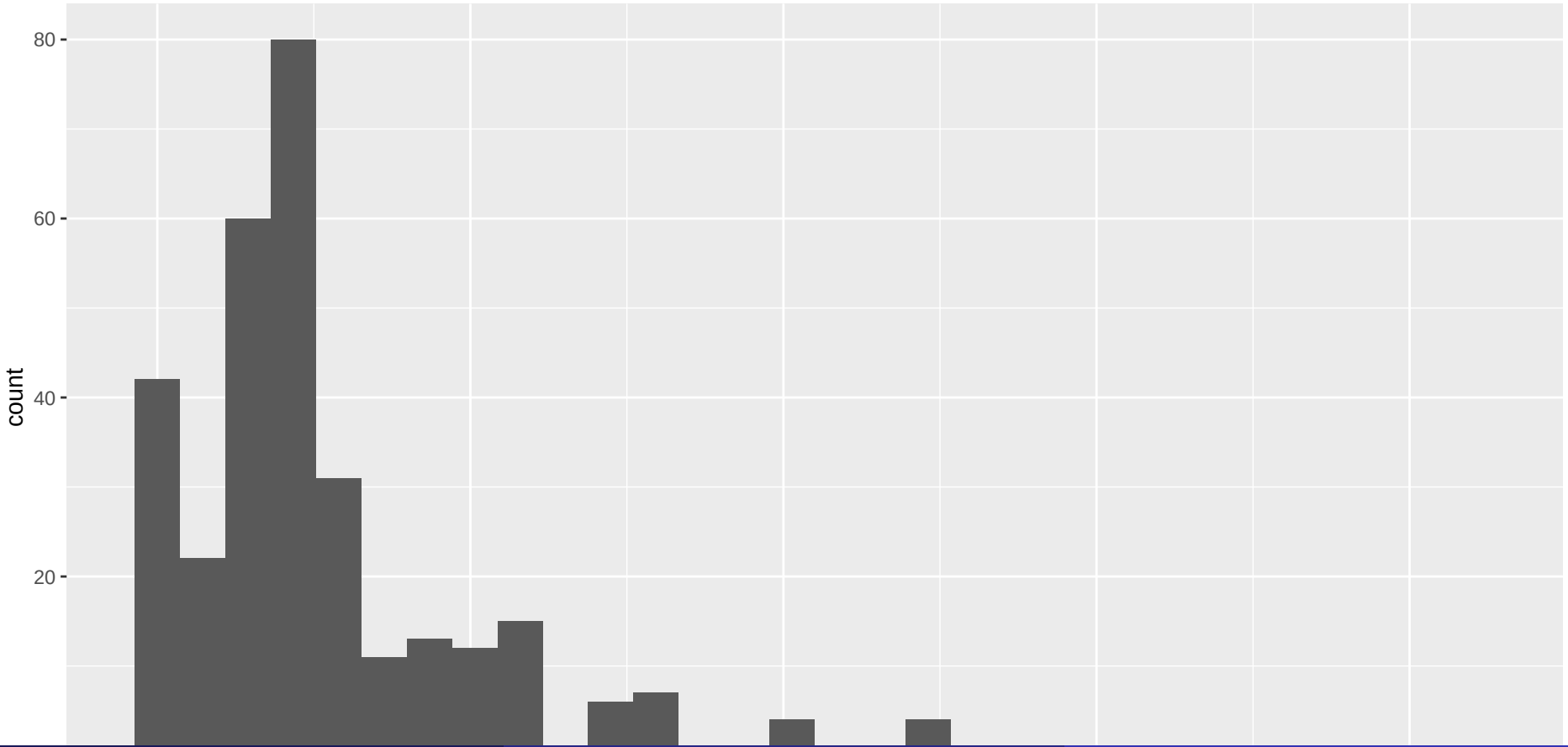
Histogram of Haircut Prices

```
ggplot(data = day1, aes(x = haircut)) +
    geom_histogram()
```

Histogram of Haircut Prices (Result)

```
`stat_bin()` using `bins = 30`. Pick better value
with `binwidth`.

Warning: Removed 4 rows containing non-finite values
(stat_bin).
```

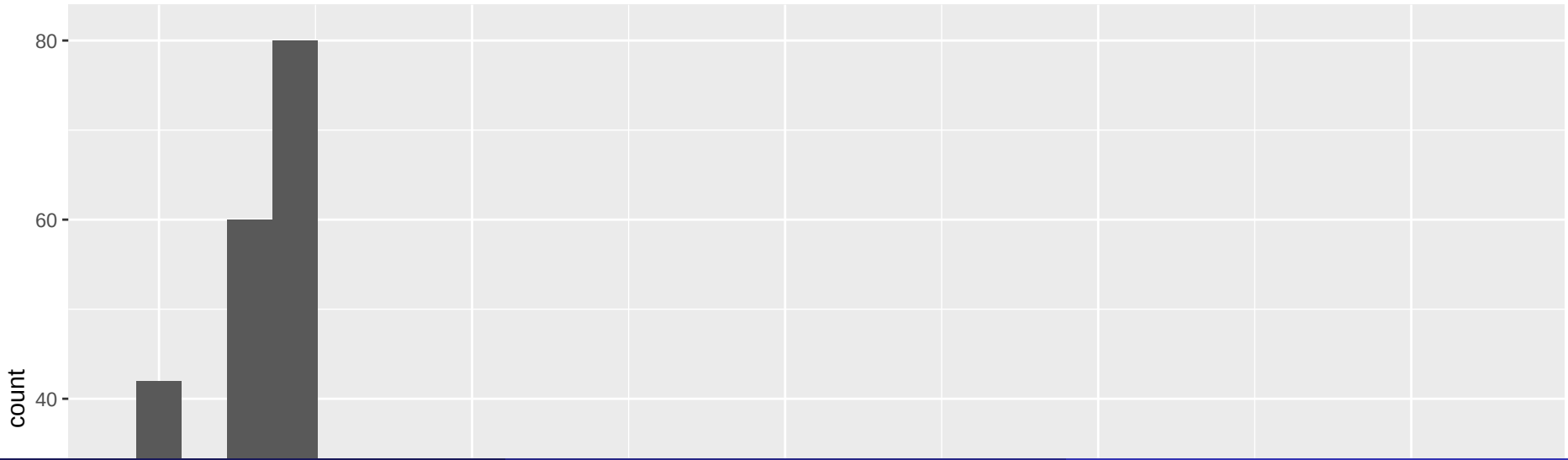


Improvements

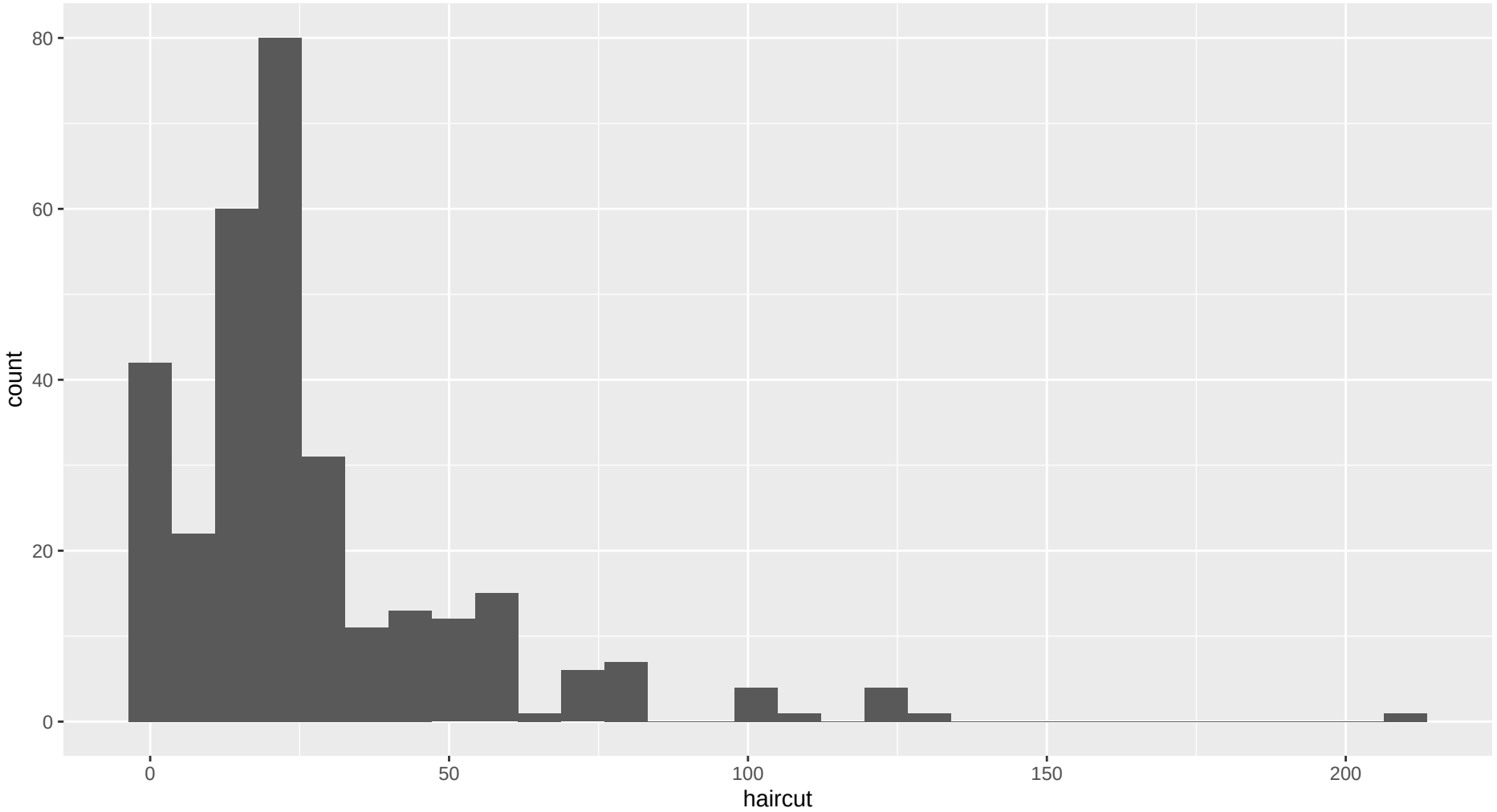
1. We'll filter the rows of the day1 tibble to include only those subjects who gave us a haircut price.

```
day1 %>%
    filter(complete.cases(haircut)) %>%
    ggplot(data = ., aes(x = haircut)) +
    geom_histogram()
```

```
`stat_bin()` using `bins = 30`. Pick better value
with `binwidth`.
```



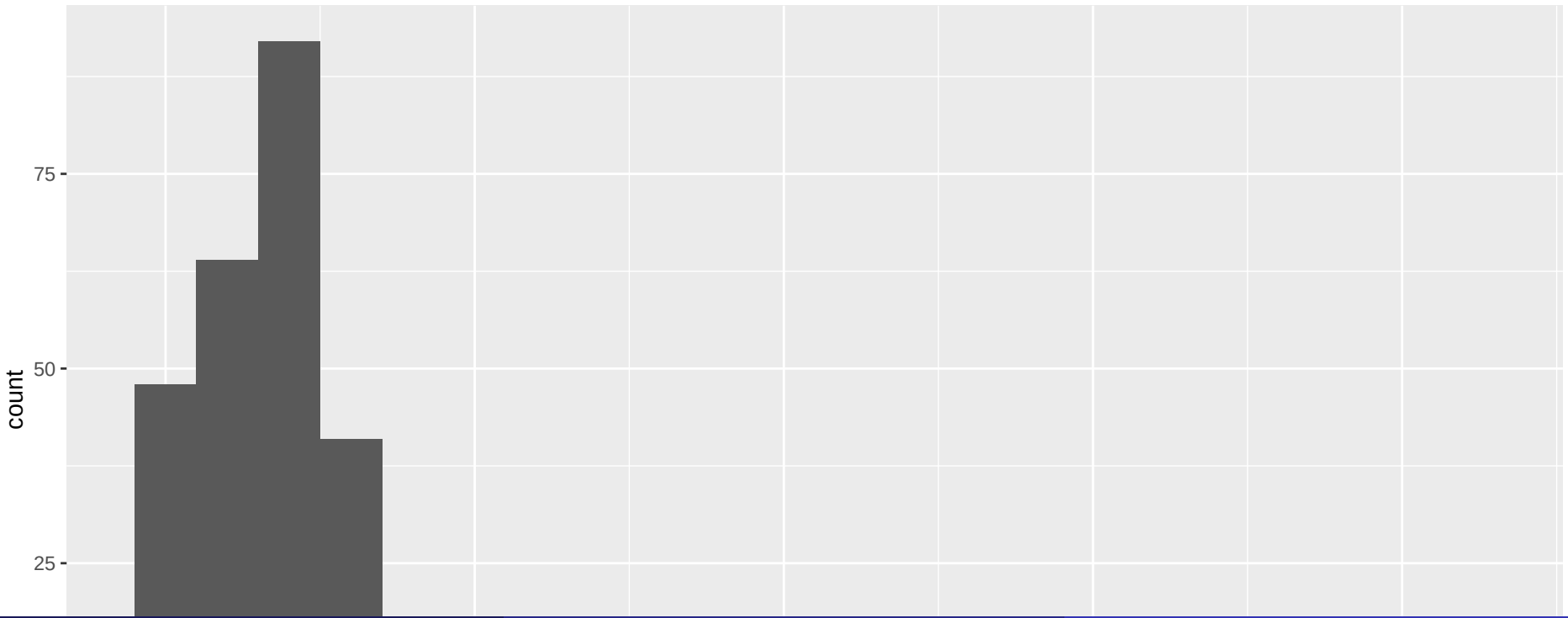
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.

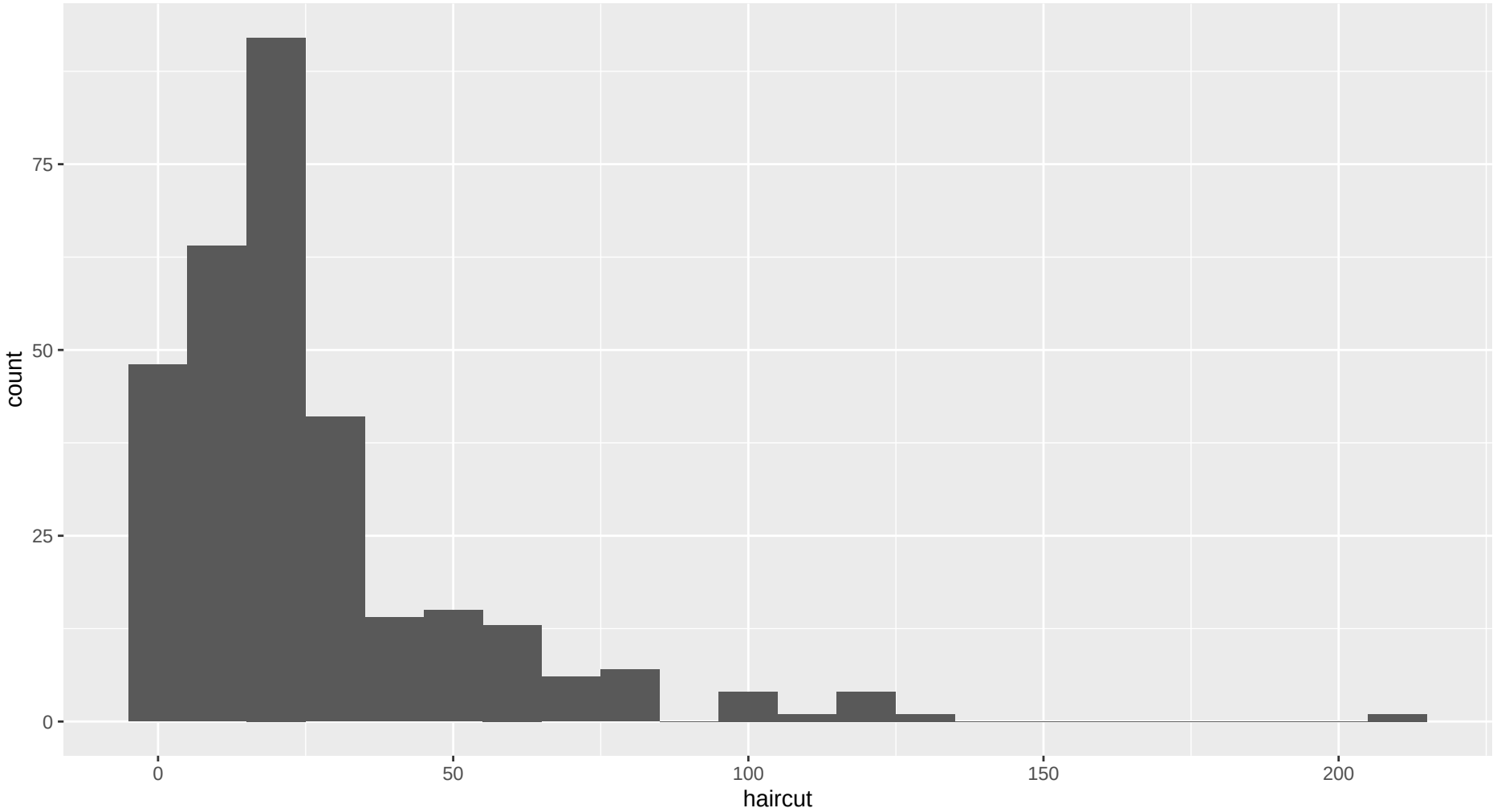


Improvements

2. We'll specify that R should create bins of width $10 (rather than the default, which creates 30 bins) for the haircut prices to fall in.

```
day1 %>%
    filter(complete.cases(haircut)) %>%
    ggplot(data = ., aes(x = haircut)) +
    geom_histogram(binwidth = 10)
```

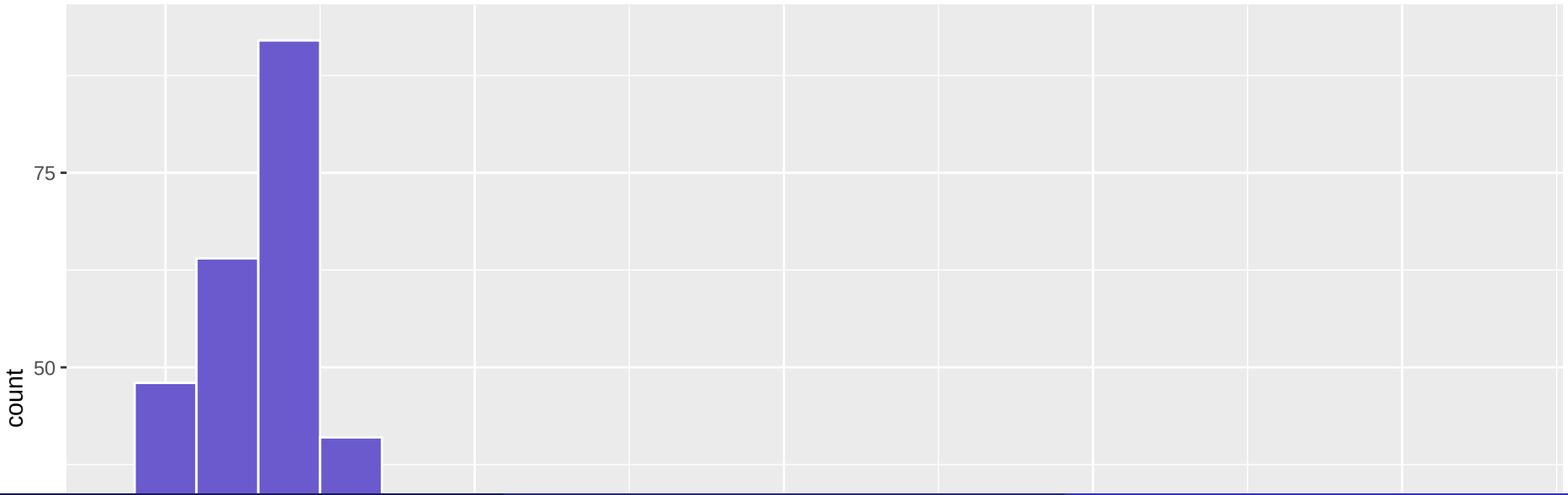


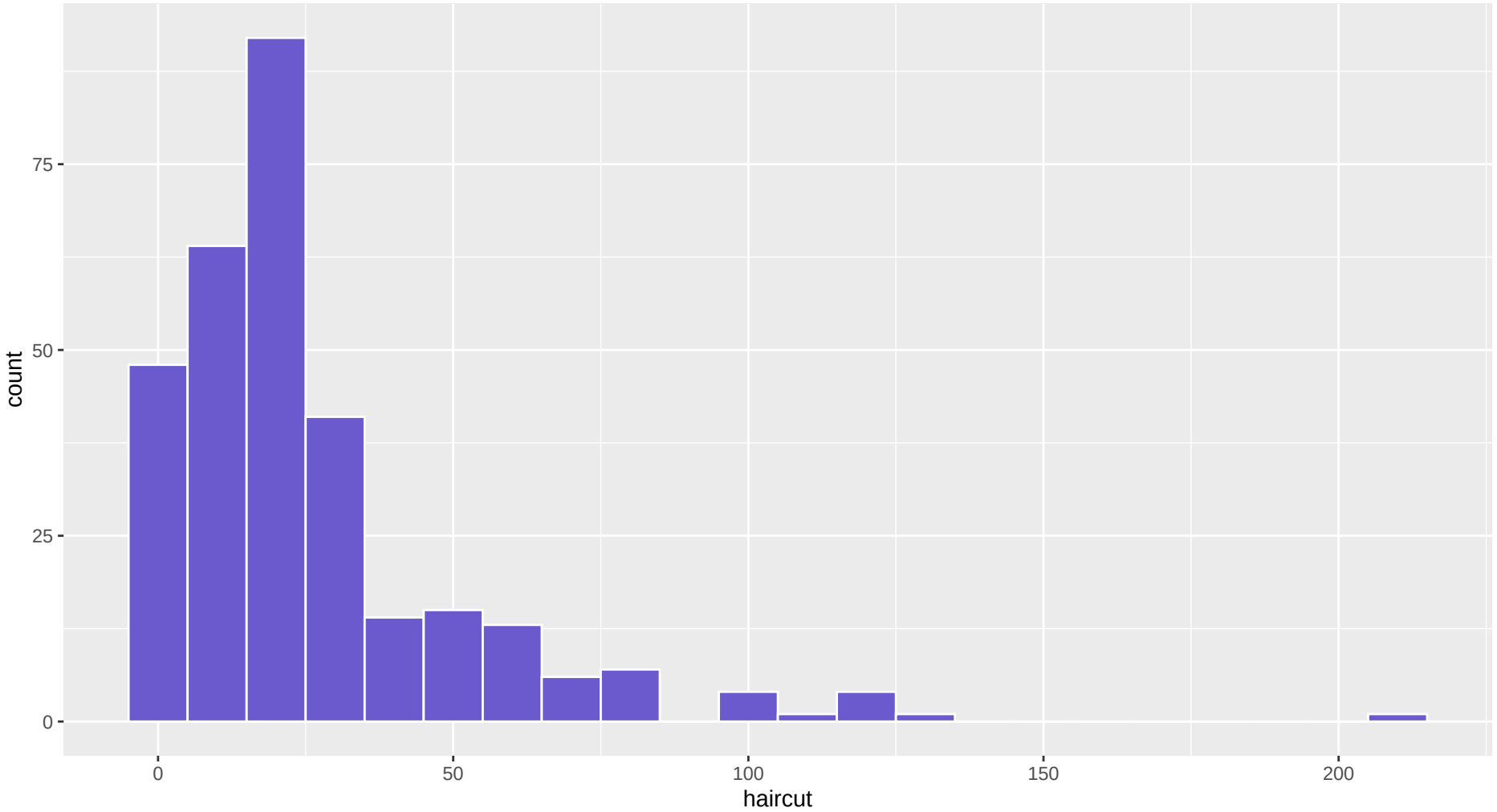

75
50
25
0
count
0
50
100
150
200
haircut

Improvements

3. We'll set the fill to be a better color - a nice resource for this is to google **Colors in R**. I'll pick "slateblue". We'll also color the outlines of the bars "white".

```
day1 %>%
    filter(complete.cases(haircut)) %>%
    ggplot(data = ., aes(x = haircut)) +
    geom_histogram(binwidth = 10,
                   fill = "slateblue", col = "white")
```

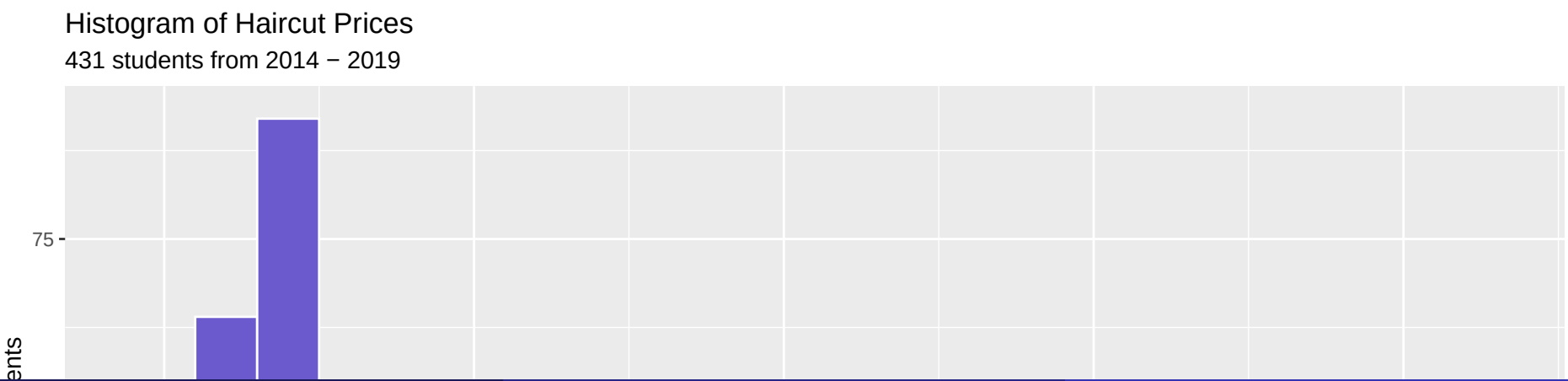



count
75
50
25
0
0
50
100
150
200
haircut

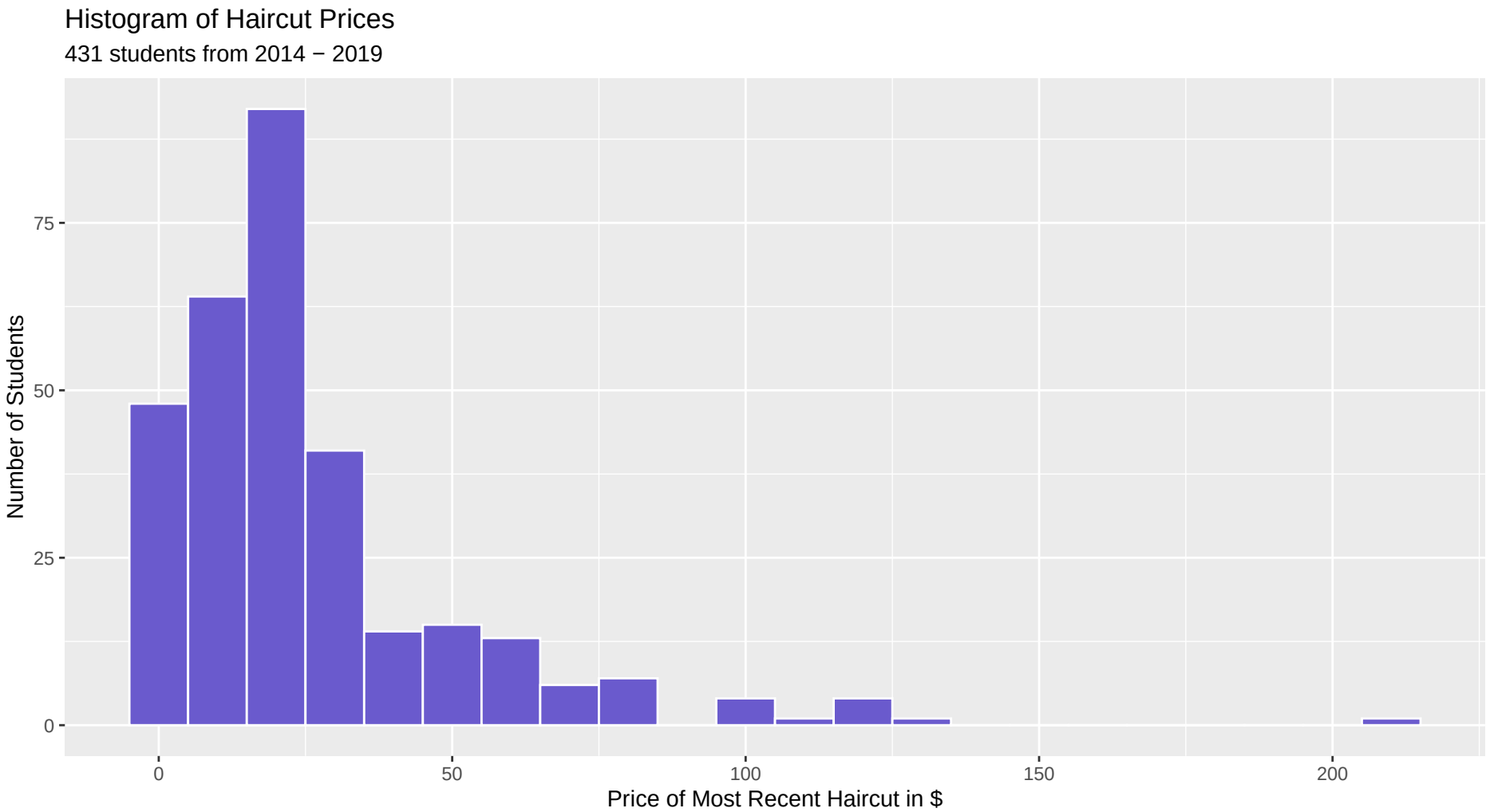
Improvements

4. We'll build a main title, subtitle and proper axis titles.

```
day1 %>%
    filter(complete.cases(haircut)) %>%
    ggplot(data = ., aes(x = haircut)) +
    geom_histogram(binwidth = 10,
                   fill = "slateblue", col = "white") +
    labs(x = "Price of Most Recent Haircut in $",
         y = "Number of Students",
         title = "Histogram of Haircut Prices",
         subtitle = "431 students from 2014 - 2019")
```



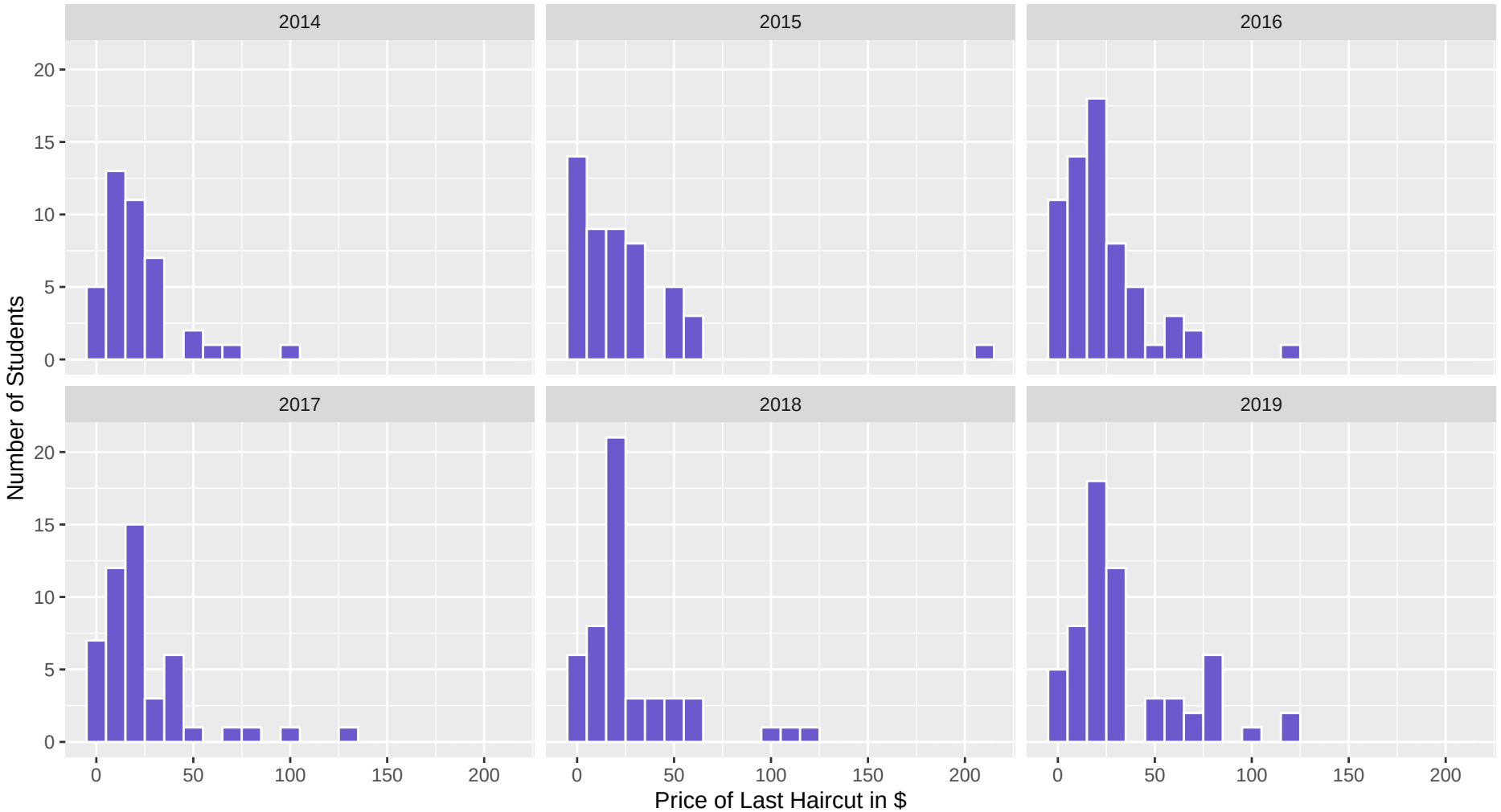
Result?


Histogram of Haircut Prices
431 students from 2014 − 2019
Number of Students
75
50
25
0
0
50
100
150
200
Price of Most Recent Haircut in $

Separate histograms for each year with faceting?

```
day1 %>%
    filter(!is.na(haircut)) %>%
    ggplot(., aes(x = haircut)) +
        geom_histogram(binwidth = 10,
                       fill = "slateblue", color = "white") +
        guides(fill = FALSE) +
        labs(x = "Price of Last Haircut in $",
                 y = "Number of Students") +
        facet_wrap(~ year)
```

Resulting (using `facet_wrap`) plot

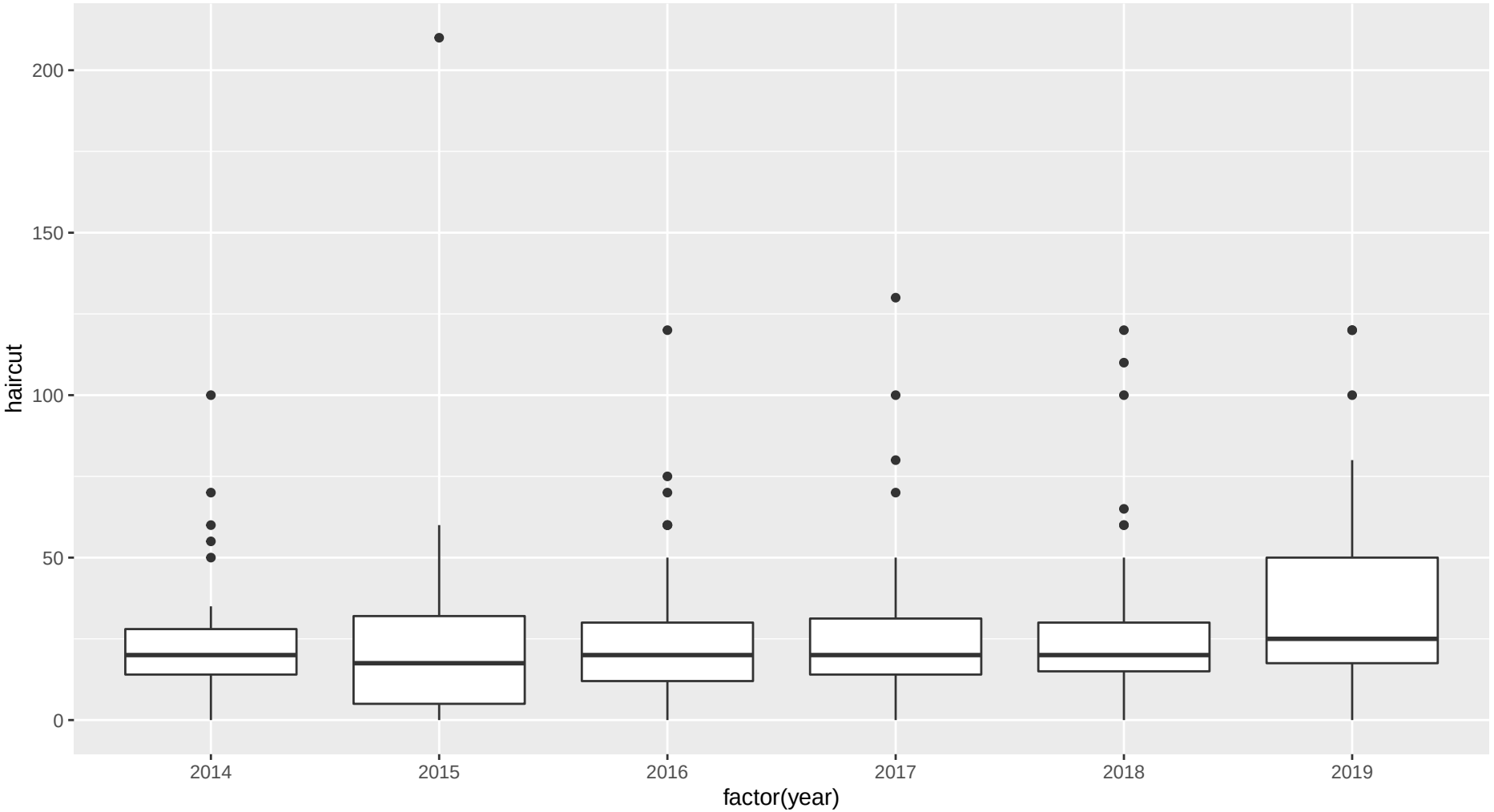


Building a Comparison Boxplot

We could use a comparison boxplot. A trick here is to specify `year` as a factor. . .

```
day1 %>%
    filter(complete.cases(haircut)) %>%
    ggplot(data = .,
           aes(x = factor(year), y = haircut)) +
    geom_boxplot()
```

Comparison Boxplot



Detailed Numerical Summary of Haircut Prices

```
day1 %>%
    select(haircut) %>%
    summary()
```

```
    haircut
 Min.   :  0.00
 1st Qu.: 14.00
 Median : 20.00
 Mean   : 27.32
 3rd Qu.: 32.00
 Max.   :210.00
 NA's   :4
```

which can also be done with

```
summary(day1$haircut)
```

The mosaic package has a useful favstats function...

```
mosaic::favstats(day1$haircut)
```

But to get this in a pipeline, you'd need the %$% operator from the magrittr package...

```
day1 %$%
    mosaic::favstats(haircut)
```

```
Registered S3 method overwritten by 'mosaic':
  method                                from
  fortify.SpatialPolygonsDataFrame ggplot2
```

```
 min Q1 median Q3 max    mean       sd   n missing
   0 14     20 32 210 27.3199 26.35565 311       4
```

The psych package has a useful describe function...

```
day1 %$%
    psych::describe(haircut)
```

```
   vars   n  mean    sd median trimmed   mad min max
X1    1 311 27.32 26.36     20   23.12 14.83   0 210
   range skew kurtosis   se
X1   210 2.38     9.02 1.49
```

The Hmisc package also has a useful describe function...

```
day1 %$%
    Hmisc::describe(haircut)
```

```
haircut
       n  missing distinct     Info     Mean      Gmd
     311        4       50    0.992    27.32    25.36
     .05      .10      .25      .50      .75      .90
       0        0       14       20       32       60
     .95
      80

lowest :   0.0   1.0   3.0   3.5   5.0
highest: 100.0 110.0 120.0 130.0 210.0
```

Numerical Summary by Year?

```
day1 %>%
    filter(!is.na(haircut)) %>%
    group_by(year) %>%
    summarize(n = n(), mean = mean(haircut),
              sd = sd(haircut), median = median(haircut))
```

Numerical Summary by Year (Result)

```
# A tibble: 6 x 5
   year      n  mean    sd median
  <int> <int> <dbl> <dbl>  <dbl>
1  2014    41  23.7  19.8    20
2  2015    49  24.7  32.8    17.5
3  2016    63  23.8  21.5    20
4  2017    48  25.9  25.3    20
5  2018    50  28.3  26.1    20
6  2019    60  35.8  29.1    25
```

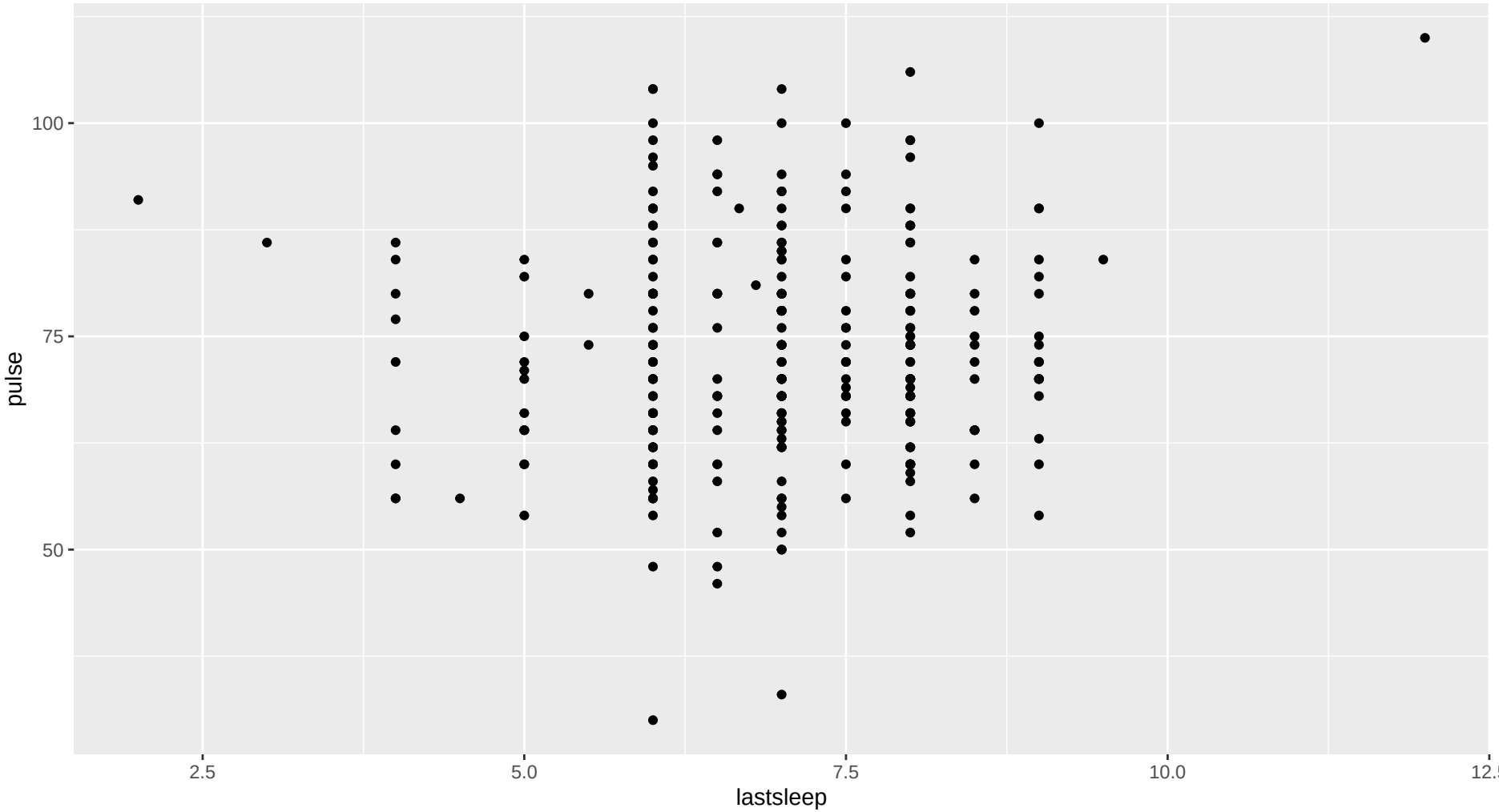
What is the relationship between 431 students' pulse rate and hours of sleep the prior night?

Here, we're looking at two quantitative variables. A **scatterplot** is usually the best choice.

```
ggplot(data = day1, aes(x = lastsleep, y = pulse)) +
    geom_point()
```

Scatterplot (Result)

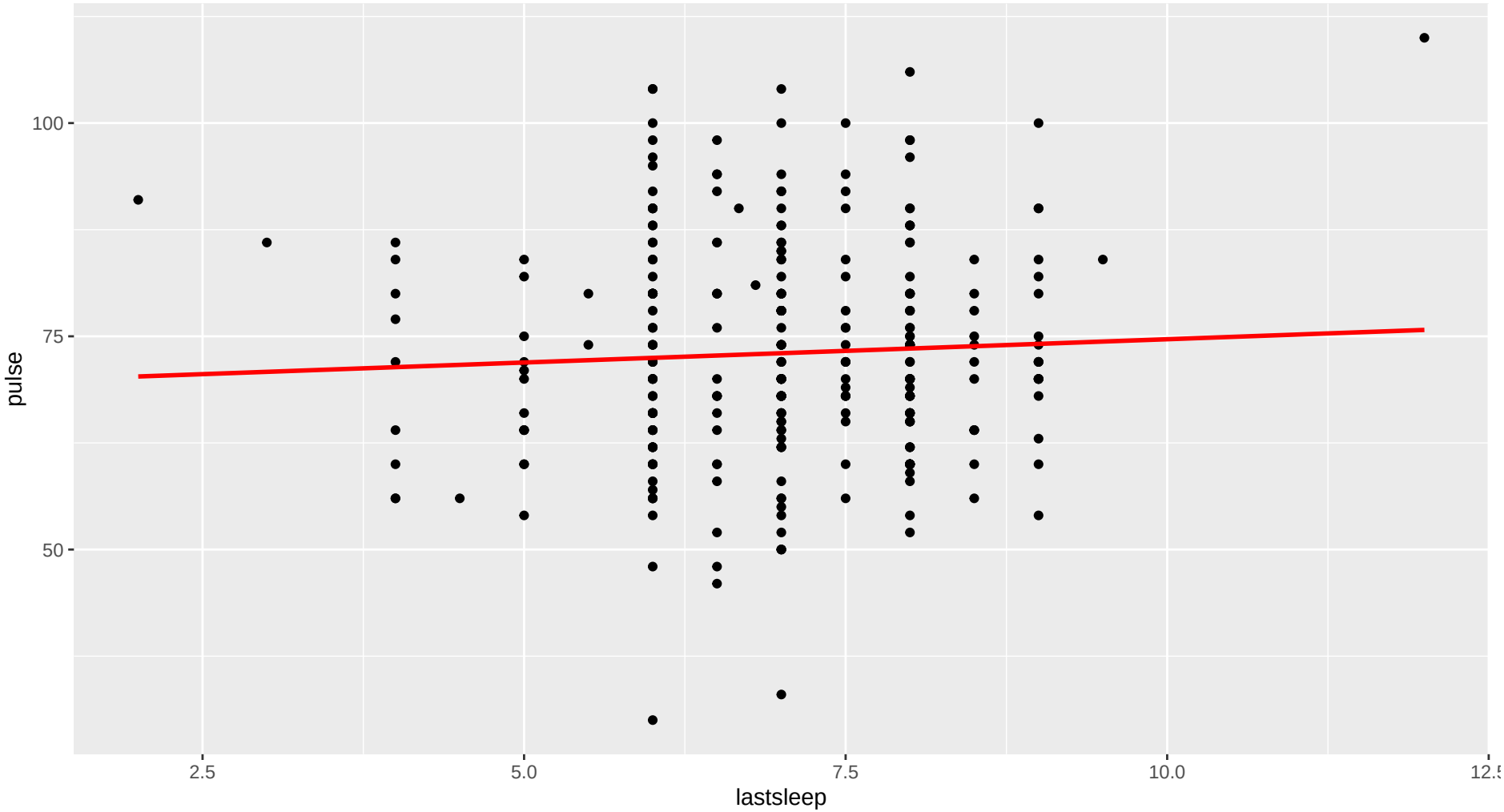
```
Warning: Removed 3 rows containing missing values
(geom_point).
```



Improving the Scatterplot

Let's filter to include only those cases with known `pulse` and known `lastsleep`, and also add a line from a linear regression model to predict pulse rate on the basis of hours of sleep the prior night.

```
day1 %>%
    filter(complete.cases(pulse, lastsleep)) %>%
    ggplot(data = ., aes(x = lastsleep, y = pulse)) +
    geom_point() +
    geom_smooth(method = "lm", se = FALSE, col = "red")
```

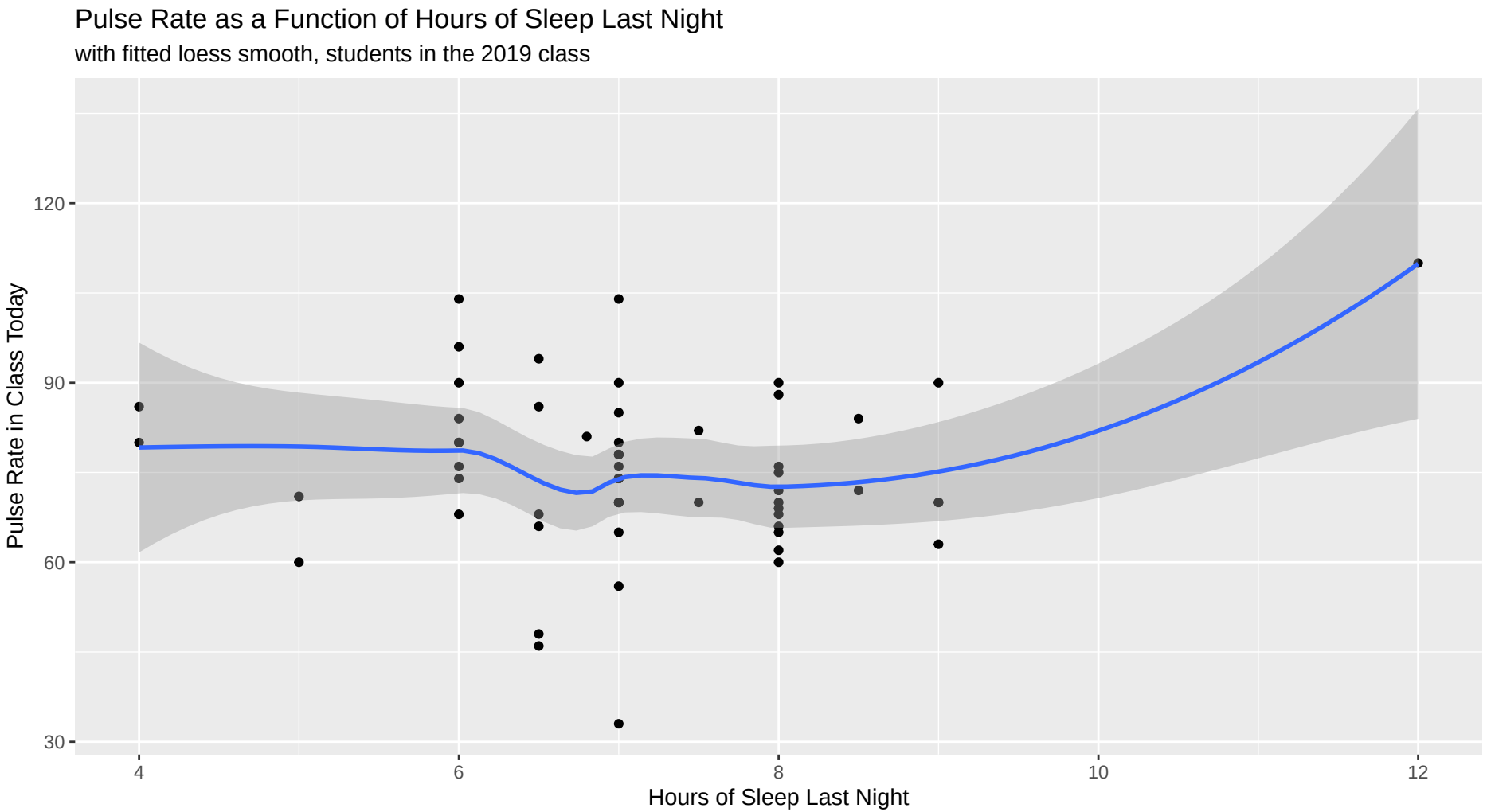

100
75
50
pulse
2.5
5.0
7.5
10.0
12.5
lastsleep

Smoothing the 2019 data

Let's look at the 2019 data only, and fit a curved (loess) smooth to predict pulse rate on the basis of hours of sleep the prior night. We'll also add a title and subtitle and retitle the axes

```
day1 %>%
    filter(year == "2019") %>%
    filter(complete.cases(pulse, lastsleep)) %>%
    ggplot(data = ., aes(x = lastsleep, y = pulse)) +
    geom_point() +
    geom_smooth(method = "loess") +
    labs(title = "Pulse Rate as a Function of Hours of Sleep I
         subtitle = "with fitted loess smooth, students in the
         x = "Hours of Sleep Last Night",
         y = "Pulse Rate in Class Today")
```

The Results

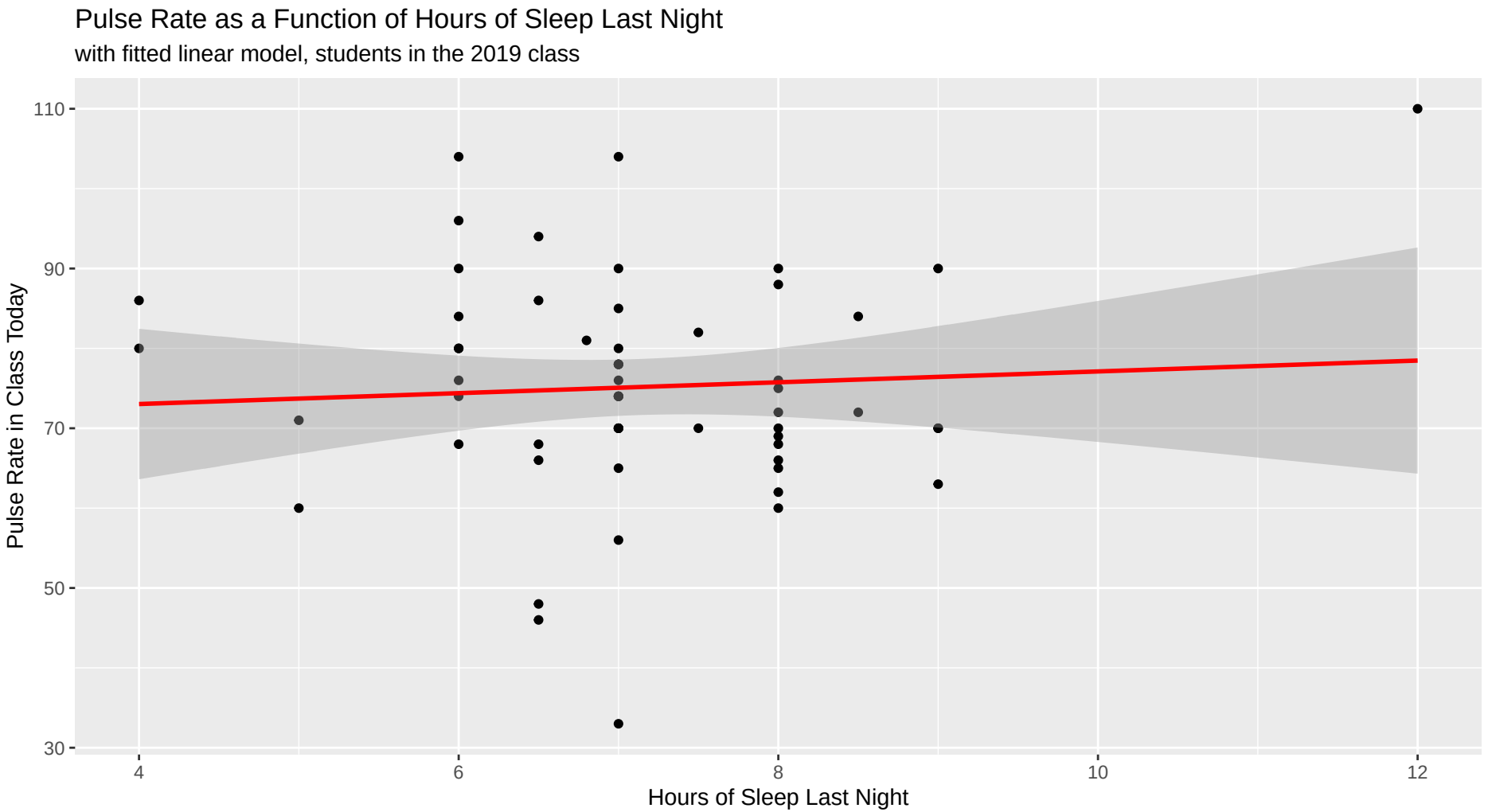

Pulse Rate as a Function of Hours of Sleep Last Night
with fitted loess smooth, students in the 2019 class
Pulse Rate in Class Today
120
90
60
30
4
6
8
10
12
Hours of Sleep Last Night

A Linear Model?

We could instead restrict ourselves to a linear model for the 2019 group.

```
day1 %>%
    filter(year == "2019") %>%
    filter(complete.cases(pulse, lastsleep)) %>%
    ggplot(data = ., aes(x = lastsleep, y = pulse)) +
    geom_point() +
    geom_smooth(method = "lm", col = "red") +
    labs(title = "Pulse Rate as a Function of Hours of Sleep l
         subtitle = "with fitted linear model, students in the
         x = "Hours of Sleep Last Night",
         y = "Pulse Rate in Class Today")
```

Linear Fit (Results)



Correlation?

The correlation of `lastsleep` and `pulse` is likely to be of some interest. Note the use of both the %>% and %$% pipes in this case.

```
day1 %>%
    filter(year == "2019") %>%
    filter(complete.cases(pulse, lastsleep)) %$%
    cor(pulse, lastsleep)
```

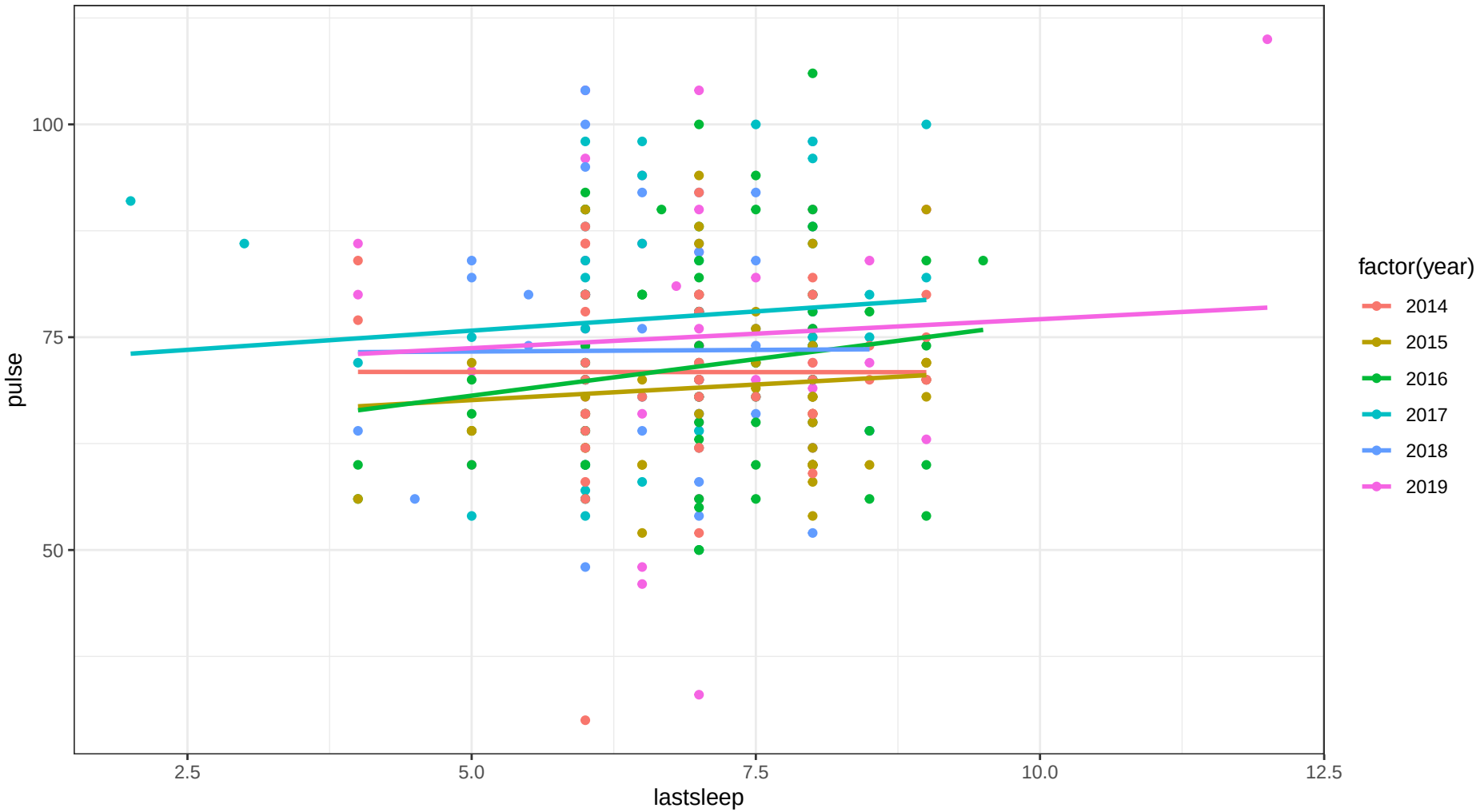
```
[1] 0.06356228
```

Does the linear model change much by year?

Here's the plot, color coding the models by year (note the use of the group as well as the color aesthetic here), and also incorporating the black-and-white theme, rather than the default.

```
day1 %>%
    filter(complete.cases(pulse, lastsleep)) %>%
    ggplot(., aes(x = lastsleep, y = pulse,
                                  color = factor(year),
                                  group = factor(year))) +
    geom_point() +
    geom_smooth(method = "lm", se = FALSE) +
      theme_bw()
```

Does the linear model change much by year?

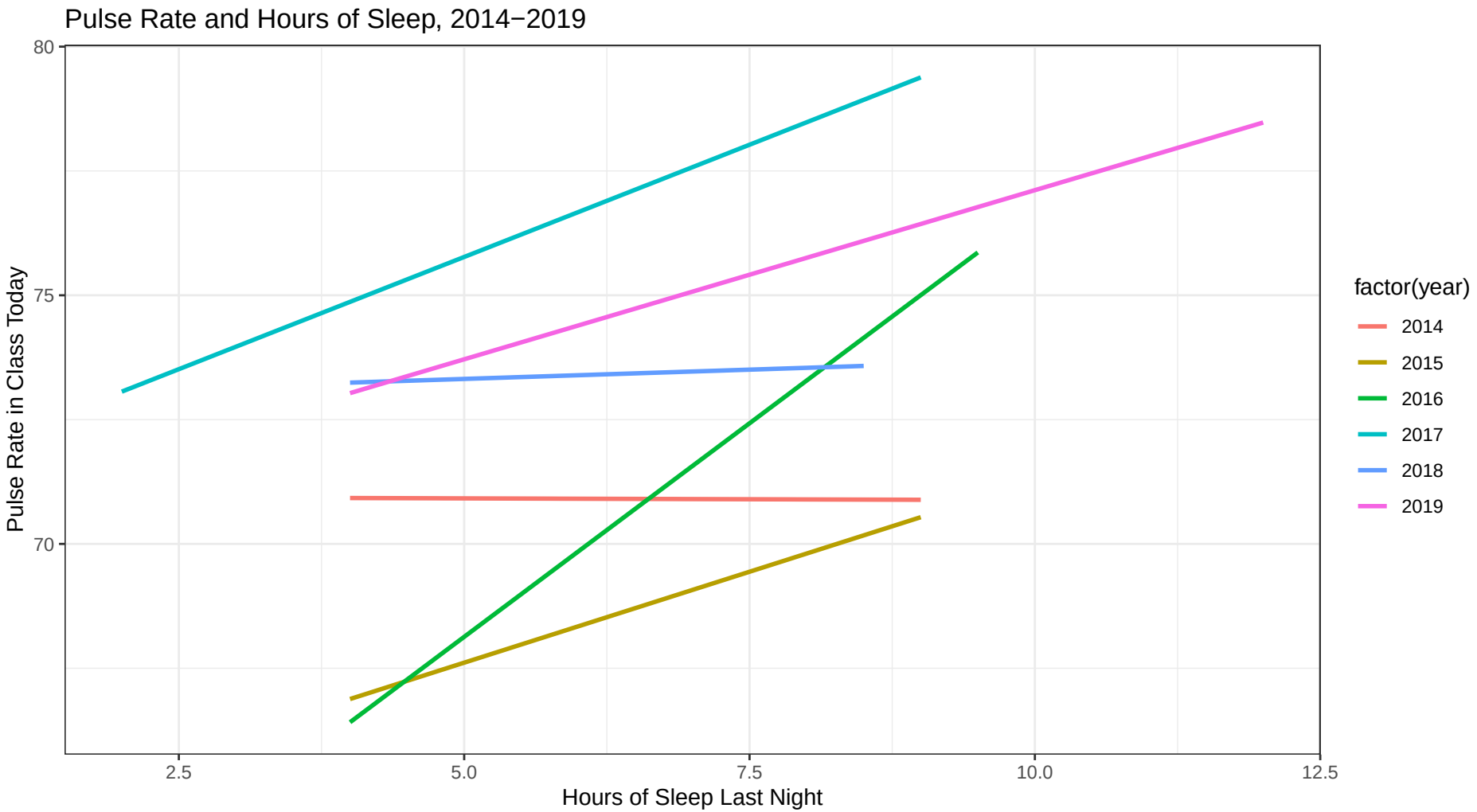


Plot of the models only

Here's the same plot of the models alone, and not showing the data (commenting out the line of code that draws the points.) We'll also improve the labeling.

```
day1 %>%
    filter(complete.cases(pulse, lastsleep)) %>%
    ggplot(., aes(x = lastsleep, y = pulse,
                                  color = factor(year),
                                  group = factor(year))) +
 #   geom_point() +
    geom_smooth(method = "lm", se = FALSE) +
    labs(title = "Pulse Rate and Hours of Sleep, 2014-2019",
         x = "Hours of Sleep Last Night",
         y = "Pulse Rate in Class Today") +
      theme_bw()
```

Plot of the models only

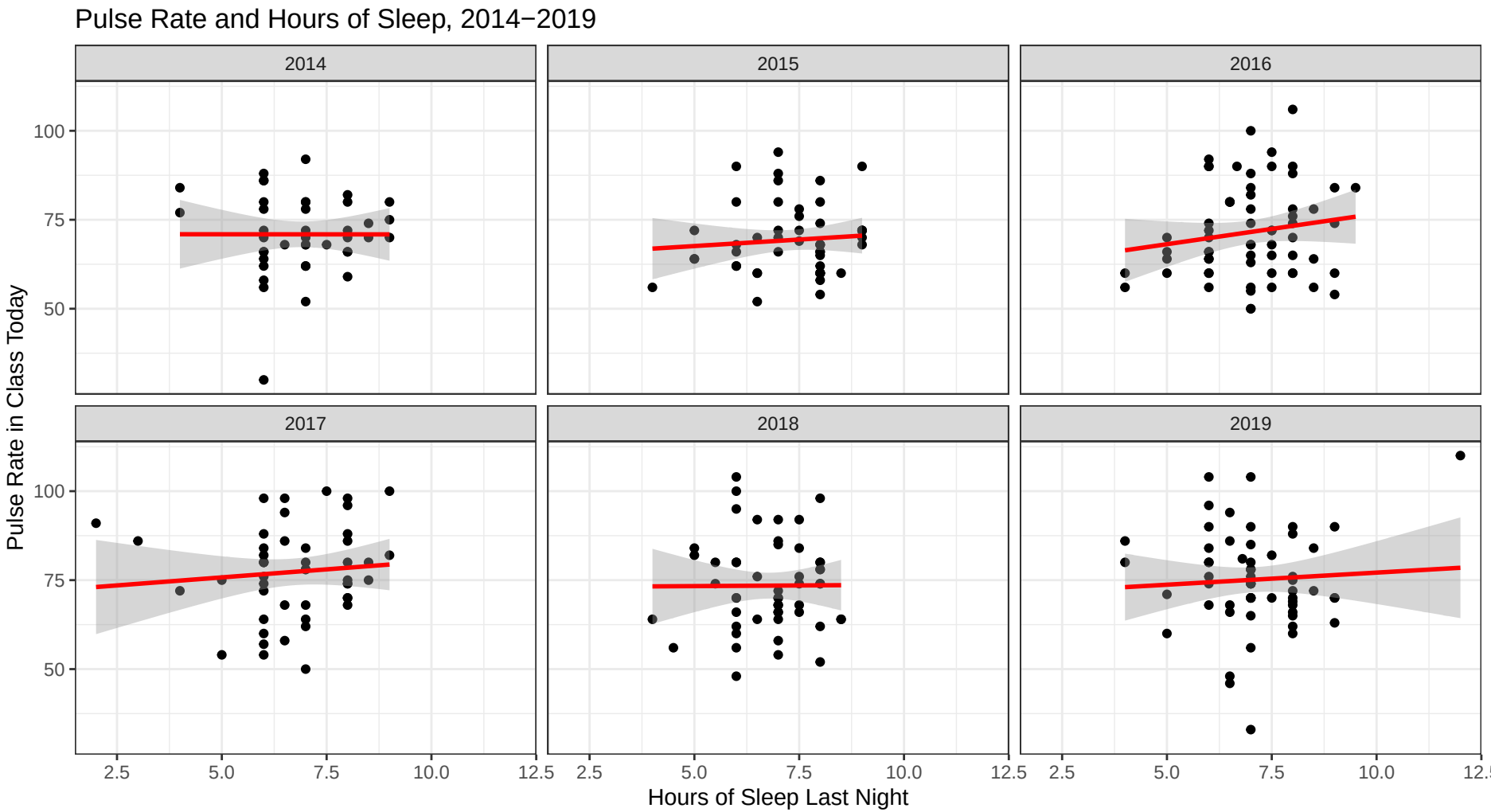


Faceting a Scatterplot

Here's the same basic information, but faceted by year.

```
day1 %>%
    filter(complete.cases(pulse, lastsleep)) %>%
    ggplot(data = ., aes(x = lastsleep, y = pulse,
                                 group = factor(year))) +
    geom_point() +
    geom_smooth(method = "lm", color = "red") +
    facet_wrap(~ year) +
    labs(title = "Pulse Rate and Hours of Sleep, 2014-2019",
         x = "Hours of Sleep Last Night",
         y = "Pulse Rate in Class Today") +
    theme_bw()
```

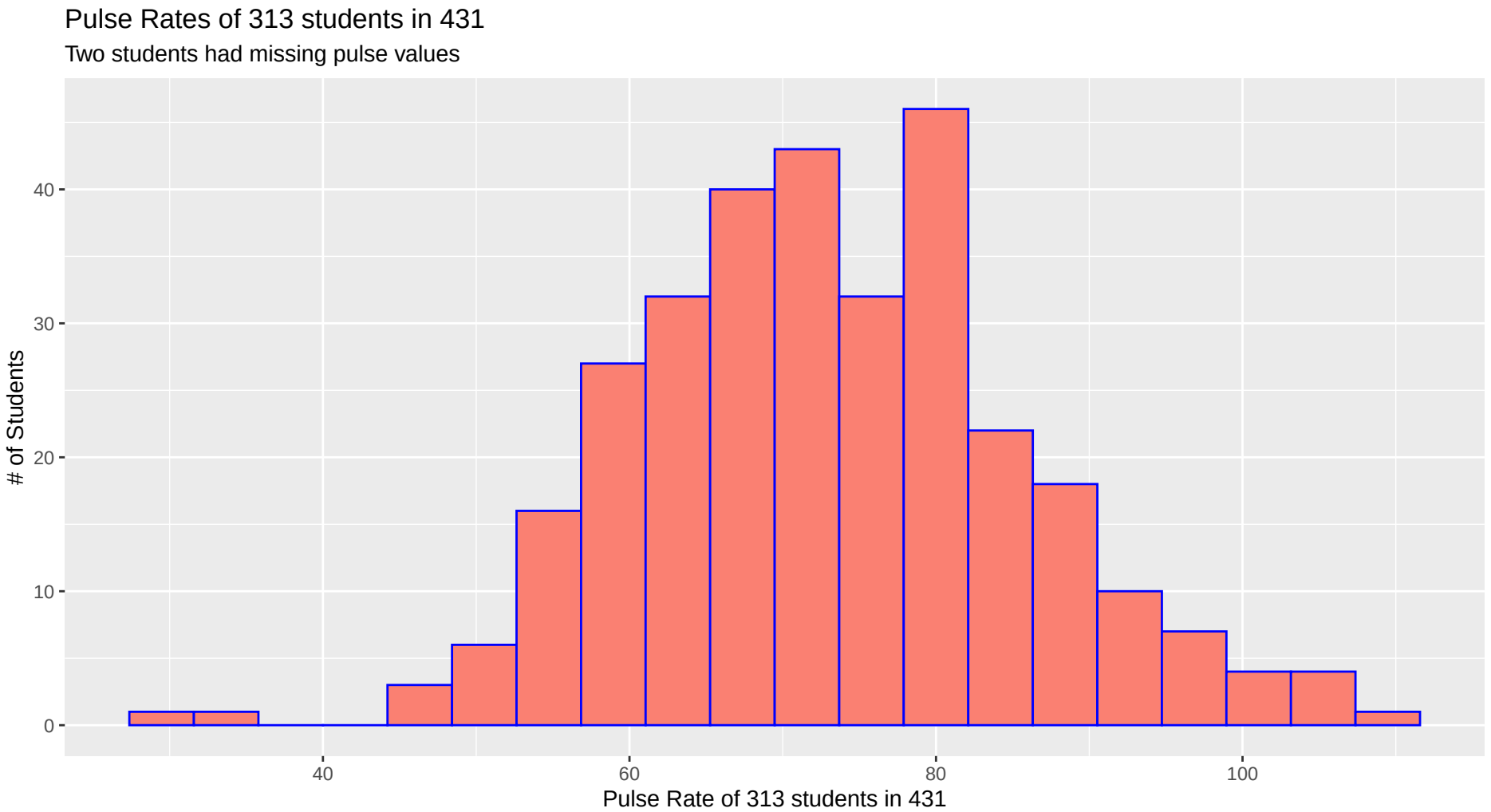
Faceting a Scatterplot



Analyzing the Survey Data - A little challenge

Can you reproduce the following. . .

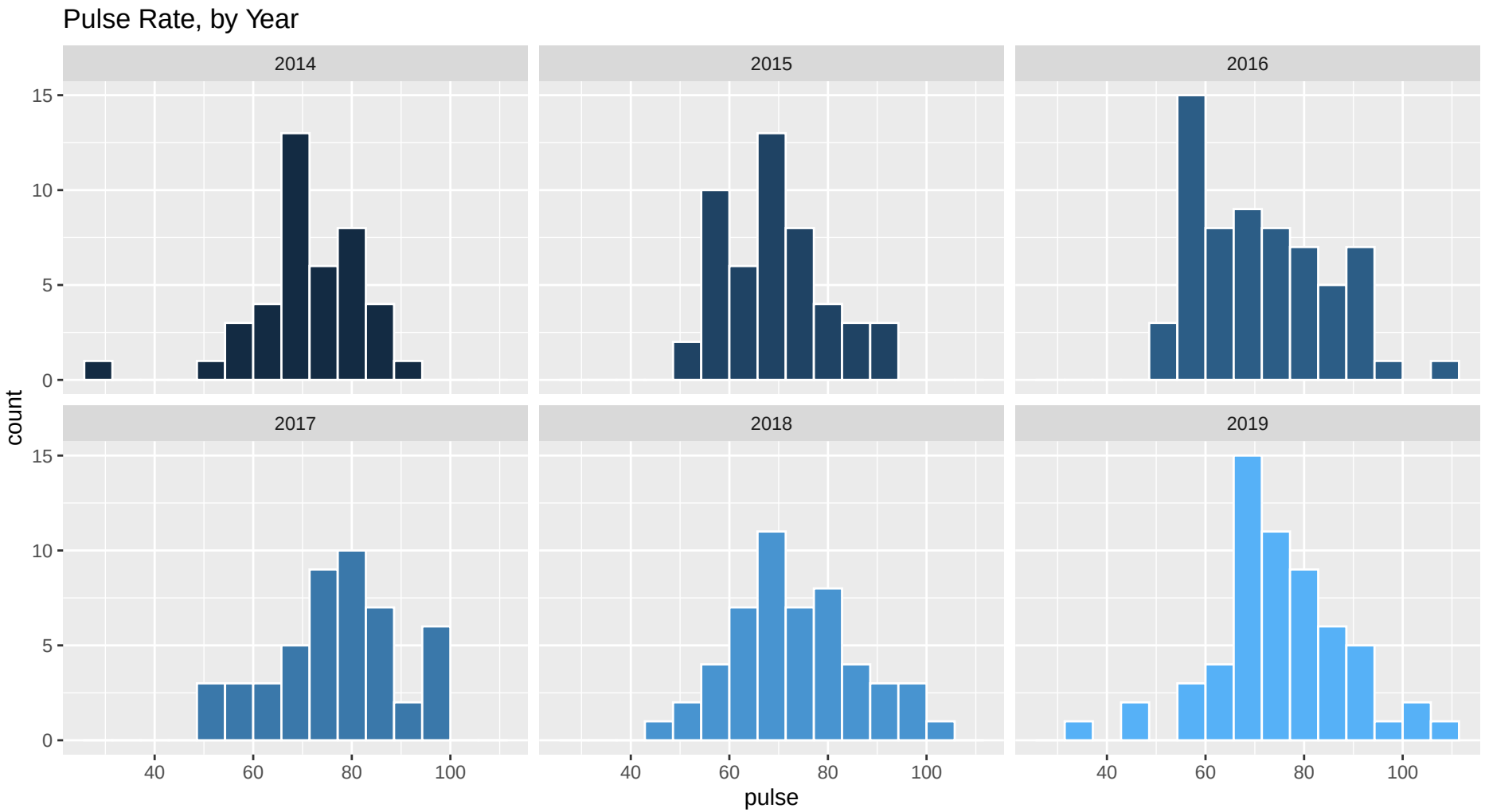
A. That fill color is called *salmon*, I used 20 bins.



Code for Part A.

```
day1 %>% filter(complete.cases(pulse)) %>%
  ggplot(data = ., aes(x = pulse)) +
  geom_histogram(bins = 20, col = "blue", fill = "salmon") +
  labs(x = "Pulse Rate of 313 students in 431",
       y = "# of Students",
       title = "Pulse Rates of 313 students in 431",
       subtitle = "Two students had missing pulse values")
```

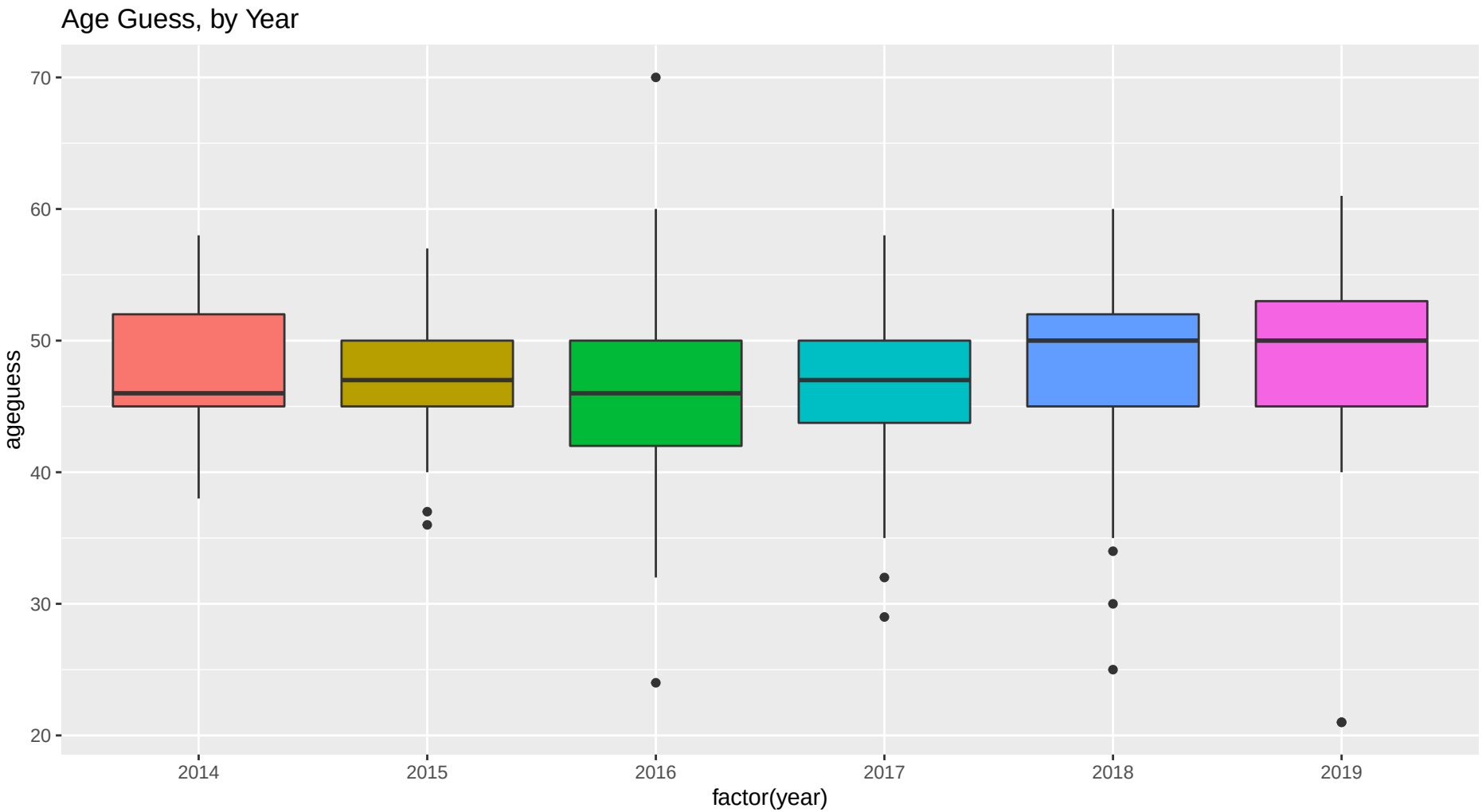
B. Histograms of Pulse Rates, Faceted by Year



Code for Plot B.

```
day1 %>% filter(complete.cases(pulse)) %>%
  ggplot(data = ., aes(x = pulse, fill = year)) +
  geom_histogram(bins = 15, col = "white") +
  facet_wrap(~ year) +
  guides(fill = FALSE) +
  labs(title = "Pulse Rate, by Year")
```

C. Boxplots of Age Guesses, by Year



Code for Plot C

```
day1 %>% filter(complete.cases(ageguess)) %>%
  ggplot(data = ., aes(x = factor(year), y = ageguess,
                       fill = factor(year))) +
  geom_boxplot() +
  guides(fill = FALSE) +
  labs(title = "Age Guess, by Year")
```

Summary Table of Age Guesses, by Year

```
# A tibble: 6 x 5
   year      n  mean    sd median
  <int> <int> <dbl> <dbl>  <dbl>
1  2014    41  47.3  5.21     46
2  2015    49  47.1  4.62     47
3  2016    61  46.0  7.00     46
4  2017    48  46.5  6.15     47
5  2018    50  48.2  6.47     50
6  2019    60  48.6  7.09     50
```

Code for Summary Table

```
day1 %>%
  filter(complete.cases(ageguess)) %>%
  group_by(year) %>%
  summarize(n = n(),
            mean = mean(ageguess, na.rm=TRUE),
            sd = sd(ageguess, na.rm=TRUE),
            median = median(ageguess, na.rm=TRUE)
            )
```