431 Class 06

Thomas E. Love, Ph.D.

2022-09-15

Our Agenda

These slides will be used in Class 05 **and** Class 06.

- Setting up the `dm431` data, including blood pressures
- Assessing center, spread, shape of a data batch effectively
- Some other key visualizations and summaries

Our R Packages

```
1 library(broom) # for neatening model results
2 library(janitor)
3 library(naniar) # although today's data are complete
4 library(patchwork)
5 library(tidyverse) # always load tidyverse last
6 
7 theme_set(theme_light()) # other TEL option: theme_bw()
```

- `broom` package will help us neaten model results
- `naniar` package helps us identify missing values
- `theme_light` this time instead of `theme_bw`
- Use `{r, message = FALSE}` in the code chunk header to silence messages about conflicts between R packages.

Code Chunk Header?

```
```{r}
library(janitor)
library(tidyverse)
```
```

vs,

```
```{r, message = FALSE}
library(janitor)
library(tidyverse)
```
```

Without `message = FALSE`?

```
Attaching package: 'janitor'
The following objects are masked from 'package:stats': chisq.test, fisher.test

Registered S3 methods overwritten by 'dbplyr':
  method         from
  print.tbl_lazy
  print.tbl_sql
── Attaching packages ─────────────────────────────────── tidyverse 1.3.2 ──
✓ tibble  3.1.8     ✓ purrr   0.3.4   ✓ tidyr   1.2.0     ✓ stringr 1.4.1
✓ readr   2.1.2     ✓ forcats 0.5.2─
── Conflicts ────── ──────────────────────────────── tidyverse_conflicts() ──
✖ tidyr::expand() masks Matrix::expand()
✖ dplyr::filter() masks stats::filter()
✖ dplyr::lag()    masks stats::lag()
✖ tidyr::pack()   masks Matrix::pack()
✖ tidyr::unpack() masks Matrix::unpack()
```

Ingesting Today's Data

```
1 dm431 <- read_csv("c05/data/dm_431.csv", show_col_types = FALSE)
```

- This is a (simulated) sample of 431 women with diabetes.
- Note the use of `read_csv` instead of `read.csv` here.
  - Can also run this without `show_col_types = FALSE` and you'll get a message (see next slide.)
  - Could instead silence message with `{r, message = FALSE}` in the code chunk header.

Without `show_col_types = FALSE`

```
Rows: 431 Columns: 16

── Column specification ────────────────────────────────────────────────────────

Delimiter: ","

chr  (6): CLASS5_ID, INSURANCE, TOBACCO, RACE_ETHNICITY, SEX, COUNTY
dbl (10): AGE, N_INCOME, HT, WT, SBP, DBP, A1C, LDL, STATIN, EYE_EXAM

ℹ Use `spec()` to retrieve the full column specification for this data.
ℹ Specify the column types or set `show_col_types = FALSE` to quiet this
message.
```

A First Look at the tibble

```
1 dm431
```

```
# A tibble: 431 × 16
   CLASS5_ID   AGE INSURANCE N_INC…¹    HT    WT   SBP   DBP   A1C   LDL
TOBACCO
   <chr>     <dbl> <chr>       <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <chr>
 1 S-001        57 Medicare    22139  1.71  91.2   120    79   6.2   148
Former
 2 S-002        63 Medicaid    39268  1.52  90.6   112    74   5.9   116 Never
 3 S-003        44 Commerci…   56837  1.6   89.0   118    74   8     134 Never
 4 S-004        56 Uninsured   39962  1.7   88.9   140    80  14.3    42
Former
 5 S-005        38 Medicaid    40228  1.67 116.    156   118   7.8    96
Current
 6 S-006        56 Commerci…   43782  1.6  100.    128    83   6      66
Former
 7 S-007        50 Medicaid    39574  1.69  80.9   136    60   6.3   110 Never
```

What is in `dm431`?

Simulated data to match Better Health Partnership specs.

- 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 16 variables included in the tibble.
- Key variables for now include `AGE`, `SBP` and `DBP`.
  - `CLASS5_ID` = identification code.

dm431 variable names

```
1 names(dm431)
```

```
 [1] "CLASS5_ID"      "AGE"            "INSURANCE"      "N_INCOME"
 [5] "HT"             "WT"             "SBP"            "DBP"
 [9] "A1C"            "LDL"            "TOBACCO"        "STATIN"
[13] "EYE_EXAM"       "RACE_ETHNICITY" "SEX"            "COUNTY"
```

Variables we'll use now: AGE, SBP and DBP, mostly.

Details on other variables to come later.

First and last few subjects

```
1 head(dm431, 3)
```

```
# A tibble: 3 × 16
  CLASS5_ID   AGE INSURANCE  N_INC…¹    HT    WT   SBP   DBP   A1C   LDL
TOBACCO
  <chr>     <dbl> <chr>        <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <chr>
1 S-001        57 Medicare     22139  1.71  91.2   120    79   6.2   148
Former
2 S-002        63 Medicaid     39268  1.52  90.6   112    74   5.9   116 Never
3 S-003        44 Commercial   56837  1.6   89.0   118    74   8     134 Never
# … with 5 more variables: STATIN <dbl>, EYE_EXAM <dbl>, RACE_ETHNICITY <chr>,
#   SEX <chr>, COUNTY <chr>, and abbreviated variable name ¹N_INCOME
```

```
1 tail(dm431, 2)
```

```
# A tibble: 2 × 16
  CLASS5_ID   AGE INSURANCE  N_INC…¹    HT    WT   SBP   DBP   A1C   LDL
TOBACCO
  <chr>     <dbl> <chr>        <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <chr>
1 S-430        59 Uninsured    60335  1.58  99.5   120    82   8.8   106 Never
2 S-431        51 Commercial   23980  1.62  88.3   121    73   6.6    96
Former
# … with 5 more variables: STATIN <dbl>, EYE_EXAM <dbl>, RACE_ETHNICITY <chr>,
#   SEX <chr>, COUNTY <chr>, and abbreviated variable name ¹N_INCOME
```

dm431 glimpse (first few values)

```
1 glimpse(dm431)
```

```
Rows: 431
Columns: 16
$ CLASS5_ID      <chr> "S-001", "S-002", "S-003", "S-004", "S-005", "S-006",
"…
$ AGE            <dbl> 57, 63, 44, 56, 38, 56, 50, 49, 47, 38, 64, 56, 38,
47,…
$ INSURANCE      <chr> "Medicare", "Medicaid", "Commercial", "Uninsured",
"Med…
$ N_INCOME       <dbl> 22139, 39268, 56837, 39962, 40228, 43782, 39574,
38676,…
$ HT             <dbl> 1.71, 1.52, 1.60, 1.70, 1.67, 1.60, 1.69, 1.71, 1.67,
1…
$ WT             <dbl> 91.22, 90.63, 88.96, 88.91, 115.76, 100.33, 80.88,
105.…
$ SBP            <dbl> 120, 112, 118, 140, 156, 128, 136, 120, 121, 131, 125,
```

What would improve our data ingest?

- Clean up the variable names so that they are lower case
  - If names have spaces or other problematic characters, replace them with underscores and also de-duplicate.
- Convert categorical variables like `insurance` we might wind up analyzing from characters to factors.
- Keep `class5_id` subject codes as characters.

Re-ingesting Today's Data

```
dm431 <- read_csv("c05/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 `across(where())` syntax tells R to change everything that gives a TRUE response to "is this a character variable?" into a factor variable.
- We want `class5_id` to be a character so we don't accidentally analyze it.
- `clean_names()` comes from the janitor package. —>

What does `clean_names()` do?

- Resulting names are unique, and use only numbers, letters and underscores.
- Accented characters are transliterated to ASCII.
- `case` parameter specifies preferences (default is snake)
  - `clean_names(case = "snake")` yields `snake_case`
  - "lower_camel" produces `lowerCamel`
  - "upper_camel" produces `UpperCamel`
  - "screaming_snake" yields `ALL_CAPS`

The dm431 data, version 2

```
1 dm431
```

```
# A tibble: 431 × 16
   class5_id   age insurance n_inc…¹    ht    wt   sbp   dbp   a1c   ldl
tobacco
   <chr>     <dbl> <fct>       <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <fct>
 1 S-001        57 Medicare    22139  1.71  91.2   120    79   6.2   148
Former
 2 S-002        63 Medicaid    39268  1.52  90.6   112    74   5.9   116 Never
 3 S-003        44 Commerci…   56837  1.6   89.0   118    74   8     134 Never
 4 S-004        56 Uninsured   39962  1.7   88.9   140    80  14.3    42
Former
 5 S-005        38 Medicaid    40228  1.67 116.    156   118   7.8    96
Current
 6 S-006        56 Commerci…   43782  1.6  100.    128    83   6      66
Former
 7 S-007        50 Medicaid    39574  1.69  80.9   136    60   6.3   110 Never
```

dm431 codebook (part 1)

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

`dm431` codebook (part 2)

Variable	Description
`ht`	height, in meters (2 decimal places)
`wt`	weight, in kilograms (2 decimal places)
`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, else 0

dm431 codebook (part 3)

Variable	Description
eye_exam	1 = diabetic eye exam in past 12m, else 0
race_ethnicity	race/ethnicity category, 3 levels
sex	all subjects turn out to be Female
county	all subjects live in Cuyahoga County

- Again, these are 431 female adults living with diabetes in Cuyahoga County within a certain age range, with complete data on the 16 variables in this codebook.

New dm431 variable structure

```
1 str(dm431)
```

```
tibble [431 × 16] (S3: tbl_df/tbl/data.frame)
 $ class5_id     : chr [1:431] "S-001" "S-002" "S-003" "S-004" ...
 $ age           : num [1:431] 57 63 44 56 38 56 50 49 47 38 ...
 $ insurance     : Factor w/ 4 levels "Medicare","Medicaid",..: 1 2 3 4 2 3 2
2 3 2 ...
 $ n_income      : num [1:431] 22139 39268 56837 39962 40228 ...
 $ ht            : num [1:431] 1.71 1.52 1.6 1.7 1.67 1.6 1.69 1.71 1.67 1.49
...
 $ wt            : num [1:431] 91.2 90.6 89 88.9 115.8 ...
 $ sbp           : num [1:431] 120 112 118 140 156 128 136 120 121 131 ...
 $ dbp           : num [1:431] 79 74 74 80 118 83 60 77 82 85 ...
 $ a1c           : num [1:431] 6.2 5.9 8 14.3 7.8 6 6.3 11.9 10.4 5.6 ...
 $ ldl           : num [1:431] 148 116 134 42 96 66 110 129 101 128 ...
 $ tobacco       : Factor w/ 3 levels "Former","Never",..: 1 2 2 1 3 1 2 2 1 2
...
 $ statin        : num [1:431] 1 0 1 1 1 1 1 1 1 1
```

Checking for missingness

```
1 miss_case_table(dm431)
```

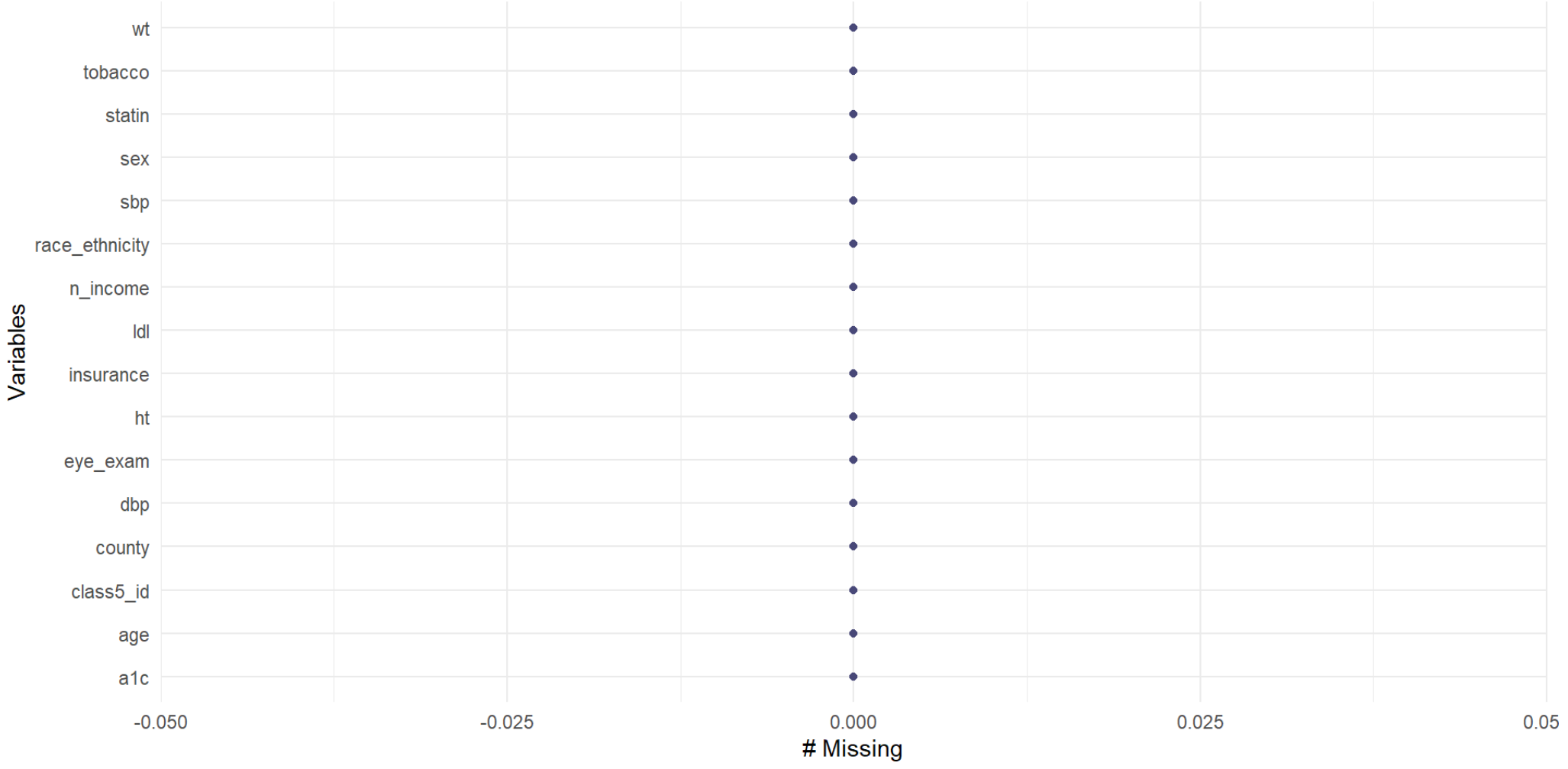
```
# A tibble: 1 × 3
  n_miss_in_case n_cases pct_cases
           <int>   <int>     <dbl>
1              0     431       100
```

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

- `miss_var_summary()` and `miss_var_table()`
- Next slide shows `gg_miss_var()` result —>

Plot of missingness in dm431 tibble

```
1 gg_miss_var(dm431)
```



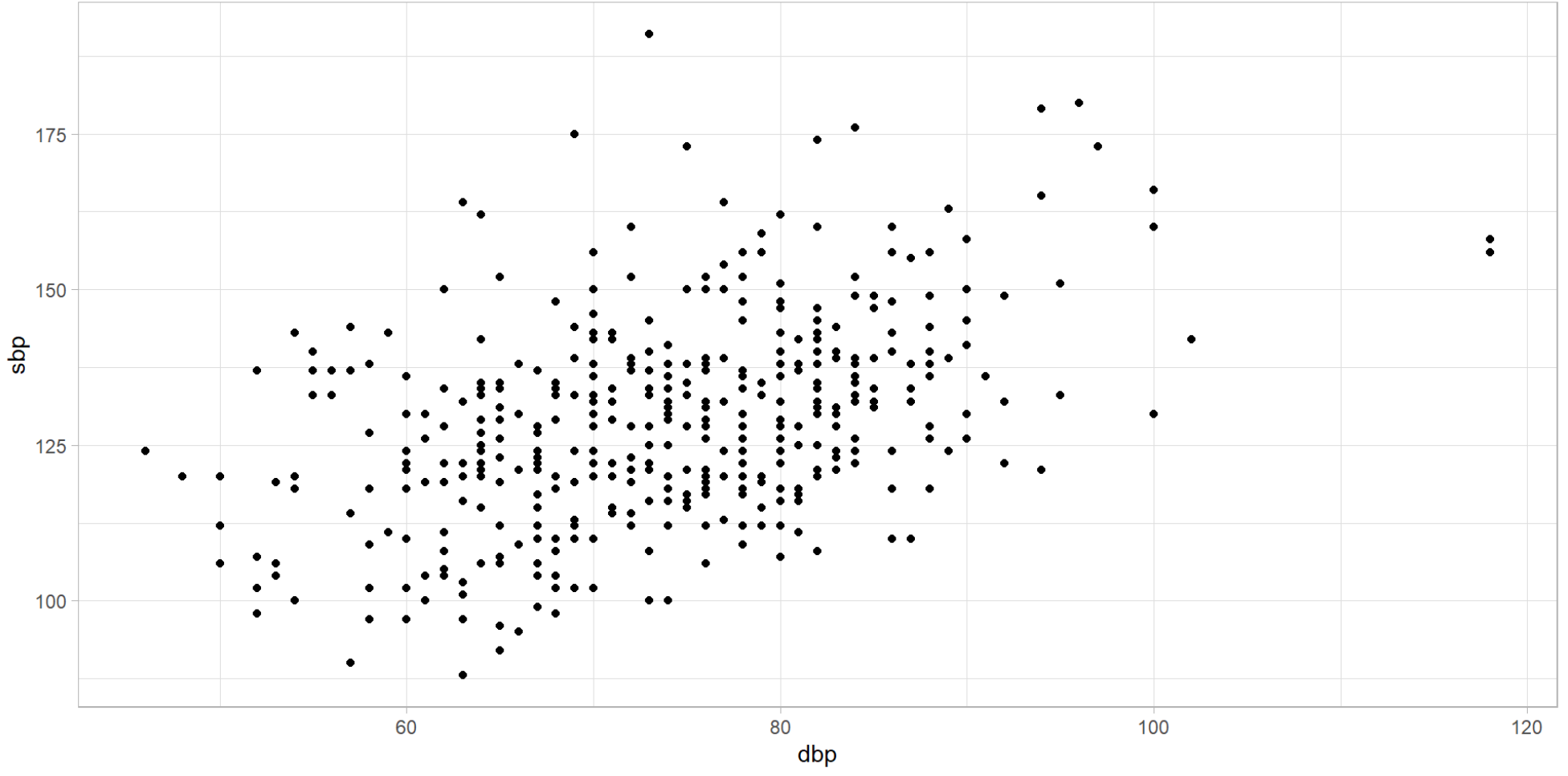
Systolic and Diastolic BP

> Systolic blood pressure, the top number, measures the force the heart exerts on the walls of the arteries each time it beats. Diastolic blood pressure, the bottom number, measures the force the heart exerts on the walls of the arteries in between beats. (Mayo Clinic)

Question: What is the nature of the relationship between SBP and DBP in the `dm431` data?

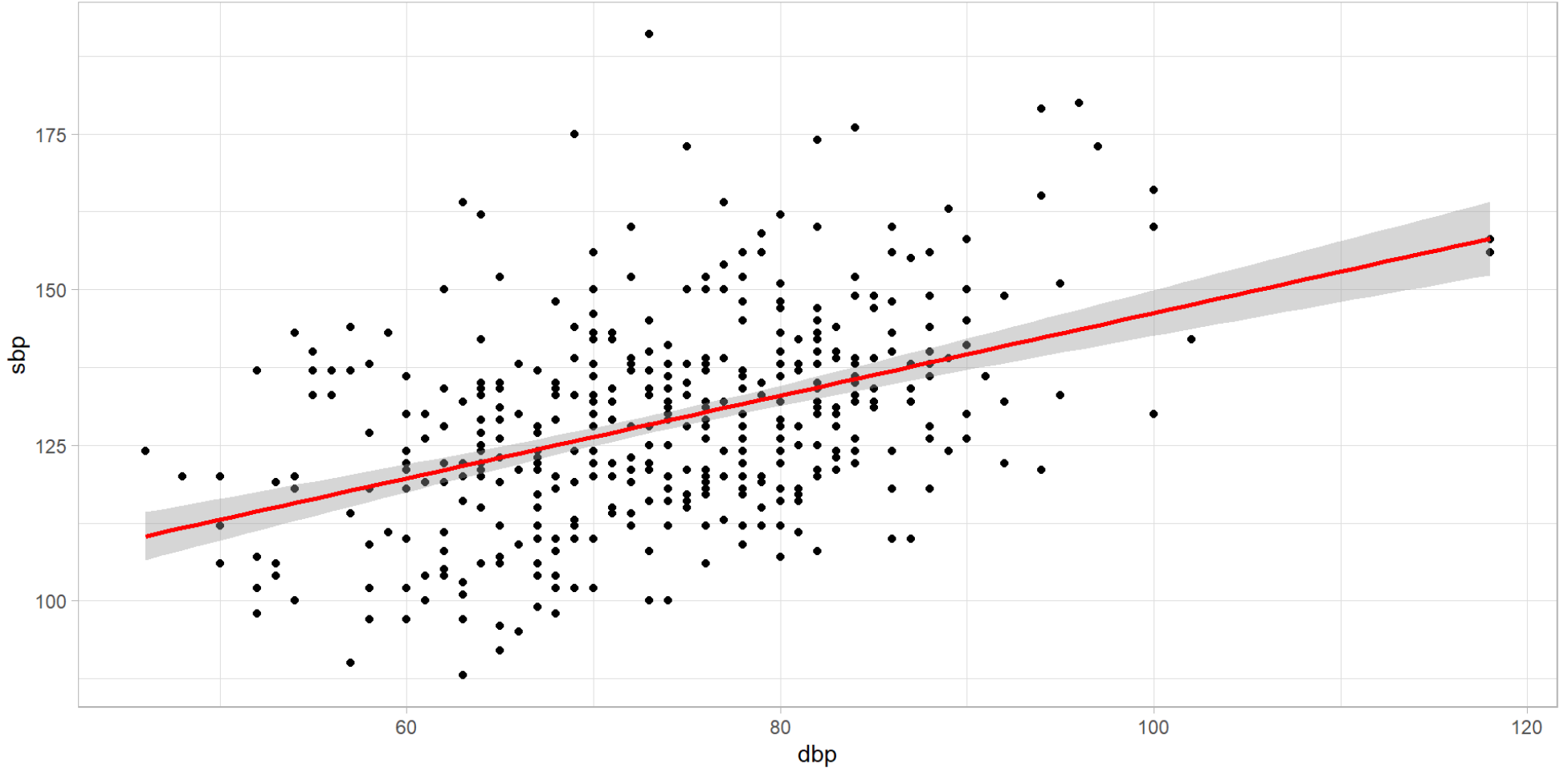
How associated are SBP and DBP?

```
1 ggplot(data = dm431, aes(x = dbp, y = sbp)) +
2   geom_point()
```



Add regression line (in red)

```
1 ggplot(data = dm431, aes(x = dbp, y = sbp)) + geom_point() +
2   geom_smooth(method = "lm", se = TRUE, formula = y ~ x, col = "red")
```



SBP and DBP association summaries

```
1 dm431 |> select(dbp, sbp) |> cor()
```

```
          dbp       sbp
dbp 1.0000000 0.4379255
sbp 0.4379255 1.0000000
```

- What does a Pearson correlation of $r = 0.44$ mean?
- R-squared = $r^2 = (0.4379)^2 \approx 0.19$ -> meaning?

```
1 lm(sbp ~ dbp, data = dm431)
```

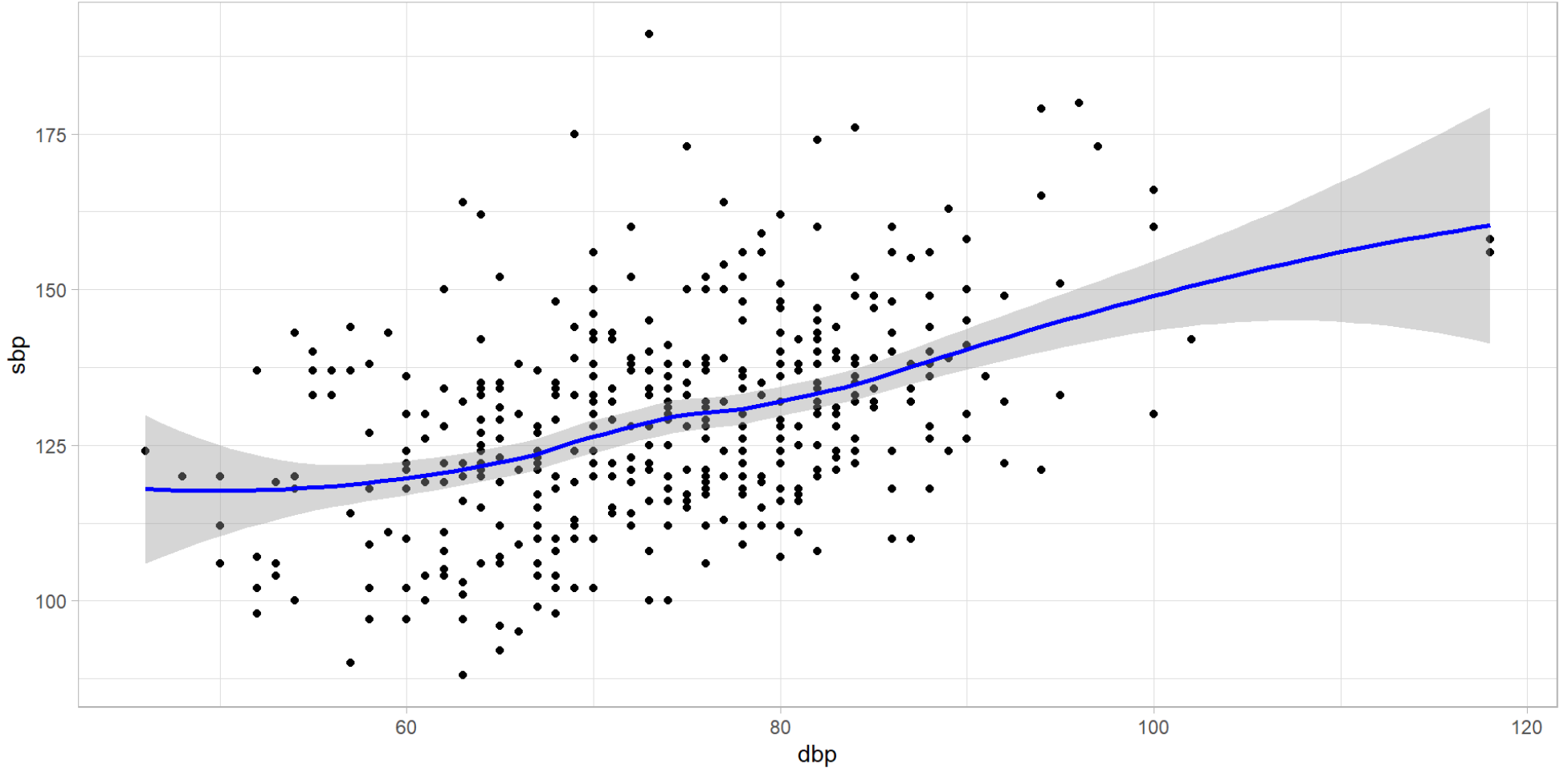
```
Call:
lm(formula = sbp ~ dbp, data = dm431)

Coefficients:
(Intercept)          dbp
     79.849        0.664
```

- What are the slope and intercept of the regression line?

Add loess smooth in blue

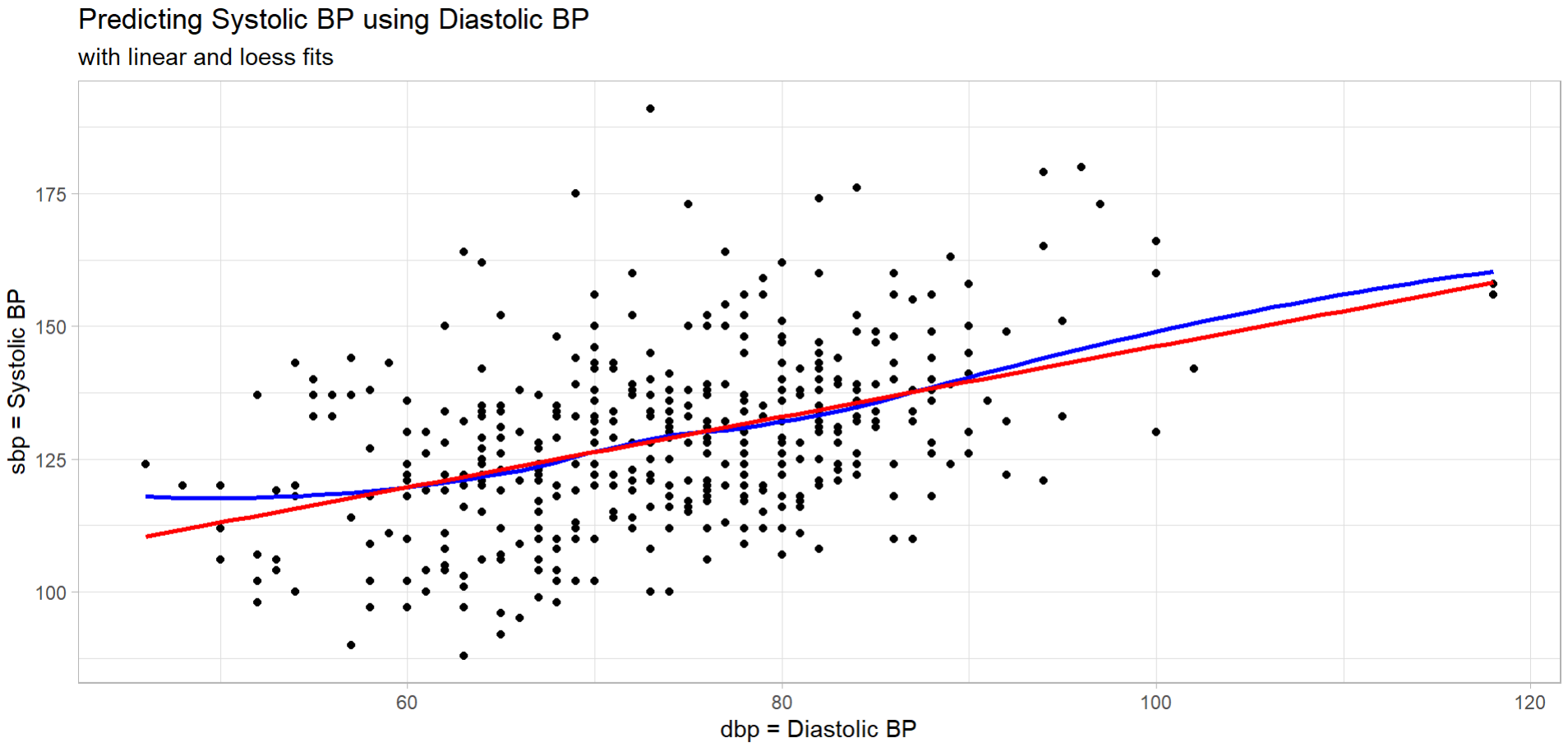
```
1 ggplot(data = dm431, aes(x = dbp, y = sbp)) + geom_point() +
2   geom_smooth(method = "loess", se = TRUE, formula = y ~ x, col = "blue")
```



Linear and loess fits, together

```
ggplot(data = dm431, aes(x = dbp, y = sbp)) +
  geom_point() +
  geom_smooth(method = "loess", se = FALSE, formula = y ~ x,
              col = "blue") +
  geom_smooth(method = "lm", se = FALSE, formula = y ~ x,
              col = "red") +
  labs(title = "Predicting Systolic BP using Diastolic BP",
       subtitle = "with linear and loess fits",
       caption = "431 women with diabetes from dm431",
       y = "sbp = Systolic BP",
       x = "dbp = Diastolic BP")
```

Linear and loess fits, together



Flip roles? Predict sbp from dbp?

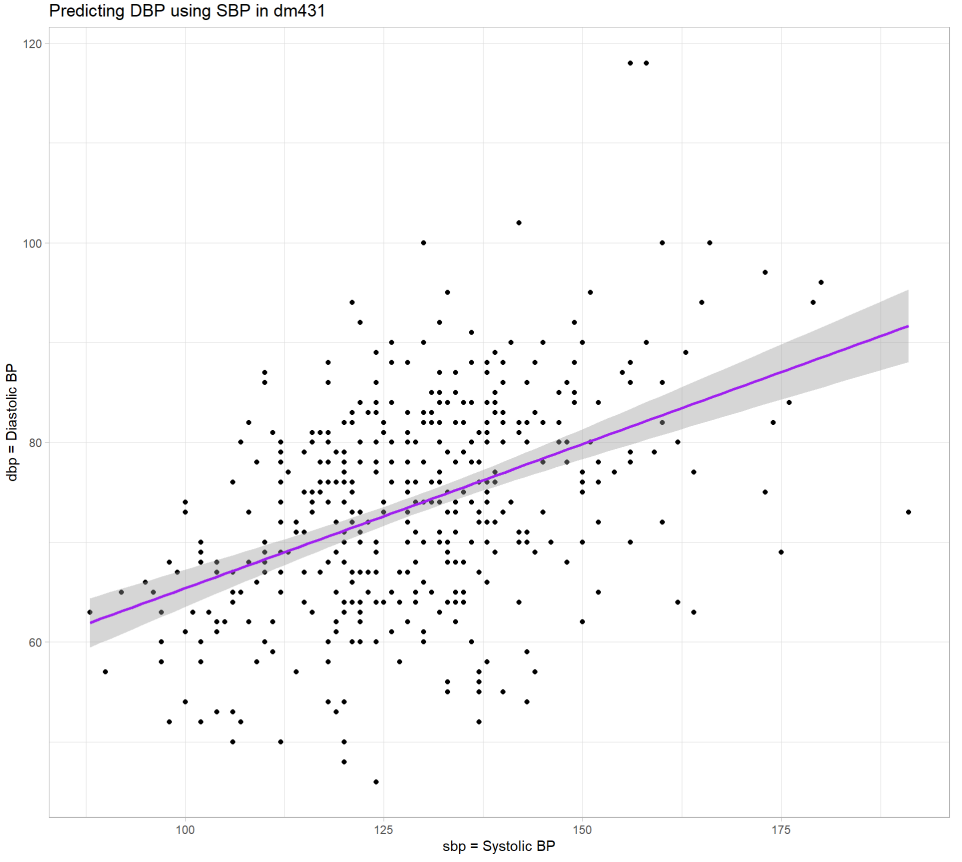
```
dm431 |>
  select(sbp, dbp) |>
  cor()
```

```
          sbp       dbp
sbp 1.0000000 0.4379255
dbp 0.4379255 1.0000000
```

```
lm(dbp ~ sbp,
   data = dm431)
```

```
Call:
lm(formula = dbp ~ sbp, data = dm431)

Coefficients:
(Intercept)          sbp
    36.5089       0.2888
```



Summarizing A Batch of Data

How old are these women?

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

```
1 dm431 |> select(age) |> summary()
```

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

- What is the age range of these women?

More numerical summaries?

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

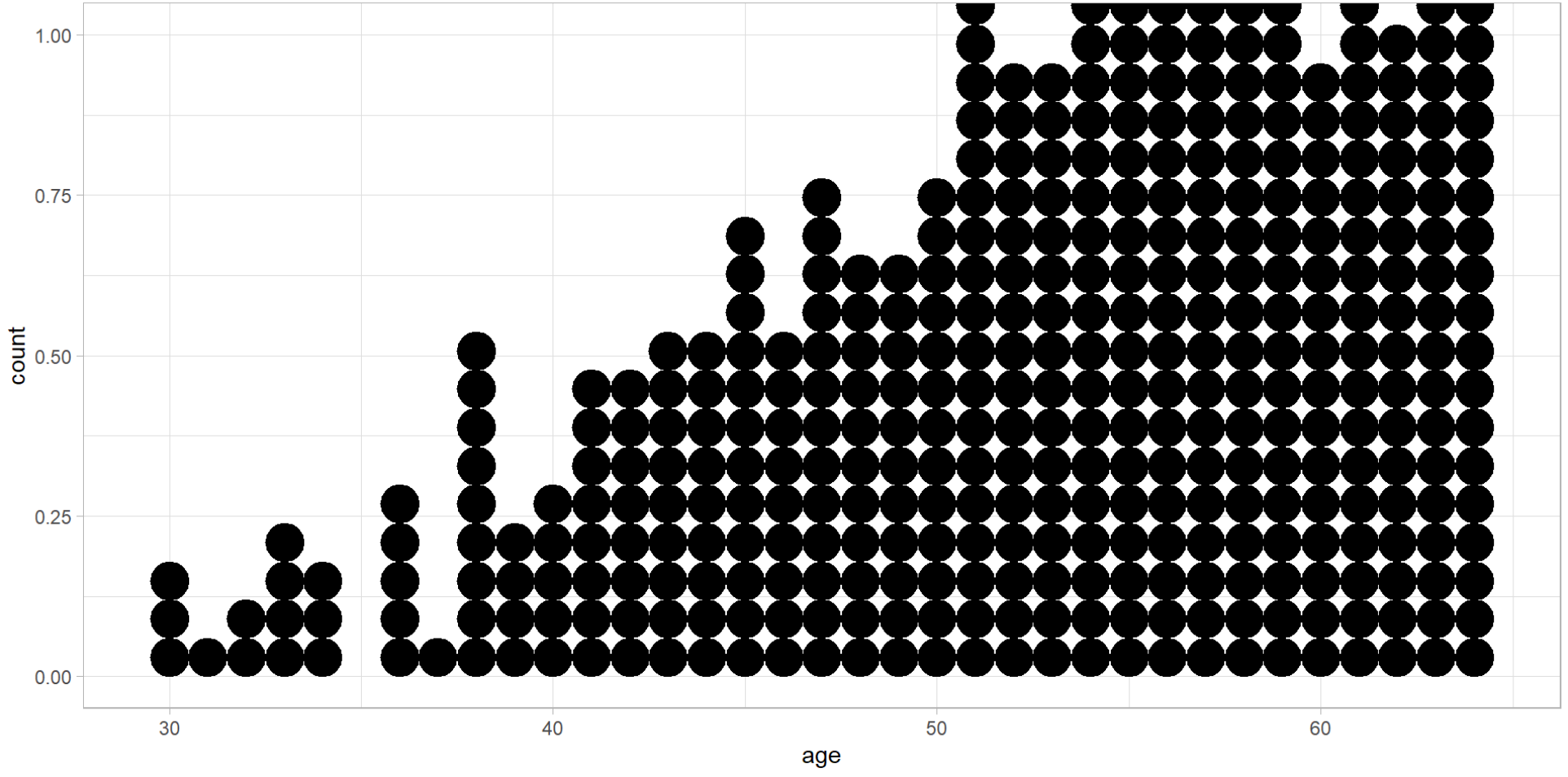
```
min Q1 median Q3 max     mean       sd   n missing
 30 48     54 59  64 52.90023 7.993414 431       0
```

- Five-number summary of quantiles: `min`, `Q1`, `median`, `Q3` and `max`
- Mean and standard deviation (`sd`) of the ages
- Sample size (non-missing) and # of missing values

Can you envision the distribution of these ages?

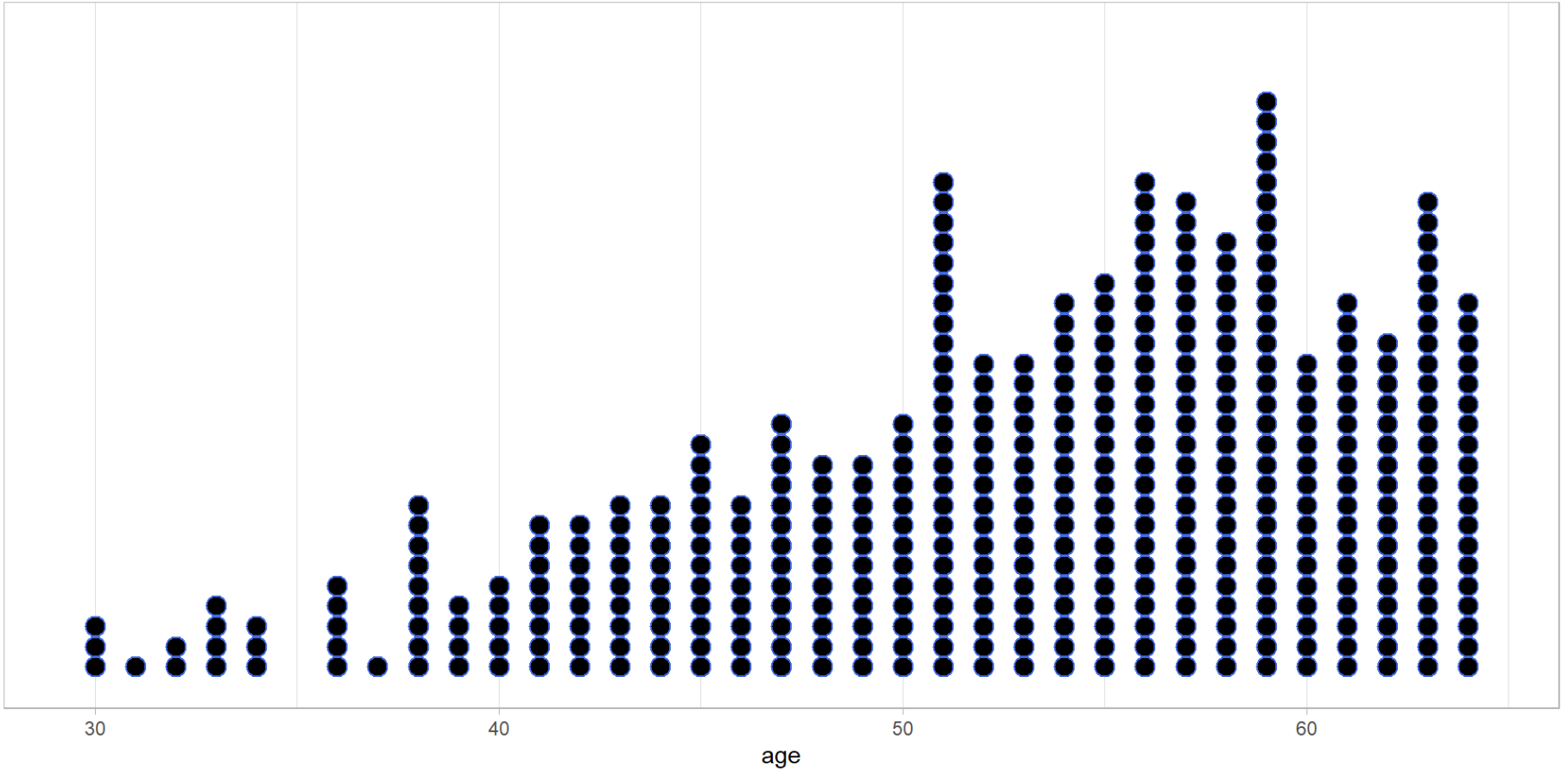
Raw Dot Plot of dm431 Ages

```
1 ggplot(data = dm431, aes(x = age)) +
2   geom_dotplot(binwidth = 1)
```



Improved Dot Plot of dm431 Ages

```
ggplot(data = dm431, aes(x = age)) +
  geom_dotplot(binwidth = 1, dotsize = 0.5, col = "royalblue") +
  scale_y_continuous(NULL, breaks = NULL)
```



Stem-and-Leaf of age values?

```
1  stem(dm431$age)
```

```
The decimal point is at the |

30 | 0000
32 | 000000
34 | 000
36 | 000000
38 | 0000000000000
40 | 0000000000000
42 | 00000000000000000
44 | 00000000000000000000
46 | 00000000000000000000
48 | 00000000000000000000
50 | 0000000000000000000000000000000000000000
52 | 00000000000000000000000000000000
```

Ages, sorted

```
1 dm431 |> select(age) |> arrange(age) |> as.vector()
```

```
$age
  [1] 30 30 30 31 32 32 33 33 33 33 34 34 34 36 36 36 36 36 37 38 38 38 38 38
38
 [26] 38 38 38 39 39 39 39 40 40 40 40 40 41 41 41 41 41 41 41 41 42 42 42 42
42
 [51] 42 42 42 43 43 43 43 43 43 43 43 43 44 44 44 44 44 44 44 44 44 44 45 45 45
45
 [76] 45 45 45 45 45 45 45 45 46 46 46 46 46 46 46 46 46 47 47 47 47 47 47 47
47
[101] 47 47 47 47 47 48 48 48 48 48 48 48 48 48 48 48 48 49 49 49 49 49 49 49 49
49
[126] 49 49 50 50 50 50 50 50 50 50 50 50 50 50 50 51 51 51 51 51 51 51 51 51
51
[151] 51 51 51 51 51 51 51 51 51 51 51 51 51 51 51 52 52 52 52 52 52 52 52 52
52
[176] 52 52 52 52 52 52 53 53 53 53 53 53 53 53 53 53 53 53 53 53 53 53 53 53 54 54
```

Using `psych::describe()`

```
1 psych::describe(dm431$age)
```

```
   vars   n mean   sd median trimmed  mad min max range  skew kurtosis   se
X1    1 431 52.9 7.99     54    53.6 7.41  30  64    34 -0.72    -0.16 0.39
```

- What's new here?
  - `trimmed` = mean of middle 80% of data
  - `mad` = median absolute deviation (measures spread)
  - `se` = standard error of the mean $= sd/\sqrt{n}$
  - `skew` and `kurtosis` not so important today

Using `Hmisc::describe()`

```
1  dm431 |> select(age) |> Hmisc::describe()
```

```
select(dm431, age)

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

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

- What's new here?
  - `distinct`, `Info`, `Gmd` —>

New `Hmisc::describe` elements

- `Hmisc::describe` treats a numeric variable as discrete if it has 10 or fewer distinct values
- `Info` is related to how "continuous" the variable is - it's a relative measure of the available information that is reduced below 1 by ties or non-distinct values
- `Gmd` = Gini's mean difference measures dispersion (spread). It is the mean absolute difference between any pairs of the 431 observations. Pronounced "Ginny".

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

Age Breakdown by Insurance

Note: Is this actually what I want?

```
1 dm431 |> tabyl(age, insurance)
```

age	Medicare	Medicaid	Commercial	Uninsured
30	0	2	0	1
31	0	0	1	0
32	1	1	0	0
33	0	4	0	0
34	0	2	0	1
36	0	4	1	0
37	0	0	1	0
38	0	8	1	0
39	0	4	0	0
40	1	2	1	1
41	1	4	3	0
42	0	4	4	0
43	1	4	3	1
44	3	2	4	0
45	1	5	4	2

Age Breakdown by Insurance (2)

Do these results make sense to you?

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

```
   insurance min    Q1 median   Q3 max     mean       sd   n missing
1   Medicare  32 52.75   58.5 62.0  64 56.59000 6.751124 100       0
2   Medicaid  30 46.00   53.0 58.0  64 51.20419 8.385776 191       0
3 Commercial  31 47.50   54.0 57.5  64 52.69369 7.212102 111       0
4  Uninsured  30 48.00   52.0 58.0  64 52.13793 8.339768  29       0
```

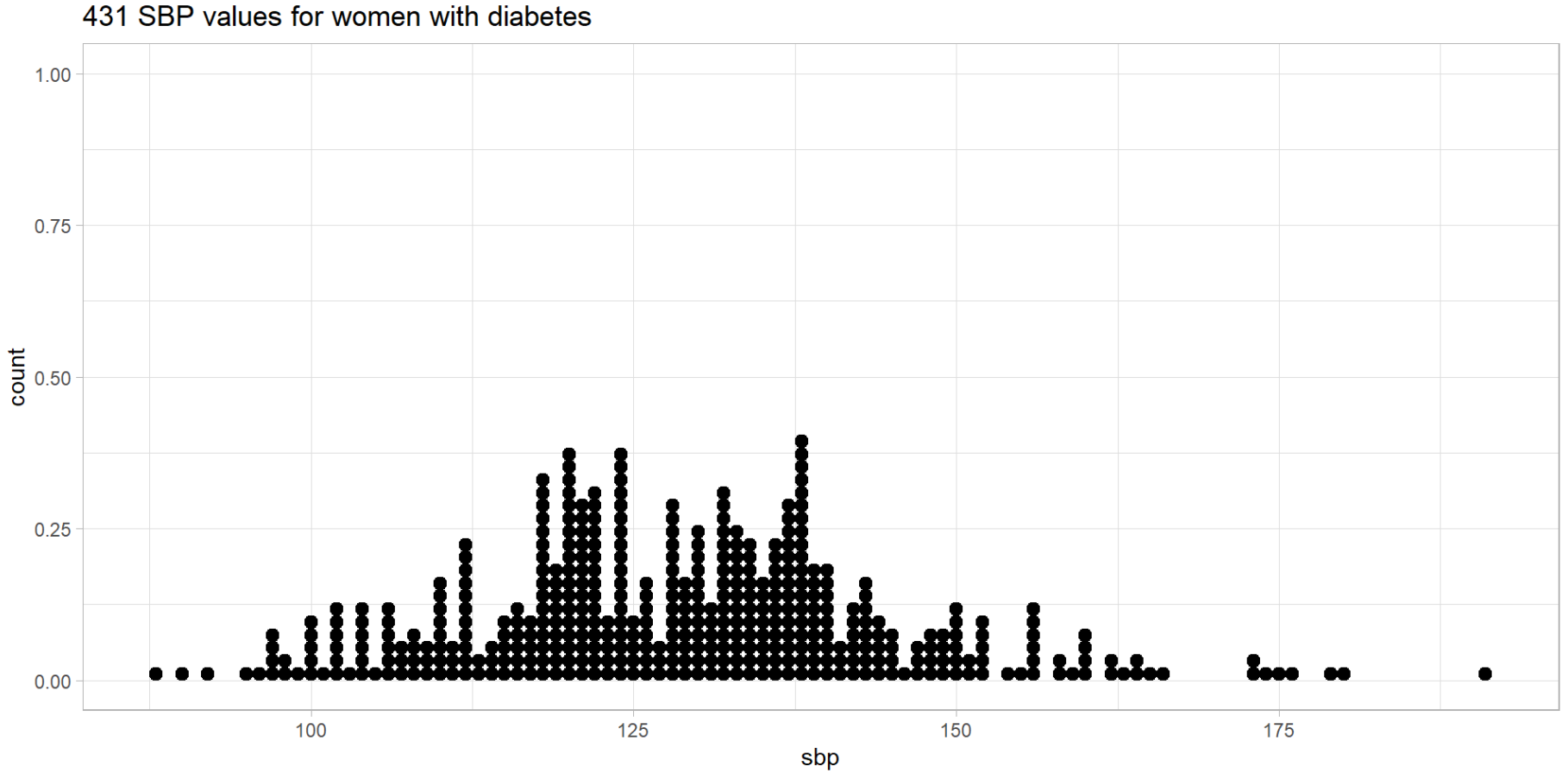
```
1 dm431 |> group_by(insurance) |>
2   summarize(n = n(), mean = round_half_up(mean(age),2),
3             median = median(age), sd = round_half_up(sd(age),2),
4             skew1 = round_half_up( (mean - median) / sd, 2))
```

```
# A tibble: 4 × 6
  insurance      n  mean median    sd skew1
  <fct>      <int> <dbl>  <dbl> <dbl> <dbl>
1 Medicare     100  56.6   58.5  6.75 -0.28
2 Medicaid     191  51.2   53    8.39 -0.21
3 Commercial   111  52.7   54    7.21 -0.18
4 Uninsured     29  52.1   52    8.34  0.02
```

Center, spread, outliers and shape of Blood Pressure Data

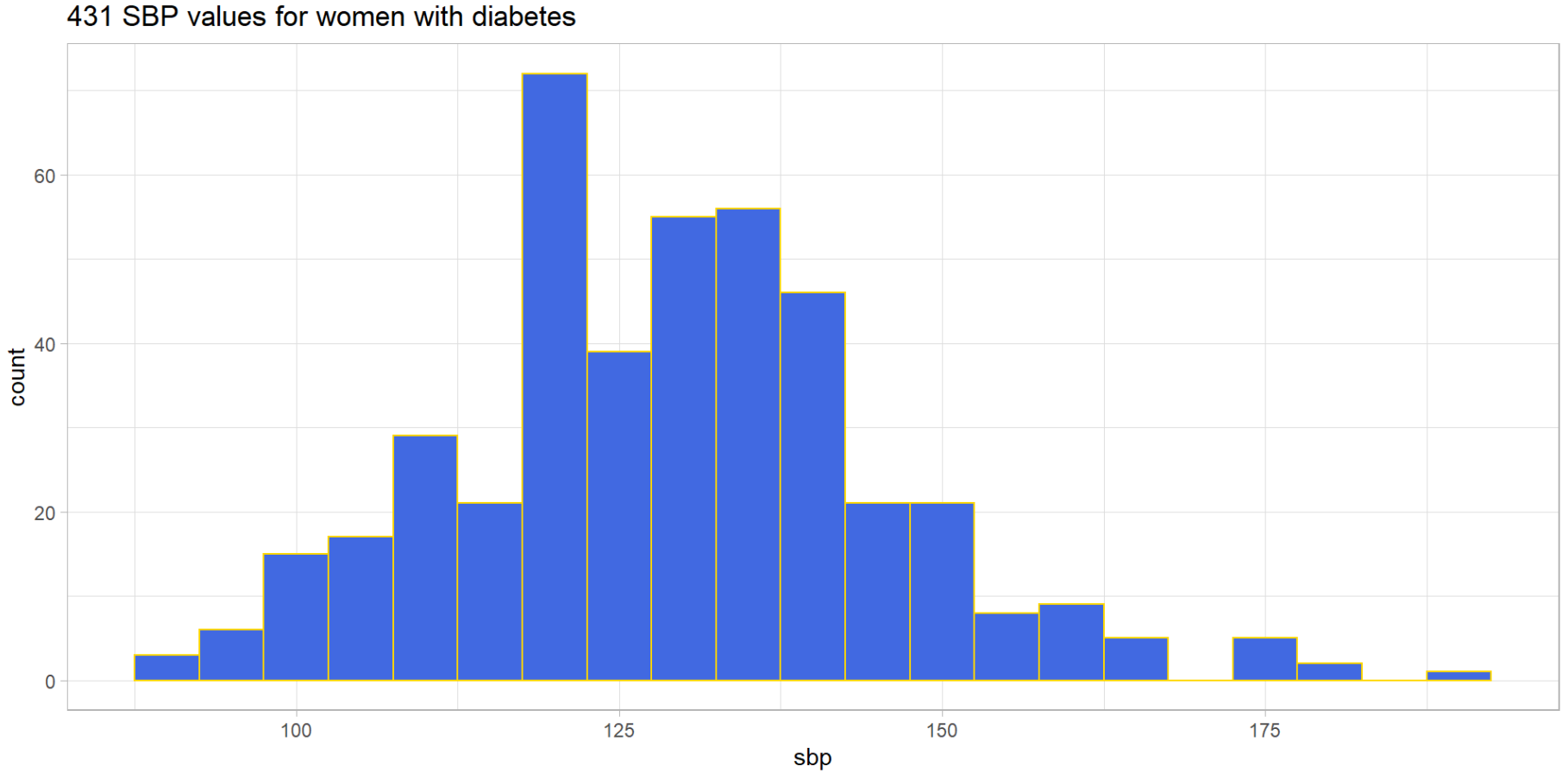
Systolic BP values from dm431 (dotplot)

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

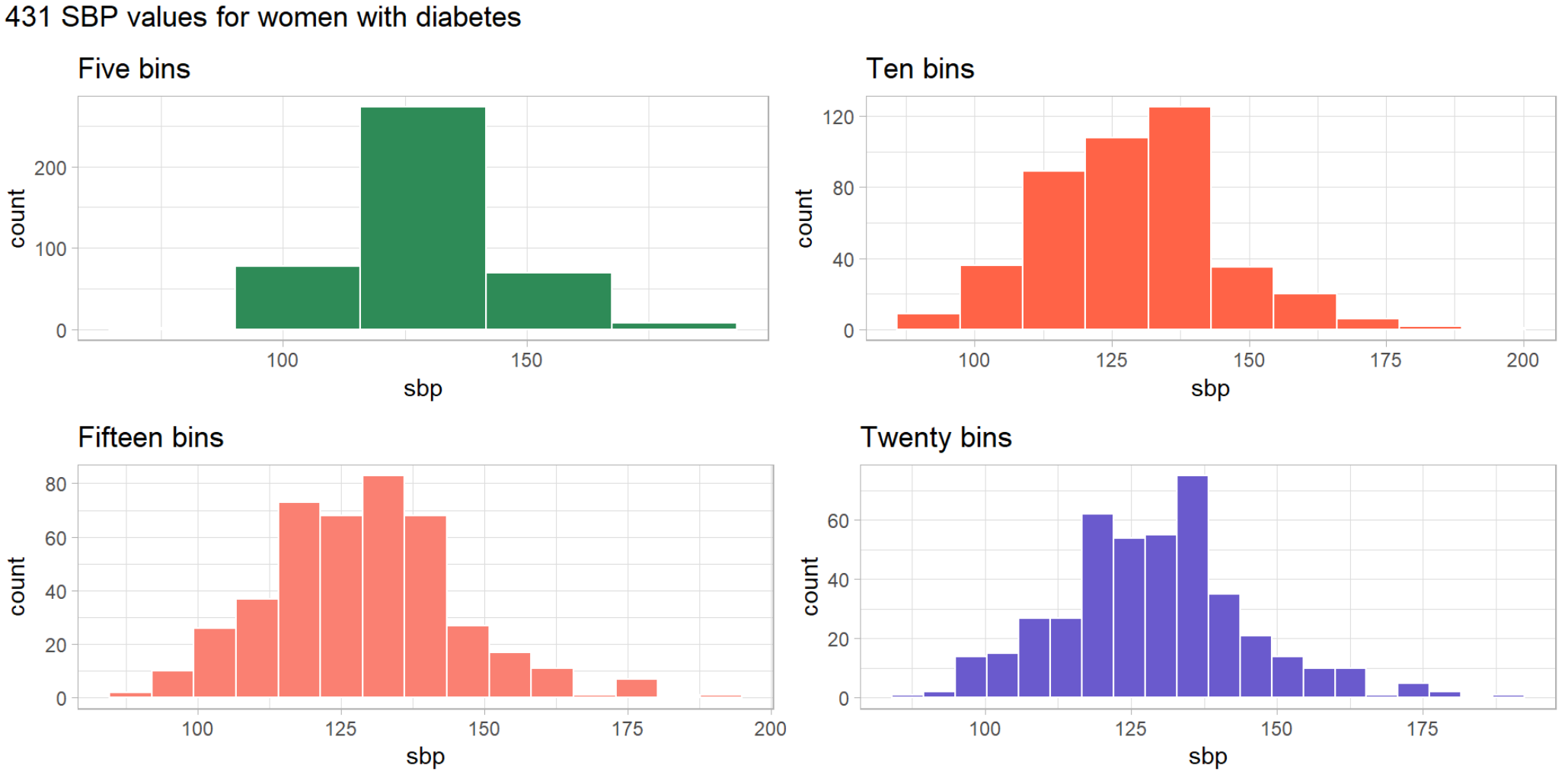


Histogram of dm431 Systolic BP

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



Number of Bins in a Histogram



"Pseudo-Code" for previous slide

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

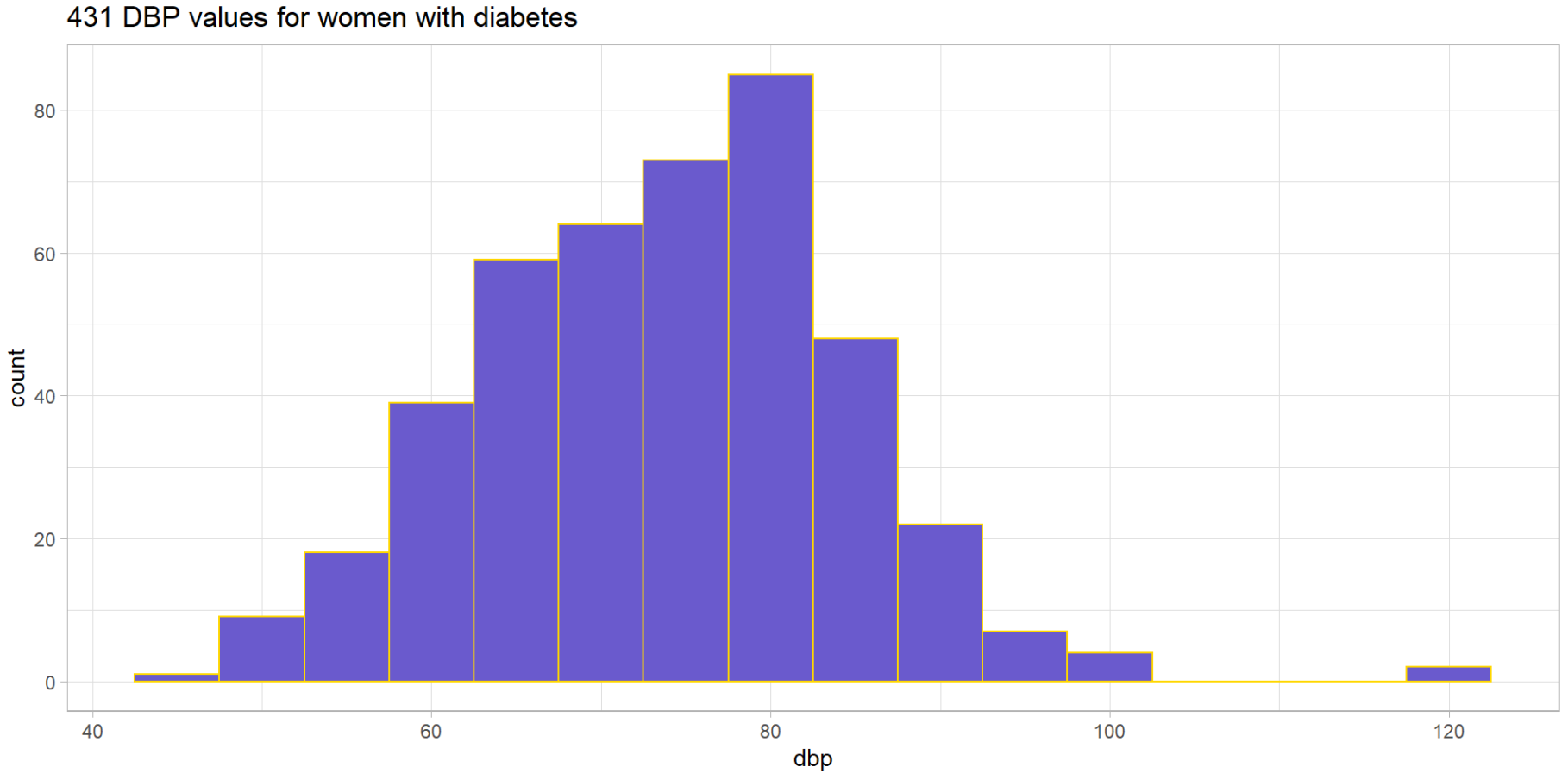
# omitting the code for plots p2-p4 in this slide,
# use bins = 10, 15 and 20, respectively, and use
# tomato, salmon and slateblue for fill, respectively

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

- You have the Quarto file for every set of slides in the README.

Histogram of dm431 Diastolic BP

```
1  ggplot(data = dm431, aes(x = dbp)) +
2    geom_histogram(binwidth = 5, fill = "slateblue", col = "gold") +
3    labs(title = "431 DBP values for women with diabetes")
```



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.

Are our data "Normal 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 designed to help us clearly identify potential problems with assuming Normality.

Are our data "Normal enough"?

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

1. Graphs
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.

Are our data "Normal enough"?

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

1. Graphs
2. Planned analyses after decision is made
3. Numerical Summaries of the data
    - Definitely the least important even though they seem "easy-to-use" and "objective".

Simulating Data from a Normal

What would a sample of 431 systolic blood pressures from a Normal distribution look like?

- 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(2022)
sim_data <- tibble(
  sbp = rnorm(n = 431, mean = mean(dm431$sbp), sd = sd(dm431$sbp)))
```

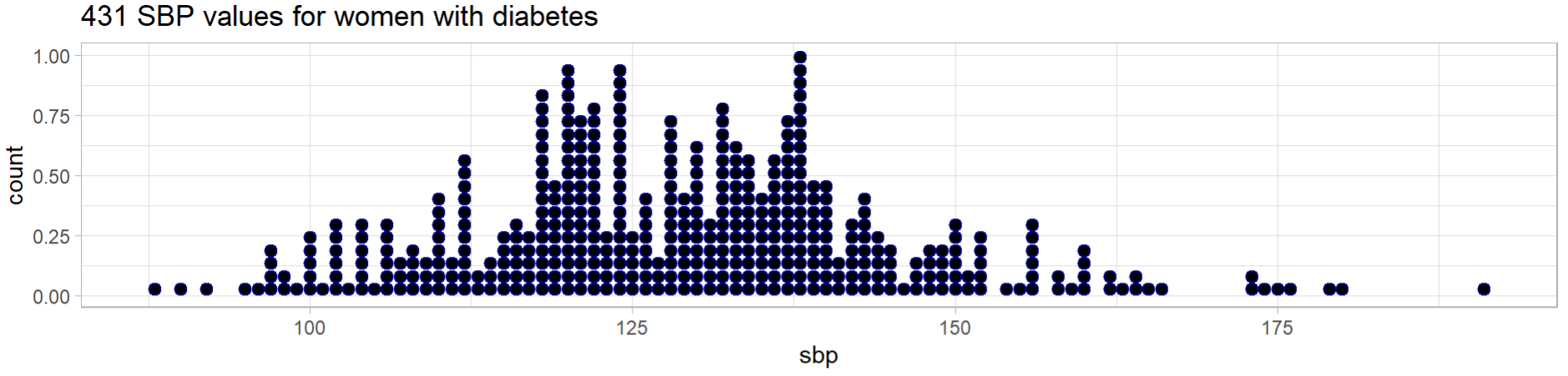
Simulated Sample vs. 431 Data

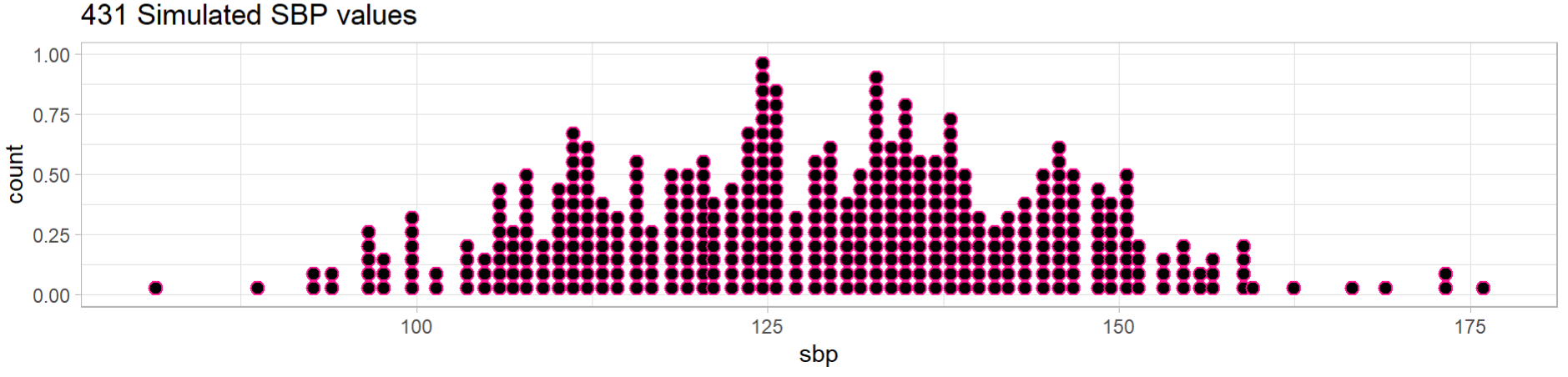
```
p1 <- ggplot(data = dm431, aes(x = sbp)) +
  geom_dotplot(binwidth = 1, col = "navy") +
  labs(title = "431 SBP values for women with diabetes")

p2 <- ggplot(data = sim_data, aes(x = sbp)) +
  geom_dotplot(binwidth = 1, col = "deeppink") +
  labs(title = "431 Simulated SBP values")

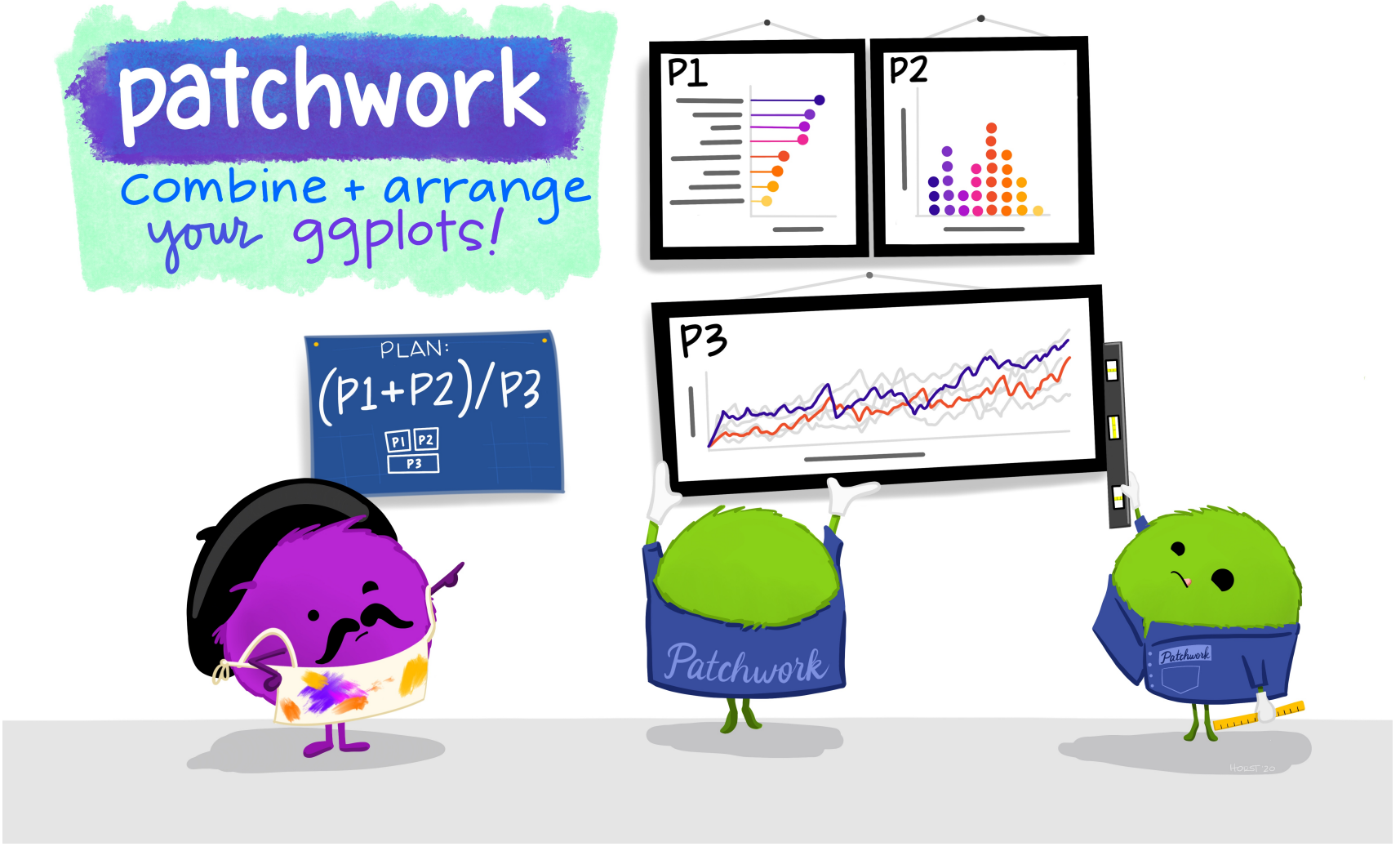
p1 / p2
```

Simulated Sample vs. 431 Data





Putting the plots together...



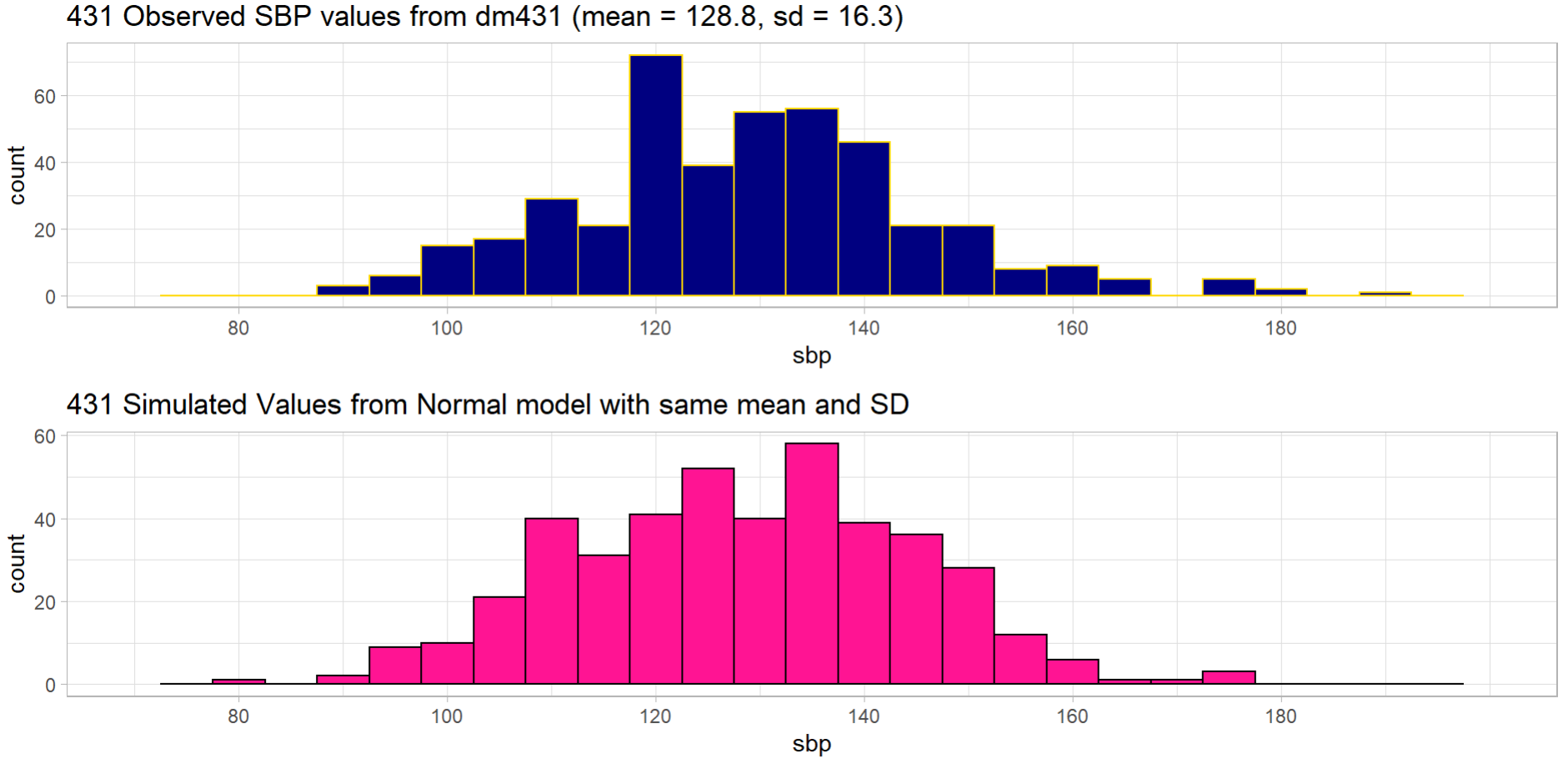
Comparing Histograms

```
p1 <- ggplot(data = dm431, aes(x = sbp)) +
  geom_histogram(binwidth = 5, fill = "navy", col = "gold") +
  scale_x_continuous(limits = c(70, 200),
                     breaks = c(80, 100, 120, 140, 160, 180)) +
  labs(title = "431 Observed SBP values from dm431 (mean = 128.8, sd = 16.3

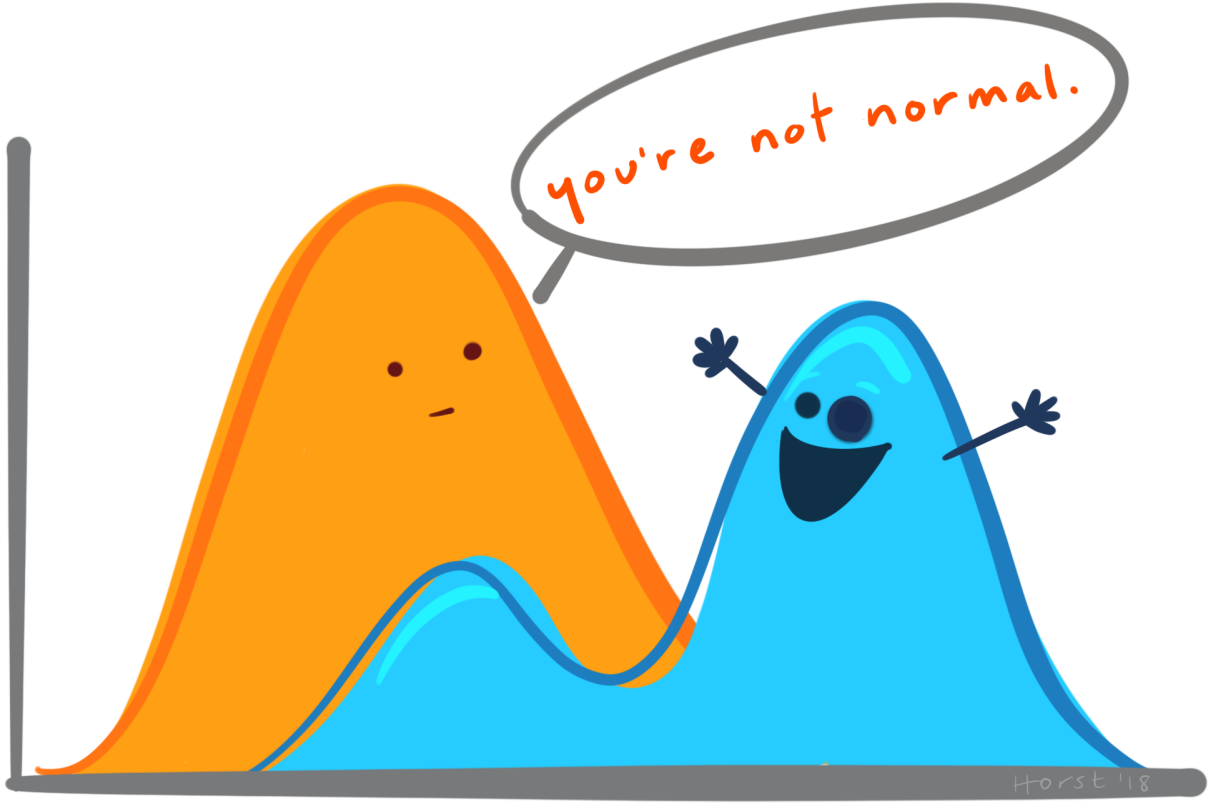
p2 <- ggplot(sim_data, aes(x = sbp)) +
  geom_histogram(binwidth = 5, fill = "deeppink", col = "black") +
  scale_x_continuous(limits = c(70, 200),
                     breaks = c(80, 100, 120, 140, 160, 180)) +
  labs(title = "431 Simulated Values from Normal model with same mean and S

p1 / p2
```

Comparing Histograms



Graphs are our most important tool!



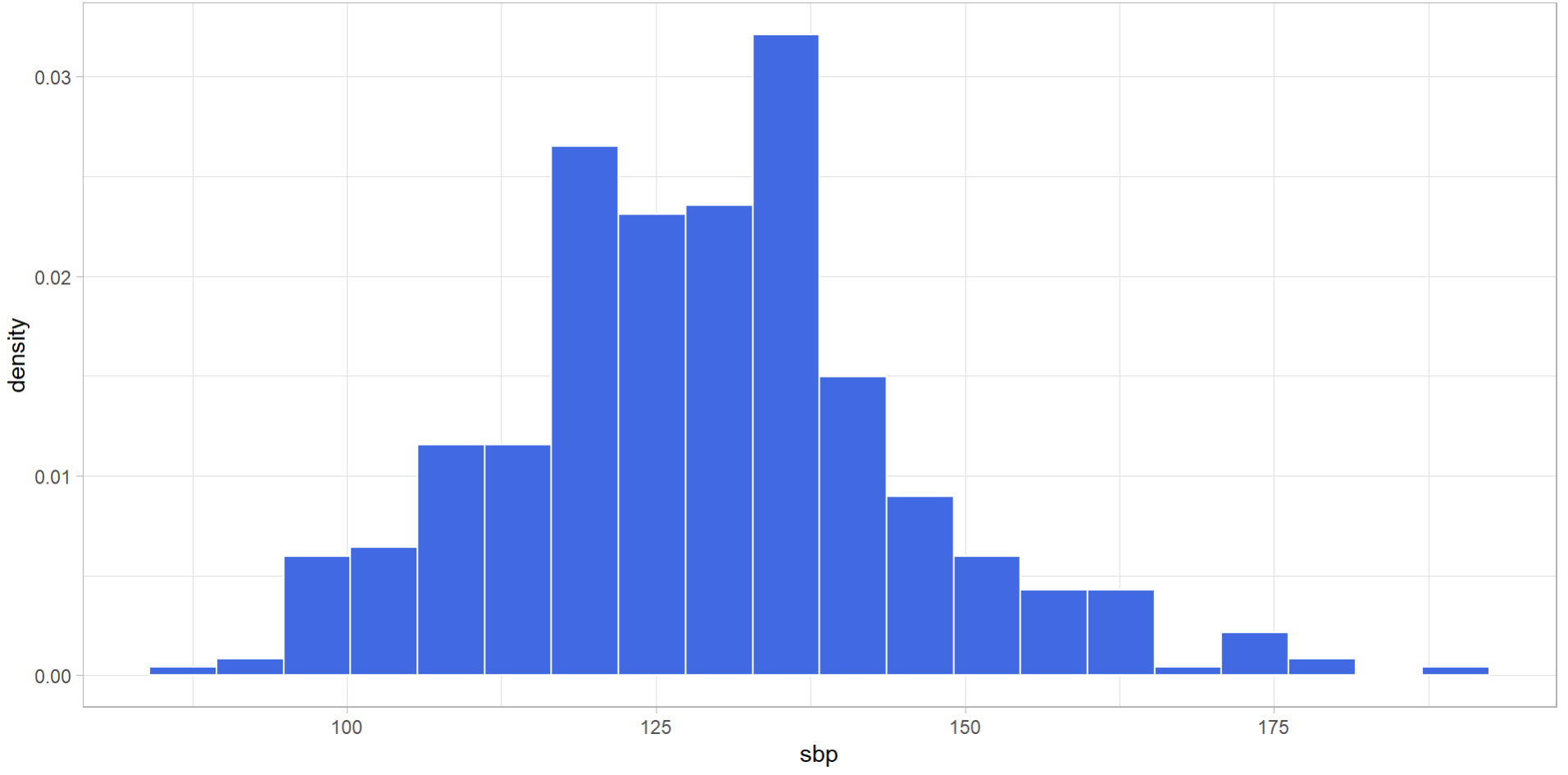
Rescale dm431 SBP histogram as density

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")
```

Rescale `dm431` SBP histogram as density

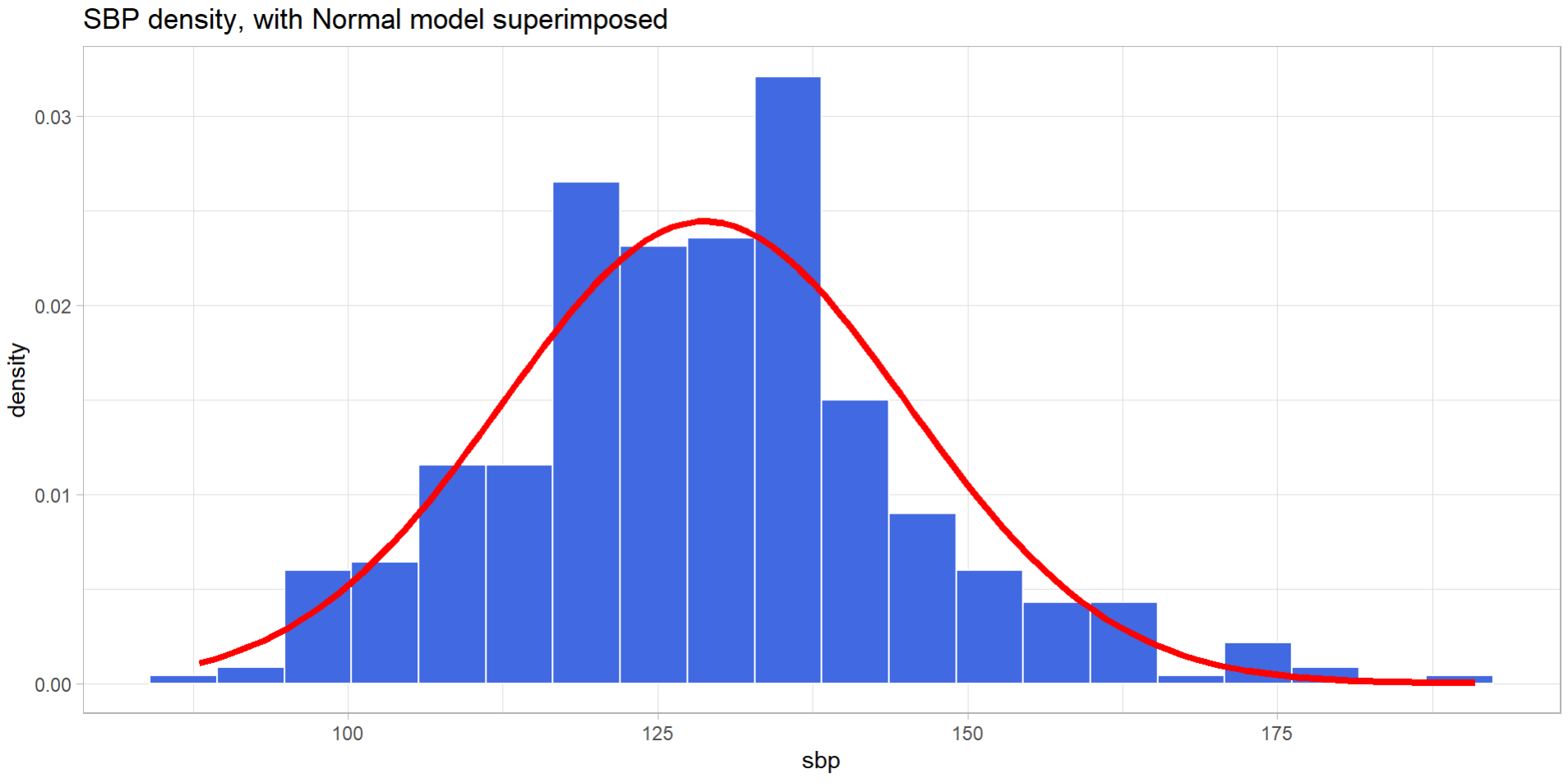


Density, with superimposed Normal

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

```
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")
```

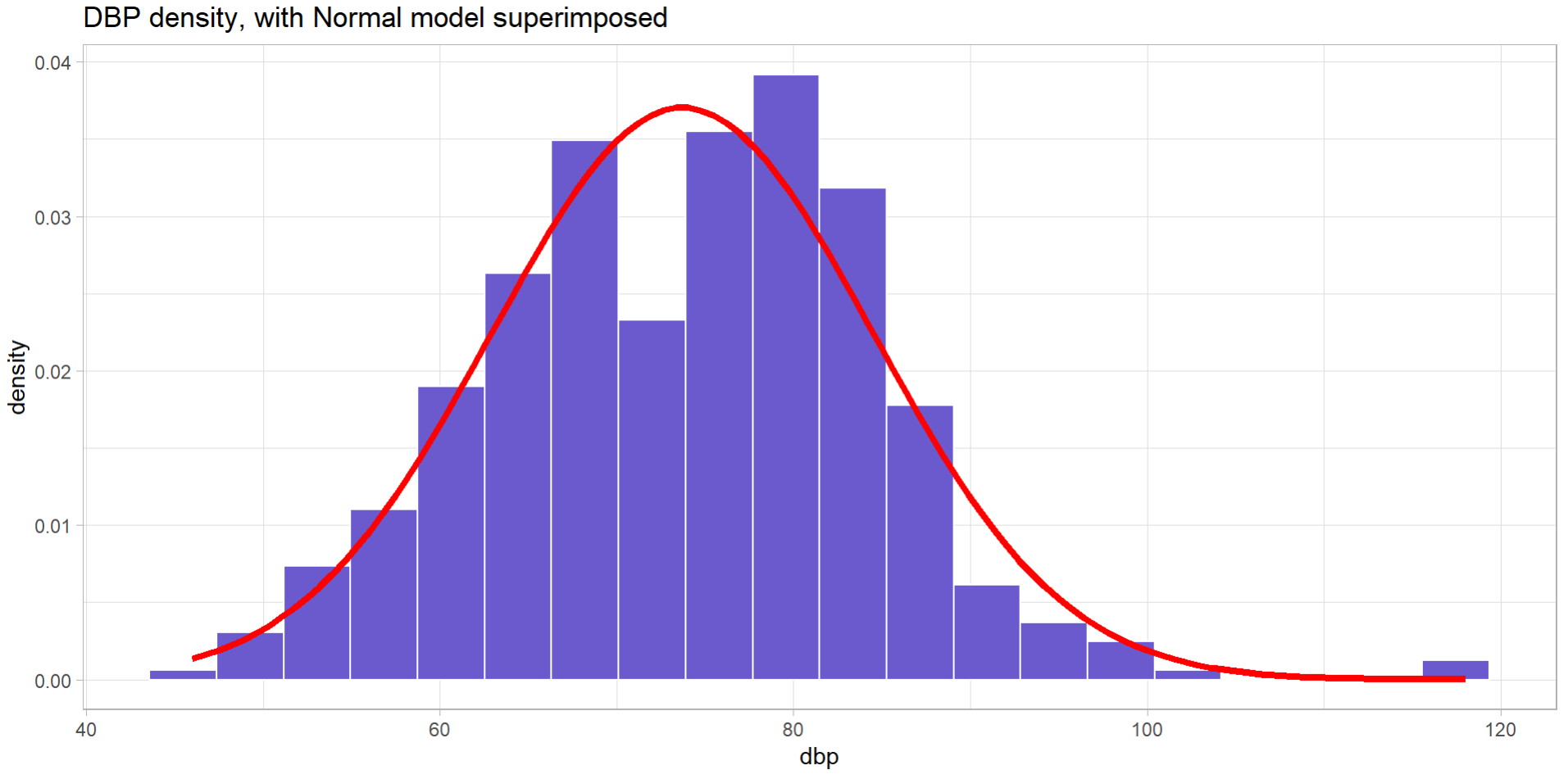
Density, with superimposed Normal



DBP: Density Curve with Normal model

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

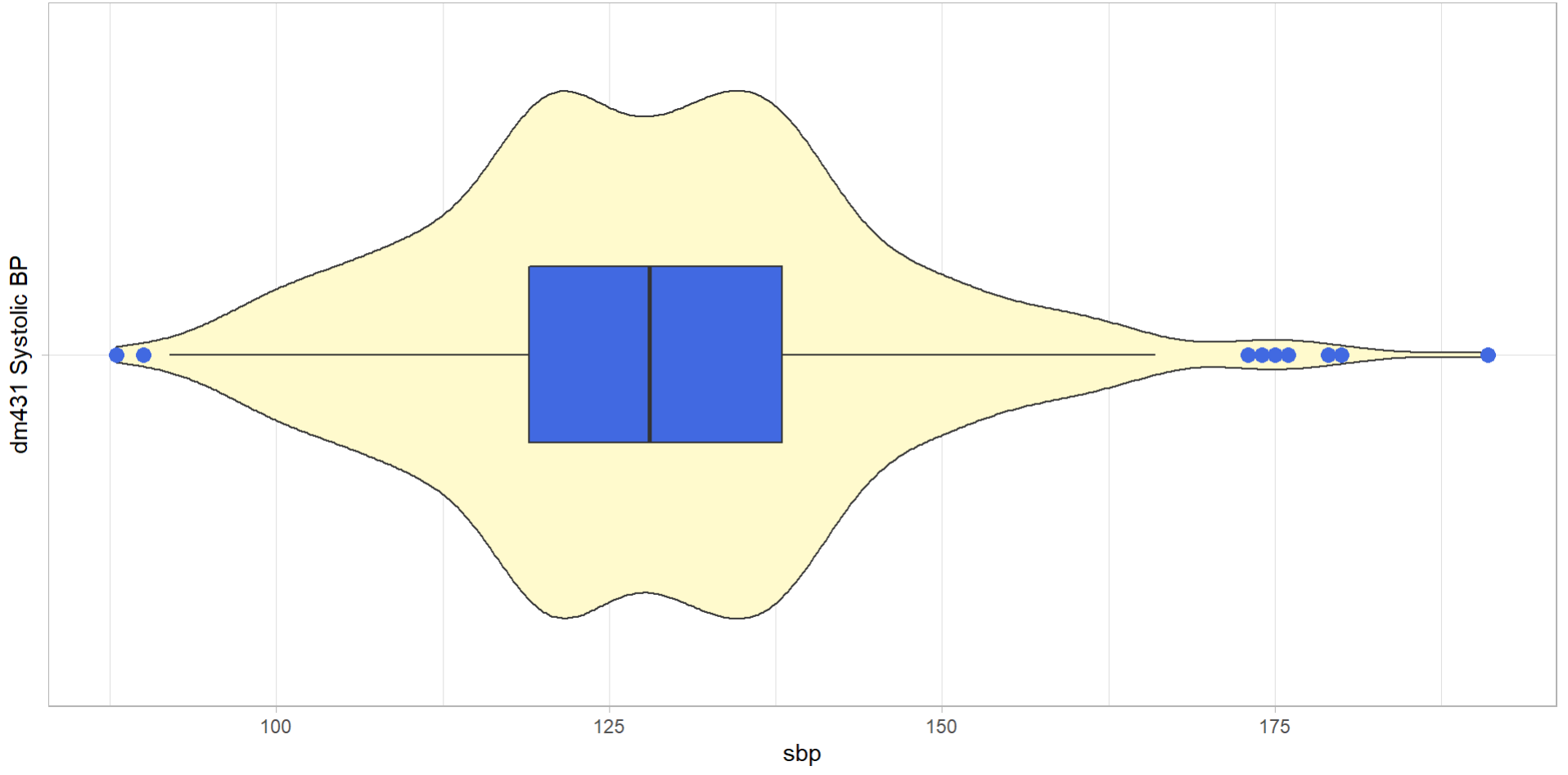
DBP: Density Curve with Normal model



Violin and Boxplot for dm431 SBP

```
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 Systolic BP")
```

Violin and Boxplot for dm431 SBP

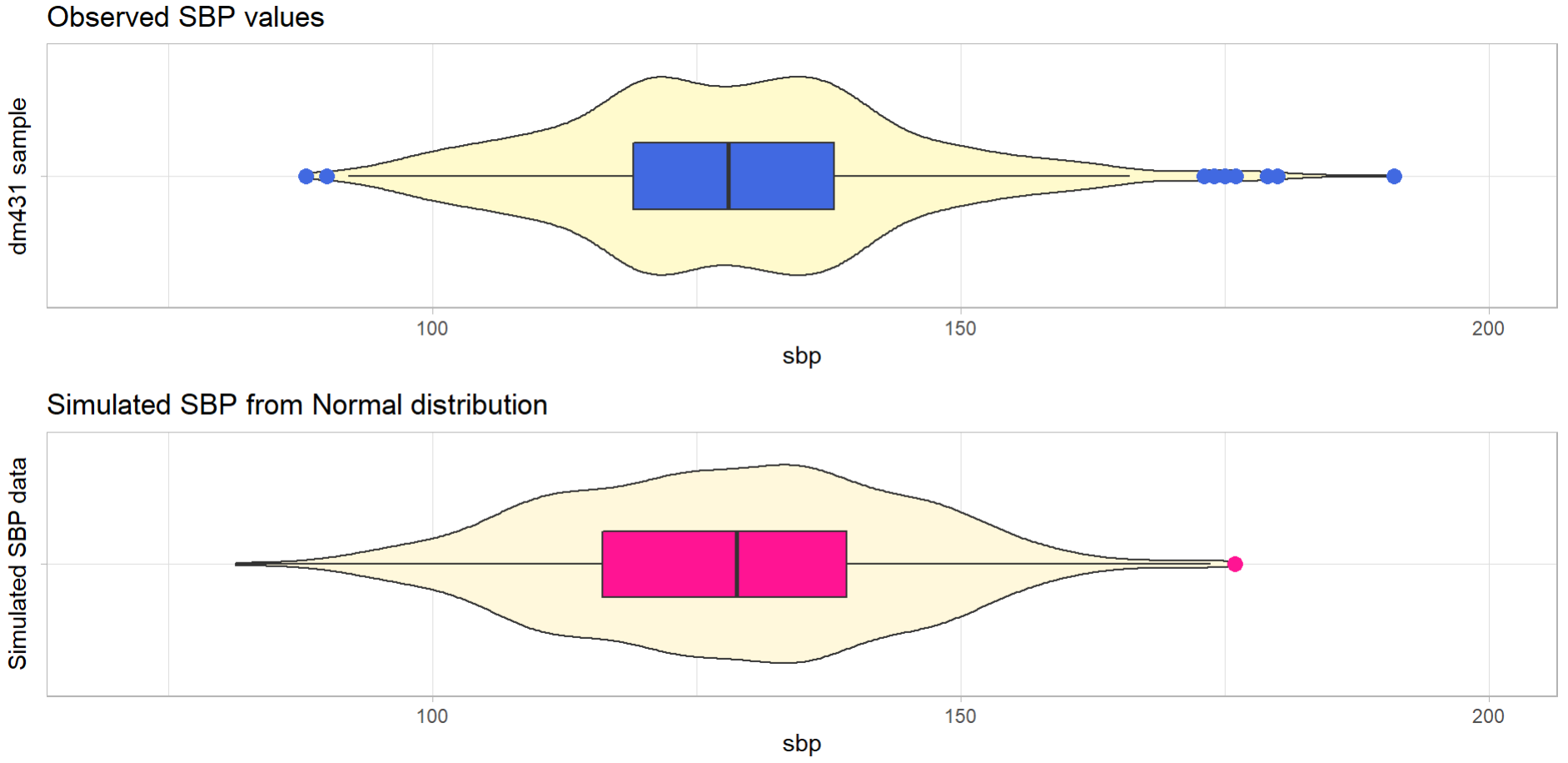


Observed vs. Simulated Systolic BPs

```
p1 <- ggplot(dm431, aes(x = "", y = sbp)) +
  geom_violin(fill = "lemonchiffon") +
  geom_boxplot(width = 0.3, fill = "royalblue",
               outlier.size = 3,
               outlier.color = "royalblue") +
  lims(y = c(70, 200)) +
  coord_flip() +
  labs(x = "dm431 sample",
       title = "Observed SBP values")

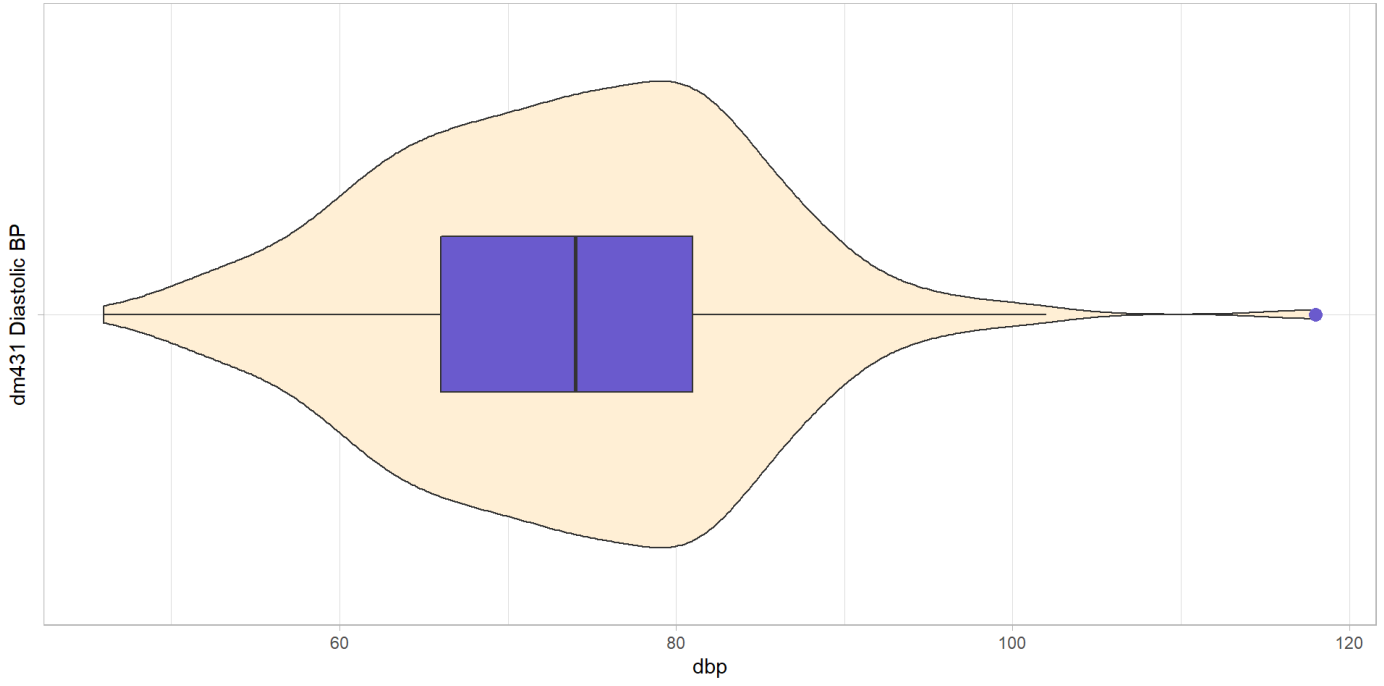
p2 <- ggplot(sim_data, aes(x = "", y = sbp)) +
  geom_violin(fill = "cornsilk") +
  geom_boxplot(width = 0.3, fill = "deeppink",
               outlier.size = 3,
               outlier.color = "deeppink") +
  lims(y = c(70, 200)) +
  coord_flip() +
  labs(x = "Simulated SBP data",
       title = "Simulated SBP from Normal distribution")
```

Observed vs. Simulated Systolic BPs



Violin and Boxplot for dm431 DBP

```
ggplot(dm431, aes(x = "", y = dbp)) +
  geom_violin(fill = "papayawhip") +
  geom_boxplot(width = 0.3, fill = "slateblue",
               outlier.size = 3,
               outlier.color = "slateblue") +
  coord_flip() +
  labs(x = "dm431 Diastolic BP")
```



This is where we'll start for Class 06.

What have I changed?

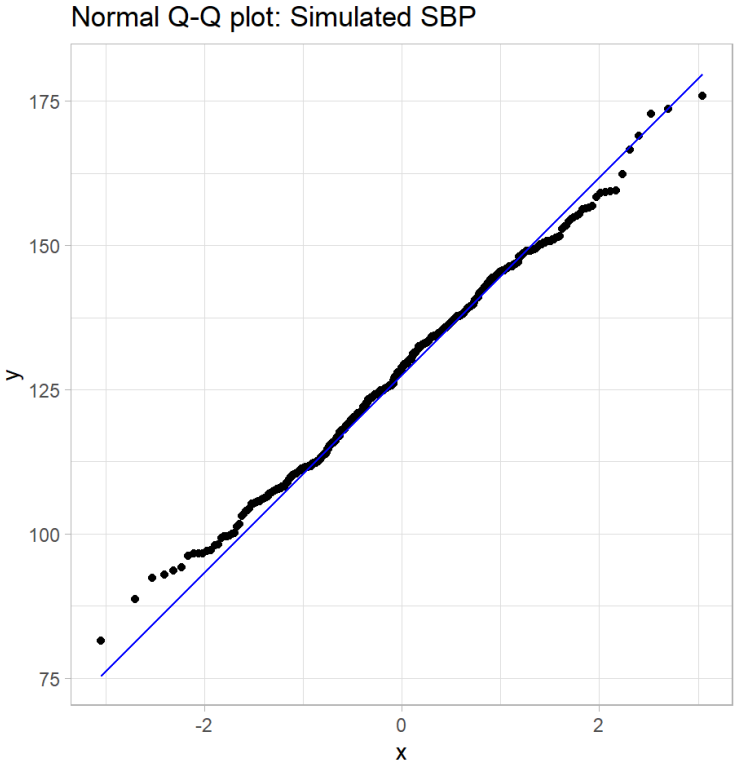
Since Tuesday, I changed the following things in the slides prior to this one…

- Changed `dm431.csv` file (and posted it on our 431-data page) to have 16 rather than 17 variables.
- Dropped note about warning silencing in slide 21.
- Corrected placement of SBP and DBP in slides 24-29 to be consistent with using DBP to predict SBP.
- Added se = TRUE to loess smooth in slide 27.
- Added slides 28-29 to show loess, linear smooths together and to show labels with title, subtitle and caption.
- Moved slide 30 to where it is now, and corrected bottom left output.
- Added slide 31 (title slide).
- Added code for slide 37.

Using a Normal Q-Q plot to assess Normality of a batch of data

Normal Q-Q plot of our simulated data

Remember that these are draws from a Normal distribution, so this is what a sample of 431 Normally distributed data points should look like.



What is a Normal Q-Q Plot? (1)

Tool to help assess whether the distribution of a single sample is well-modeled by the Normal.

- Suppose we have N data points in our sample.
- Normal Q-Q plot will plot N points, on a scatterplot.
  - Y value is the observed data value.
  - X value is the expected value for that point in a Normal distribution with mean 0 and standard deviation 1.

What is a Normal Q-Q Plot? (2)

Given a sample of size N, R calculates what the minimum value would be expected to be for a standard Normal distribution (a Normal with mean 0 and standard deviation 1.) Then it calculates what the next smallest value would be, and so forth all the way up to the maximum value.

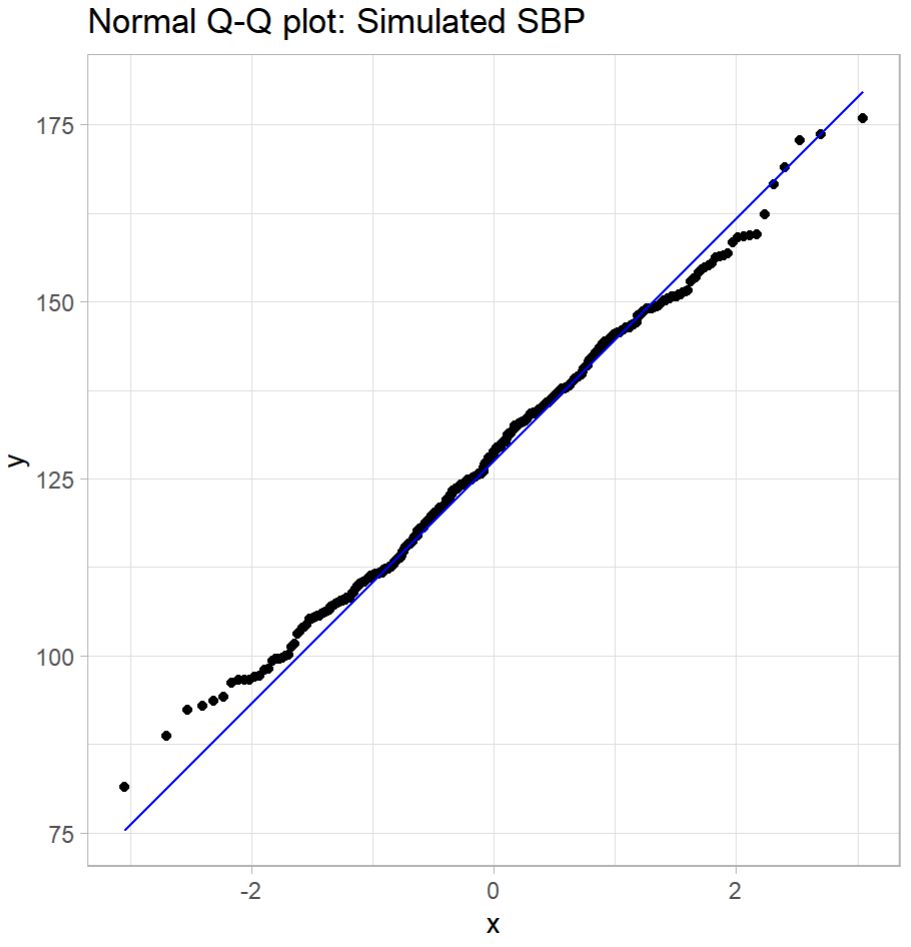
- X value in the Normal Q-Q plot is the value that a Normal(0,1) distribution would take for that rank in the data.
- We draw a line through Y = X, and points close to the line therefore match what we'd expect from a Normal distribution.

How do we create a Normal Q-Q plot?

For our simulated blood pressure data

```
ggplot(sim_data, aes(sample = sbp)) +
  geom_qq() + # plot the points
  geom_qq_line(col = "blue") + # plot the Y = X line
  theme(aspect.ratio = 1) + # make the plot square
  labs(title = "Normal Q-Q plot: Simulated SBP")
```

How do we create a Normal Q-Q plot?



Interpreting the Normal Q-Q plot? (1)

The Normal Q-Q plot can help us identify data as well approximated by a Normal distribution, or not, because of:

- **skew** (including distinguishing between right skew and left skew)
- behavior in the **tails** (which could be heavy-tailed [more outliers than expected] or light-tailed)

1. Normally distributed data are indicated by close adherence of the points to the diagonal reference line.

Interpreting the Normal Q-Q plot? (2)

2. **Skew** is indicated by substantial curving (on both ends of the distribution) in the points away from the reference line (if both ends curve up, we have right skew; if both ends curve down, this indicates left skew)
3. An abundance or dearth of **outliers** (as compared to the expectations of a Normal model) are indicated in the tails of the distribution by an "S" shape or reverse "S" shape in the points.

Examples coming up next —>

These next few slides –> Six Examples

We'll next build useful visualizations and numeric summaries, describing…

1. Data sampled from a Normal distribution
2. … a left-skewed distribution
3. … a right-skewed distribution
4. … a symmetric, discrete distribution
5. … a uniform distribution
6. … a symmetric, outlier-prone distribution

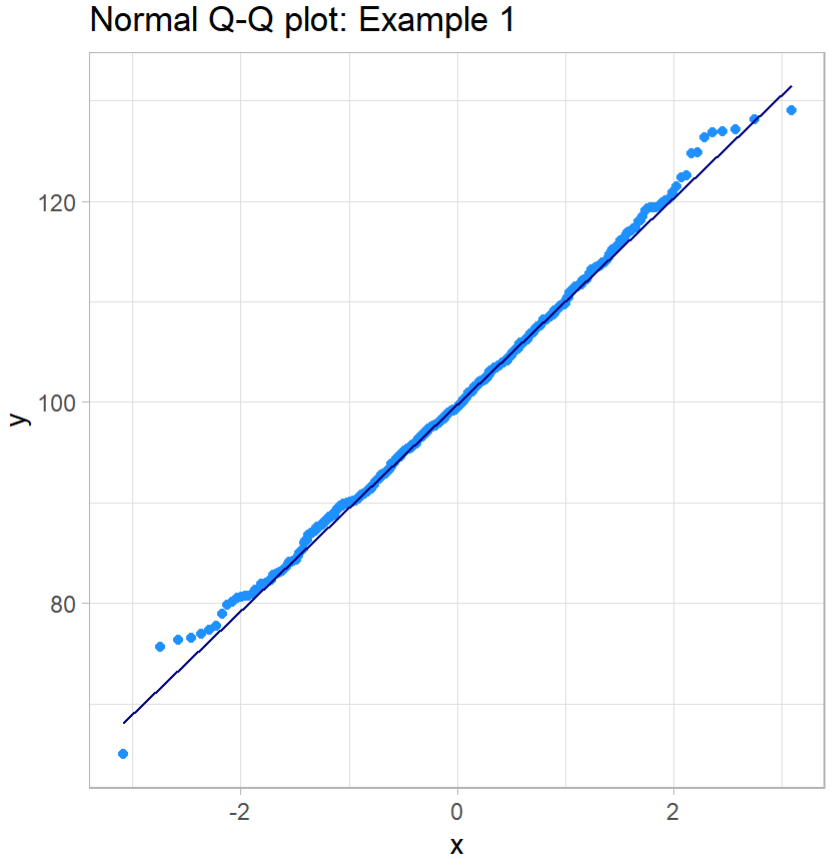
```
# code for Example 1
# plotting data sampled from
# a Normal distribution

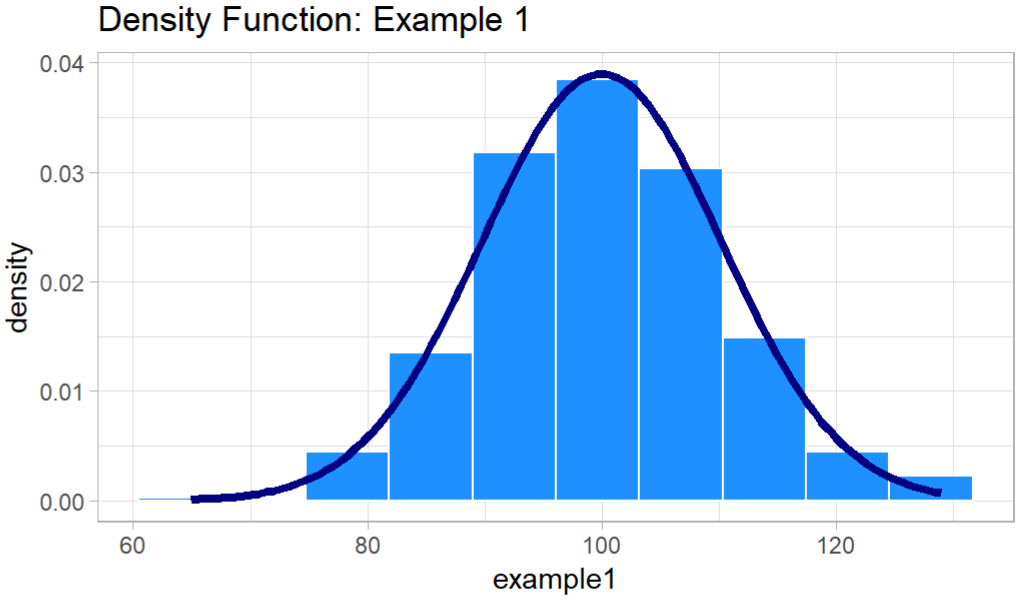
set.seed(431)
example1 <- rnorm(n = 500, mean = 100, sd = 10)
sim_study <- tibble(example1)

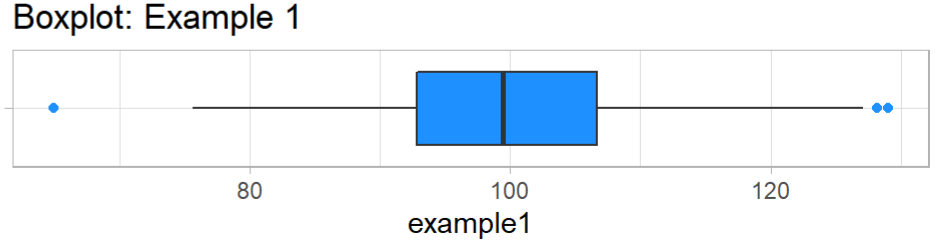
p1 <- ggplot(sim_study, aes(sample = example1)) +
  geom_qq(col = "dodgerblue") + geom_qq_line(col = "navy") +
  theme(aspect.ratio = 1) +
  labs(title = "Normal Q-Q plot: Example 1")

p2 <- ggplot(sim_study, aes(x = example1)) +
  geom_histogram(aes(y = stat(density)),
                 bins = 10, fill = "dodgerblue", col = "white") +
  stat_function(fun = dnorm,
                args = list(mean = mean(sim_study$example1),
```

Example 1. Sample from Normal model








```
1 mosaic::favstats(~ example1, data = sim_study)
```

```
     min        Q1    median        Q3       max      mean        sd   n missing
64.93932 92.84206 99.40395 106.6913 129.0048 99.97668 10.23073 500       0
```

Does a Normal model fit well?

1. Is a Normal Q-Q plot showing something close to a straight line, without clear signs of skew or indications of lots of outliers (heavy-tailedness)?
2. Does a boxplot, violin plot and/or histogram also show a symmetric distribution, where both the number of outliers is modest, and the distance of those outliers from the mean is modest?
3. Do numerical measures match up with the expectations of a normal model?

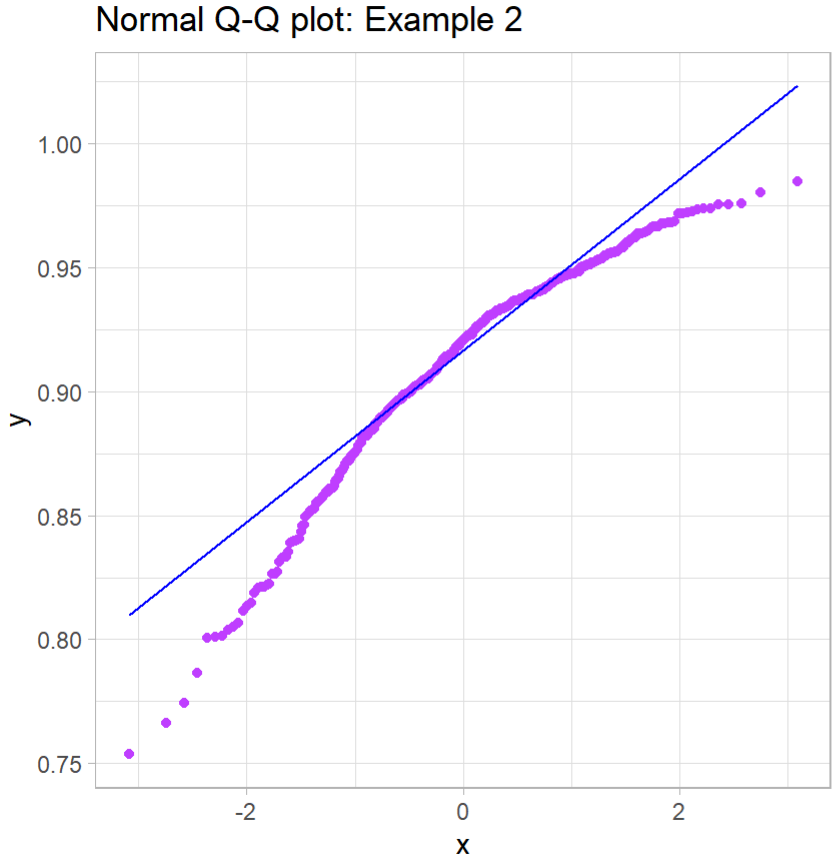
```
# code for Example 2
# plotting data sampled from
# a left-skewed distribution

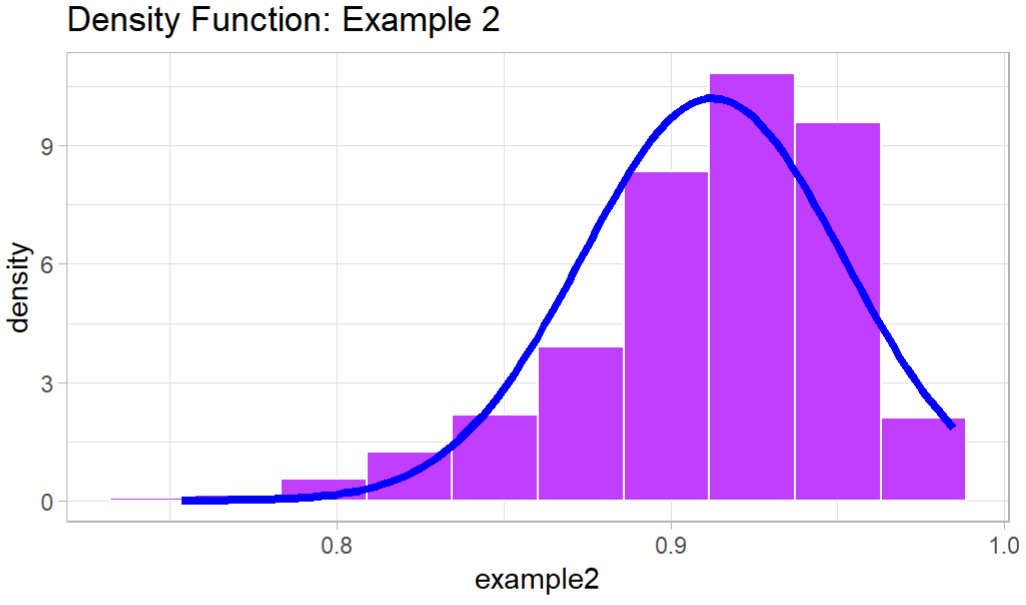
set.seed(431)
sim_study$example2 <- rbeta(n = 500, shape = 2, shape2 = 5, ncp = 100)

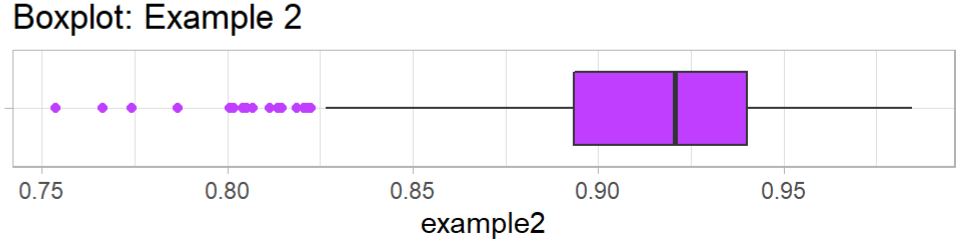
p1 <- ggplot(sim_study, aes(sample = example2)) +
  geom_qq(col = "darkorchid1") + geom_qq_line(col = "blue") +
  theme(aspect.ratio = 1) +
  labs(title = "Normal Q-Q plot: Example 2")

p2 <- ggplot(sim_study, aes(x = example2)) +
  geom_histogram(aes(y = stat(density)),
                 bins = 10, fill = "darkorchid1", col = "white") +
  stat_function(fun = dnorm,
                args = list(mean = mean(sim_study$example2),
                            sd = sd(sim_study$example2)),
                col = "blue", lwd = 1.5) +
```

Example 2. Left-Skewed Sample








```
1 mosaic::favstats(~ example2, data = sim_study)
```

```
      min         Q1    median        Q3       max      mean          sd   n
0.7535839 0.8934259 0.920609 0.939979 0.9847001 0.9125466 0.03907996 500
missing
      0
```

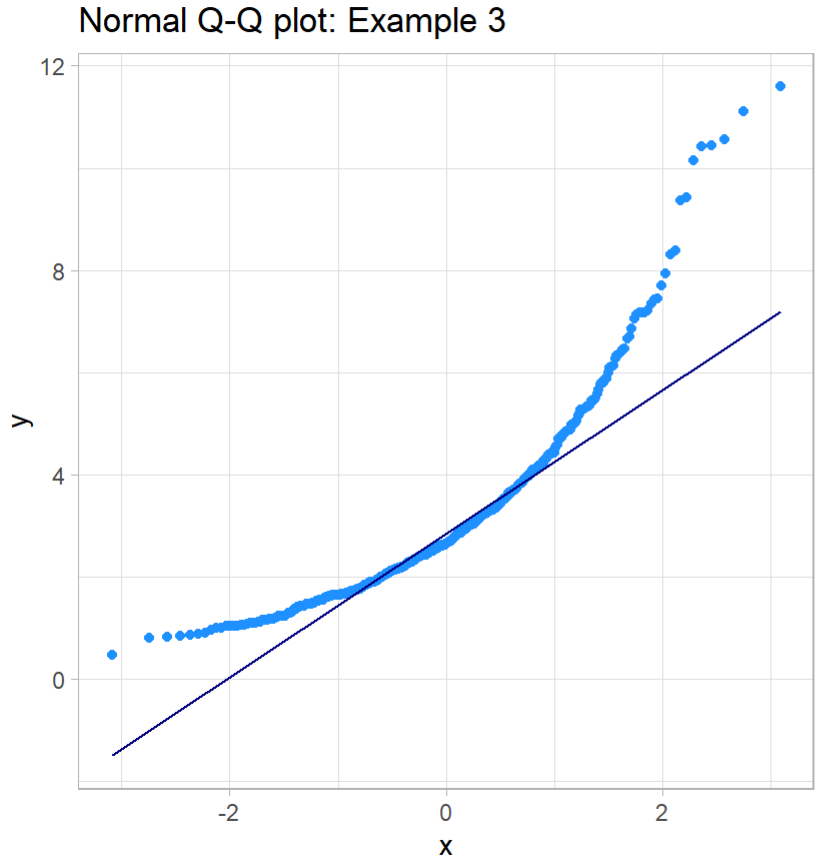
```
# code for Example 3
# plotting data sampled from
# a right-skewed distribution

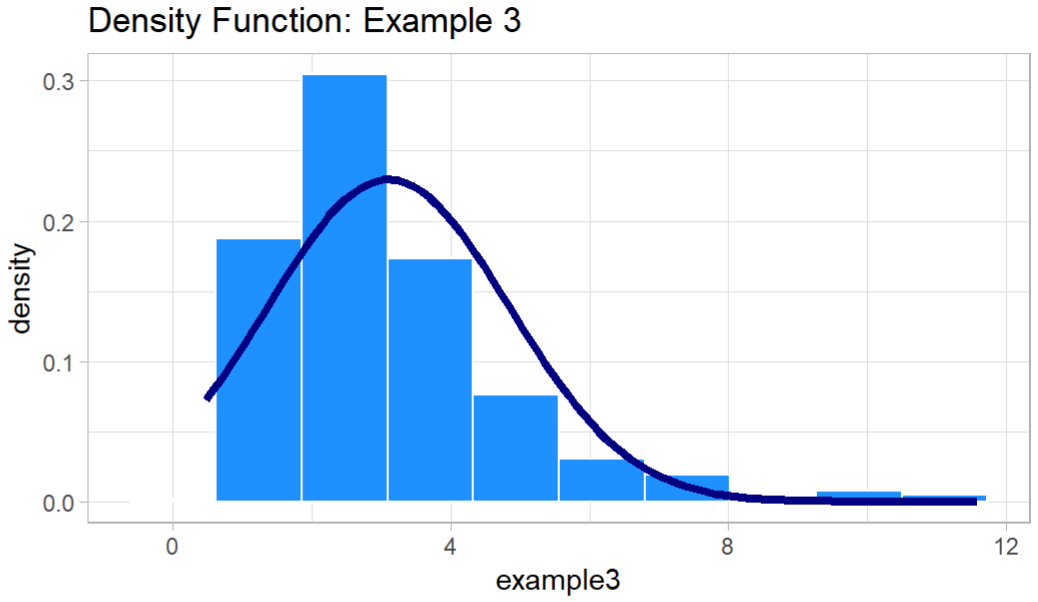
set.seed(431)
sim_study$example3 <- exp(rnorm(n = 500, mean = 1, sd = 0.5))

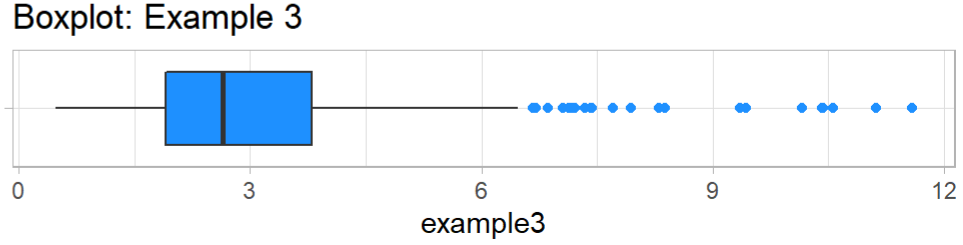
p1 <- ggplot(sim_study, aes(sample = example3)) +
  geom_qq(col = "dodgerblue") + geom_qq_line(col = "navy") +
  theme(aspect.ratio = 1) +
  labs(title = "Normal Q-Q plot: Example 3")

p2 <- ggplot(sim_study, aes(x = example3)) +
  geom_histogram(aes(y = stat(density)),
                 bins = 10, fill = "dodgerblue", col = "white") +
  stat_function(fun = dnorm,
                args = list(mean = mean(sim_study$example3),
                            sd = sd(sim_study$example3)),
```

Example 3. Right-Skewed Sample





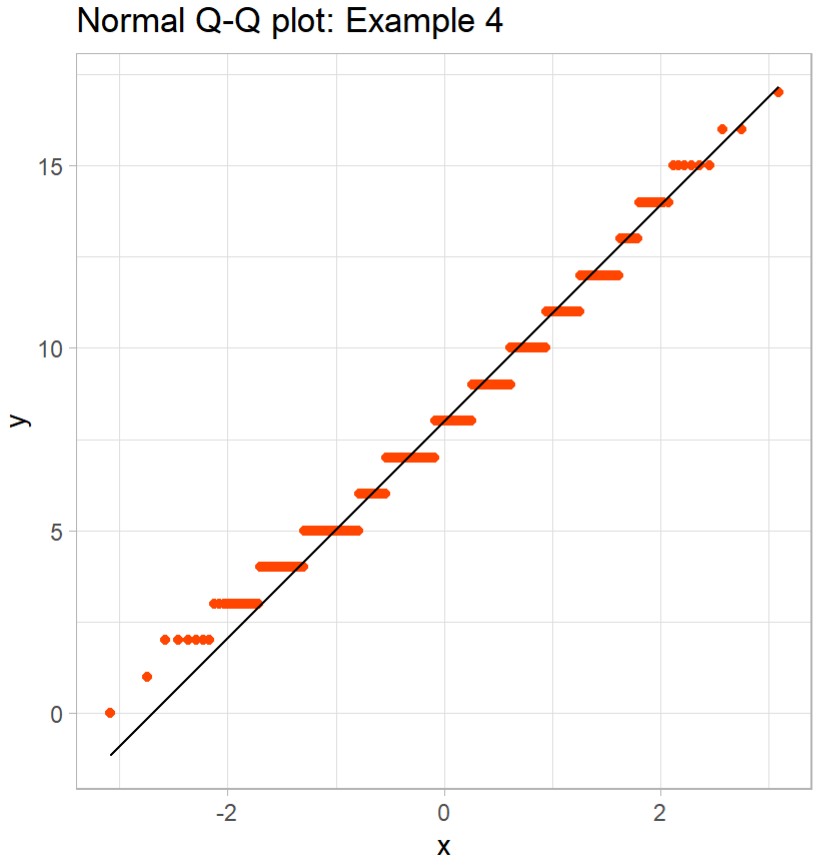


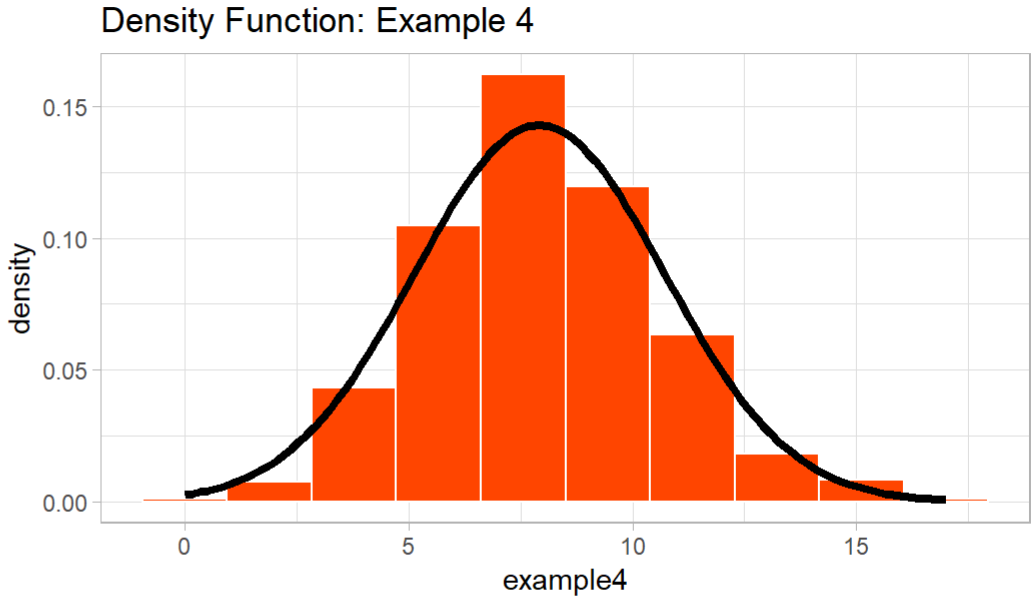

```
1 mosaic::favstats(~ example3, data = sim_study)
```

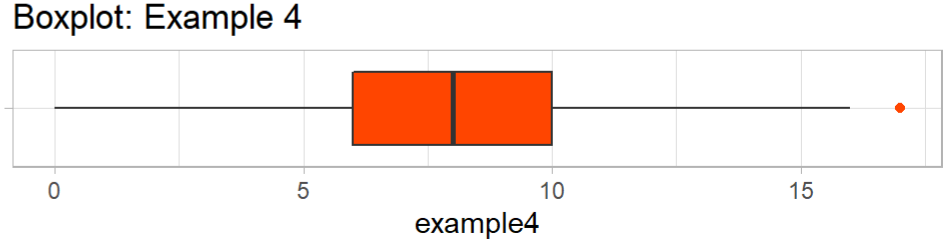
```
      min       Q1   median       Q3      max     mean       sd   n missing
0.4709357 1.900474 2.638468 3.798358 11.59111 3.101597 1.737721 500       0
```

```r
# code for Example 4
# plotting data sampled from
# a symmetric, but discrete distribution

set.seed(431)
sim_study$example4 <- rpois(n = 500, lambda = 8)

p1 <- ggplot(sim_study, aes(sample = example4)) +
  geom_qq(col = "orangered") + geom_qq_line(col = "black") +
  theme(aspect.ratio = 1) +
  labs(title = "Normal Q-Q plot: Example 4")

p2 <- ggplot(sim_study, aes(x = example4)) +
  geom_histogram(aes(y = stat(density)),
                 bins = 10, fill = "orangered", col = "white") +
  stat_function(fun = dnorm,
                args = list(mean = mean(sim_study$example4),
                            sd = sd(sim_study$example4)),
                col = "black", lwd = 1.5) +
```

Example 4. Symmetric Discrete Sample





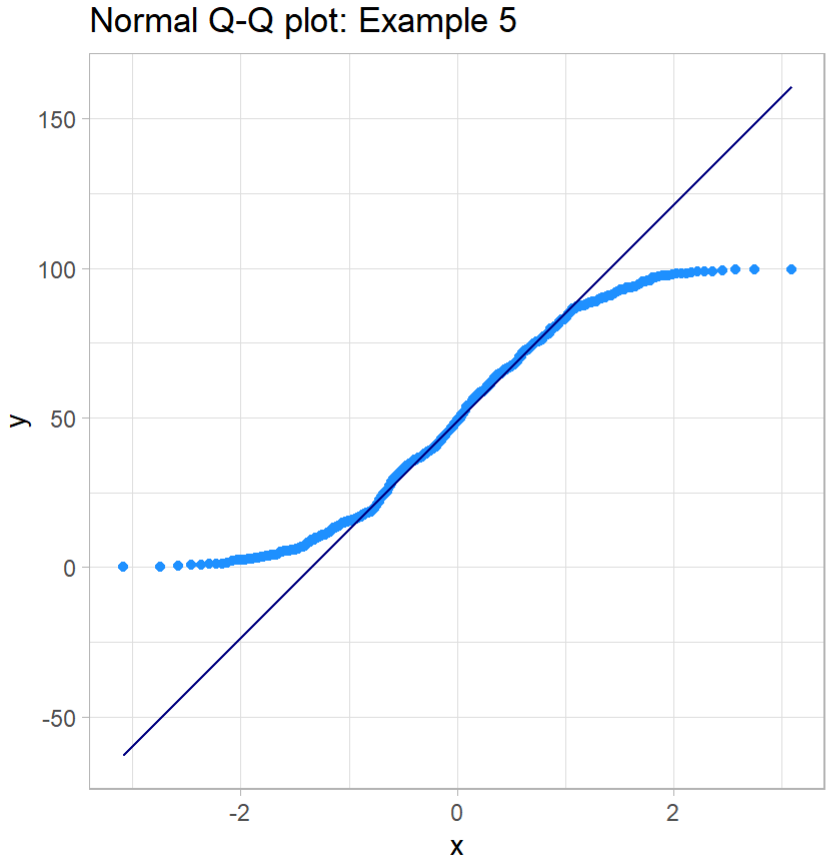


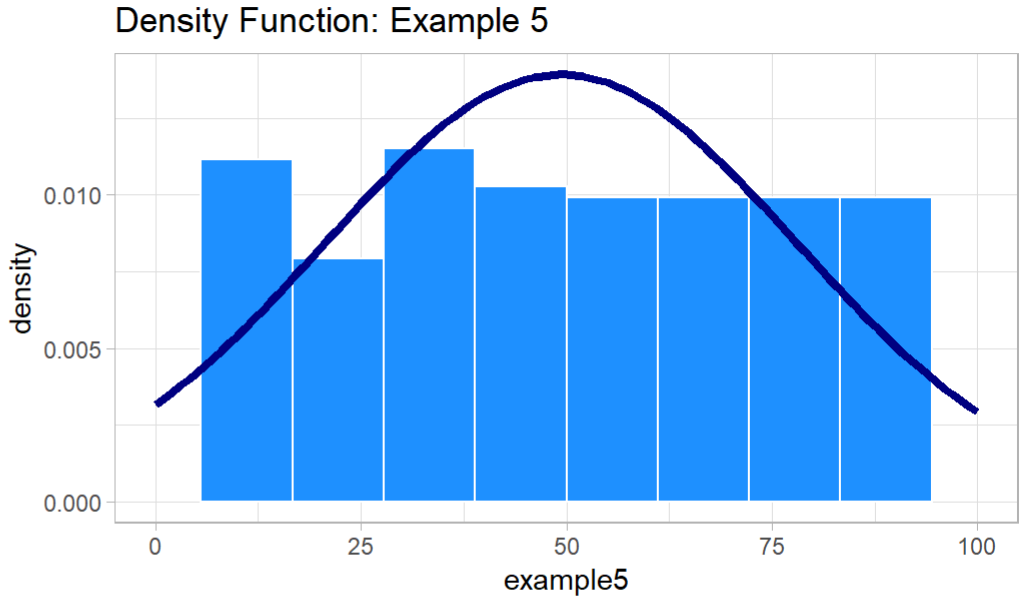

```
1 mosaic::favstats(~ example4, data = sim_study)
```

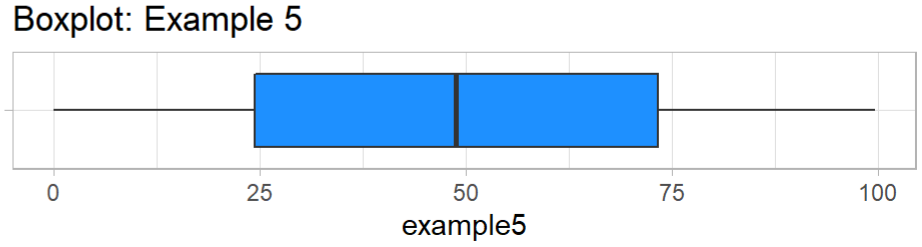
```
min Q1 median Q3 max  mean       sd   n missing
  0  6      8 10  17 7.916 2.792946 500       0
```

```
# code for Example 5
# plotting data sampled from
# a uniform distribution

set.seed(431)
sim_study$example5 <- runif(n = 500, min = 0, max = 100)

p1 <- ggplot(sim_study, aes(sample = example5)) +
  geom_qq(col = "dodgerblue") + geom_qq_line(col = "navy") +
  theme(aspect.ratio = 1) +
  labs(title = "Normal Q-Q plot: Example 5")

p2 <- ggplot(sim_study, aes(x = example5)) +
  geom_histogram(aes(y = stat(density)),
                 bins = 10, fill = "dodgerblue", col = "white") +
  stat_function(fun = dnorm,
                args = list(mean = mean(sim_study$example5),
                            sd = sd(sim_study$example5)),
```

Example 5. Uniform Sample





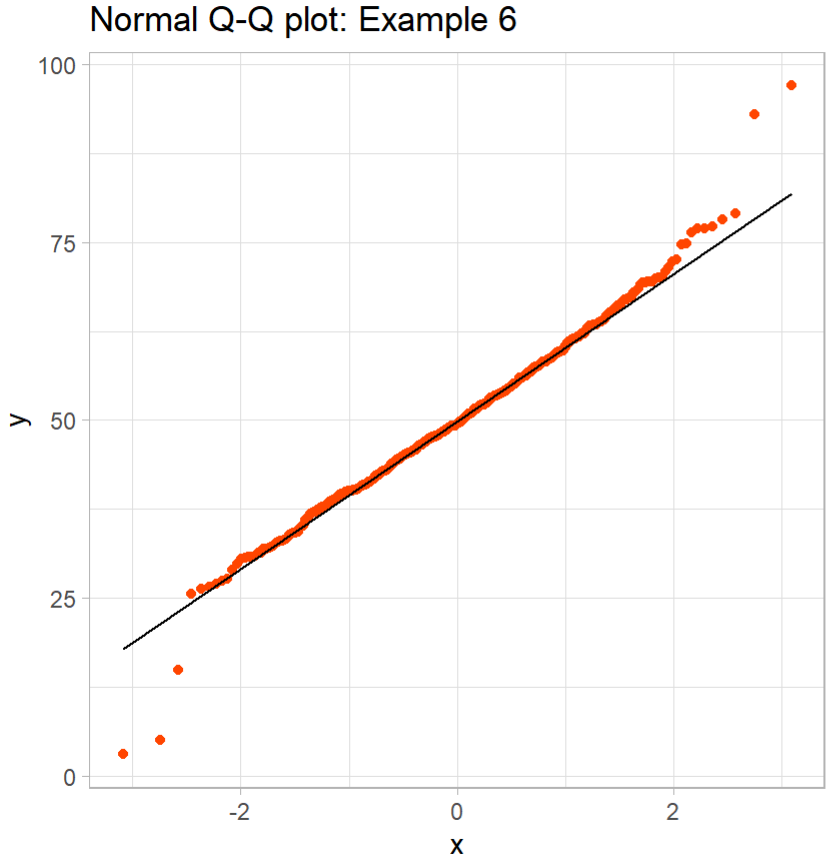


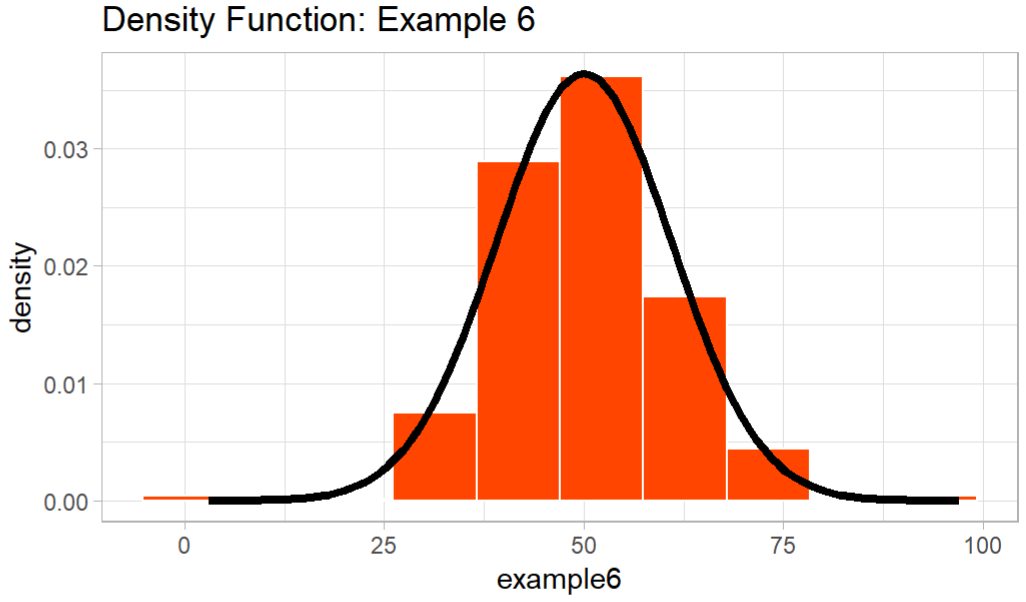

```
1 mosaic::favstats(~ example5, data = sim_study)
```

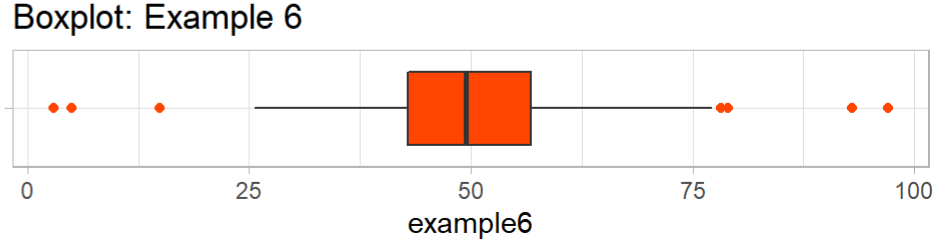
```
        min        Q1    median        Q3       max      mean        sd   n missing
0.02273887 24.46615 48.80411 73.2494 99.6397 49.34273 28.6585 500       0
```

```
# code for Example 6
# plotting data sampled from
# a symmetric, but heavy-tailed (outlier-prone) distribution

set.seed(431)
sim_study$example6 <- rnorm(n = 500, mean = 50, sd = 10)
sim_study$example6[14] <- 5
sim_study$example6[15] <- 3
sim_study$example6[39] <- 93
sim_study$example6[38] <- 97

p1 <- ggplot(sim_study, aes(sample = example6)) +
  geom_qq(col = "orangered") + geom_qq_line(col = "black") +
  theme(aspect.ratio = 1) +
  labs(title = "Normal Q-Q plot: Example 6")

p2 <- ggplot(sim_study, aes(x = example6)) +
  geom_histogram(aes(y = stat(density)),
```

Example 6. Symmetric, Outlier-Prone



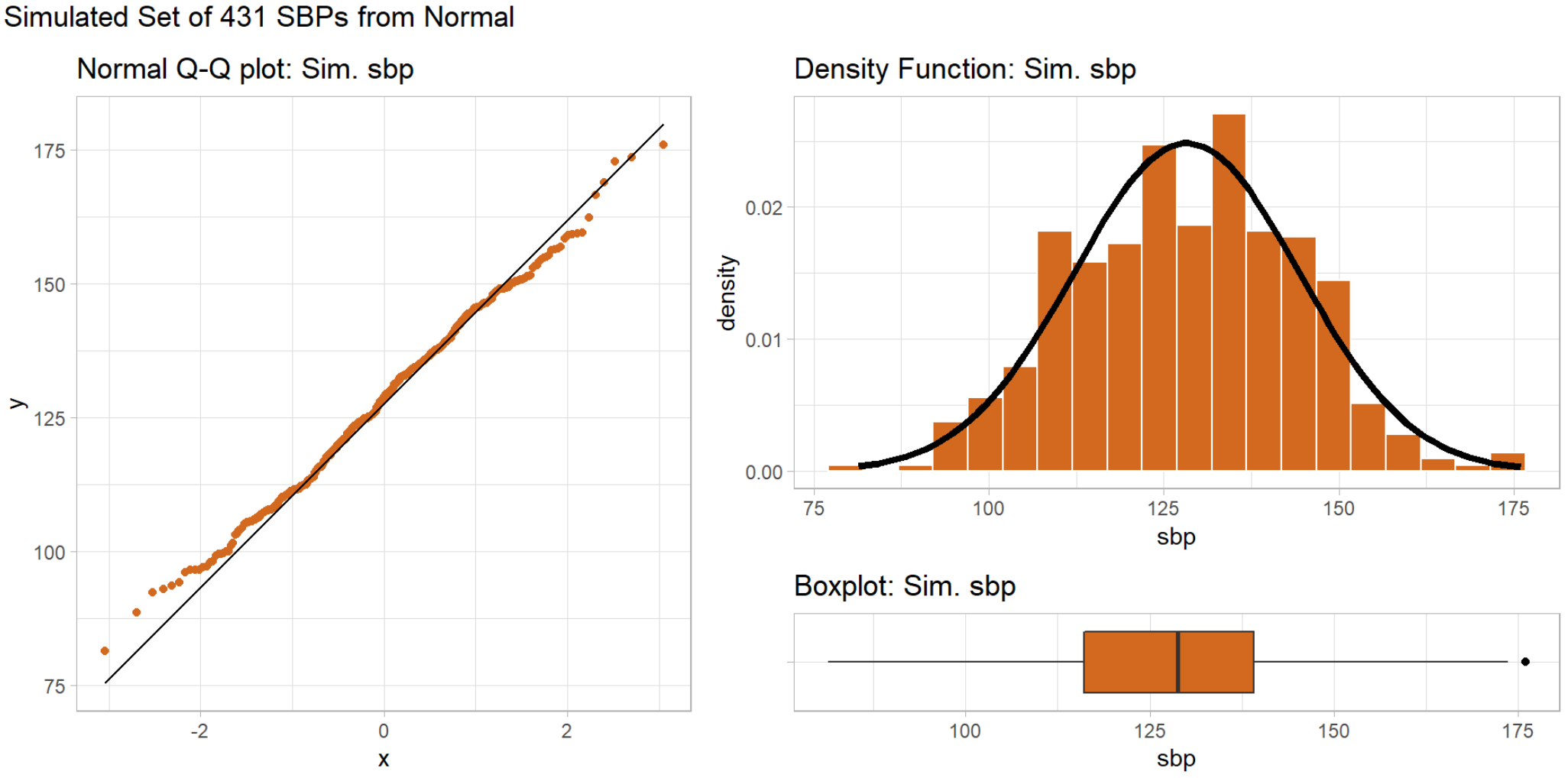




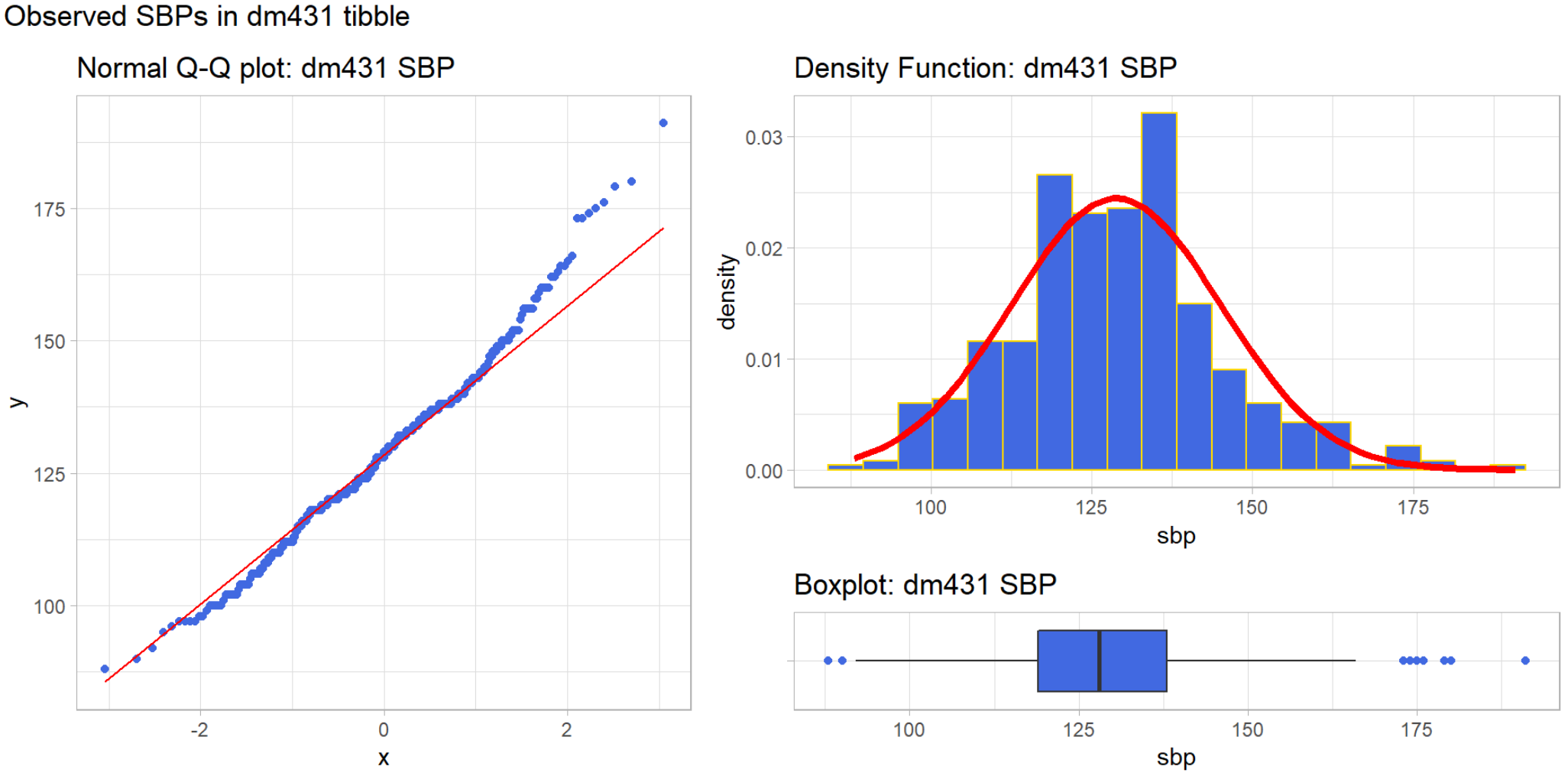

```
1 mosaic::favstats(~ example6, data = sim_study)
```

```
min       Q1   median       Q3 max     mean       sd   n missing
  3 42.84206 49.40395 56.8067  97 50.01256 10.96276 500       0
```

Our 431 Simulated Systolic BPs



Observed Systolic BP values in dm431



What Summaries to Report

It is usually helpful to focus on the **shape**, **center** and **spread** of a distribution.
Bock, Velleman and DeVeaux suggest:

- If the data are skewed, report the median and IQR (or the three middle quantiles).
  - You may want to include the mean and standard deviation, but you should point out why the mean and median differ.
  - The fact that the mean and median do not agree is a sign that the distribution may be skewed. A histogram will help you make that point.
- If the data are symmetric, report the mean and standard deviation, and possibly the median and IQR as well.
- If there are clear outliers and you are reporting the mean and standard deviation, report them with the outliers present and with the outliers removed.
  - The differences may be revealing. The median and IQR are not likely to be seriously affected by outliers.

Which summaries for sbp

Should we focus on, in light of our visualizations?

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

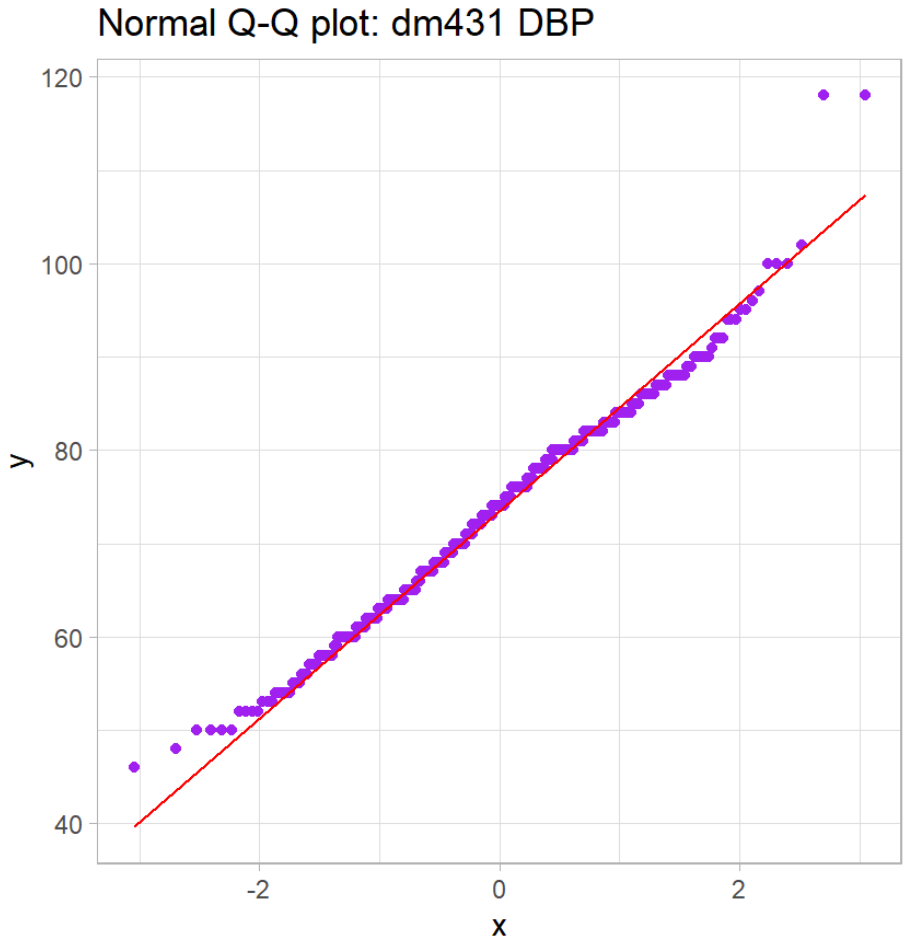
```
 min  Q1 median  Q3 max     mean       sd   n missing
  88 119    128 138 191 128.7889 16.33058 431       0
```

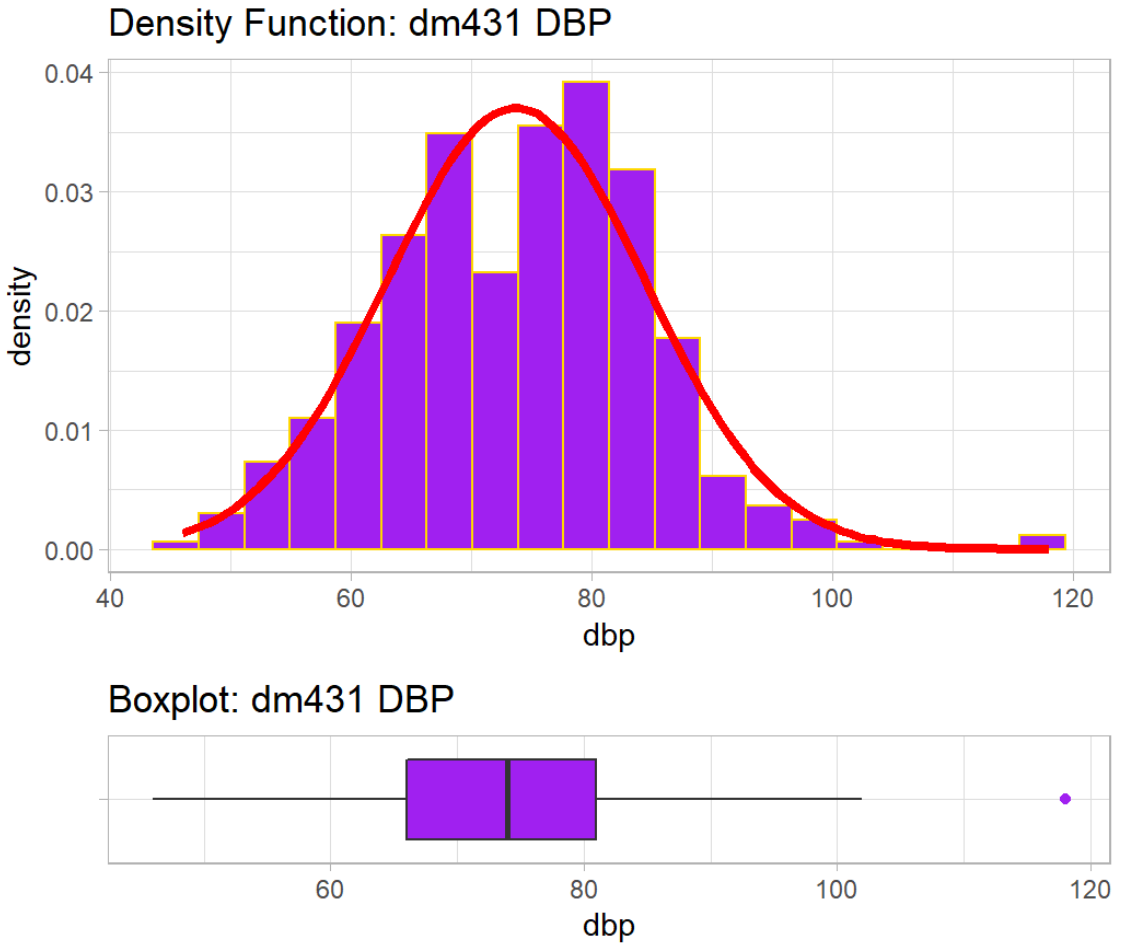
```
1 dm431 |> select(sbp) |> Hmisc::describe()
```

```
select(dm431, sbp)

 1  Variables      431  Observations
---------------------------------------------------------------------------
--
sbp
       n  missing distinct     Info     Mean      Gmd      .05      .10
     431        0       79    0.999    128.8    18.16      102      108
     .25      .50      .75      .90      .95
     119      128      138      149      157

lowest :  88  90  92  95  96, highest: 175 176 179 180 191
---------------------------------------------------------------------------
--
```

Observed Diastolic BP values in dm431

Observed DBPs in dm431 tibble





Stem-and-Leaf of dbp values?

1. Do we see any implausible diastolic blood pressures here?

```
1  stem(dm431$dbp, scale = 0.6, width = 55)
```

```
The decimal point is 1 digit(s) to the right of the |

 4 | 68
 5 | 000022223334444455566677778888888899
 6 | 00000000000011111112222222223333333344444+58
 7 | 000000000000000111111111222222222222333+83
 8 | 00000000000000000000000011111111111222222+64
 9 | 0000001224445567
10 | 0002
11 | 88
```

I've specified `scale` and `width` just for this slide.

Extreme dbp values?

Which are the subjects with unusual values of dbp?

```
dm431 |>
  filter(dbp < 50 | dbp > 110) |>
  select(class5_id, sbp, dbp)
```

```
# A tibble: 4 × 3
  class5_id   sbp   dbp
  <chr>     <dbl> <dbl>
1 S-005       156   118
2 S-202       124    46
3 S-219       120    48
4 S-240       158   118
```

Numerical Summaries for dbp?

Which summaries seem most useful for the dm431 dbp data?

```
1 mosaic::favstats(~ dbp, data = dm431)
```

```
 min Q1 median Q3 max     mean       sd   n missing
  46 66     74 81 118 73.70766 10.77089 431       0
```

```
1 Hmisc::describe(dm431$dbp)
```

```
dm431$dbp
       n  missing distinct     Info     Mean      Gmd      .05      .10
     431        0       51    0.999    73.71    12.08       56       60
     .25      .50      .75      .90      .95
      66       74       81       86       90

lowest :  46  48  50  52  53, highest:  96  97 100 102 118
```

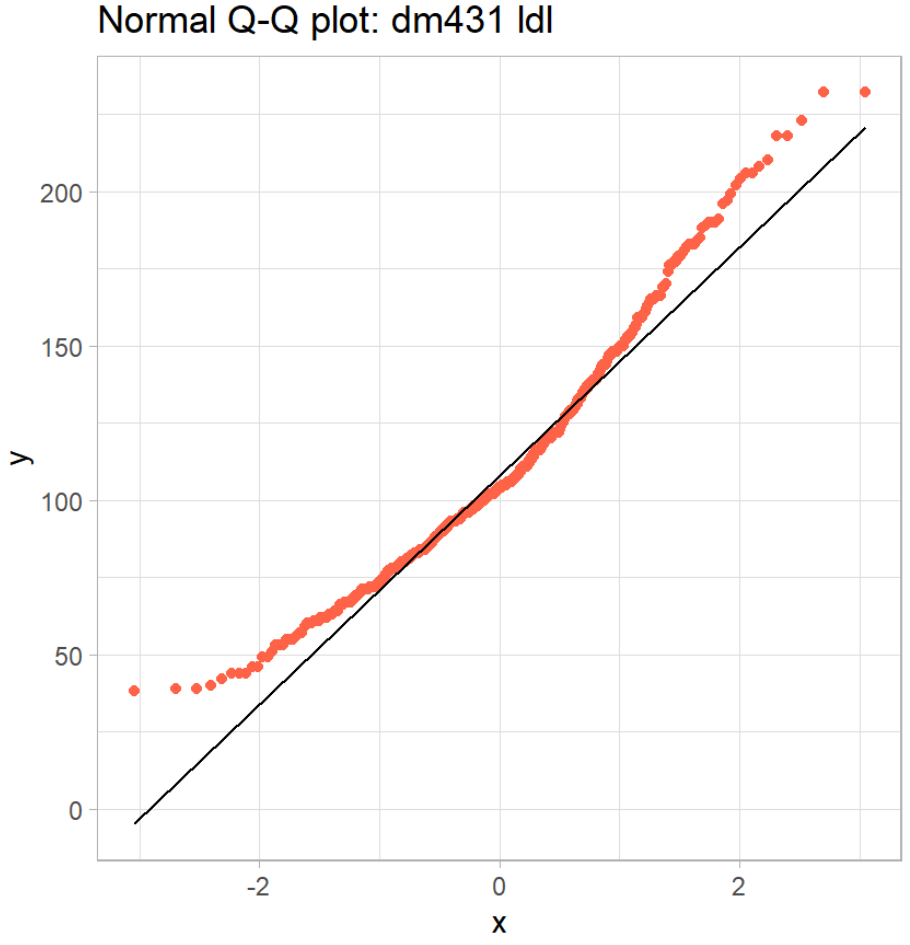
Does a Normal Model fit well?

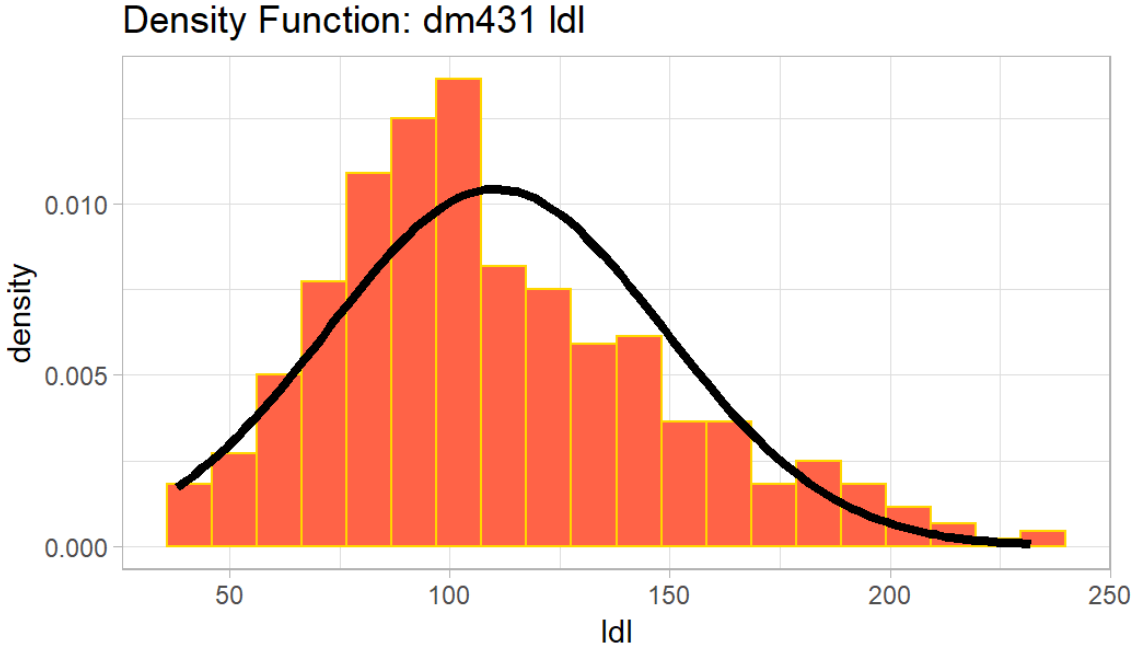
If a Normal model fits our data well, then we should see the following graphical indications:

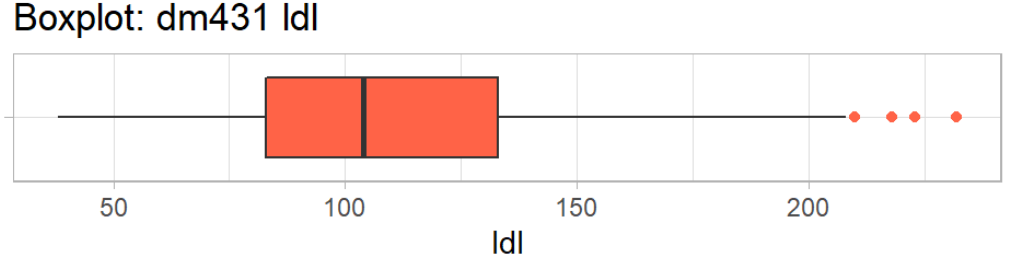
1. A histogram that is symmetric and bell-shaped.
2. A boxplot where the box is symmetric around the median, as are the whiskers, without serious outliers.
3. A normal Q-Q plot that essentially falls on a straight line.

LDL in dm431?

Observed LDL in dm431 tibble

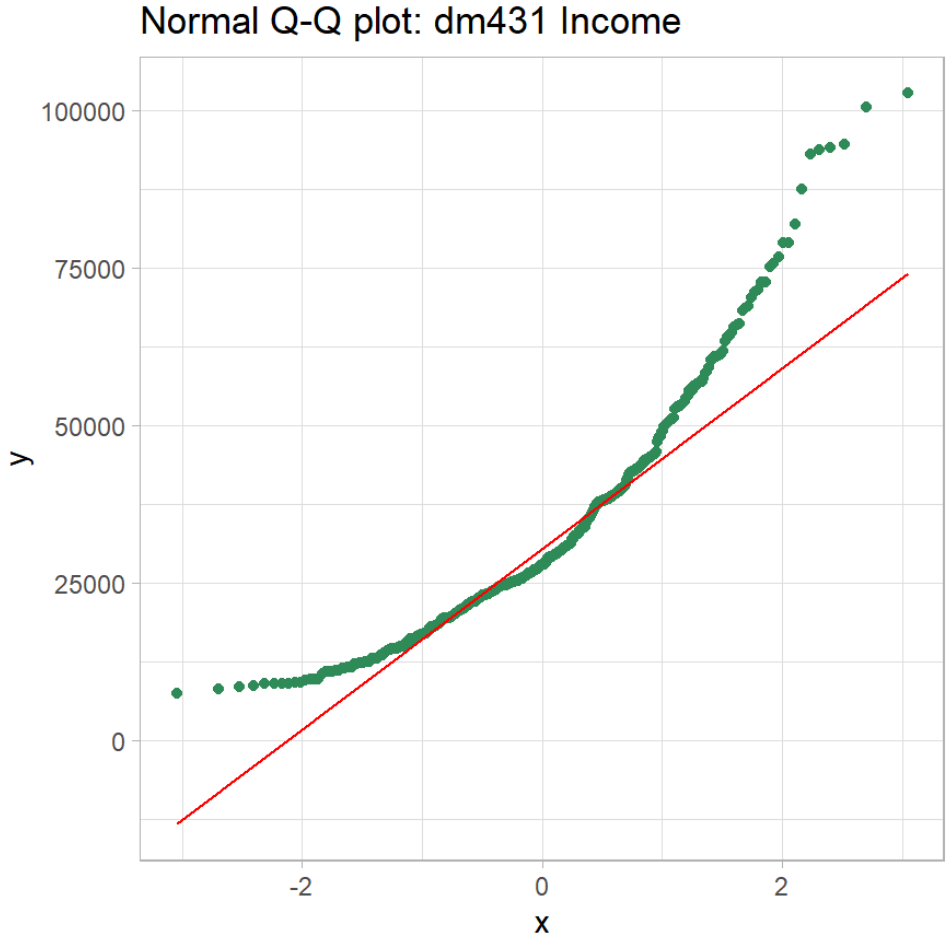


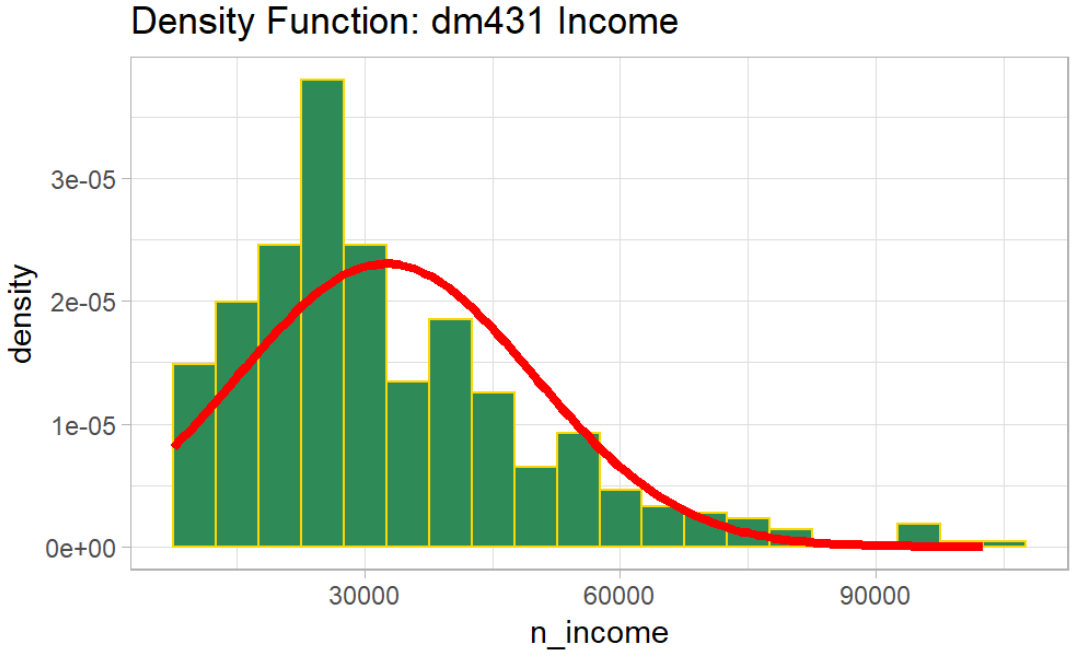


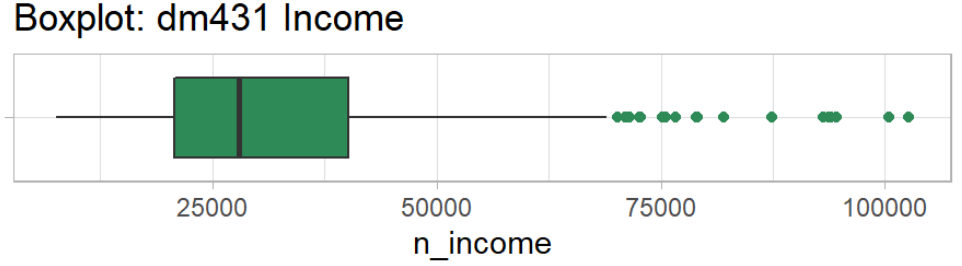


dm431: Neighborhood Income

Observed n_income in dm431





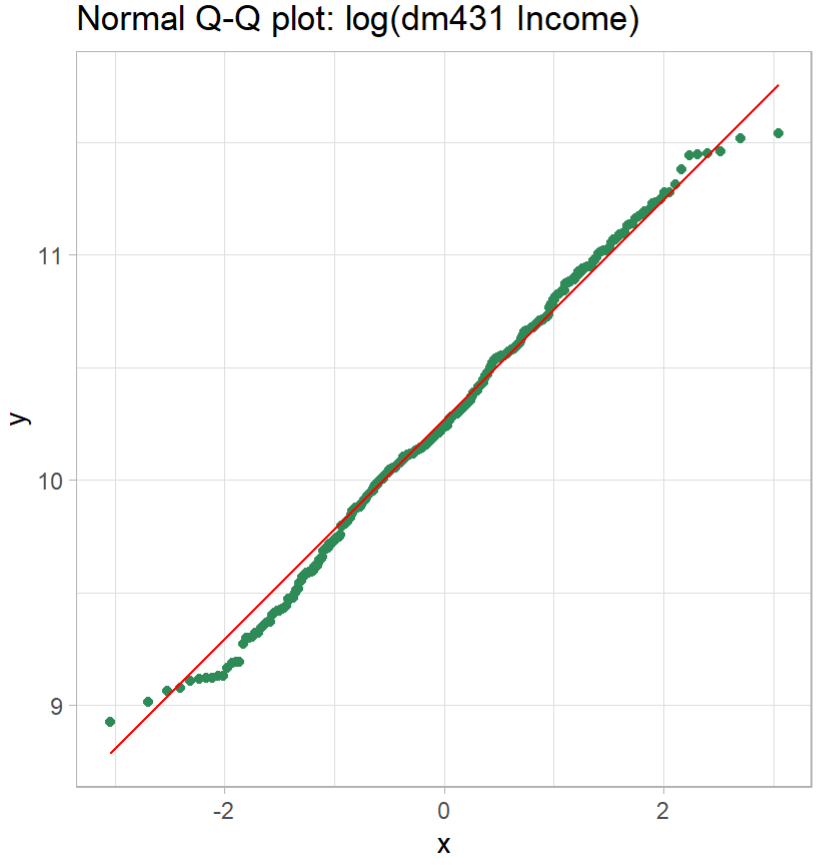


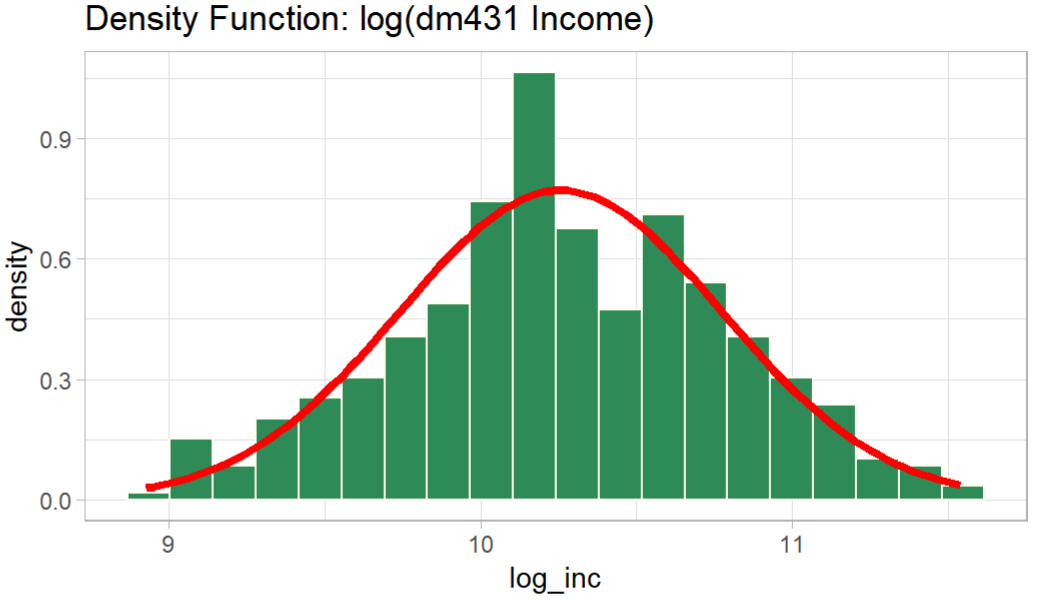
dm431: Natural Logarithm of Income

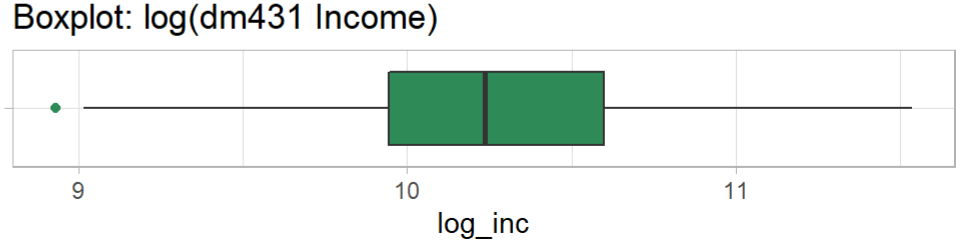
```
dm431 <- dm431 |> mutate(log_inc = log(n_income))

p1 <- ggplot(dm431, aes(sample = log_inc)) +
  geom_qq(col = "seagreen") + geom_qq_line(col = "red") +
  theme(aspect.ratio = 1) +
  labs(title = "Normal Q-Q plot: log(dm431 Income)")

p2 <- ggplot(dm431, aes(x = log_inc)) +
  geom_histogram(aes(y = stat(density)),
                 bins = 20, fill = "seagreen", col = "ivory") +
  stat_function(fun = dnorm,
                args = list(mean = mean(dm431$log_inc),
                            sd = sd(dm431$log_inc)),
                col = "red", lwd = 1.5) +
  labs(title = "Density Function: log(dm431 Income)")

p3 <- ggplot(dm431, aes(x = log_inc, y = "")) +
  geom_boxplot(fill = "seagreen", outlier.color = "seagreen") +
  labs(title = "Boxplot: log(dm431 Income)", y = "")
```

dm431: Natural Logarithm of Income

Observed log(n_income) in dm431





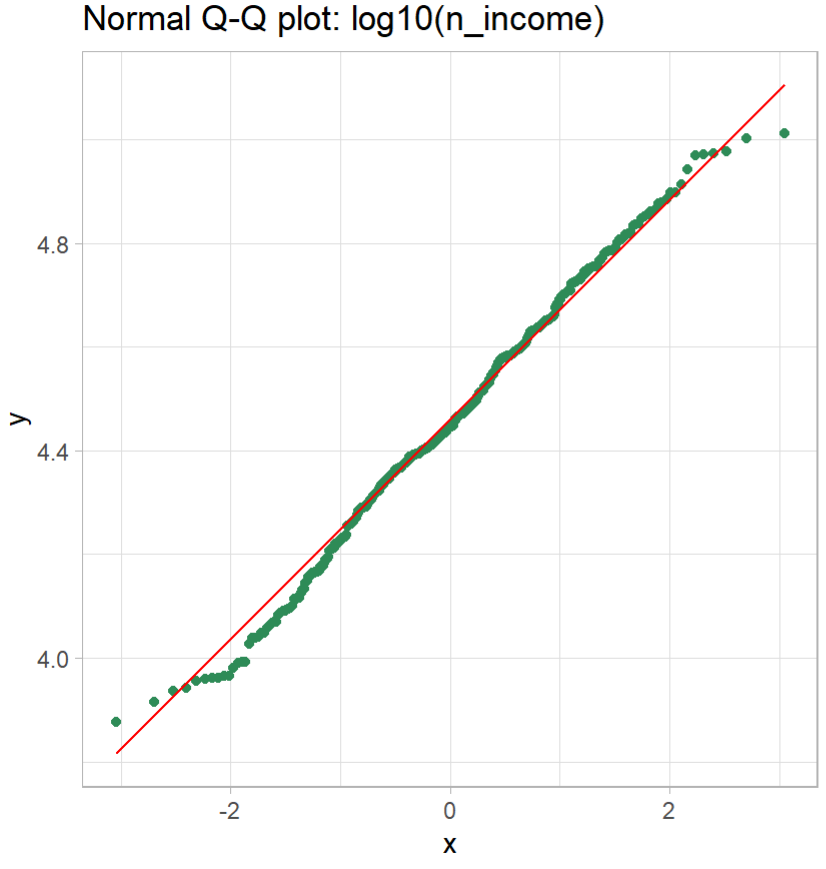


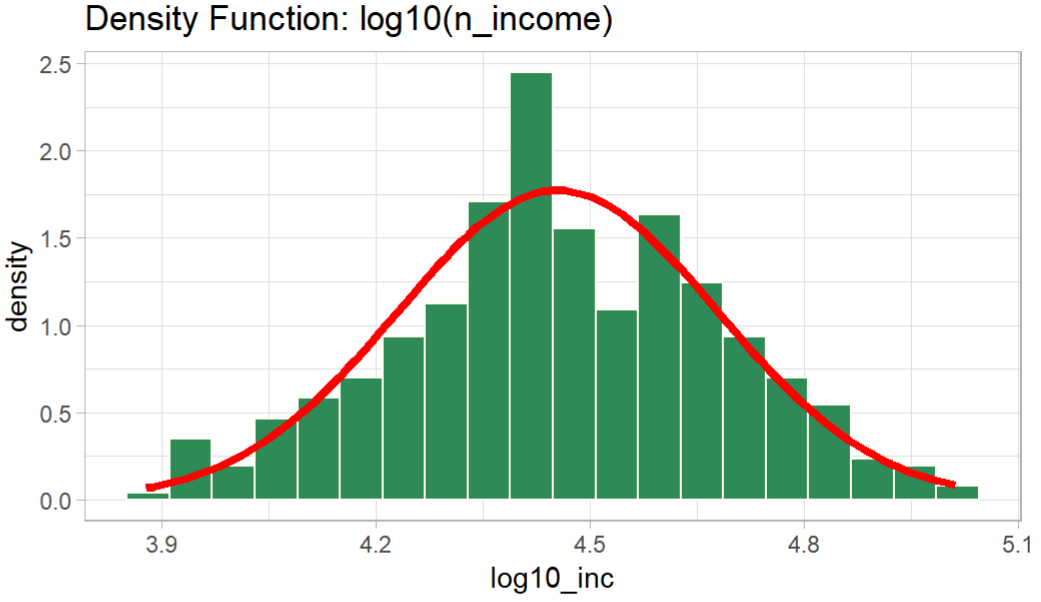
dm431: Base-10 Logarithm of Income

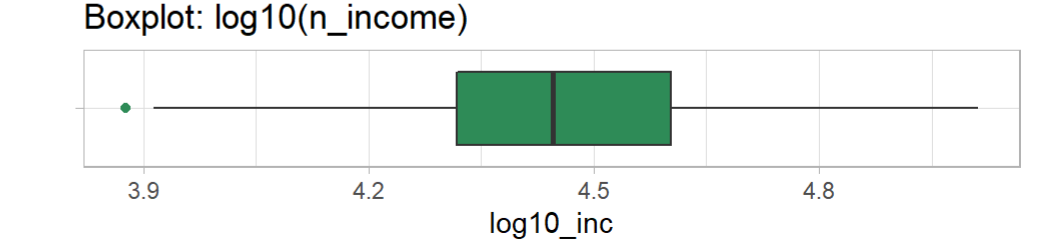
```
dm431 <- dm431 |> mutate(log10_inc = log10(n_income))

p1 <- ggplot(dm431, aes(sample = log10_inc)) +
  geom_qq(col = "seagreen") + geom_qq_line(col = "red") +
  theme(aspect.ratio = 1) +
  labs(title = "Normal Q-Q plot: log10(n_income)")

p2 <- ggplot(dm431, aes(x = log10_inc)) +
  geom_histogram(aes(y = stat(density)),
                 bins = 20, fill = "seagreen", col = "ivory") +
  stat_function(fun = dnorm,
                args = list(mean = mean(dm431$log10_inc),
                            sd = sd(dm431$log10_inc)),
                col = "red", lwd = 1.5) +
  labs(title = "Density Function: log10(n_income)")

p3 <- ggplot(dm431, aes(x = log10_inc, y = "")) +
  geom_boxplot(fill = "seagreen", outlier.color = "seagreen") +
  labs(title = "Boxplot: log10(n_income)", y = "")
```

dm431: Base-10 Logarithm of Income

Observed log10(n_income) in dm431







Using Numerical Summaries to Assess Normality: A Good Idea?

Comparing Numerical Summaries

```
1 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
```

```
1 mosaic::favstats(~ sbp, data = sim_data)
```

```
      min       Q1  median       Q3      max     mean       sd   n missing
 81.41991 116.0579 128.777 139.1657 175.9422 128.2372 16.05684 431       0
```

What can we learn from these comparisons…

- about the center of the data?
- about the spread of the data?
- about the shape of the data?
- about whether a Normal model fits well?

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 skew in dm431 SBP?

```
1 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
```

```
1 dm431 |>
2   summarize(skew1 = (mean(sbp) - median(sbp))/sd(sbp))
```

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

What does this suggest?

$skew_1$ for other dm431 variables

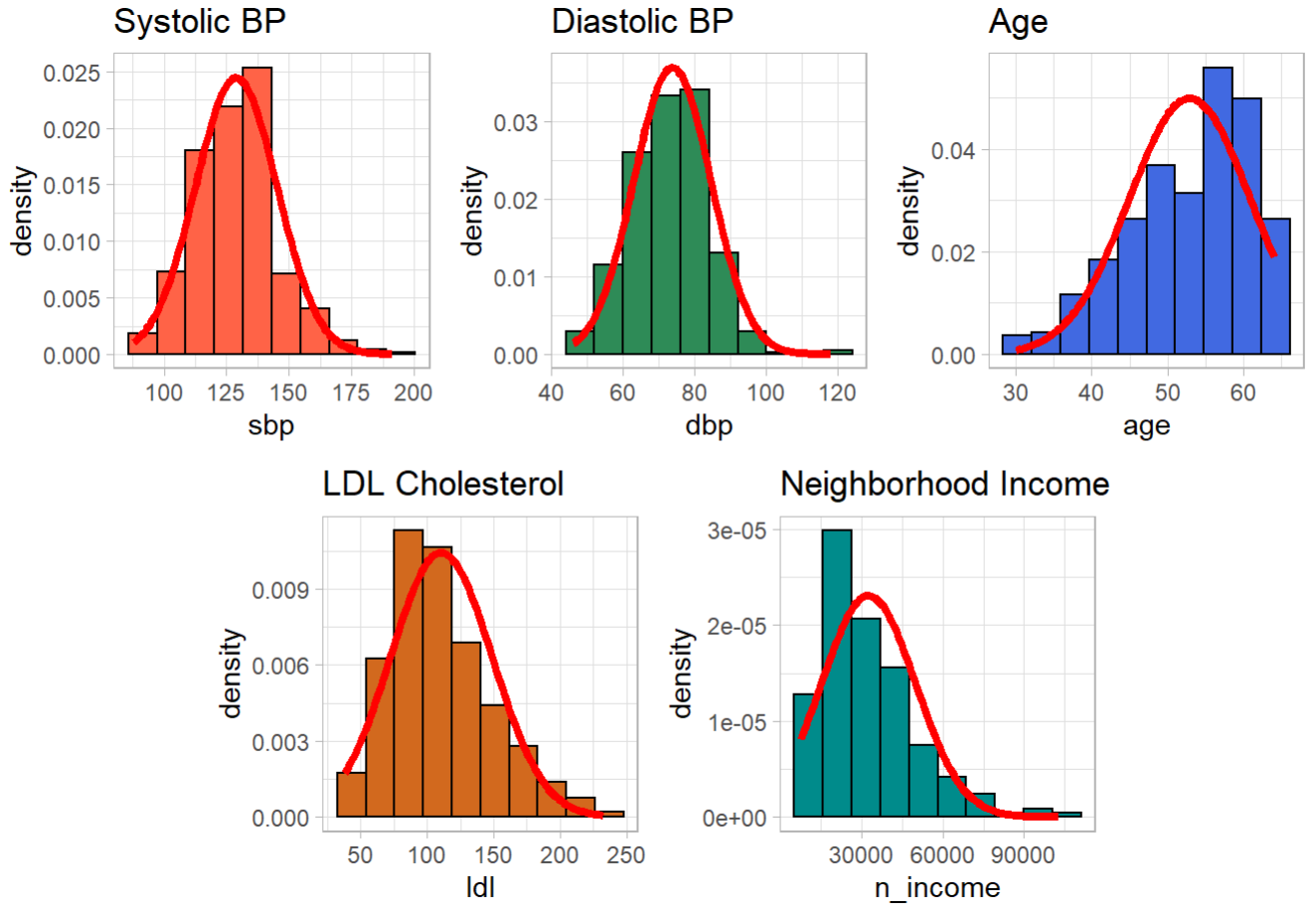
Variable	$\bar{x}$ = mean	median	s = SD	$skew_1$
sbp	128.8	128	16.3	0.05
dbp	73.7	74	10.8	-0.03
age	52.9	54	8	-0.14
ldl	110.5	104	38.3	0.17
n_income	3.2514^{4}	2.7903^{4}	1.7295^{4}	0.27

- Don't draw conclusions without a plot!
- Does this tell us anything about outliers?

Histograms for dm431

```
p1a <- ggplot(dm431, aes(x = sbp)) +
  geom_histogram(aes(y = stat(density)),
                 bins = 10, fill = "tomato", col = "black") +
  stat_function(fun = dnorm,
                args = list(mean = mean(dm431$sbp),
                            sd = sd(dm431$sbp)),
                col = "red", lwd = 1.5) +
  theme(aspect.ratio = 1) +
  labs(title = "Systolic BP")

p1b <- ggplot(dm431, aes(x = dbp)) +
  geom_histogram(aes(y = stat(density)),
                 bins = 10, fill = "seagreen", col = "black") +
  stat_function(fun = dnorm,
                args = list(mean = mean(dm431$dbp),
                            sd = sd(dm431$dbp)),
                col = "red", lwd = 1.5) +
  theme(aspect.ratio = 1) +
  labs(title = "Diastolic BP")
```

Histograms for dm431



For any data set...

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).

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

SBPs within 1 SD of the mean?

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

```
# A tibble: 3 × 3
  `sbp > mean(sbp) - sd(sbp)` `sbp < mean(sbp) + sd(sbp)`     n
  <lgl>                       <lgl>                       <int>
1 FALSE                       TRUE                           70
2 TRUE                        FALSE                          55
3 TRUE                        TRUE                          306
```

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

SBP and $\bar{x} \pm 2s$ rule?

```
1 dm431 |>
2   count(sbp > mean(sbp) - 2*sd(sbp),
3         sbp < mean(sbp) + 2*sd(sbp))
```

```
# A tibble: 3 × 3
  `sbp > mean(sbp) - 2 * sd(sbp)` `sbp < mean(sbp) + 2 * sd(sbp)`     n
  <lgl>                           <lgl>                           <int>
1 FALSE                           TRUE                                5
2 TRUE                            FALSE                              15
3 TRUE                            TRUE                              411
```

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

Coverage Probabilities in dm431

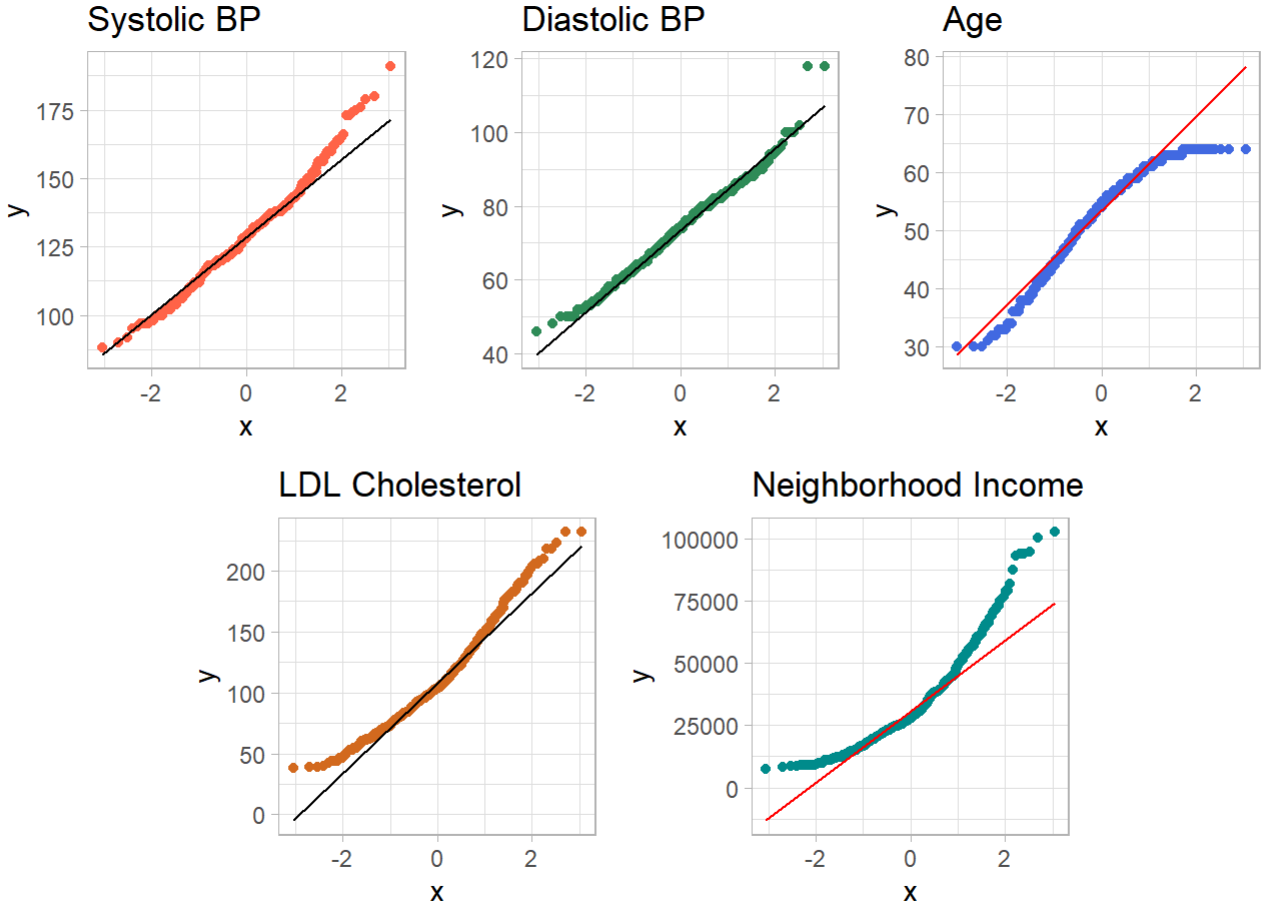
Variable	$\bar{x}$	s = SD	$\bar{x} \pm s$	$\bar{x} \pm 2s$	$\bar{x} \pm 3s$
sbp	128.8	16.3	71%	95.4%	99.3%
dbp	73.7	10.8	71%	95.8%	99.5%
age	52.9	8	65.2%	95.8%	100%
ldl	110.5	38.3	68.4%	95.4%	99.5%
n_income	32514	17295	72.2%	95.1%	98.4%

- Conclusions about utility of the Normal model?
- Do these match the conclusions from the plots? —>

Normal Q-Q plots for dm431

```
p1a <- ggplot(dm431, aes(sample = sbp)) +
  geom_qq(col = "tomato") + geom_qq_line(col = "black") +
  theme(aspect.ratio = 1) +
  labs(title = "Systolic BP")

p1b <- ggplot(dm431, aes(sample = dbp)) +
  geom_qq(col = "seagreen") + geom_qq_line(col = "black") +
  theme(aspect.ratio = 1) +
  labs(title = "Diastolic BP")

p1c <- ggplot(dm431, aes(sample = age)) +
  geom_qq(col = "royalblue") + geom_qq_line(col = "red") +
  theme(aspect.ratio = 1) +
  labs(title = "Age")

p1d <- ggplot(dm431, aes(sample = ldl)) +
  geom_qq(col = "chocolate") + geom_qq_line(col = "black") +
  theme(aspect.ratio = 1) +
  labs(title = "LDL Cholesterol")
```

Normal Q-Q plots for dm431



Should we use hypothesis tests to assess Normality?

Hypothesis Testing to assess Normality

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

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

```
    Shapiro-Wilk normality test

data:  dm431$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? Alas, not actually useful.

Other Hypothesis Tests of Normality

The `nortest` package, which I don't even install as part of our 431 packages, includes many other possible tests of Normality for a batch of data, including:

Why not test for Normality? (1)

There are multiple hypothesis testing schemes 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.

Why not test for Normality? (2)

- 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.

Does a Normal Model fit well?

Do we have…

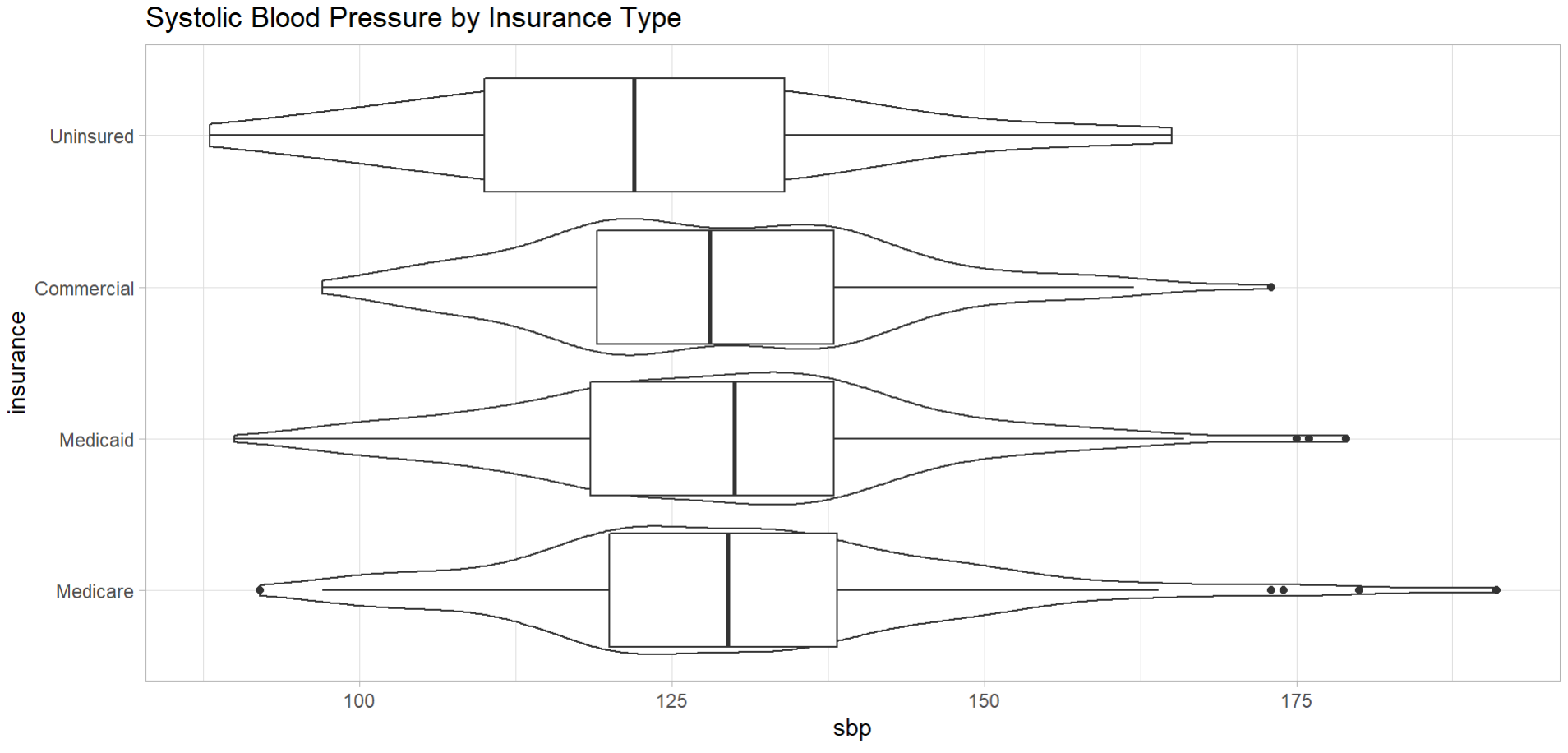
1. A histogram that is symmetric and bell-shaped.
2. A boxplot where the box is symmetric around the median, as are the whiskers, without severe outliers.
3. A normal Q-Q plot that essentially falls on a straight line.
4. If in doubt, maybe compare mean to median re: skew, and consider Empirical Rule to help make tough calls.

Big issue: why do you need to assume a Normal model?

Comparing SBP by Insurance, Attempt 1

```
ggplot(data = dm431, aes(x = sbp, y = insurance)) +
  geom_violin() +
  geom_boxplot() +
  labs(title = "Systolic Blood Pressure by Insurance Type",
       caption = "dm431 tibble")
```

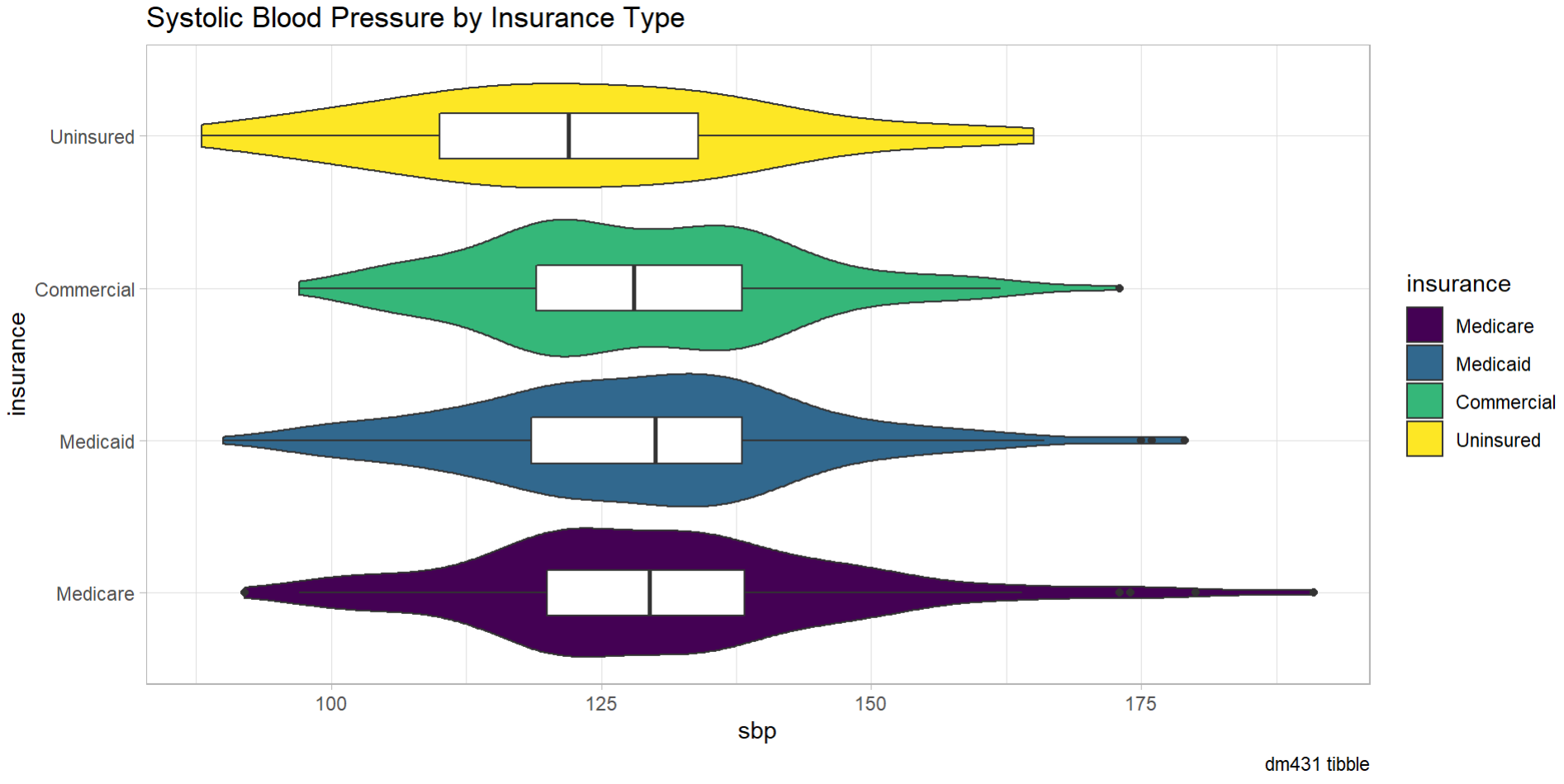
Comparing SBP by Insurance, Attempt 1



Comparing SBP by Insurance, Attempt 2

```
ggplot(data = dm431, aes(x = sbp, y = insurance)) +
  geom_violin(aes(fill = insurance)) +
  geom_boxplot(width = 0.3) +
  scale_fill_viridis_d() +
  labs(title = "Systolic Blood Pressure by Insurance Type",
       caption = "dm431 tibble")
```

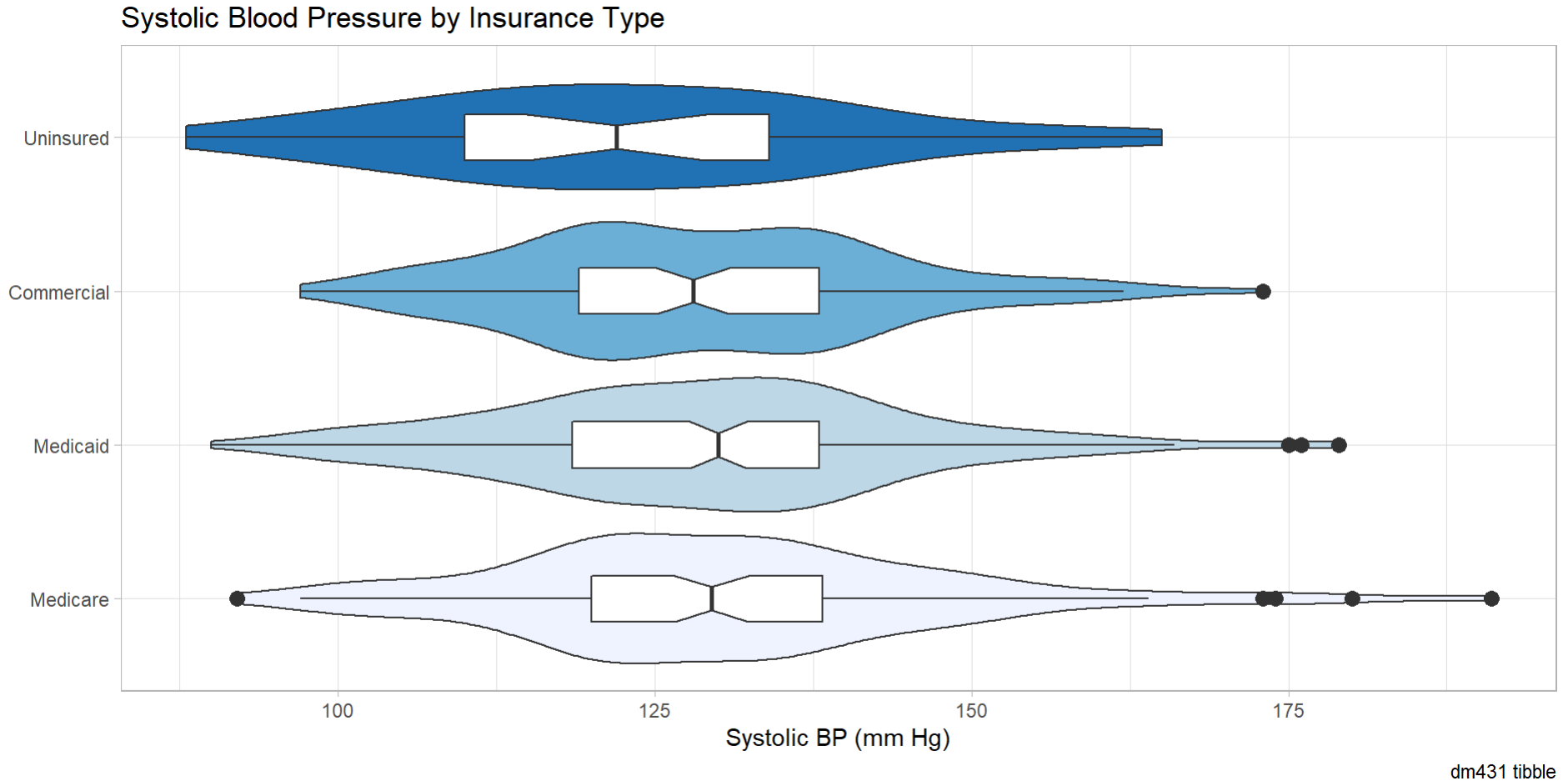
Comparing SBP by Insurance, Attempt 2



Comparing SBP by Insurance, Attempt 3

```
ggplot(data = dm431, aes(x = sbp, y = insurance)) +
  geom_violin(aes(fill = insurance)) +
  geom_boxplot(width = 0.3, notch = TRUE, outlier.size = 3) +
  scale_fill_brewer() +
  guides(fill = "none", col = "none") +
  labs(title = "Systolic Blood Pressure by Insurance Type",
       caption = "dm431 tibble",
       y = "", x = "Systolic BP (mm Hg)")
```

Comparing SBP by Insurance, Attempt 3



Numerical Summaries: SBP by Insurance

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

```
   insurance min    Q1 median     Q3 max     mean       sd   n missing
1   Medicare  92 120.0  129.5 138.25 191 130.5700 17.89885 100       0
2   Medicaid  90 118.5  130.0 138.00 179 128.8325 16.12919 191       0
3 Commercial  97 119.0  128.0 138.00 173 128.6216 14.50834 111       0
4  Uninsured  88 110.0  122.0 134.00 165 123.0000 18.01190  29       0
```

```
mod1 <- lm(sbp ~ insurance, data = dm431)

anova(mod1) # only makes sense if comparing means makes sense
```

```
Analysis of Variance Table

Response: sbp
            Df Sum Sq Mean Sq F value Pr(>F)
insurance    3   1293  430.84  1.6226 0.1834
Residuals  427 113383  265.53
```

Neighborhood Income and Eye Exam

```
1 dm431 |> select(n_income, eye_exam) |> summary()
```

```
   n_income         eye_exam      
Min.   :  7534   Min.   :0.0000  
1st Qu.: 20794   1st Qu.:0.0000  
Median : 27903   Median :1.0000  
Mean   : 32514   Mean   :0.6079  
3rd Qu.: 40128   3rd Qu.:1.0000  
Max.   :102672   Max.   :1.0000  
```

Need to convert `eye_exam` to a factor

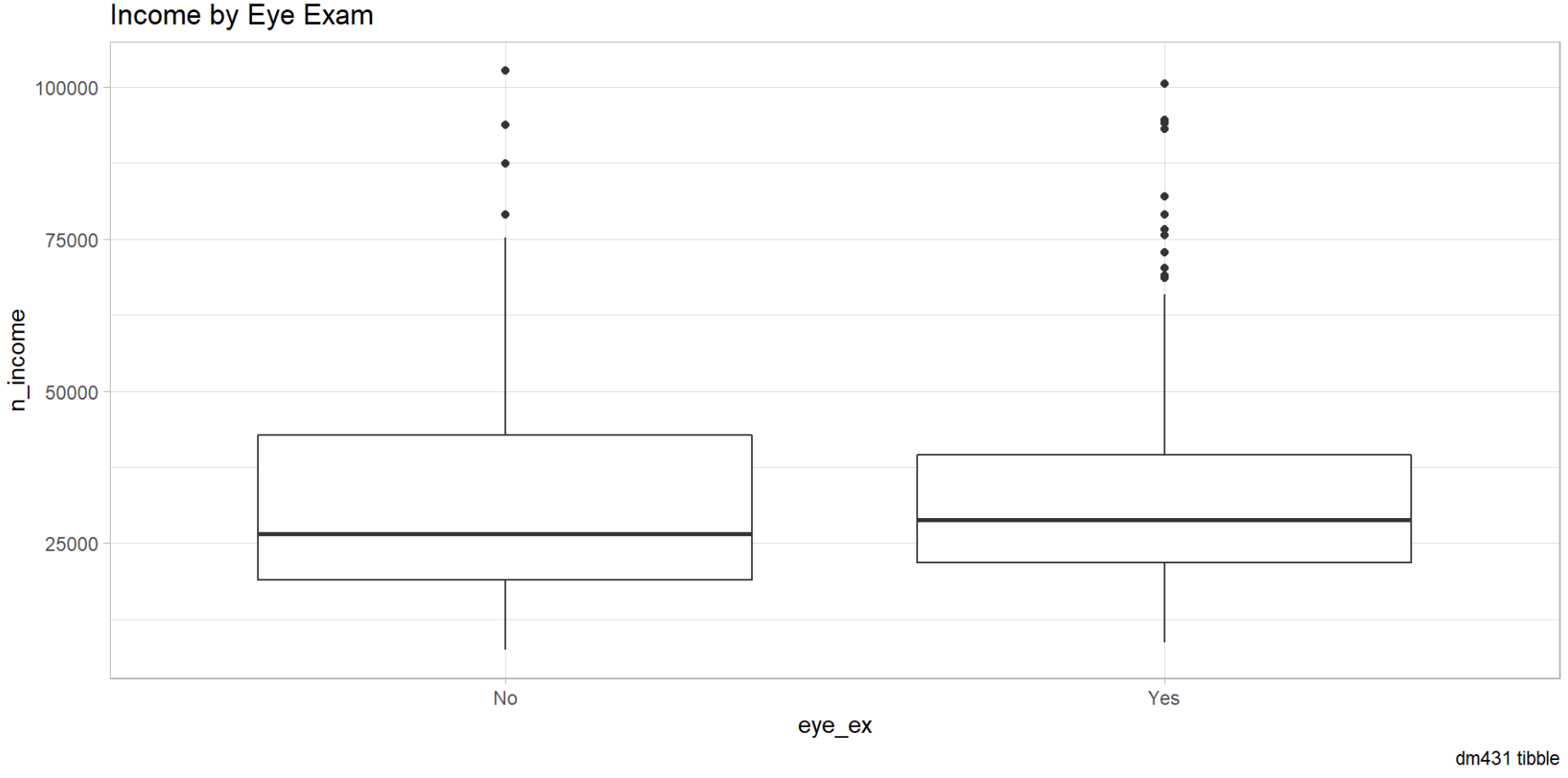
Also, we want to create levels Yes (1) and No (0).

```
dm431 <- dm431 |>
  mutate(eye_ex = as_factor(eye_exam),
         eye_ex =
           fct_recode(eye_ex, "Yes" = "1", "No" = "0"))

dm431 |> count(eye_exam, eye_ex)
```

```
# A tibble: 2 × 3
  eye_exam eye_ex     n
     <dbl> <fct>  <int>
1        0 No       169
2        1 Yes      262
```

Income by Eye Exam, version 1

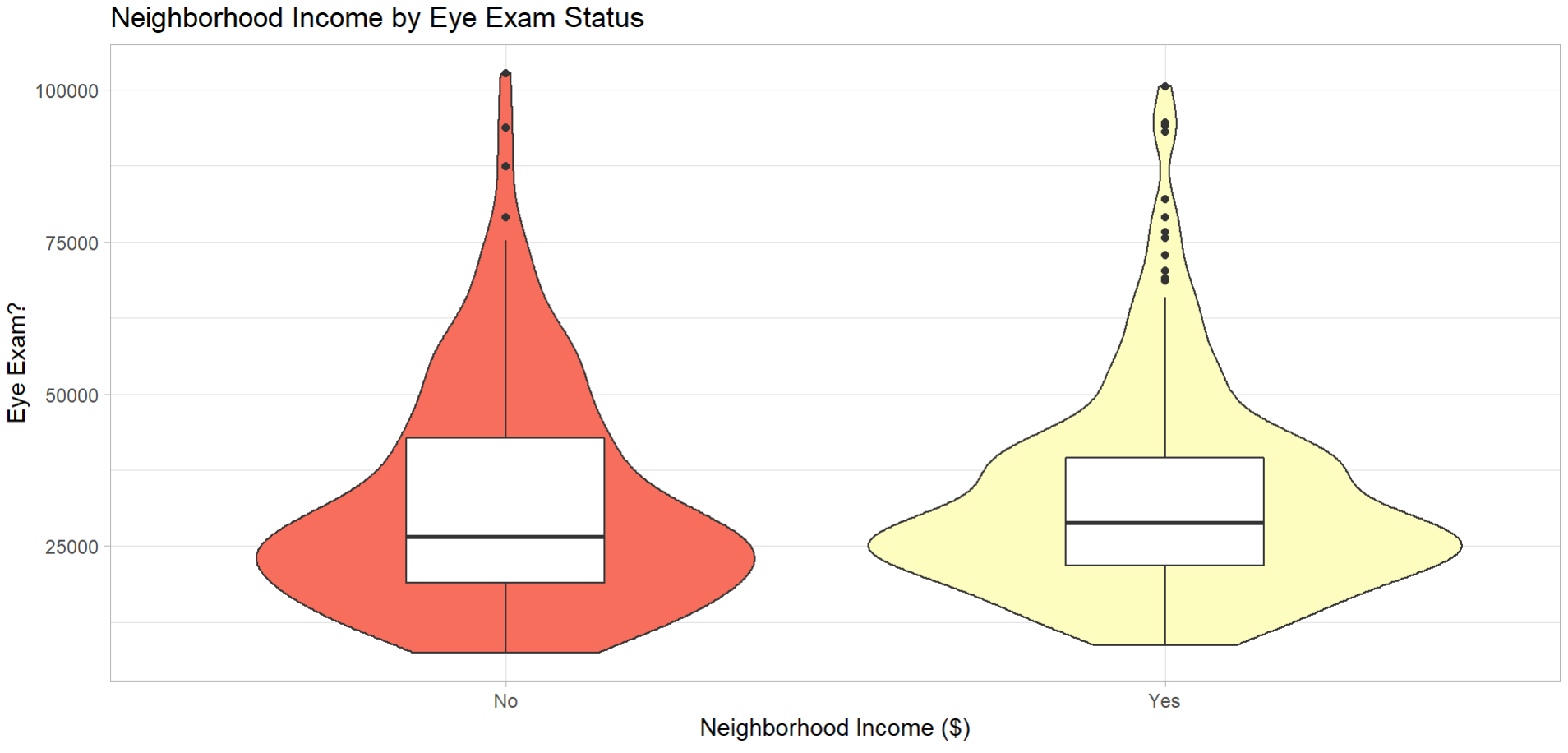
```
1 ggplot(data = dm431, aes(x = eye_ex, y = n_income)) +
2   geom_boxplot() +
3   labs(title = "Income by Eye Exam", caption = "dm431 tibble")
```



Income by Eye Exam, 2

```
ggplot(data = dm431, aes(x = eye_ex, y = n_income)) +
  geom_violin(aes(fill = eye_ex)) +
  geom_boxplot(width = 0.3) +
  scale_fill_viridis_d(begin = 0.7, option = "A") +
  guides(fill = "none", col = "none") +
  labs(title = "Neighborhood Income by Eye Exam Status",
       caption = "dm431 tibble",
       y = "Eye Exam?", x = "Neighborhood Income ($)")
```

Income by Eye Exam, 2



Income by Eye Exam and Model

```
1 mosaic::favstats(n_income ~ eye_ex, data = dm431)
```

```
  eye_ex  min       Q1  median       Q3    max     mean       sd   n missing
1     No 7534 19013.00 26521.0 42788.00 102672 32270.93 18310.77 169       0
2    Yes 8641 21785.75 28803.5 39484.75 100437 32670.58 16640.23 262       0
```

What if we fit a regression model here?

```
1 mod2 <- lm(n_income ~ eye_ex, data = dm431)
2
3 tidy(mod2, conf.int = TRUE, conf.level = 0.90)
```

```
# A tibble: 2 × 7
  term        estimate std.error statistic  p.value conf.low conf.high
  <chr>          <dbl>     <dbl>     <dbl>    <dbl>    <dbl>     <dbl>
1 (Intercept)   32271.     1332.    24.2   2.37e-82   30076.    34466.
2 eye_exYes       400.     1708.     0.234 8.15e- 1   -2416.     3215.
```

Income by Eye Exam t test?

```
1 t.test(n_income ~ eye_ex, data = dm431, var.eq = TRUE, conf.level = 0.90)
```

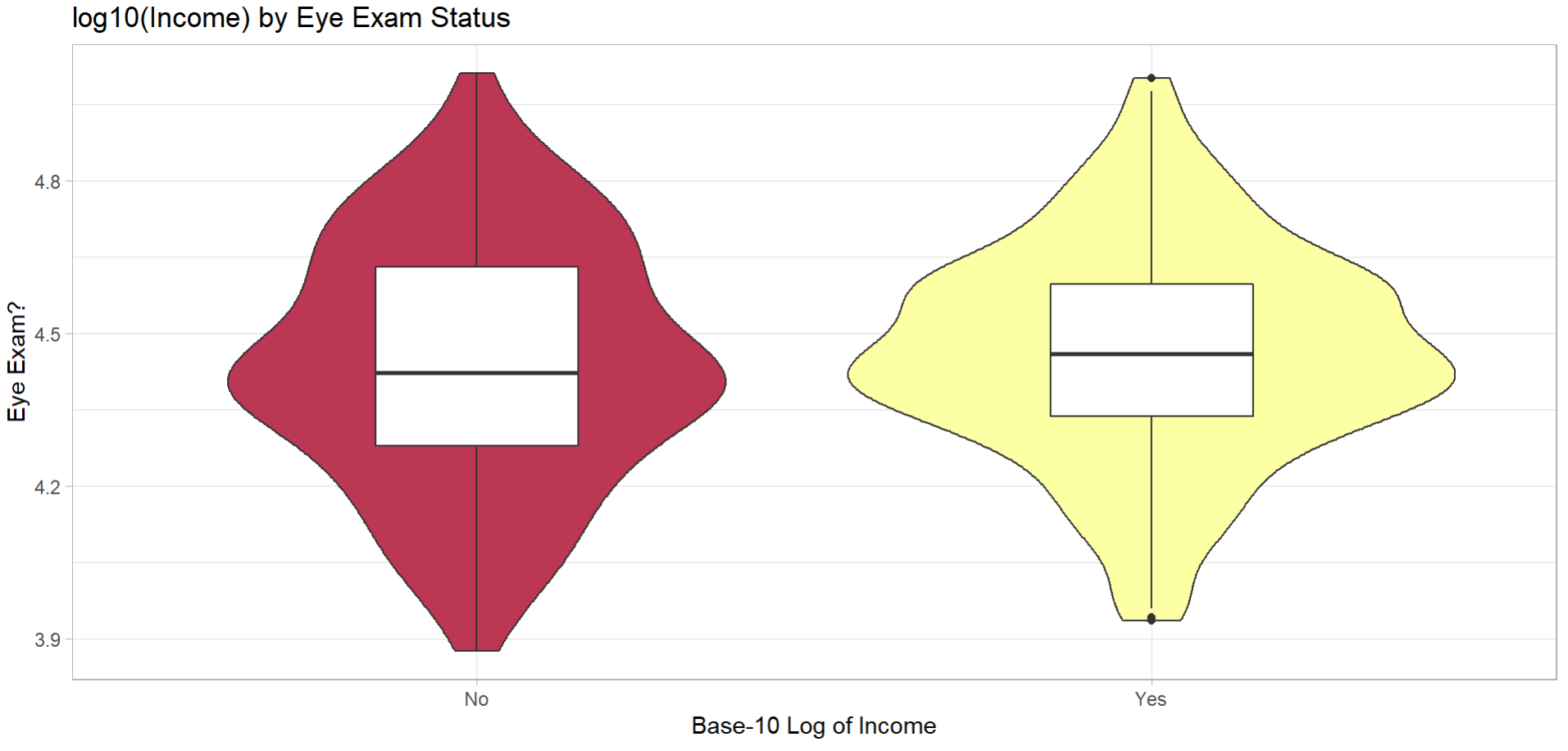
```
    Two Sample t-test

data:  n_income by eye_ex
t = -0.23396, df = 429, p-value = 0.8151
alternative hypothesis: true difference in means between group No and group
Yes is not equal to 0
90 percent confidence interval:
 -3215.431  2416.133
sample estimates:
 mean in group No mean in group Yes
        32270.93          32670.58
```

Does a t test make any sense given non-Normal distribution of `n_income`?

log10(Income) by Eye Exam

```
ggplot(data = dm431, aes(x = eye_ex, y = log10(n_income))) +
  geom_violin(aes(fill = eye_ex)) +
  geom_boxplot(width = 0.3) +
  scale_fill_viridis_d(begin = 0.5, option = "B") +
  guides(fill = "none", col = "none") +
  labs(title = "log10(Income) by Eye Exam Status",
       caption = "dm431 tibble",
       y = "Eye Exam?", x = "Base-10 Log of Income")
```

log10(Income) by Eye Exam



log10(Income) by Eye Exam

```
1 mosaic::favstats(log10(n_income) ~ eye_ex, data = dm431)
```

```
  eye_ex      min       Q1   median       Q3      max     mean         sd   n
1     No 3.877026 4.279051 4.423590 4.631322 5.011452 4.442798 0.2426069 169
2    Yes 3.936564 4.338172 4.459439 4.596428 5.001894 4.462694 0.2130671 262
  missing
1       0
2       0
```

What if we fit a regression model here?

```
1 mod3 <- lm(log10(n_income) ~ eye_ex, data = dm431)
2
3 tidy(mod3, conf.int = TRUE, conf.level = 0.90)
```

```
# A tibble: 2 × 7
  term        estimate std.error statistic p.value conf.low conf.high
  <chr>          <dbl>     <dbl>     <dbl>   <dbl>    <dbl>     <dbl>
1 (Intercept)   4.44      0.0173   257.       0        4.41      4.47
2 eye_exYes     0.0199    0.0222     0.896    0.371   -0.0167    0.0565
```

log10(Income) by Eye Exam t test?

```
t.test(log10(n_income) ~ eye_ex, data = dm431,
       var.eq = TRUE, conf.level = 0.90)
```

```
    Two Sample t-test

data:  log10(n_income) by eye_ex
t = -0.89589, df = 429, p-value = 0.3708
alternative hypothesis: true difference in means between group No and group
Yes is not equal to 0
90 percent confidence interval:
 -0.05650466  0.01671222
sample estimates:
 mean in group No mean in group Yes
         4.442798           4.462694
```

Is using this t test on log10(income) a sensible choice?

Session Information

```
1 sessionInfo()
```

```
R version 4.2.1 (2022-06-23 ucrt)
Platform: x86_64-w64-mingw32/x64 (64-bit)
Running under: Windows 10 x64 (build 22000)

Matrix products: default

locale:
[1] LC_COLLATE=English_United States.utf8
[2] LC_CTYPE=English_United States.utf8
[3] LC_MONETARY=English_United States.utf8
[4] LC_NUMERIC=C
[5] LC_TIME=English_United States.utf8

attached base packages:
[1] stats     graphics  grDevices utils     datasets  methods   base
```