

431 Class 07

`thomaseLove.github.io/431`

2021-09-14

Package Setup

```
library(janitor)
library(knitr)
library(magrittr)
library(modelsummary) # new today
library(naniar)
library(patchwork)
library(tidyverse)

theme_set(theme_bw())
```

Loading Some New Data

```
dm1000 <- read_csv("data/dm_1000.csv",  
                  show_col_types = FALSE) %>%  
  clean_names() %>%  
  mutate(across(where(is.character), as_factor)) %>%  
  mutate(subject = as.character(subject))
```

- 1000 (simulated) patients with diabetes between the ages of 31 and 75 who live in Cuyahoga County and are in one of four race-ethnicity categories, as well as one of four insurance categories.
- Same variables we saw in dm431 last week, but 1000 new subjects, and one new variable (residence)

Variable	Description
subject	subject code (M-0001 through M-1000)
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)
a1c	most recent Hemoglobin A1c (% , with one decimal)
ldl	most recent LDL cholesterol level (mg/dl)

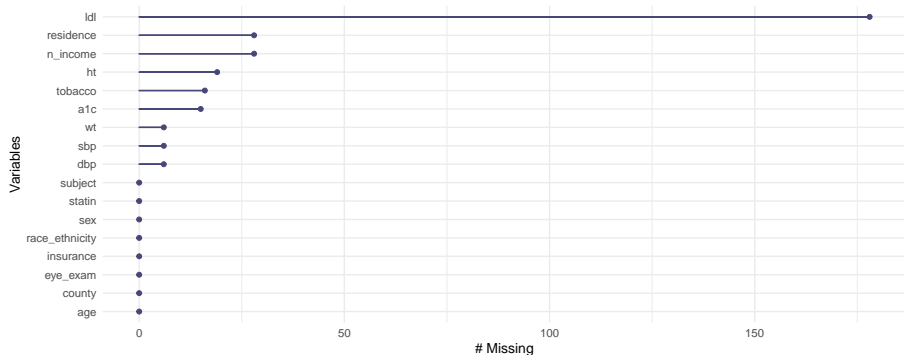
with seven more variables shown on the next slide. . .

Remainder of dm1000 codebook

Variable	Description
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	Female or Male
county	all subjects turn out to be in Cuyahoga County
residence	Cleveland or Suburbs

Any Missing Data?

```
gg_miss_var(dm1000)
```



- For now, `gg_miss_var()` throws a warning, which I've suppressed.

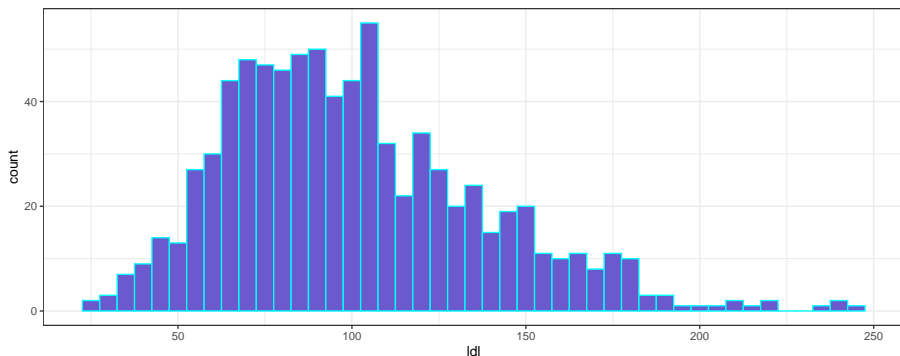
How should we summarize data with missing values?

- If you are providing a data summary, then you should summarize the complete cases, and specify the number of missing values.
- If you are intending to use the sample you've collected to make an inference about a process or population or to build a model, then you may want to consider whether or not a complete-case analysis will introduce bias.

What do graphs do with missing data?

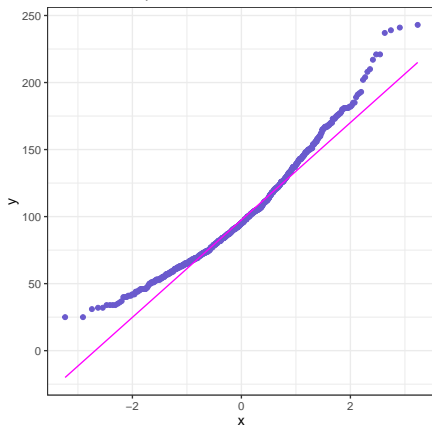
```
ggplot(data = dm1000, aes(x = ldl)) +  
  geom_histogram(binwidth = 5,  
                 fill = "slateblue", col = "cyan")
```

Warning: Removed 178 rows containing non-finite values (stat_bin).

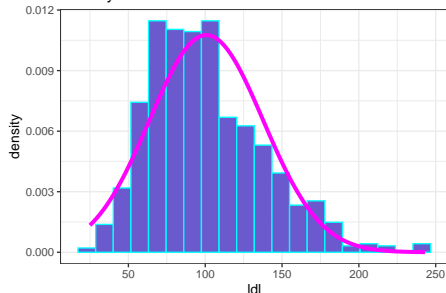


Exploring ldl in dm1000 (warnings suppressed)

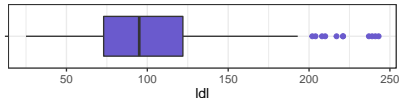
Normal Q-Q plot: dm1000 LDL



Density Function: dm1000 LDL



Boxplot: dm1000 LDL



min	Q1	median	Q3	max	mean	sd	n	missing
25	73	95	122	243	100.7	37.1	821	178

Numerical Summaries with missing values...

```
summary(dm1000$a1c)
```

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
4.100	6.400	7.200	7.854	8.900	16.900
NA's					
15					

```
mosaic::favstats(~ sbp, data = dm1000)
```

min	Q1	median	Q3	max	mean	sd	n	missing
84	122	132	142	209	132.7825	17.95949	993	6





Summarizing dm1000 with tools from modelsummary

Note that `modelsummary` really works well for HTML output but not for these PDF slides, so I am using images from an HTML knitting of the code below to get the results you're seeing.

```
dm1000 %>% select(age, ht, sbp, a1c) %>%  
  datasummary_skim(.)
```

- I've used `{r, eval = FALSE}` in the code chunk header to get R to print the code but not evaluate it in this slide.
- Results on next slide

```
dm1000 %>% select(age, ht, sbp, a1c) %$$
datasummary_skim(.)
```

	Unique (#)	Missing (%)	Mean	SD	Min	Median	Max	
age	45	0	57.2	10.1	31.0	58.0	75.0	
ht	56	2	1.7	0.1	1.4	1.7	2.0	
sbp	100	1	132.8	18.0	84.0	132.0	209.0	
a1c	99	2	7.9	2.0	4.1	7.2	16.9	

Summarizing age within insurance type

```
mosaic::favstats(age ~ insurance, data = dm1000) %>%  
  kable(digits = 0)
```

insurance	min	Q1	median	Q3	max	mean	sd	n	missing
Medicaid	31	46	52	58	64	51	8	329	0
Commercial	33	50	56	60	72	55	8	196	0
Medicare	32	58	66	70	75	63	9	432	0
Uninsured	33	51	56	59	64	54	8	42	0

datasummary: ages within insurance types

```
datasummary(insurance * age ~  
             (N + mean + sd + min + P50 + max),  
             data = dm1000, histogram = FALSE)
```

- Results on next slide

```
datasummary(insurance * age ~
  (N + mean + sd + min + P50 + max),
  data = dm1000)
```

insurance		N	mean	sd	min	P50	max
Medicaid	age	329	51.19	8.39	31.00	52.00	64.00
Commercial	age	196	55.05	7.62	33.00	56.00	72.00
Medicare	age	432	63.05	9.28	32.00	66.00	75.00
Uninsured	age	42	54.10	7.69	33.00	56.50	64.00

Mean of sbp

```
mean(dm1000$sbp)
```

```
[1] NA
```

```
mean(dm1000$sbp, na.rm = TRUE)
```

```
[1] 132.7825
```


Summarizing sbp in dm1000

```
mosaic::favstats(~ sbp, data = dm1000) %>%  
  kable(digits = 1)
```

min	Q1	median	Q3	max	mean	sd	n	missing
84	122	132	142	209	132.8	18	993	6

```
Hmisc::describe(dm1000$sbp)
```

dm1000\$sbp

n	missing	distinct	Info	Mean	Gmd
993	6	99	1	132.8	19.87
.05	.10	.25	.50	.75	.90
105.0	110.0	122.0	132.0	142.0	154.8
.95					
165.0					

Summarizing age and sbp with datasummary

```
datasummary((age + sbp) ~  
             (mean + sd + min + P50 + max),  
             data = dm1000)
```

- Results on next slide

```
datasummary((age + sbp) ~  
             (mean + sd + min + P50 + max),  
             data = dm1000)
```

	mean	sd	min	P50	max
age	57.20	10.11	31.00	58.00	75.00
sbp				132.00	

Applying `na.rm = TRUE` to multiple summaries

```
datasummary((age + sbp + ldl + ht) ~  
             (mean + sd + min + P50 + max) *  
             Arguments(na.rm = TRUE),  
             data = dm1000)
```

- Results on next slide

```
datasummary((age + sbp + ldl + ht) ~  
             (mean + sd + min + P50 + max) *  
             Arguments(na.rm = TRUE),  
             data = dm1000)
```











	mean	sd	min	P50	max
age	57.20	10.11	31.00	58.00	75.00
sbp	132.78	17.96	84.00	132.00	209.00
ldl	100.69	37.07	25.00	95.00	243.00
ht	1.69	0.10	1.36	1.69	1.97

datasummary_skim on the whole tibble

```
datasummary_skim(dm1000)
```

- Again, results on next slide
- Which variables are NOT included?

```
datasummary_skim(dm1000)
```

	Unique (#)	Missing (%)	Mean	SD	Min	Median	Max	
age	45	0	57.2	10.1	31.0	58.0	75.0	
n_income	964	3	35187.3	18783.3	2279.0	30594.0	129549.0	
ht	56	2	1.7	0.1	1.4	1.7	2.0	
wt	885	1	95.0	25.9	26.0	92.4	218.2	
sbp	100	1	132.8	18.0	84.0	132.0	209.0	
dbp	72	1	74.5	12.4	41.0	75.0	137.0	
a1c	99	2	7.9	2.0	4.1	7.2	16.9	
ldl	159	18	100.7	37.1	25.0	95.0	243.0	
statin	2	0	0.8	0.4	0.0	1.0	1.0	
eye_exam	2	0	0.6	0.5	0.0	1.0	1.0	

Using a Sample to Estimate a Population Mean

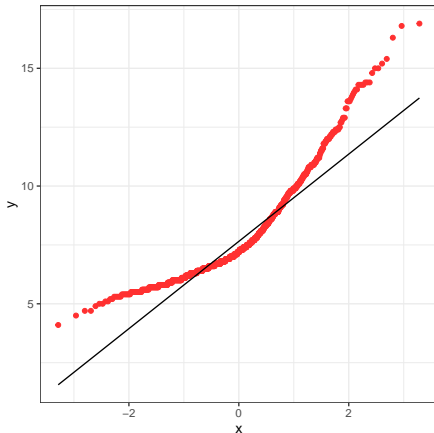
Suppose our sample in `dm1000` is a random sample from the population of all Cuyahoga County residents between the ages of 31-75 receiving care for diabetes.

What's a good estimate for the mean Hemoglobin A1c of the people in that population?

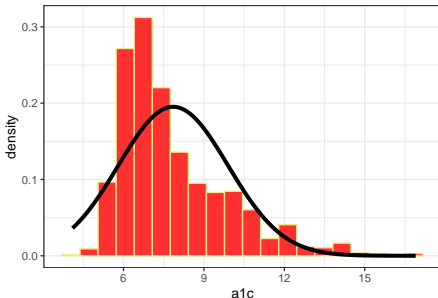
- How would we make this estimate using our data?
- What would we want to know about the data?
- DTDP

Hemoglobin A1c in the dm1000 sample

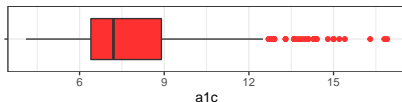
Normal Q-Q plot: dm1000 a1c



Density Function: dm1000 a1c



Boxplot: dm1000 a1c



min	Q1	median	Q3	max	mean	sd	n	missing
4.1	6.4	7.2	8.9	16.9	7.85	2.04	984	15

What if we assume the A1cs are Normally distributed?

Then we could use a linear regression model to obtain a 95% confidence interval for the population mean.

```
m1 <- lm(a1c ~ 1, data = dm1000)
tidy(m1, conf.int = TRUE, conf.level = 0.95) %>%
  select(estimate, conf.low, conf.high) %>%
  kable(digits = 3)
```

estimate	conf.low	conf.high
7.854	7.726	7.982

What is the model we've fit here?

```
m1
```

Call:

```
lm(formula = a1c ~ 1, data = dm1000)
```

Coefficients:

```
(Intercept)  
      7.854
```

- This “intercept only” model simply predicts the mean value of our outcome, a1c.

How do we interpret our 95% confidence interval?

We are estimating the mean of the **population** based on a mean from our *sample* of 1000 observations taken (at random, we assume) from that population.

- Our 95% confidence interval is (7.73, 7.98).
 - Sadly, this **doesn't** mean we're 95% confident that the actual population mean is in that range, even though lots and lots of people (incorrectly) assume it does.
- Essentially, we have 95% confidence in the **process** of fitting confidence intervals this way. If we fit 100 such confidence intervals to a variety of data sets, we have some reason to anticipate that 95 of them will contain the actual unknown value of the population mean.
 - That's oversimplifying a little, and a particular concern in this case is that the data are somewhat skewed (at any rate, not very close to Normally distributed) and this may impact our ability to generate accurate and efficient confidence intervals via a linear model like this.

An Equivalent Approach

We could use a t test to obtain a 95% confidence interval for the population mean.

```
tt <- dm1000 %$% t.test(a1c, conf.level = 0.95)
tidy(tt) %>% select(estimate, conf.low, conf.high) %>%
  kable(digits = 3)
```

estimate	conf.low	conf.high
7.854	7.726	7.982

This is exactly the same result as we obtain from the linear model `m1`.

Another Equivalent Approach

We could use a function called `smean.cl.normal()` from the `Hmisc` package to obtain the same 95% confidence interval for the population mean.

```
dm1000 %$% Hmisc::smean.cl.normal(a1c, conf.int = 0.95)
```

Mean	Lower	Upper
7.853963	7.726125	7.981802

Again, this is the same result as we have seen previously.

What if we weren't willing to assume that the A1c values came from a Normal distribution?

- My first instinct would be to use a bootstrap confidence interval to estimate the population mean with a 95% confidence interval.

```
set.seed(20210914)  # why do we set a seed here?  
dm1000 %$% Hmisc::smean.cl.boot(a1c, conf.int = 0.95)
```

Mean	Lower	Upper
7.853963	7.733633	7.987215

```
set.seed(431)  # what happens if we change the seed  
dm1000 %$% Hmisc::smean.cl.boot(a1c, conf.int = 0.95)
```

Mean	Lower	Upper
7.853963	7.734726	7.981847

95% Confidence Intervals for Population Mean A1c

Approach	Estimate	95% CI	Assume Normality?
Linear Model (t test)	7.85	(7.73, 7.98)	Yes
Bootstrap	7.85	(7.73, 7.99)	No

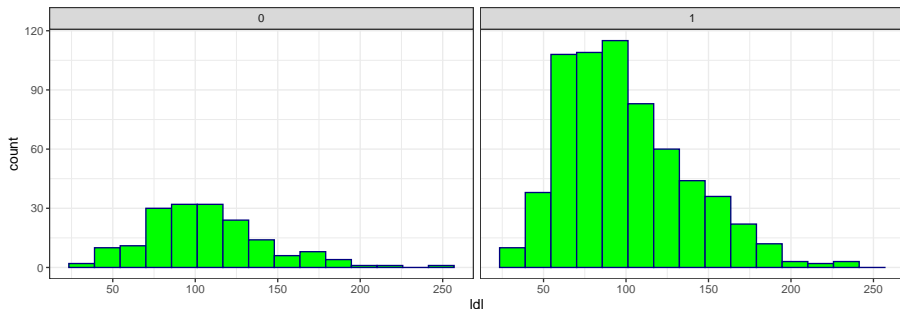
- How does this match up with our understanding of the distribution of the A1c data?
- Might the fairly large sample size ($n = 984$ non-missing) have something to do with these results?
- More on using regression models (and the bootstrap) to compare summaries is coming in Classes 13-15.

Comparing Two Distributions

Is LDL higher or lower among adults with diabetes who have a statin prescription?

```
ggplot(data = dm1000, aes(x = ldl)) +  
  geom_histogram(bins = 15, fill = "green", col = "navy") +  
  facet_wrap(~ statin)
```

Warning: Removed 178 rows containing non-finite values (stat_bin).

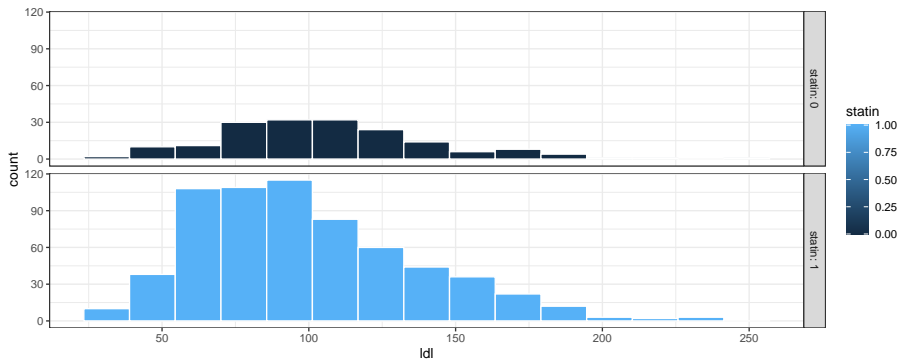


How might we improve this plot?

- ➊ Remove the warning about non-finite (missing) values.
- ➋ Place the histograms vertically to ease comparisons.
- ➌ Fill the histograms differently for statin and no statin.
- ➍ Augment the labels (0 and 1) to show they identify statin use.

LDL stratified by statin status (First Try)

```
dm1000 %>% filter(complete.cases(ldl, statin)) %>%  
  ggplot(data = ., aes(x = ldl, fill = statin)) +  
  geom_histogram(bins = 15, col = "white") +  
  facet_grid(statin ~ ., labeller = "label_both")
```

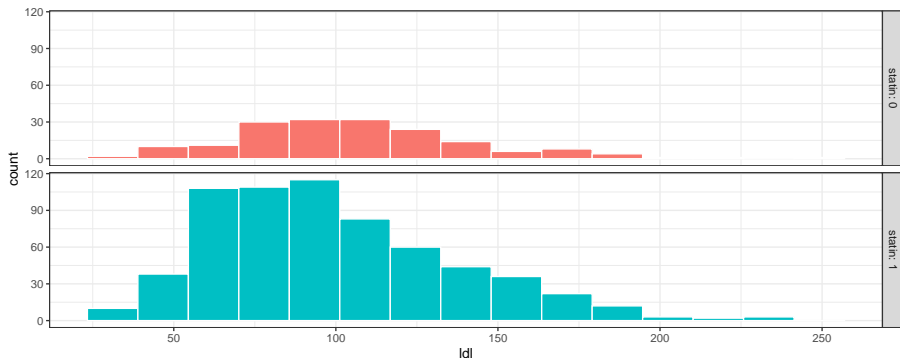


Problems with the previous plot

- 1 Statin is actually a two-category variable (1 and 0 are just codes) but the legend is treating it as if it was a numeric variable.
- 2 Do we actually need the legend (called a guide in R) or can we remove it?

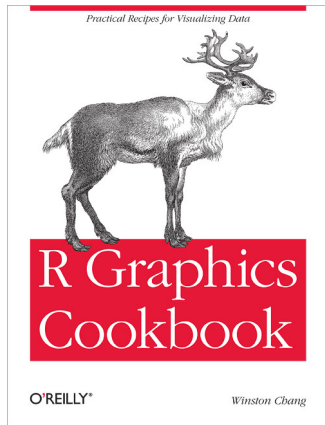
But statin is categorical? (Second Try)

```
dm1000 %>% filter(complete.cases(ldl, statin)) %>%  
  ggplot(data = ., aes(x = ldl, fill = factor(statin))) +  
  geom_histogram(bins = 15, col = "white") +  
  facet_grid(statin ~ ., labeller = "label_both") +  
  guides(fill = "none")
```



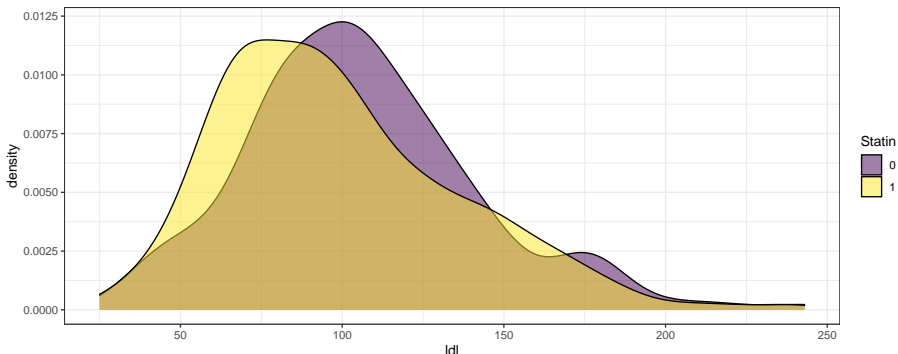
My Main Source for ggplot2 Visualization Recipes

<https://r-graphics.org/>



Comparison of densities (ignores relative frequency)

```
dm1000 %>% filter(complete.cases(ldl, statin)) %>%  
  ggplot(data = ., aes(x = ldl, fill = factor(statin))) +  
  geom_density(alpha = 0.5) +  
  scale_fill_viridis_d() +  
  labs(fill = "Statin")
```



Numerical Summaries comparing Two Groups

```
dm1000 %>% filter(complete.cases(statin, ldl)) %>%  
  group_by(statin) %>%  
  summarize(n = n(), min = min(ldl), med = median(ldl),  
            max = max(ldl), mean = mean(ldl),  
            sd = sd(ldl)) %>%  
  kable(digits = 2)
```

statin	n	min	med	max	mean	sd
0	176	34	102	243	106.22	36.05
1	645	25	92	241	99.19	37.23

- What is the difference in mean(LDL) between the two samples?

Using favstats to compare LDL by Statin group

```
dm1000 %$%  
  mosaic::favstats(ldl ~ statin)
```

	statin	min	Q1	median	Q3	max	mean	sd	n
1	0	34	82	102	126	243	106.22159	36.04619	176
2	1	25	71	92	122	241	99.18605	37.22776	645
	missing								
1		66							
2		112							

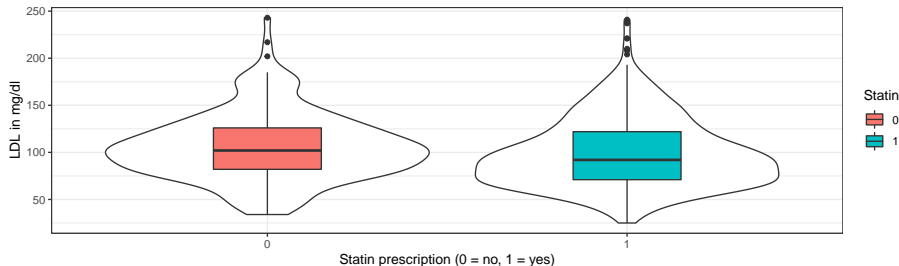
We would have obtained the same result with:

```
mosaic::favstats(ldl ~ statin, data = dm1000)
```

Comparison Boxplot with Violins (LDL and statin)

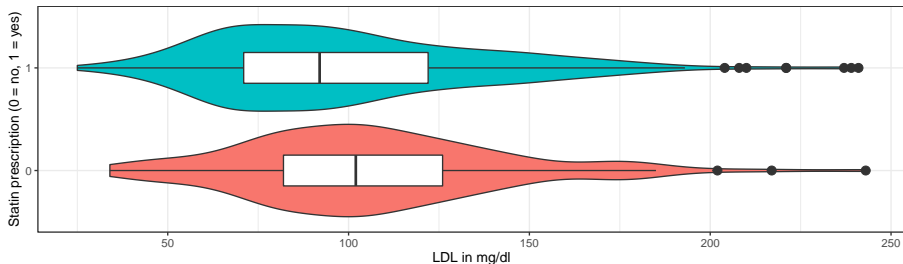
Here's a first attempt...

```
dm1000 %>% filter(complete.cases(ldl, statin)) %>%  
  ggplot(data = ., aes(x = factor(statin), y = ldl)) +  
  geom_violin() +  
  geom_boxplot(aes(fill = factor(statin)), width = 0.3) +  
  labs(x = "Statin prescription (0 = no, 1 = yes)",  
       y = "LDL in mg/dl", fill = "Statin")
```



Try 2: Boxplot with Violins for LDL and statin

```
dm1000 %>% filter(complete.cases(ldl, statin)) %>%  
  ggplot(data = ., aes(x = factor(statin), y = ldl)) +  
  geom_violin(aes(fill = factor(statin))) +  
  geom_boxplot(width = 0.3, outlier.size = 3) +  
  coord_flip() +  
  guides(fill = "none") +  
  labs(x = "Statin prescription (0 = no, 1 = yes)",  
       y = "LDL in mg/dl")
```



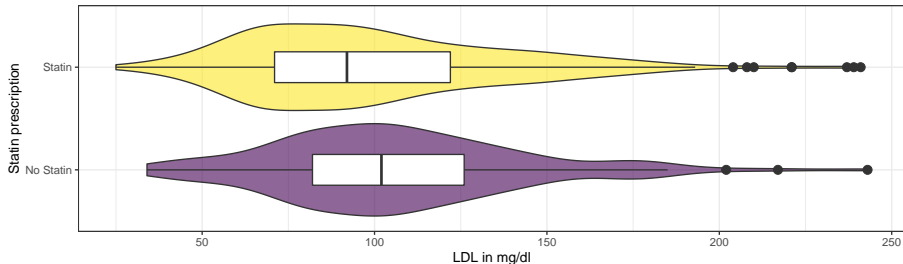
Setting Up Third Try

```
dm_for_boxplot <- dm1000 %>%  
  filter(complete.cases(statin, ldl)) %>%  
  mutate(statin_f = fct_recode(factor(statin),  
                                "No Statin" = "0",  
                                "Statin" = "1")) %>%  
  select(subject, ldl, statin_f, statin)  
  
head(dm_for_boxplot, 3) # print first three rows
```

```
# A tibble: 3 x 4  
  subject    ldl statin_f  statin  
  <chr>    <dbl> <fct>    <dbl>  
1 M-0001    221 Statin      1  
2 M-0002    116 No Statin    0  
3 M-0003     52 Statin      1
```

Third Try on Boxplot for LDL by Statin Use

```
ggplot(data = dm_for_boxplot, aes(x = statin_f, y = ldl)) +  
  geom_violin(aes(fill = statin_f)) +  
  geom_boxplot(width = 0.3, outlier.size = 3) +  
  coord_flip() + guides(fill = "none") +  
  scale_fill_viridis_d(alpha = 0.6) +  
  labs(x = "Statin prescription",  
       y = "LDL in mg/dl")
```



95% confidence interval for difference between population mean LDL WITH statin and population mean LDL WITHOUT statin

If we are willing to assume that LDL follows a Normal distribution in each statin group, then we can use the following linear model with one predictor.

```
m2 <- lm(ldl ~ statin, data = dm1000)
tidy(m2, conf.int = TRUE, conf.level = 0.95) %>%
  select(term, estimate, conf.low, conf.high) %>%
  kable(digits = 3)
```

term	estimate	conf.low	conf.high
(Intercept)	106.222	100.750	111.693
statin	-7.036	-13.208	-0.863

Alternative Approach to get same result

```
tt <- t.test(ldl ~ statin, data = dm1000,  
             var.equal = TRUE, conf.level = 0.95)  
tidy(tt) %>% select(estimate, conf.low, conf.high) %>%  
  kable(digits = 3)
```

estimate	conf.low	conf.high
7.036	0.863	13.208

95% confidence interval for difference between population mean LDL WITH statin and population mean LDL WITHOUT statin

If we are not willing to assume a Normal distribution for LDL in either the statin or the “no statin” group, then we could use a bootstrap approach.

```
source("data/Love-boost.R")
set.seed(20210914)
dm1000 %>% bootdif(y = ldl,
                   g = factor(statin),
                   conf.level = 0.95)
```

Mean Difference	0.025	0.975
-7.035544	-12.887674	-1.287711

95% confidence intervals for $\mu_{NoStatin} - \mu_{Statin}$

Approach	Estimate	95% CI	Normality Assumption?
linear model	7.04	(0.86, 13.21)	Yes
bootstrap	7.04	(1.29, 12.89)	No

Assumptions these intervals share:

- random samples from the populations of interest
- independent samples (samples aren't paired or matched)

Additional assumptions for linear model:

- Normal distribution in each group (statin and “no statin”)
- variance in each group (statin and “no statin”) is equal

Next Time

- Visualizing Comparisons across 3+ batches
- Correlation and Scatterplots