431 Class 05

thomaselove.github.io/431

2021-09-07

You have R Markdown code for all slides

Today's R Packages

```
library(janitor)
library(knitr)
library(magrittr)
library(naniar) # although today's data are complete
library(patchwork)
library(tidyverse) # always load tidyverse last

theme_set(theme_light()) # other TEL option: theme_bw()
```

- I used `{r, message = FALSE}` in the code chunk header to suppress some messages about conflicts between R packages.
- By loading `tidyverse` last, things should work as I expect them to.

Ingesting Today's Data

```
dm431 <- read_csv("data/dm_431.csv",
                  show_col_types = FALSE)
```

- This is a sample of 431 people from a larger pool of data which we'll study later in the term.
- Note the use of `read_csv` instead of `read.csv` here.
- Updating R to version 4.1.1 and updating your packages may help if you're getting an error
- Can also run this without `show_col_types = FALSE` and you'll just get a message which you can suppress by using `{r, message = FALSE}` as the header of your code chunk.

A First Look

```
dm431
```

```
# A tibble: 431 x 17
   CLASS5_ID   AGE INSURANCE N_INCOME    HT    WT   SBP
   <chr>     <dbl> <chr>         <dbl> <dbl> <dbl> <dbl>
 1 S-001        57 Medicare      22139  1.71  91.2   120
 2 S-002        63 Medicaid      39268  1.52  90.6   112
 3 S-003        44 Commerci~     56837  1.6   89.0   118
 4 S-004        56 Uninsured     39962  1.7   88.9   140
 5 S-005        38 Medicaid      40228  1.67 116.    156
 6 S-006        56 Commerci~     43782  1.6  100.    128
 7 S-007        50 Medicaid      39574  1.69  80.9   136
 8 S-008        49 Medicaid      38676  1.71 106.    120
 9 S-009        47 Commerci~     71494  1.67  74.2   121
10 S-010        38 Medicaid      11690  1.49  81.8   131
# ... with 421 more rows, and 10 more variables:
#   DBP <dbl>, A1C <dbl>, LDL <dbl>, TOBACCO <chr>,
#   STATIN <dbl>, EYE_EXAM <dbl>
```

dm431 glimpse at each variable's first few values

```
glimpse(dm431)
```

```
Rows: 431
Columns: 17
$ CLASS5_ID      <chr> "S-001", "S-002", "S-003", "S-~
$ AGE            <dbl> 57, 63, 44, 56, 38, 56, 50, 49~
$ INSURANCE      <chr> "Medicare", "Medicaid", "Comme~
$ N_INCOME       <dbl> 22139, 39268, 56837, 39962, 40~
$ HT             <dbl> 1.71, 1.52, 1.60, 1.70, 1.67, ~
$ WT             <dbl> 91.22, 90.63, 88.96, 88.91, 11~
$ SBP            <dbl> 120, 112, 118, 140, 156, 128, ~
$ DBP            <dbl> 79, 74, 74, 80, 118, 83, 60, 7~
$ A1C            <dbl> 6.2, 5.9, 8.0, 14.3, 7.8, 6.0,~
$ LDL            <dbl> 148, 116, 134, 42, 96, 66, 110~
$ TOBACCO        <chr> "Former", "Never", "Never", "F~
$ STATIN         <dbl> 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, ~
$ EYE_EXAM       <dbl> 0, 1, 0, 0, 1, 1, 1, 1, 0, 0, ~
```

dm431 variable specifications

```
spec(dm431)
```

```
cols(
  CLASS5_ID = col_character(),
  AGE = col_double(),
  INSURANCE = col_character(),
  N_INCOME = col_double(),
  HT = col_double(),
  WT = col_double(),
  SBP = col_double(),
  DBP = col_double(),
  A1C = col_double(),
  LDL = col_double(),
  TOBACCO = col_character(),
  STATIN = col_double(),
  EYE_EXAM = col_double(),
  RACE_ETHNICITY = col_character(),
```

What would improve our data ingest?

- Clean up the variable names so that they are lower case (and, if they had any spaces or other problematic characters, replace those with underscores while also de-duplicating)
- Convert the categorical variables like `insurance` we might wind up analyzing from characters to factors
- Keep the `class5_id` variable (subject codes) as a character variable

Re-ingesting Today's Data

```
dm431 <- read_csv("data/dm_431.csv",
                  show_col_types = FALSE) %>%
  clean_names() %>%
  mutate(across(where(is.character), as_factor)) %>%
  mutate(class5_id = as.character(class5_id))
```

The dm431 data: second time around

```
dm431
```

```
# A tibble: 431 x 17
   class5_id   age insurance n_income    ht    wt   sbp
   <chr>     <dbl> <fct>        <dbl> <dbl> <dbl> <dbl>
 1 S-001        57 Medicare     22139  1.71  91.2   120
 2 S-002        63 Medicaid     39268  1.52  90.6   112
 3 S-003        44 Commerci~    56837  1.6   89.0   118
 4 S-004        56 Uninsured    39962  1.7   88.9   140
 5 S-005        38 Medicaid     40228  1.67 116.    156
 6 S-006        56 Commerci~    43782  1.6  100.    128
 7 S-007        50 Medicaid     39574  1.69  80.9   136
 8 S-008        49 Medicaid     38676  1.71 106.    120
 9 S-009        47 Commerci~    71494  1.67  74.2   121
10 S-010        38 Medicaid     11690  1.49  81.8   131
# ... with 421 more rows, and 10 more variables:
#   dbp <dbl>, a1c <dbl>, ldl <dbl>, tobacco <fct>,
#   statin <dbl>, eye_exam <dbl>
```

dm431 codebook (part 1)

Simulated data to match Better Health Partnership specs.

Variable	Description
class5_id	subject code (S-001 through S-431)
age	subject's age, in years
insurance	primary insurance, 4 levels
n_income	neighborhood median income, in $
ht	height, in meters (2 decimal places)
wt	weight, in kilograms (2 decimal places)
sbp	most recent systolic blood pressure (mm Hg)
dbp	most recent diastolic blood pressure (mm Hg)

dm431 codebook (part 2)

Variable	Description
a1c	most recent Hemoglobin A1c (%, with one decimal)
ldl	most recent LDL cholesterol level (mg/dl)
tobacco	most recent tobacco status, 3 levels
statin	1 = prescribed a statin in past 12m, 0 = not
eye_exam	1 = diabetic eye exam in past 12m, 0 = not
race_ethnicity	race/ethnicity category, 3 levels
sex	all subjects turn out to be Female
county	all subjects turn out to be in Cuyahoga County

- This sample includes 431 female adults living with diabetes in Cuyahoga County who are within a certain age range, and who have complete data on all of the variables listed in this codebook.

Checking for missingness

```
miss_case_table(dm431)
```

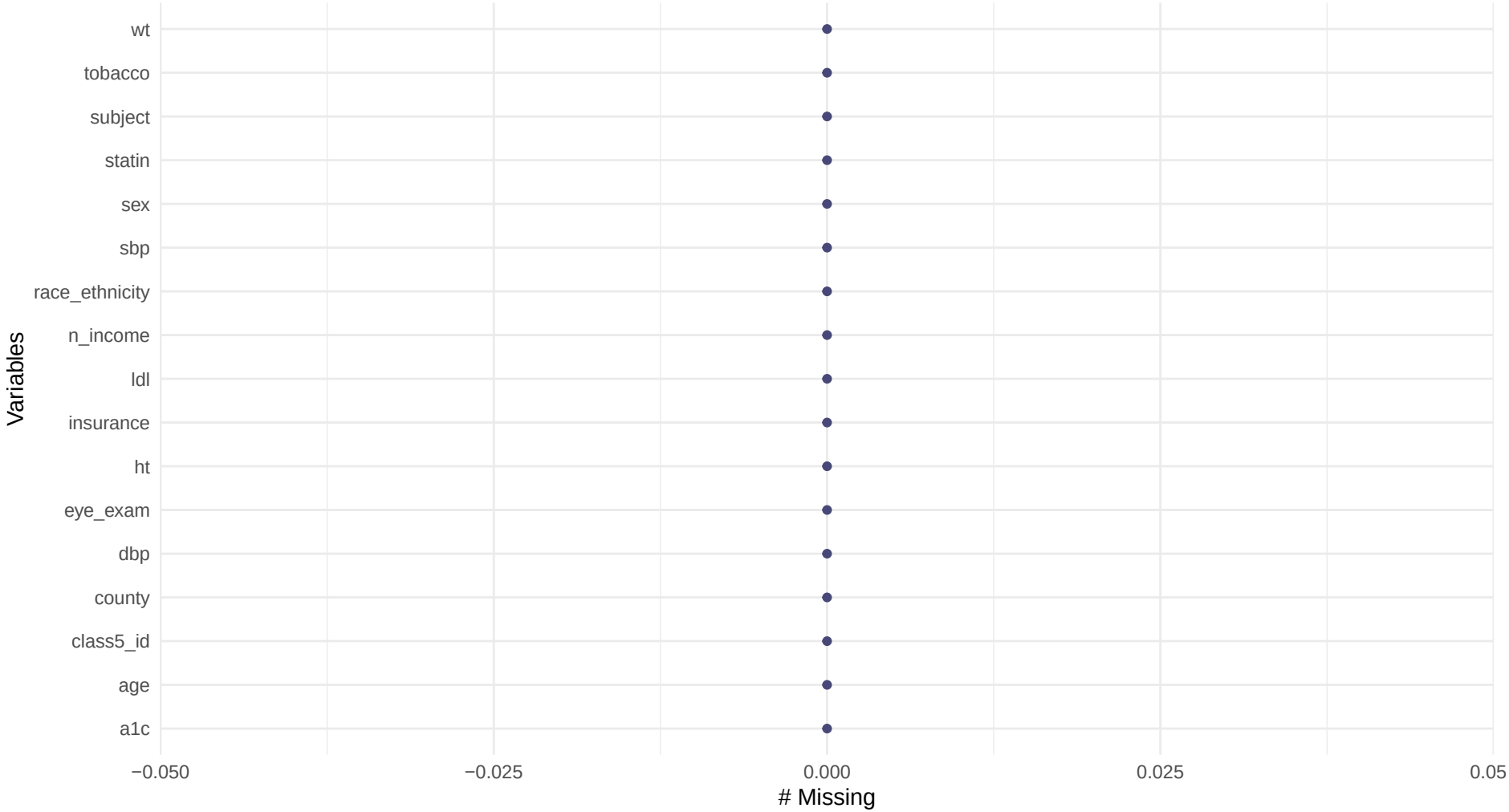
```
# A tibble: 1 x 3
  n_miss_in_case n_cases pct_cases
           <int>   <int>     <dbl>
1              0     431       100
```

Can also use other functions from the `naniar` package to understand and cope with missing values:

- `miss_var_summary()` and `miss_var_table()`
- `gg_miss_var()` as shown on next screen, although that will throw a warning message you can suppress by using `{r, warning = FALSE}` in the chunk header, as I have done.

Plot of missingness in dm431 tibble

```
gg_miss_var(dm431)
```



How old are these women?

- We want to describe the **center**, **spread** (dispersion) and **shape** (symmetry, outliers) of these 431 ages. How do these summaries help?

```
dm431 %$% summary(age)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   30.0    48.0    54.0    52.9    59.0    64.0
```

```
mosaic::favstats(~ age, data = dm431)
```

```
 min Q1 median Q3 max     mean       sd   n missing
  30 48     54 59  64 52.90023 7.993414 431       0
```

```
mosaic::favstats(~ age, data = dm431) %>% kable(digits = 2)
```

min	Q1	median	Q3	max	mean	sd	n	missing
30	48	54	59	64	52.9	7.99	431	0

Summarizing Age with `Hmisc::describe` (1/2)

```
dm431 %$% Hmisc::describe(age)
```

```
age
       n  missing distinct     Info     Mean      Gmd
     431        0       34    0.998     52.9    8.944
     .05      .10      .25      .50      .75      .90
      38       41       48       54       59       62
     .95
      63

lowest : 30 31 32 33 34, highest: 60 61 62 63 64
```

- `Info` is related to how "continuous" the variable is - it's a relative measure of the available information that is reduced below 1 by having lots of ties or non-distinct values
- `Hmisc::describe` treats a numeric variable as discrete if it has 10 or fewer distinct values

CONTINUOUS

measured data, can have ∞ values within possible range.

I AM 3.1" TALL

I WEIGH 34.16 grams

DISCRETE

observations can only exist at limited values, often counts.

I HAVE 8 LEGS and 4 SPOTS!

@allison_horst

Frank Harrell's Hmisc::describe (2/2)

```
dm431 %$% Hmisc::describe(age)
```

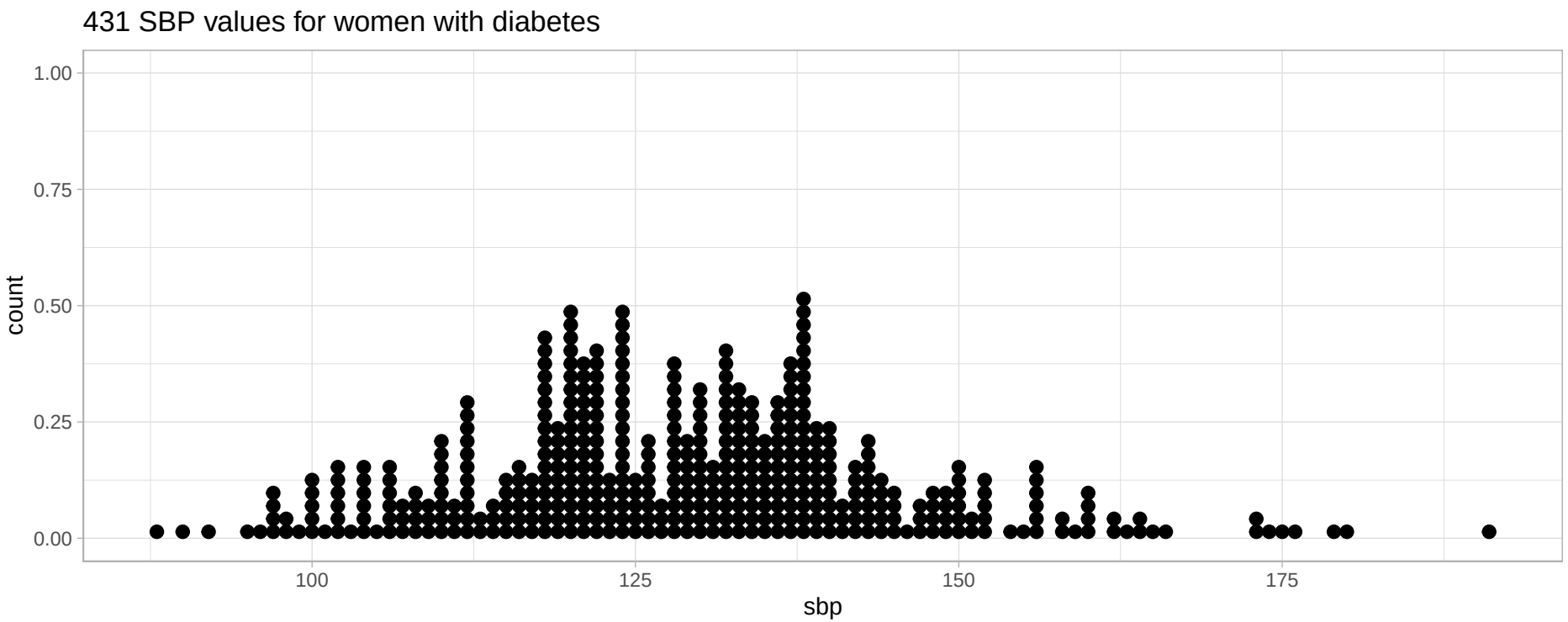
```
age
       n  missing distinct     Info     Mean      Gmd
     431        0       34    0.998     52.9    8.944
     .05      .10      .25      .50      .75      .90
      38       41       48       54       59       62
     .95
      63

lowest : 30 31 32 33 34, highest: 60 61 62 63 64
```

- Gmd = Gini's mean difference is a measure of dispersion (spread) that some people like to use rather than a standard deviation in specific settings. It is the mean absolute difference between any pairs of the 431 observations.

Plotting the `sbp` data to learn about center, spread, outliers and shape

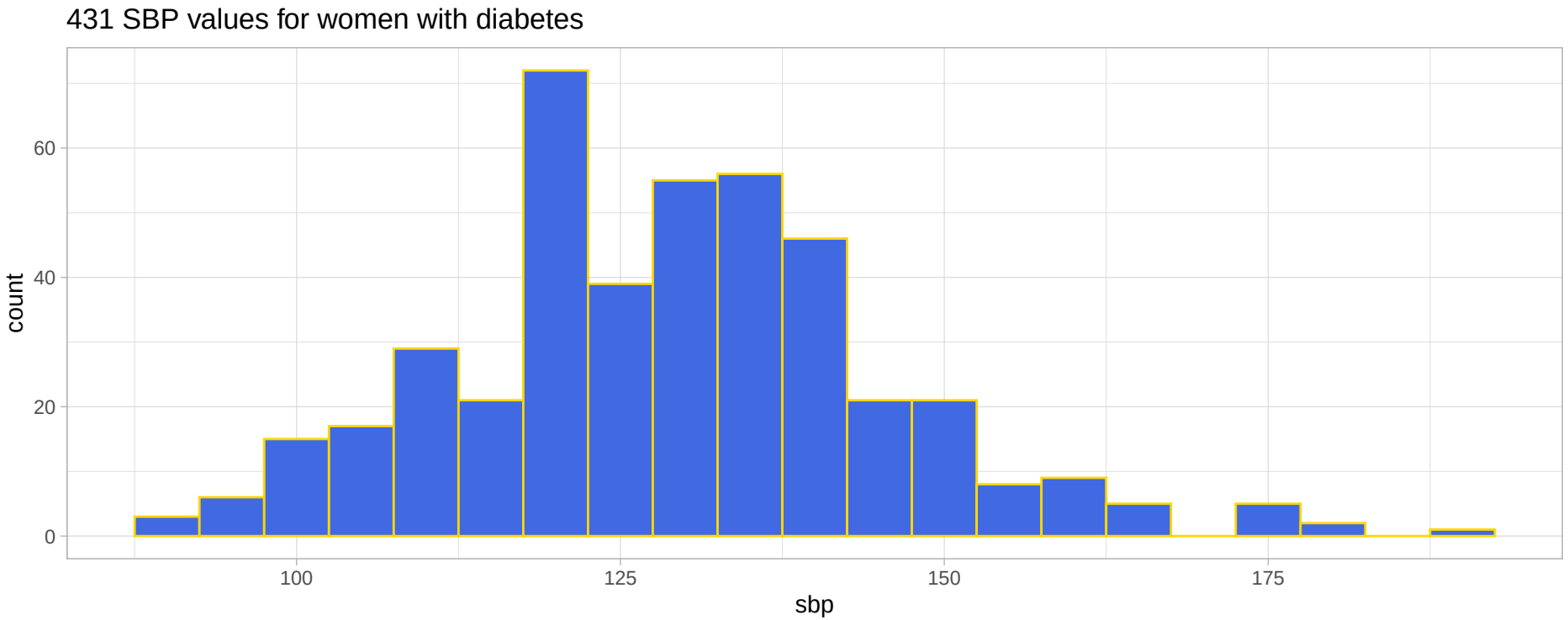
Systolic BP values from dm431 (dotplot)

```
ggplot(data = dm431, aes(x = sbp)) +
  geom_dotplot(binwidth = 1) +
  labs(title = "431 SBP values for women with diabetes")
```



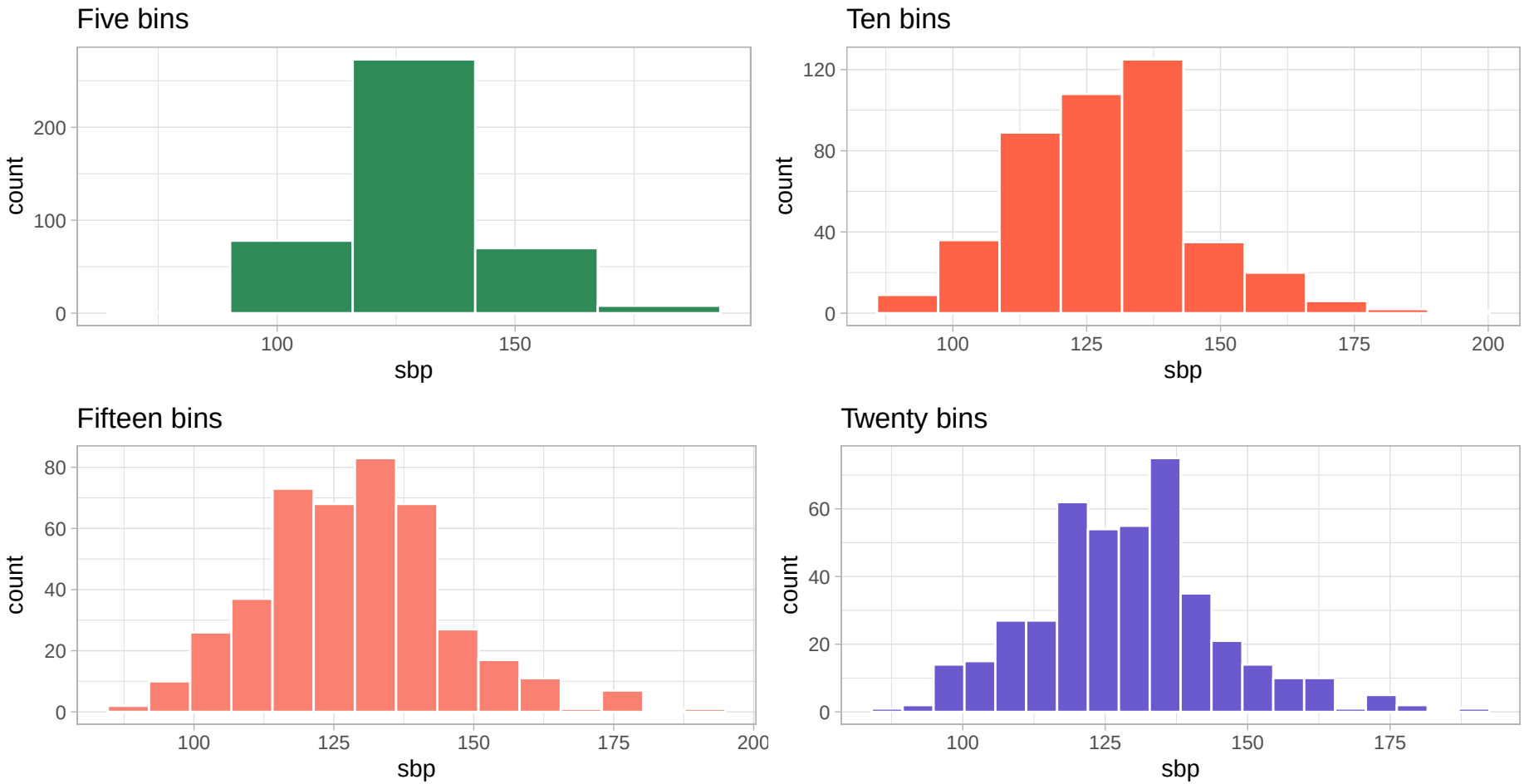
Systolic BP values from dm431 (histogram)

```
ggplot(data = dm431, aes(x = sbp)) +
  geom_histogram(binwidth = 5, fill = "royalblue",
                 col = "gold") +
  labs(title = "431 SBP values for women with diabetes")
```



Number of Bins in a Histogram

431 SBP values for women with diabetes



Code for previous slide

```
p1 <- ggplot(data = dm431, aes(x = sbp)) +
  geom_histogram(bins = 5, fill = "seagreen",
                 col = "white") +
  labs(title = "Five bins")

# omitting the code for plots p2-p4 in this slide,
# use bins = 10, 15 and 20, respectively, and use
# tomato, salmon and slateblue for fill, respectively

(p1 + p2) / (p3 + p4) +
  plot_annotation(
    title = "431 SBP values for women with diabetes")
```

- Remember that you have the R Markdown file for every set of slides.

Can we describe these data as being well-approximated by a Normal model?

What is a Normal Model?

By a Normal model, we mean that the data are assumed to be the result of selecting at random from a probability distribution called the Normal (or Gaussian) distribution, which is characterized by a bell-shaped curve.

- The Normal model is defined by establishing the values of two parameters: the mean and the standard deviation.

When is it helpful to assume our data follow a Normal model?

- When summarizing the data (especially if we want to interpret the mean and standard deviation)
- When creating inferences about populations from samples (as in a t test, or ANOVA)
- When creating regression models, it will often be important to make distributional assumptions about errors, for instance, that they follow a Normal model.

Does a Normal model fit our data "well enough"?

We evaluate whether a Normal model fits sufficiently well to our data on the basis of (in order of importance):

1. Graphs (**DTDP**) are the most important tool we have
   - There are several types of graphs available that are designed to (among other things) help us identify clearly several of the potential problems with assuming Normality.
2. Planned analyses after a Normal model decision is made
   - How serious the problems we see in graphs need to be before we worry about them changes substantially depending on how closely the later analyses we plan to do rely on the assumption of Normality.
3. Numerical Summaries are by far the least important even though they seem "easy-to-use" and "objective".

Simulating Normal data with same Mean and SD

Simulate a sample of 431 observations from a Normal model with mean and standard deviation equal to the mean and standard deviation of our `dm431` systolic BPs.

```
set.seed(20210907)
sim_data <- tibble(
  sbp = rnorm(n = 431,
              mean = mean(dm431$sbp),
              sd = sd(dm431$sbp)))
```

Comparing Summary Statistics

```
mosaic::favstats(~ sbp, data = dm431) %>% kable(dig = 1)
```

min	Q1	median	Q3	max	mean	sd	n	missing
88	119	128	138	191	128.8	16.3	431	0

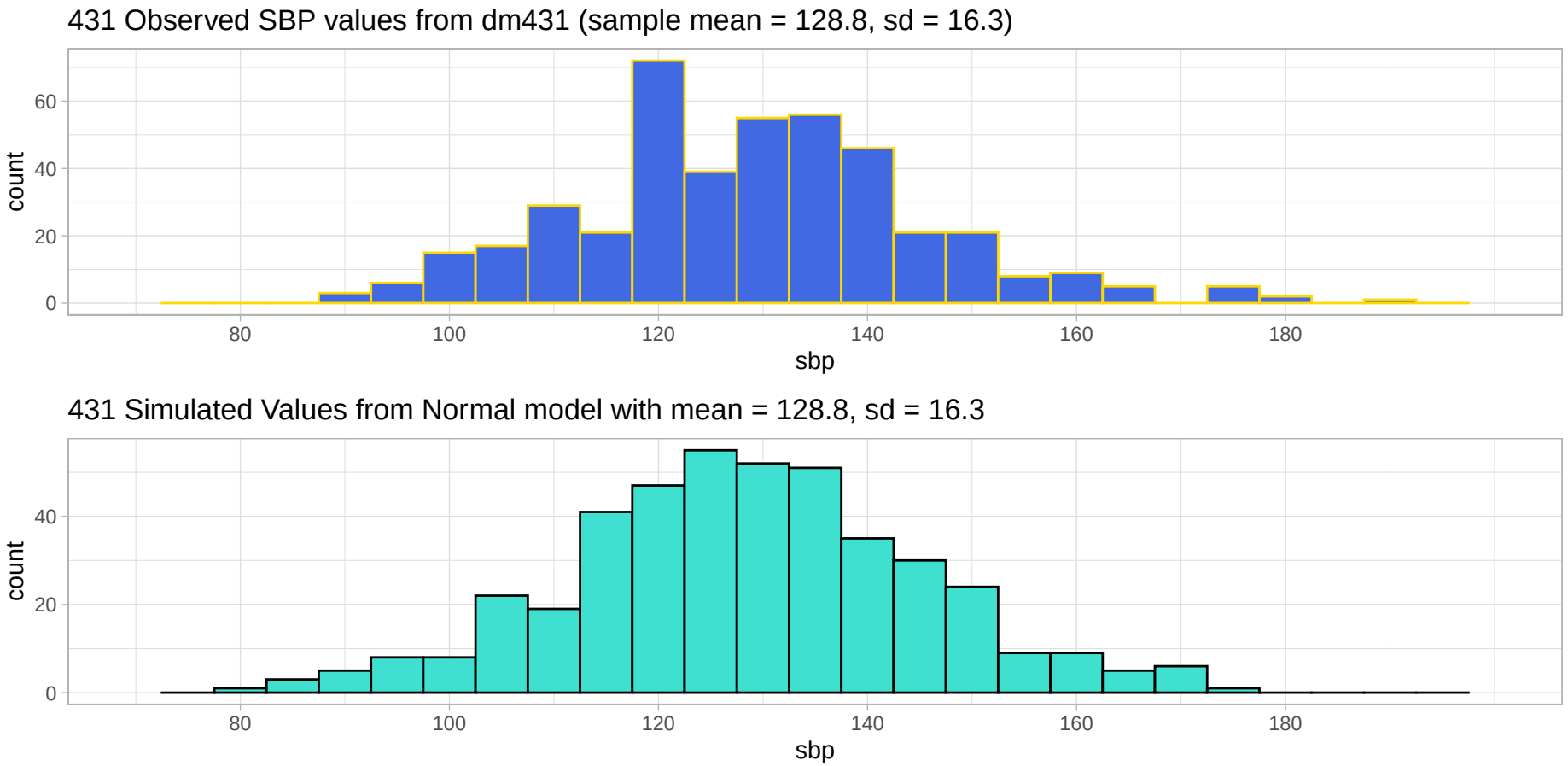
```
mosaic::favstats(~ sbp, data = sim_data) %>% kable(dig = 1)
```

min	Q1	median	Q3	max	mean	sd	n	missing
77.8	117.6	127.9	138.6	175.4	128.2	16.7	431	0

What can we learn from this comparison?

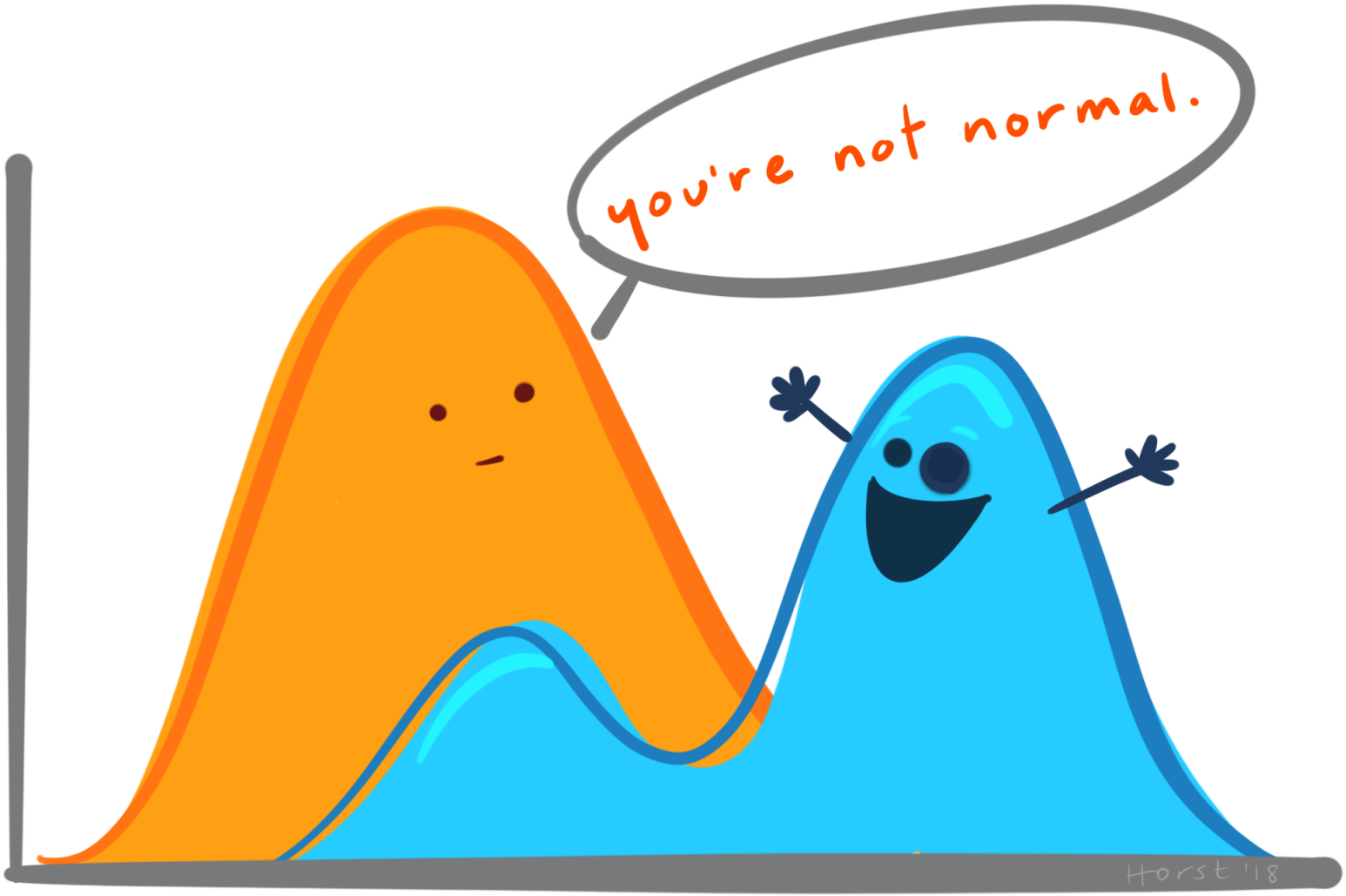
- about the center of the data?
- about the spread of the data?
- about the shape of the data?

Comparing histograms of dm431 and simulated SBP



- Does a Normal model look appropriate for describing the SBP in dm431?

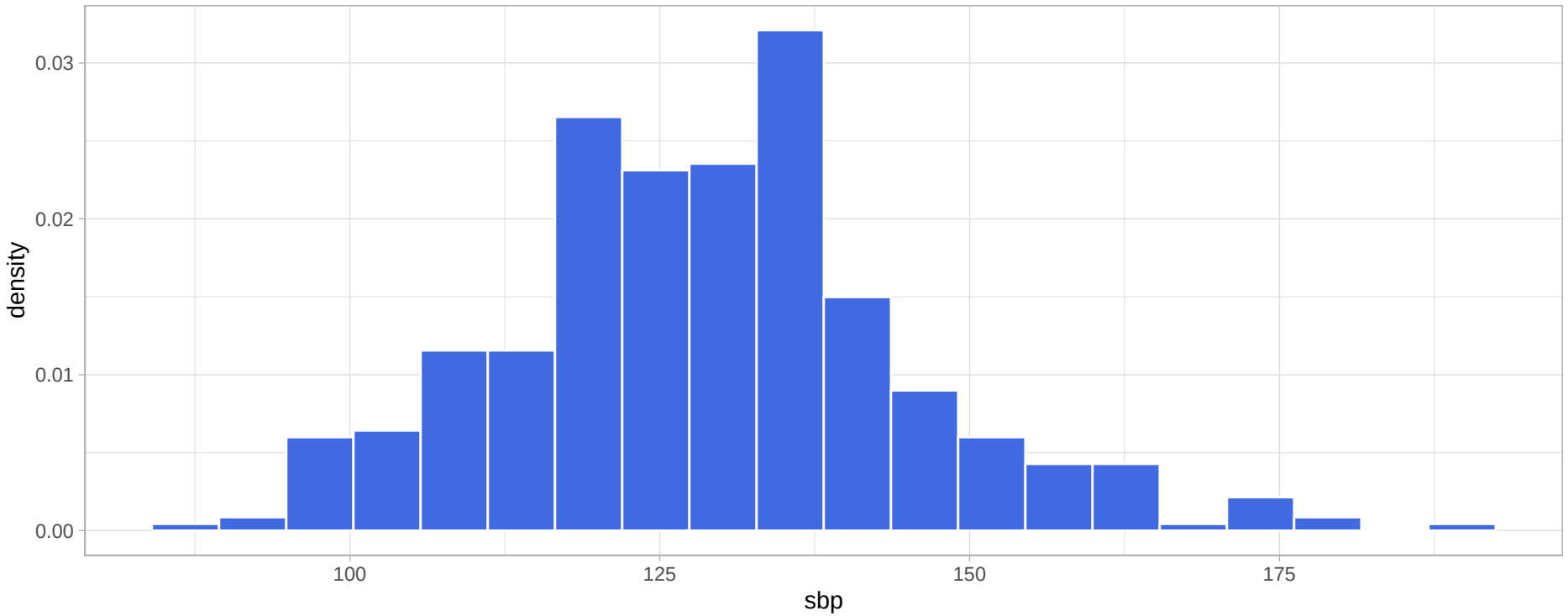
Graphs are our most important tool!



Rescale nh3 SBP histogram as density

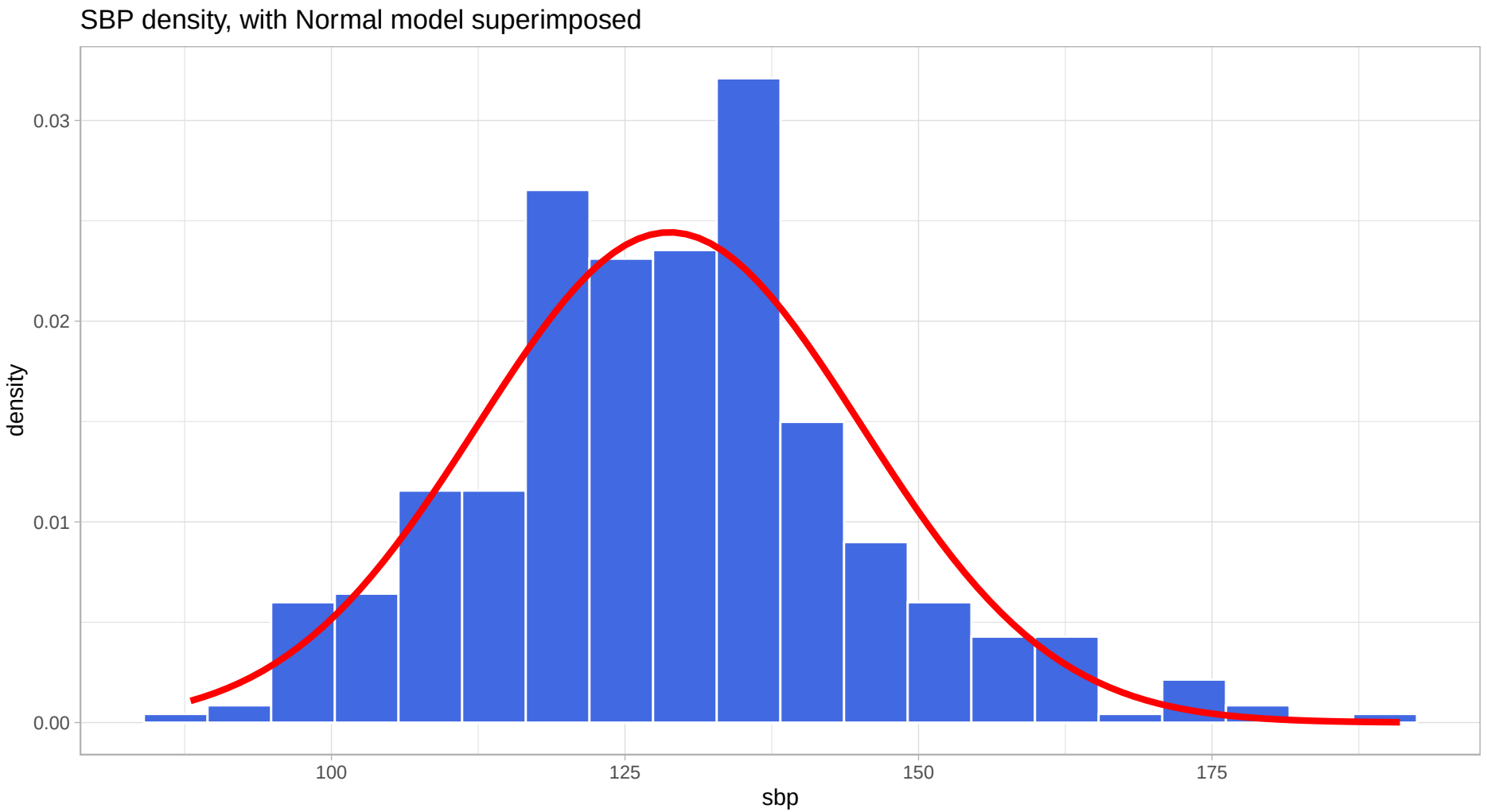
Suppose we want to rescale the histogram counts so that the bar areas integrate to 1. This will let us overlay a Normal density onto the results.

```
ggplot(dm431, aes(x = sbp)) +
  geom_histogram(aes(y = stat(density)), bins = 20,
                 fill = "royalblue", col = "white")
```



Density Function, with Normal superimposed

Now we can draw a Normal density curve on top of the rescaled histogram.

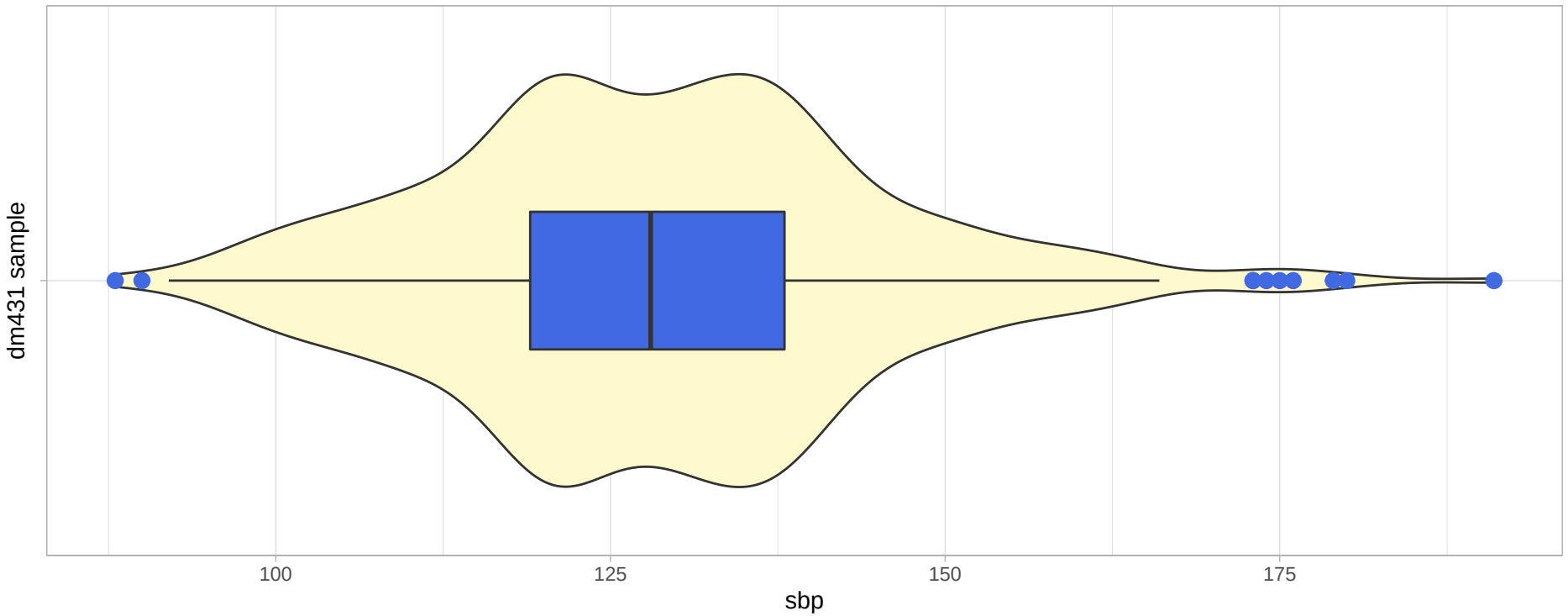


Code for plotting Histogram as Density function

Including the superimposition of a Normal density on top of the histogram.

```
ggplot(dm431, aes(x = sbp)) +
  geom_histogram(aes(y = stat(density)), bins = 20,
                 fill = "royalblue", col = "white") +
  stat_function(fun = dnorm,
                args = list(mean = mean(dm431$sbp),
                            sd = sd(dm431$sbp)),
                col = "red", lwd = 1.5) +
  labs(title = "SBP density, with Normal model superimposed")
```

Violin and Boxplot for `dm431` Systolic BP values

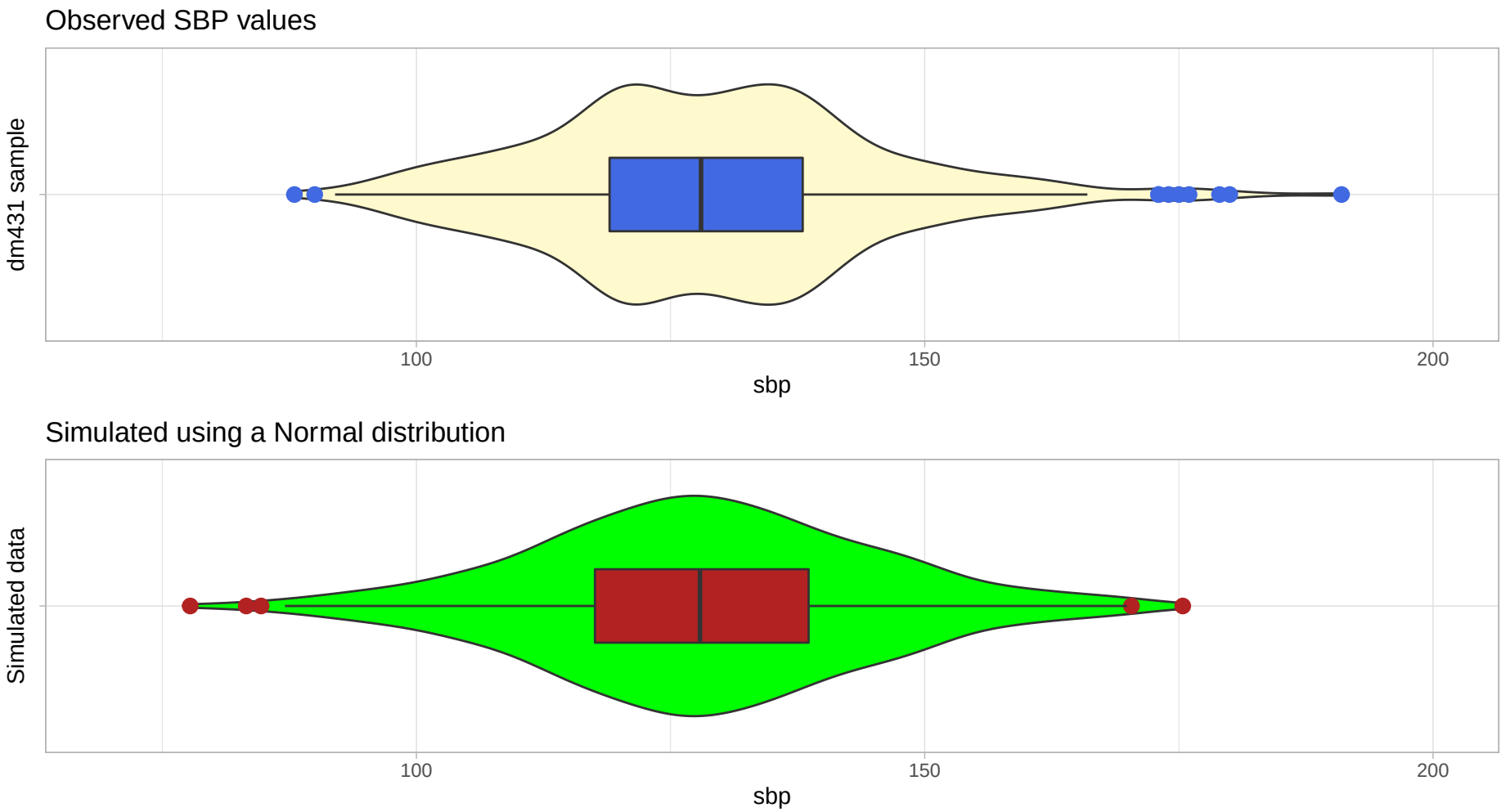


min	Q1	median	Q3	max	mean	sd	n	missing
88	119	128	138	191	128.8	16.3	431	0

Code for Previous Slide

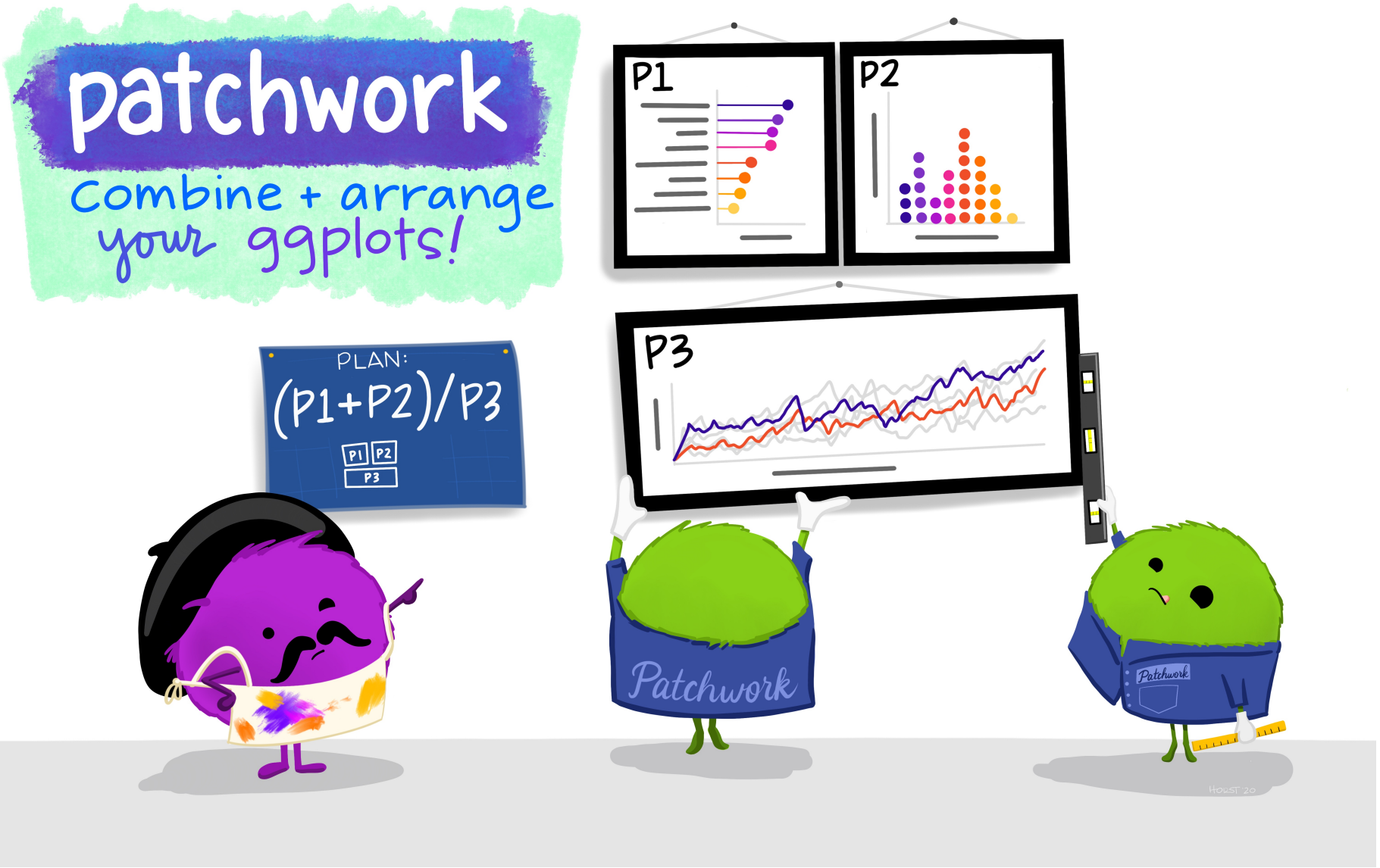
```
ggplot(dm431, aes(x = "", y = sbp)) +
  geom_violin(fill = "lemonchiffon") +
  geom_boxplot(width = 0.3, fill = "royalblue",
               outlier.size = 3,
               outlier.color = "royalblue") +
  coord_flip() +
  labs(x = "dm431 sample")

mosaic::favstats(~ sbp, data = dm431) %>%
  kable(digits = 1)
```

Observed vs. Simulated Systolic BPs



Does a Normal model look appropriate for describing the SBP in `dm431`?

Putting the plots together...



Using Numerical Summaries to Assess Normality: A Good Idea?

Does a Normal model fit well for my data?

The least important approach (even though it is seemingly the most objective) is the calculation of various numerical summaries.

Semi-useful summaries help us understand whether they match up well with the expectations of a normal model:

1. Assessing skewness with $skew_1$ (is the mean close to the median?)
2. Assessing coverage probabilities (do they match the Normal model?)

Quantifying skew with $skew_1$

$$skew_1 = \frac{mean - median}{standard\ \ deviation}$$

Interpreting $skew_1$ (for unimodal data)

- $skew_1 = 0$ if the mean and median are the same
- $skew_1 > 0.2$ indicates fairly substantial right skew
- $skew_1 < -0.2$ indicates fairly substantial left skew

Measuring skewness in the SBP values: dm431?

```
mosaic::favstats(~ sbp, data = dm431)
```

```
 min  Q1 median  Q3 max     mean       sd   n missing
  88 119    128 138 191 128.7889 16.33058 431       0
```

```
dm431 %>%
  summarize(skew1 = (mean(sbp) - median(sbp))/sd(sbp))
```

```
# A tibble: 1 x 1
   skew1
   <dbl>
1 0.0483
```

What does this suggest?

Empirical Rule for a Normal Model

If the data followed a Normal distribution, perfectly, then about:

- 68% of the data would fall within 1 standard deviation of the mean
- 95% of the data would fall within 2 standard deviations of the mean
- 99.7% of the data would fall within 3 standard deviations of the mean

Remember that, regardless of the distribution of the data:

- Half of the data will fall below the median, and half above it.
- Half of the data will fall in the Interquartile Range (IQR).

How many SBPs are within 1 SD of the mean?

```
dm431 %>%
  count(sbp > mean(sbp) - sd(sbp),
        sbp < mean(sbp) + sd(sbp)) %>%
  kable()
```

sbp > mean(sbp) - sd(sbp)	sbp < mean(sbp) + sd(sbp)	n
FALSE	TRUE	70
TRUE	FALSE	55
TRUE	TRUE	306

- Note that $306/431 = 0.71$, approximately.
- How does this compare to the expectation under a Normal model?

SBP and the mean ± 2 standard deviations rule?

The total sample size here is 431

```
dm431 %>%
  count(sbp > mean(sbp) - 2*sd(sbp),
        sbp < mean(sbp) + 2*sd(sbp)) %>%
  kable()
```

sbp > mean(sbp) - 2 * sd(sbp)	sbp < mean(sbp) + 2 * sd(sbp)	n
FALSE	TRUE	5
TRUE	FALSE	15
TRUE	TRUE	411

- Note that 411/431 = 0.95, approximately.
- How does this compare to the expectation under a Normal model?

Should we use hypothesis tests to assess Normality?

Hypothesis Testing to assess Normality

Don't. Graphical approaches are **far** better than hypothesis tests. . .

```
dm431 %$% shapiro.test(sbp)
```

```
    Shapiro-Wilk normality test

data:  sbp
W = 0.98636, p-value = 0.0004525
```

The very small p value indicates that the test finds some indications **against** adopting a Normal model for these data.

- Exciting, huh? But not actually all that useful, alas.

Why not test for Normality?

There are multiple hypothesis testing schemes (Kolmogorov-Smirnov, etc.) and each looks for one specific violation of a Normality assumption. None can capture the wide range of issues our brains can envision, and none by itself is great at its job.

- With any sort of reasonable sample size, the test is so poor at detecting non-normality compared to our eyes, that it finds problems we don't care about and ignores problems we do care about.
- And without a reasonable sample size, the test is essentially useless.

Whenever you *can* avoid hypothesis testing and instead actually plot the data, you should plot the data. Sometimes you can't plot (especially with really big data) but the test should be your very last resort.

Next Time

That's it for today. Coming Up Next...

- Please complete the **Minute Paper** by noon Wednesday.
- Next time, we'll discuss several things, including...
  - Normal Q-Q plots
  - Building confidence intervals for numerical summaries of a single batch of quantitative data
  - Comparing distributions from two batches of quantitative data