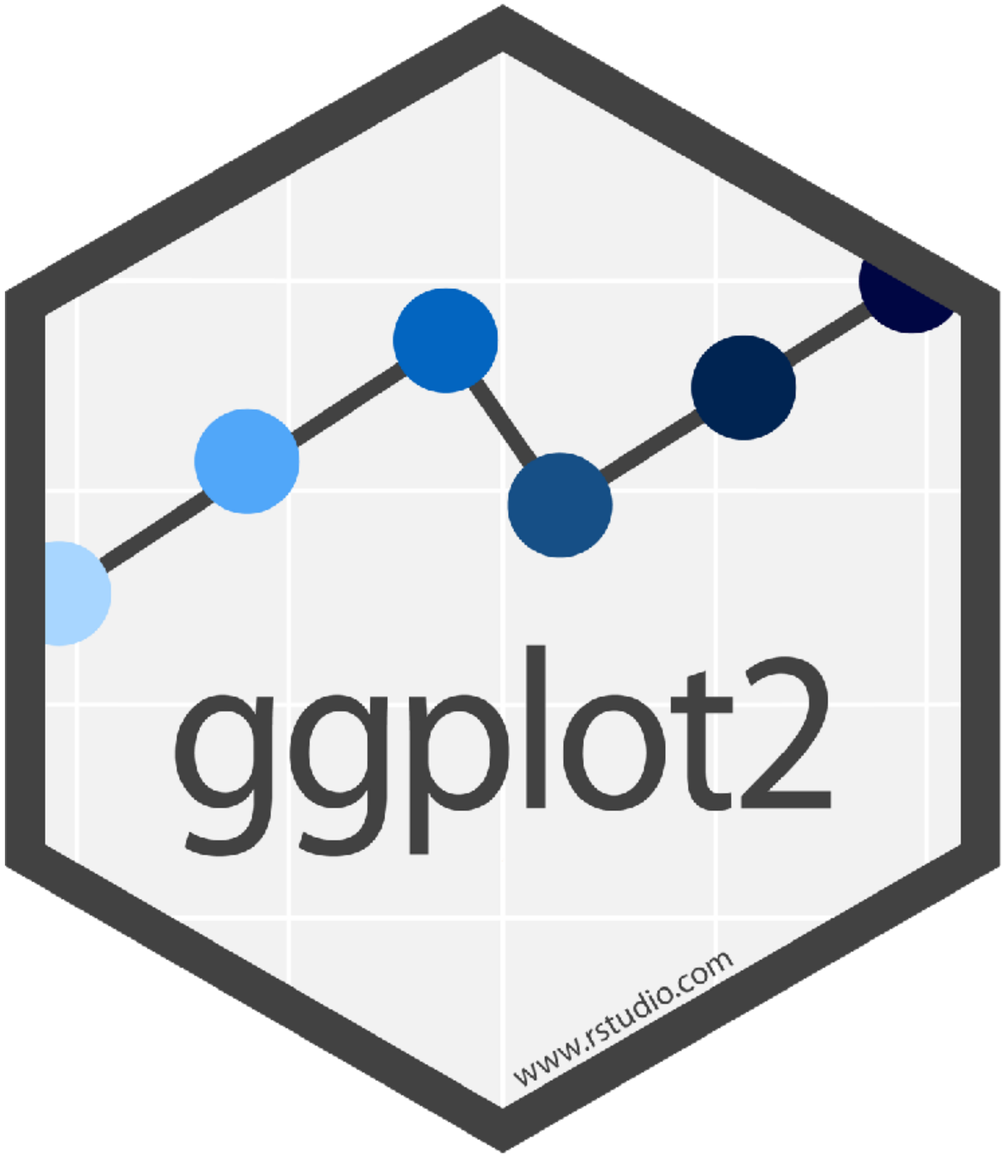
Data Visualization

Jake Thompson
wjakethompson.com
@wjakethompson

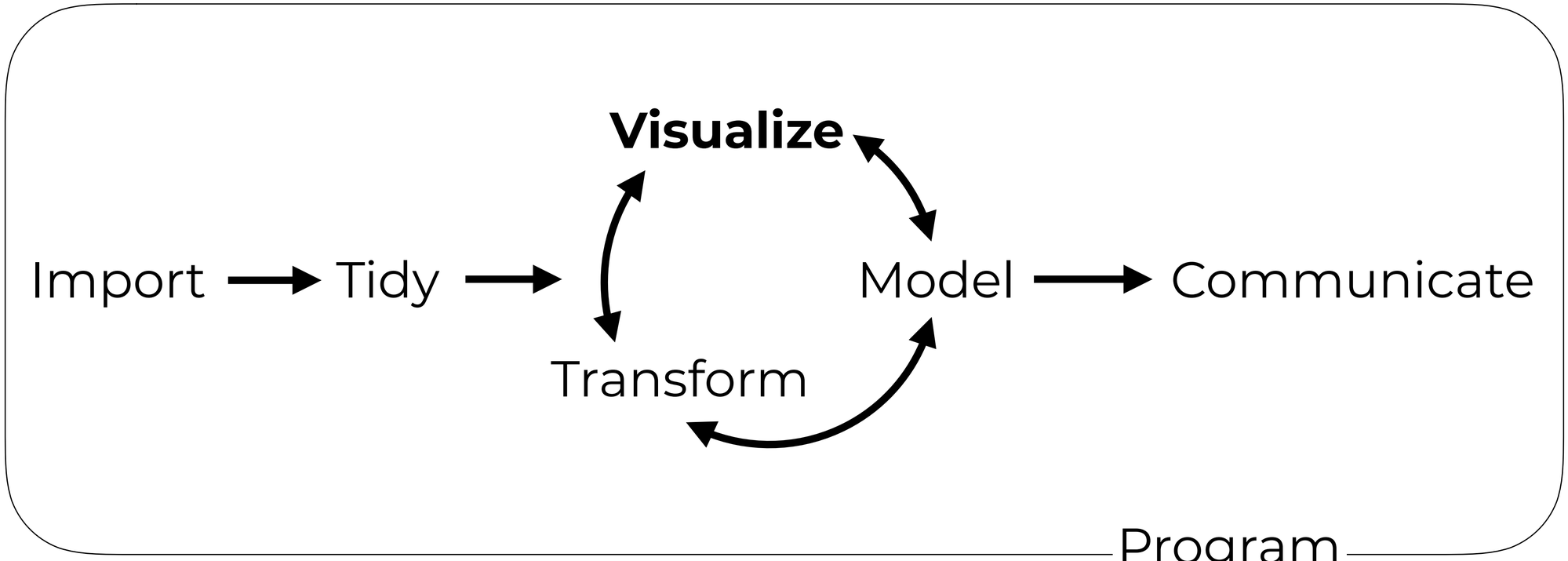
ATLAS
Accessible Teaching, Learning, and Assessment Systems



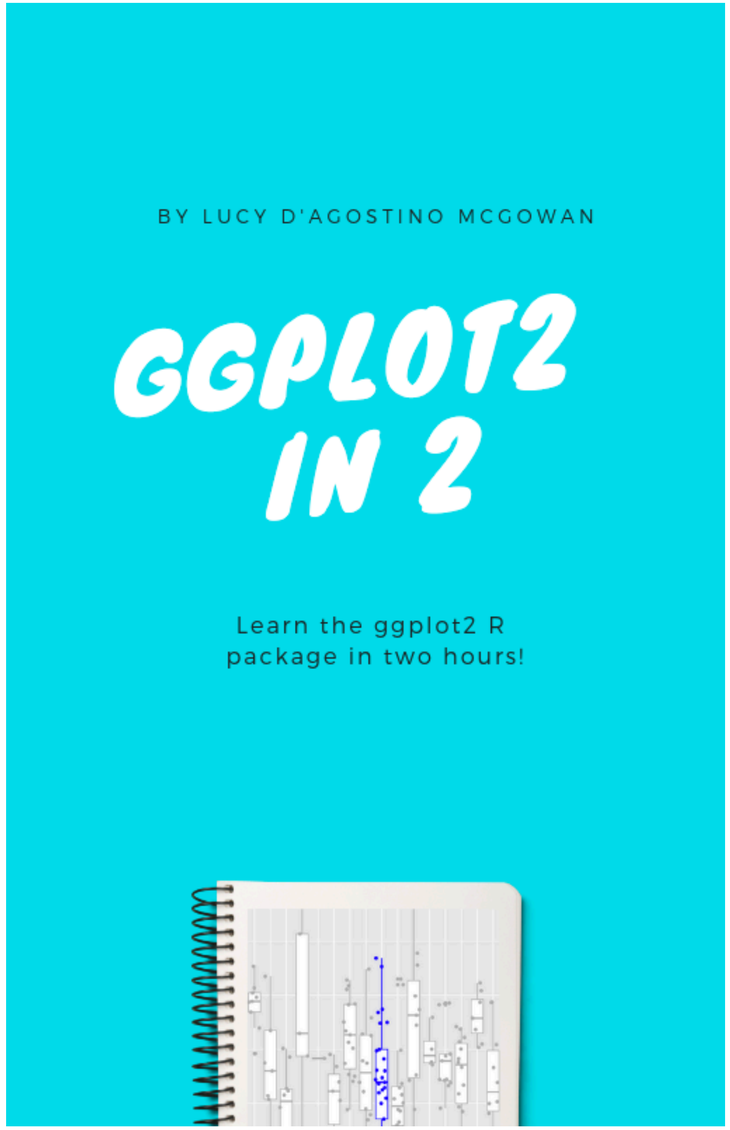
In R4DS

Data Visualisation

ATLAS

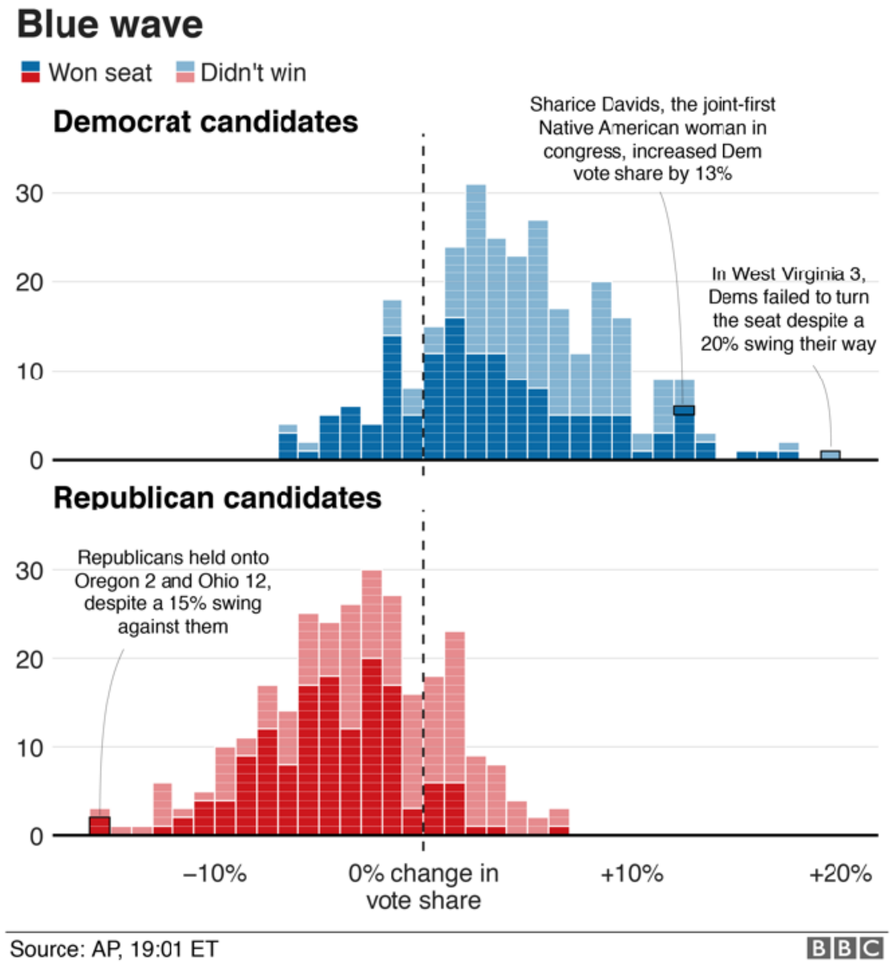


Lucy D'Agostino McGowan
Postdoctoral Researcher
Johns Hopkins University
@LucyStats

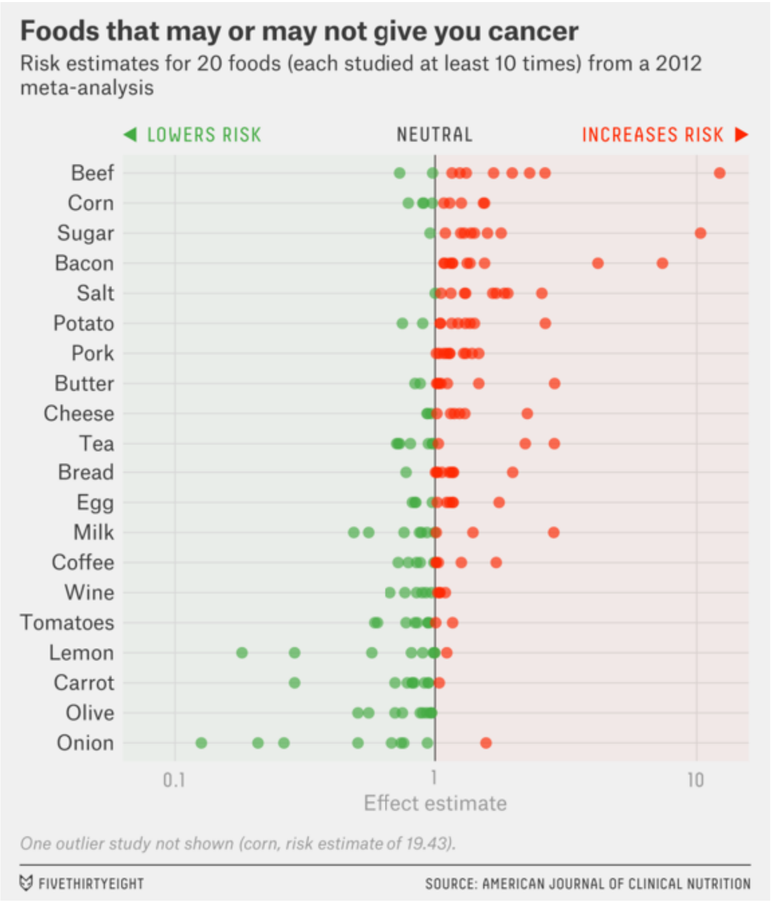


ATLAS

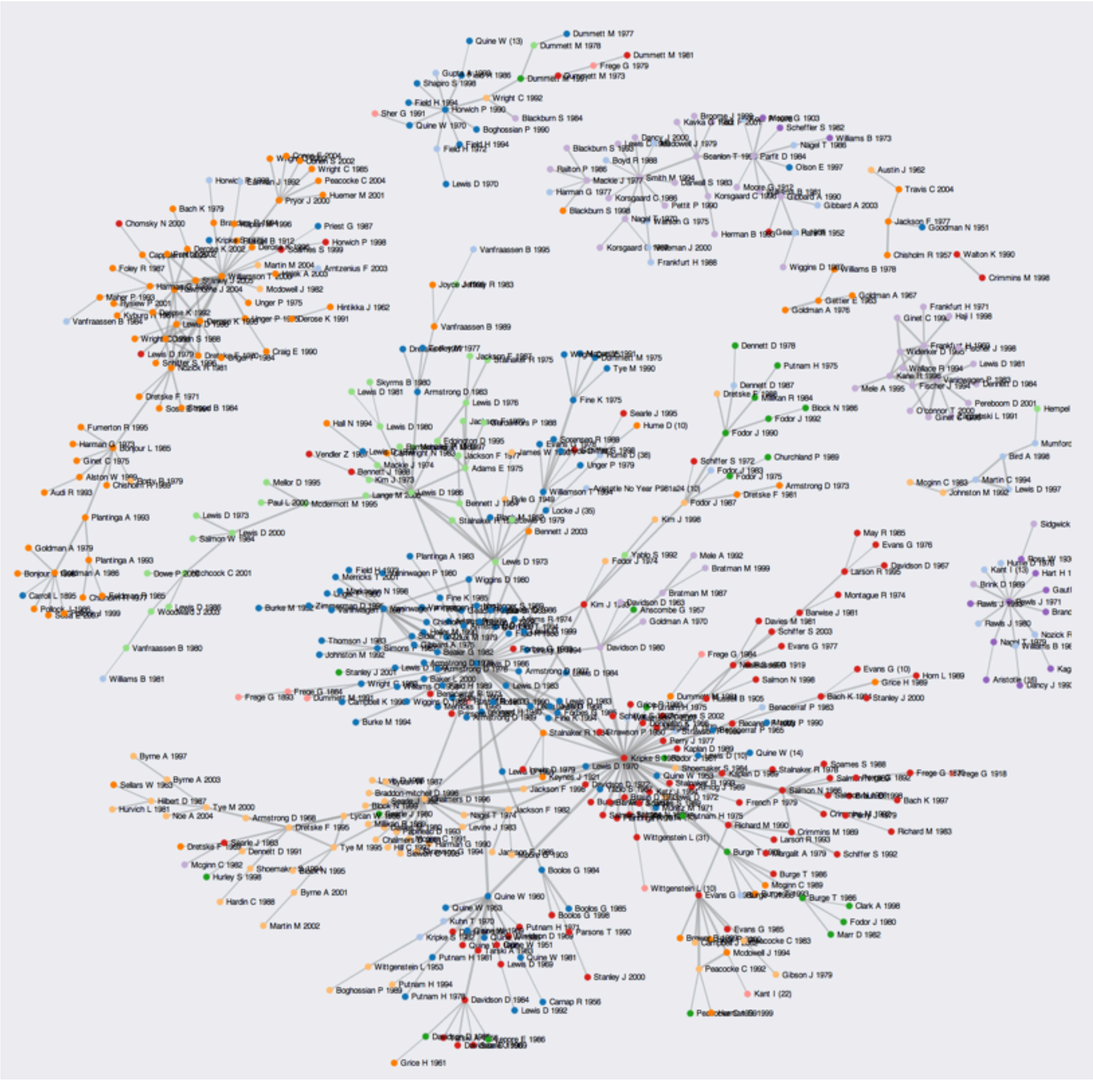
ggplot2



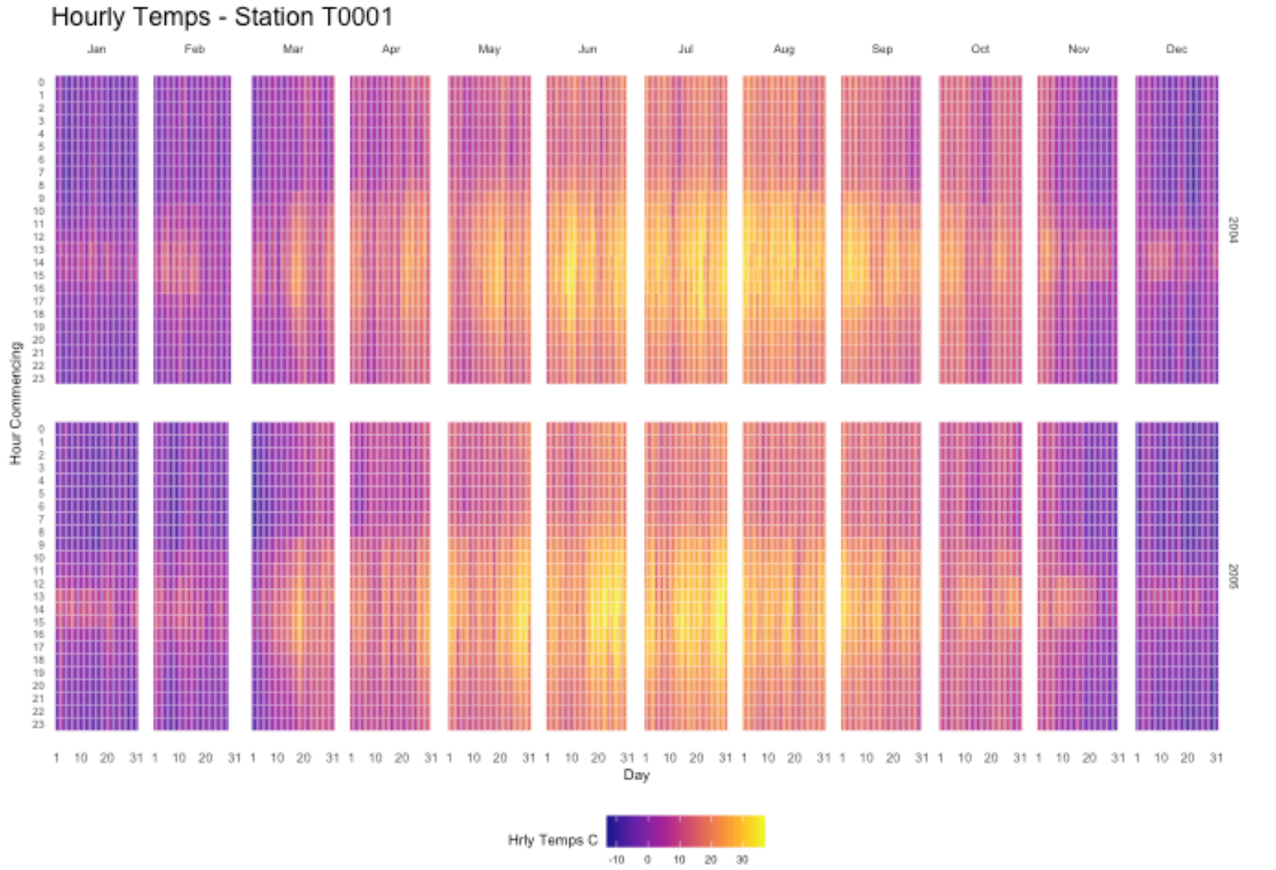
BBC News



FiveThirtyEight

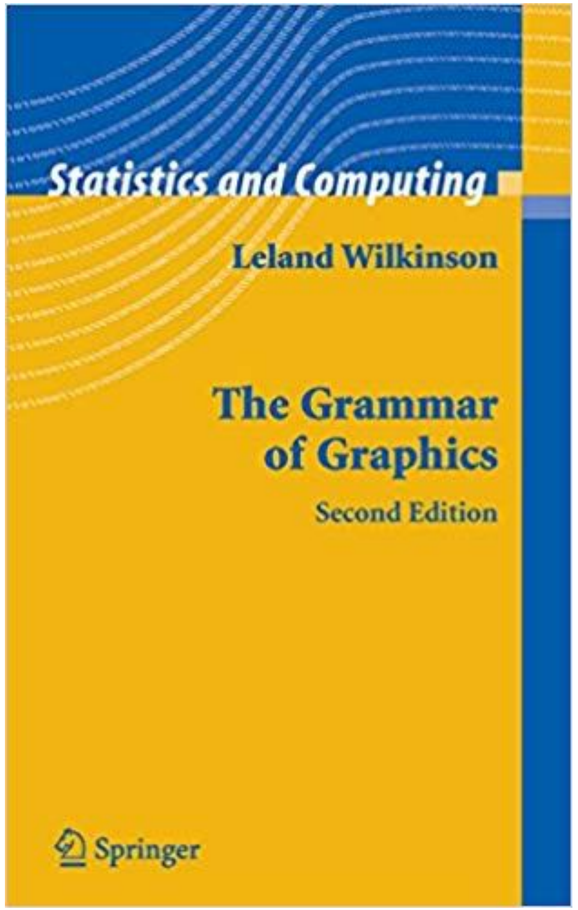


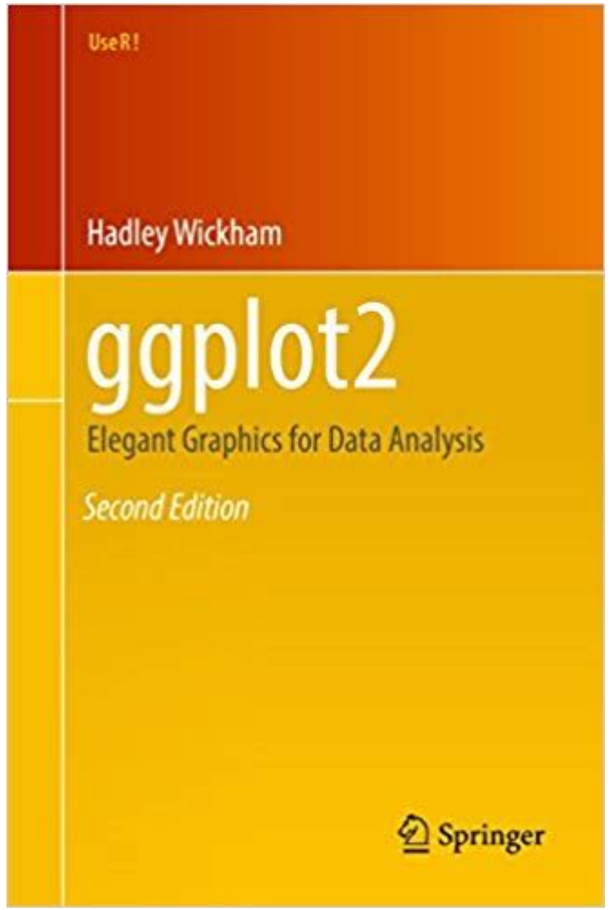
Philosophy Co-Citation Network, Kieran Healy



R Graph Gallery, John MacKintosh

Grammar of Graphics





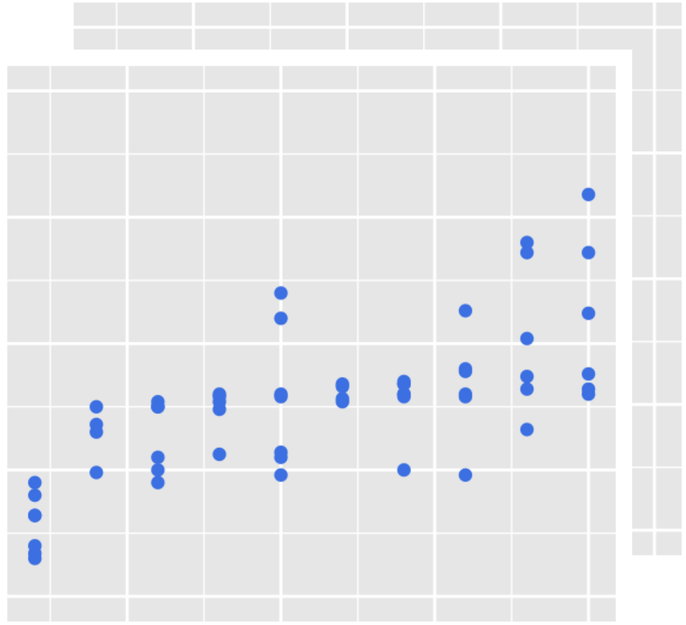
ggplot2

- **data** maps to
- **aes**thetics in
- layers

ggplot2 layers

ggplot2 layers

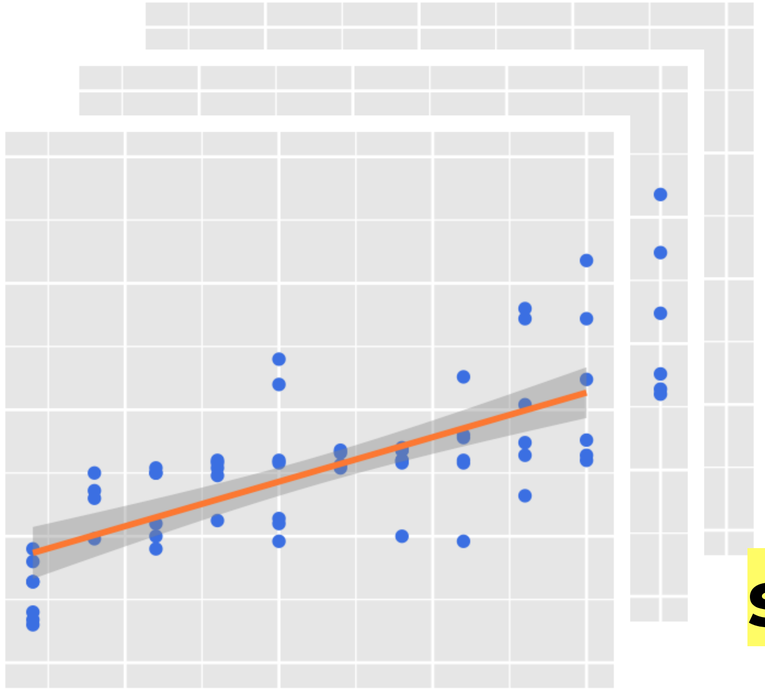
+



geometric objects

ggplot2 layers

+

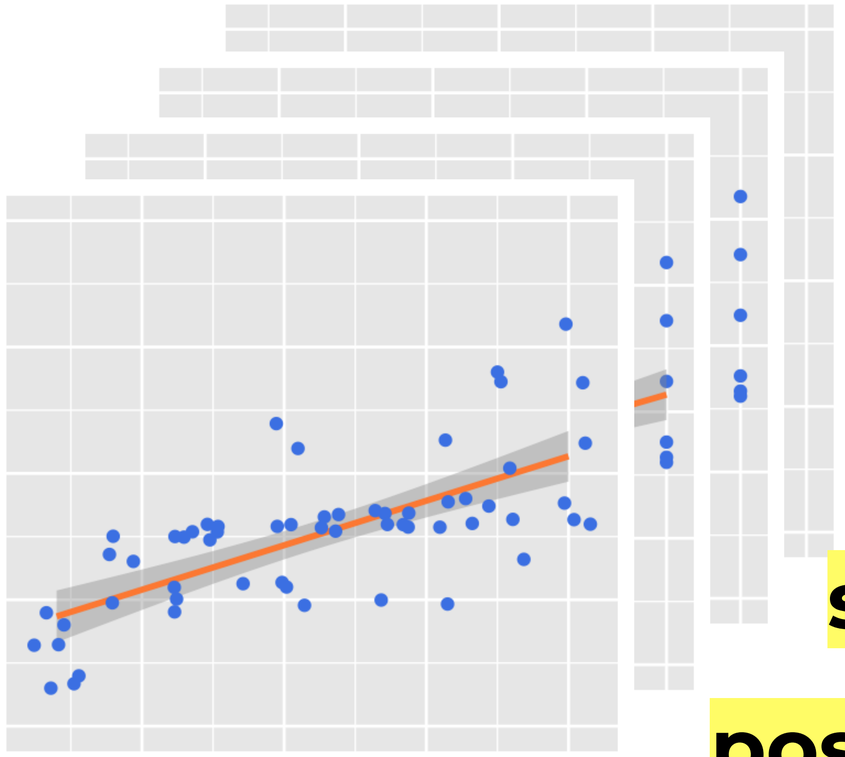


geometric objects

statistical transformations

ggplot2 layers

+

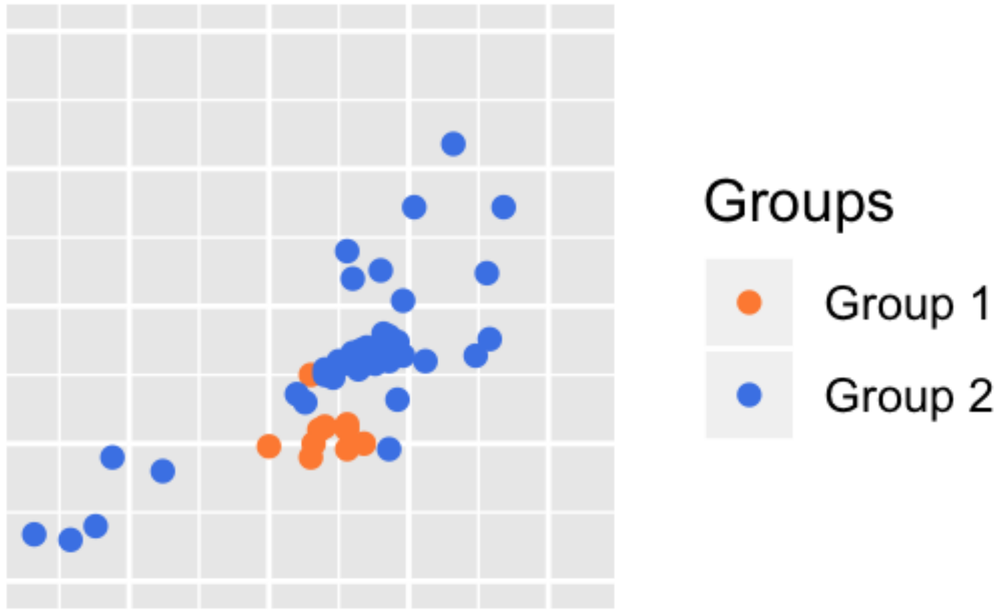


geometric objects

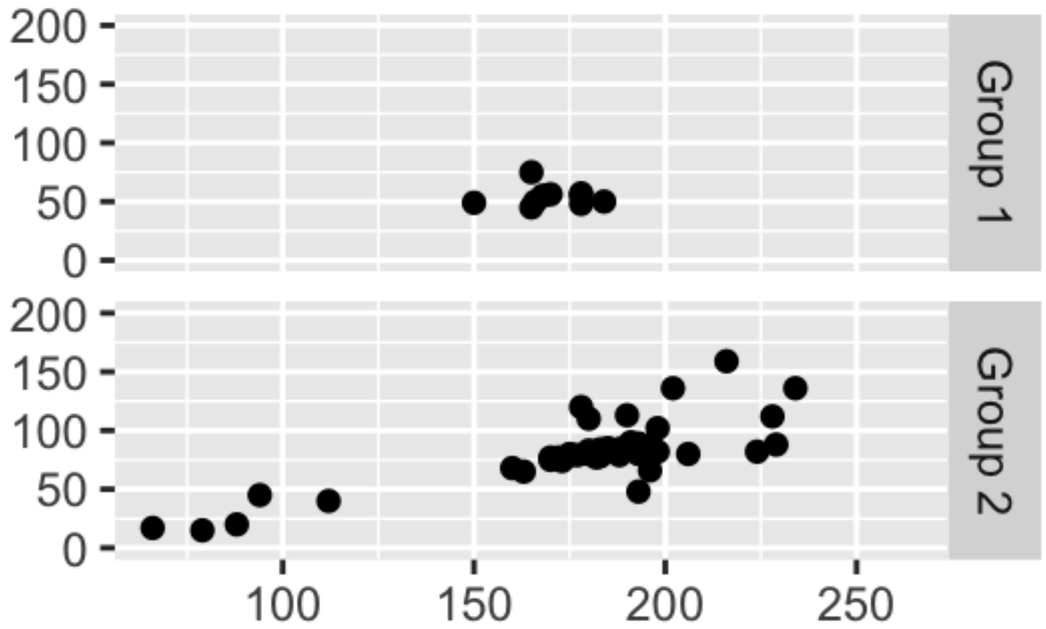
statistical transformations

position adjustments

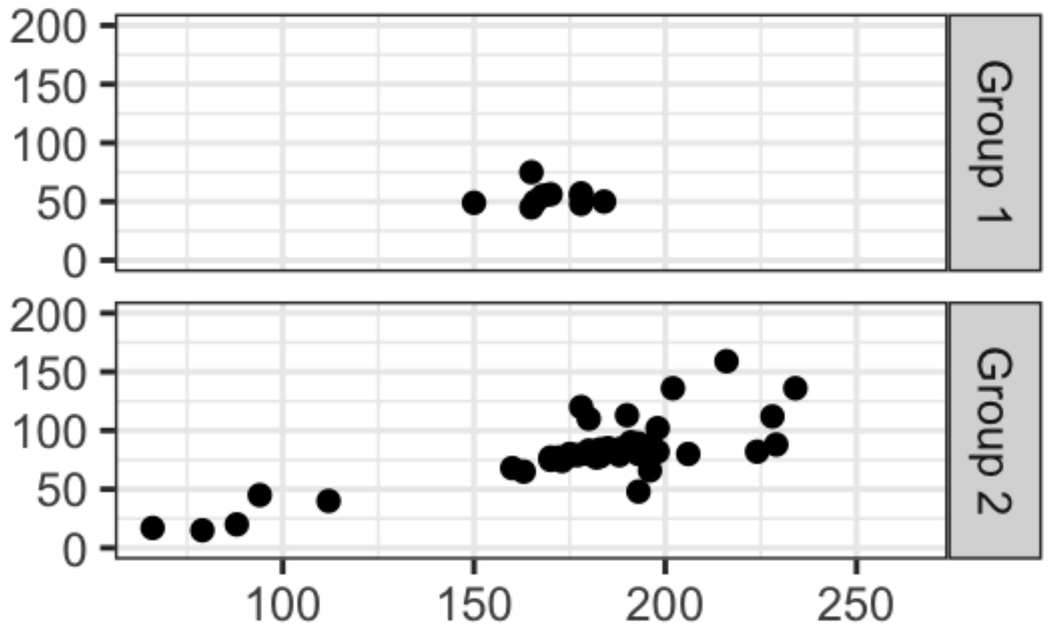
ggplot2 scales



ggplot2 facets



ggplot2 themes



bechdel

```
bechdel
```

year <int>	imdb <chr>	title <chr>	test <chr>	clean_test <ord>	binary <chr>	budget <int>	domgross <dbl>	intgross <dbl>	code <chr>
2013	tt1711425	21 & Over	notalk	notalk	FAIL	13000000	25682380	42195766	2013FAIL
2012	tt1343727	Dredd 3D	ok-disagree	ok	PASS	45000000	13414714	40868994	2012PASS
2013	tt2024544	12 Years a Slave	notalk-disagree	notalk	FAIL	20000000	53107035	158607035	2013FAIL
2013	tt1272878	2 Guns	notalk	notalk	FAIL	61000000	75612460	132493015	2013FAIL
2013	tt0453562	42	men	men	FAIL	40000000	95020213	95020213	2013FAIL
2013	tt1335975	47 Ronin	men	men	FAIL	225000000	38362475	145803842	2013FAIL
2013	tt1606378	A Good Day to Die Hard	notalk	notalk	FAIL	92000000	67349198	304249198	2013FAIL
2013	tt2194499	About Time	ok-disagree	ok	PASS	12000000	15323921	87324746	2013PASS
2013	tt1814621	Admission	ok	ok	PASS	13000000	18007317	18007317	2013PASS
2013	tt1815862	After Earth	notalk	notalk	FAIL	130000000	60522097	244373198	2013FAIL

1–10 of 1,794 rows | 1–10 of 15 columns

Previous 1 2 3 4 5 6 … 100 Next

ATLAS

Bechdel Test

1. At least 2 named women in the movie,
2. They have at least 1 conversation with each other, and
3. That conversation isn't about a man

Bechdel Test Outcomes

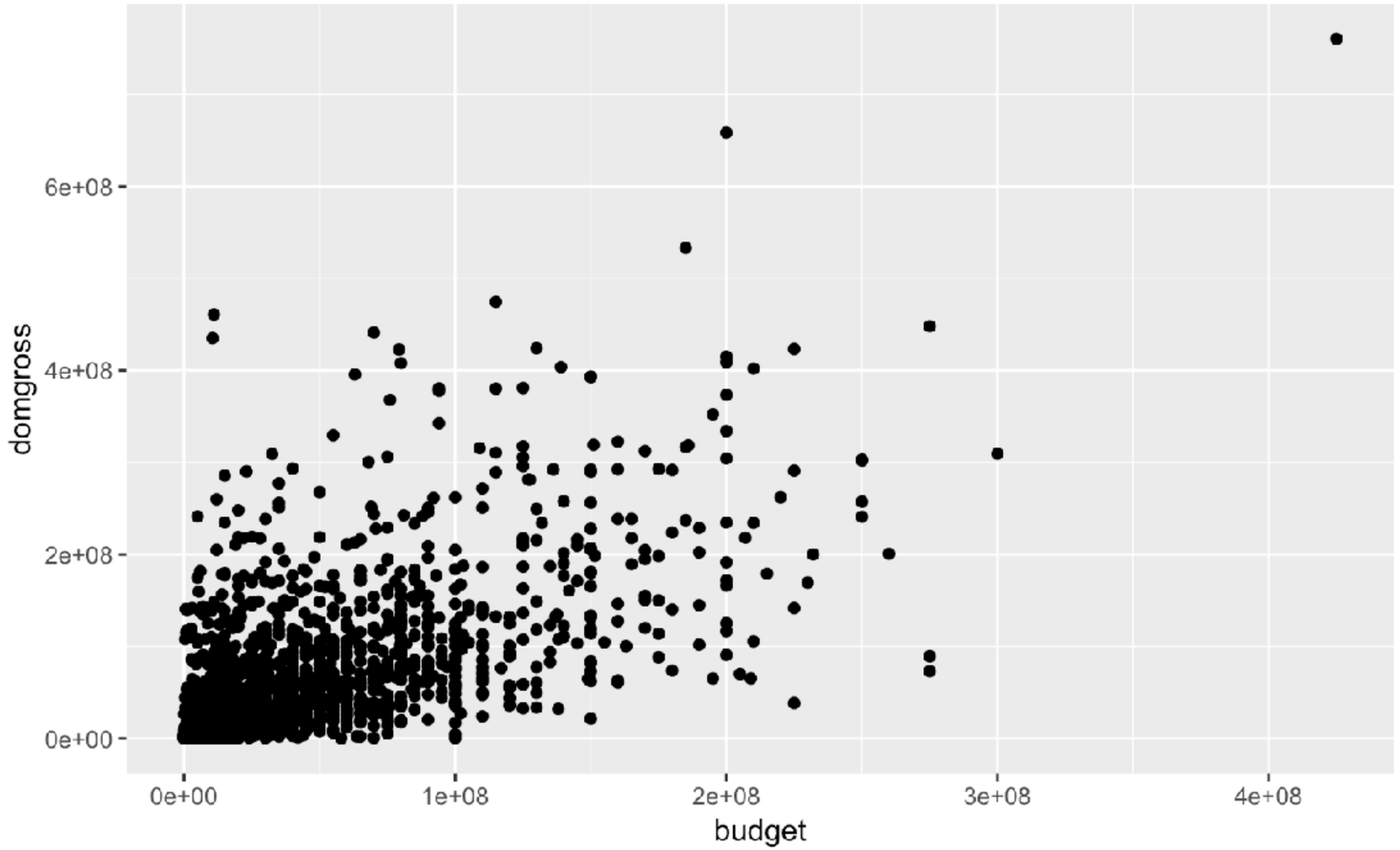
- Binary outcome (Pass/Fail)
  - Does the movie meet all three criteria?

- 5 outcomes based on severity (`clean_test`)
  - **nowomen** - Fails first criteria; less than 2 named female characters
  - **notalk** - Fails second criteria; multiple named females that don't talk to each other
  - **men** - Fails third criteria; named females talk, but only about men
  - **dubious** - Experts divided on whether third criteria is met
  - **ok** - Meets all criteria

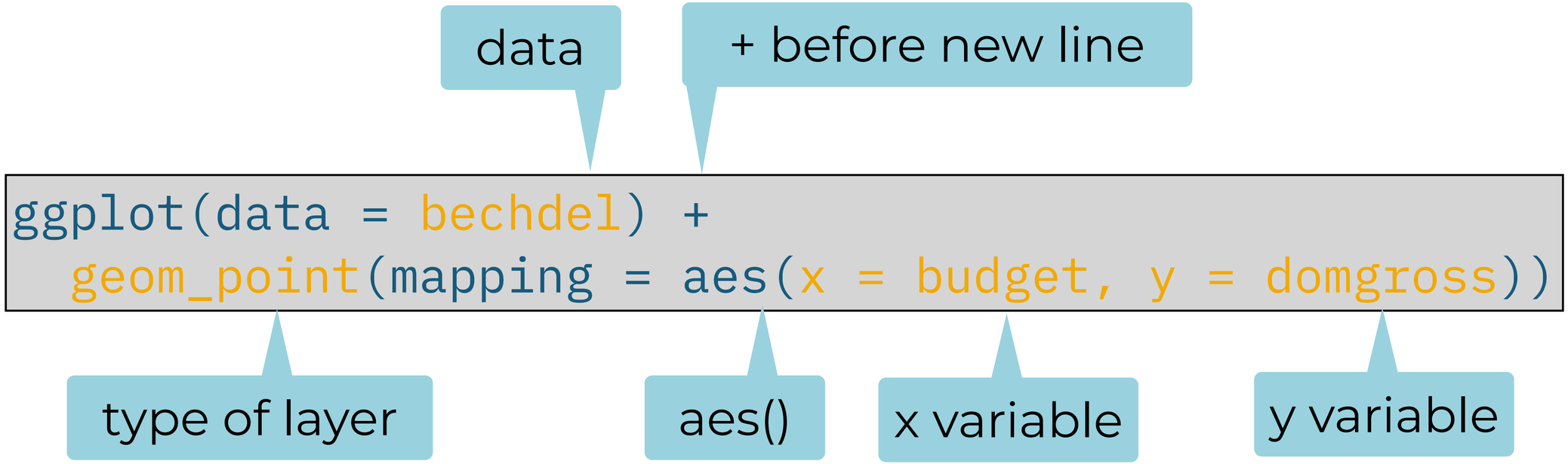
Your Turn 1

- Open the R Notebook **01-Visualize.Rmd**
- Let's look at the **bechdel** data set
- Run this code to make a graph:

```
ggplot(data = bechdel) +
  geom_point(mapping = aes(x = budget, y = domgross))
```




```
ggplot(data = bechdel) +
  geom_point(mapping = aes(x = budget, y = domgross))
```

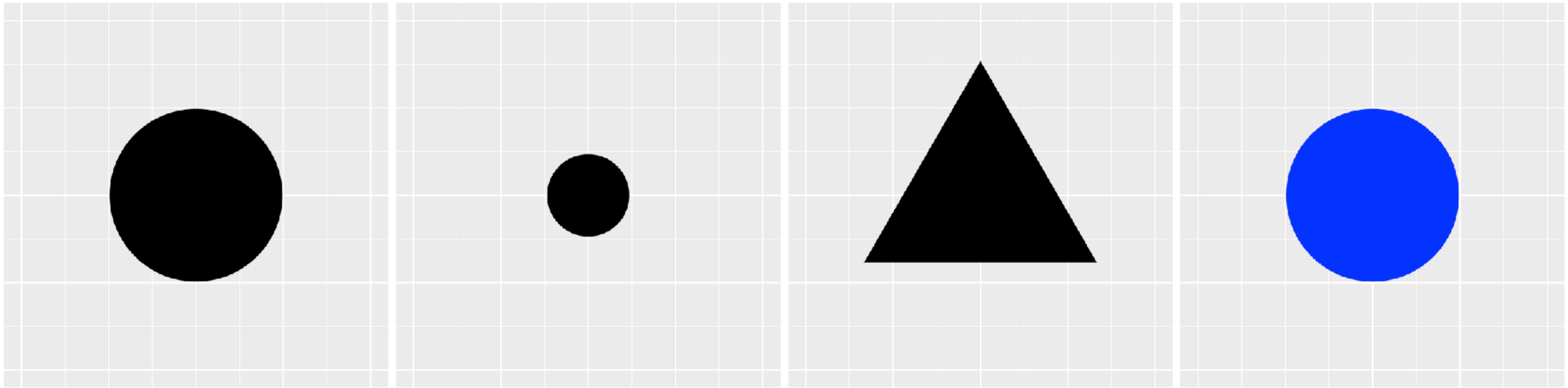


ggplot2 template

```
ggplot(data = <DATA>) +
  <GEOM_FUNCTION>(mapping = aes(<MAPPINGS>))
```

Aesthetics

Aesthetics



Aesthetics

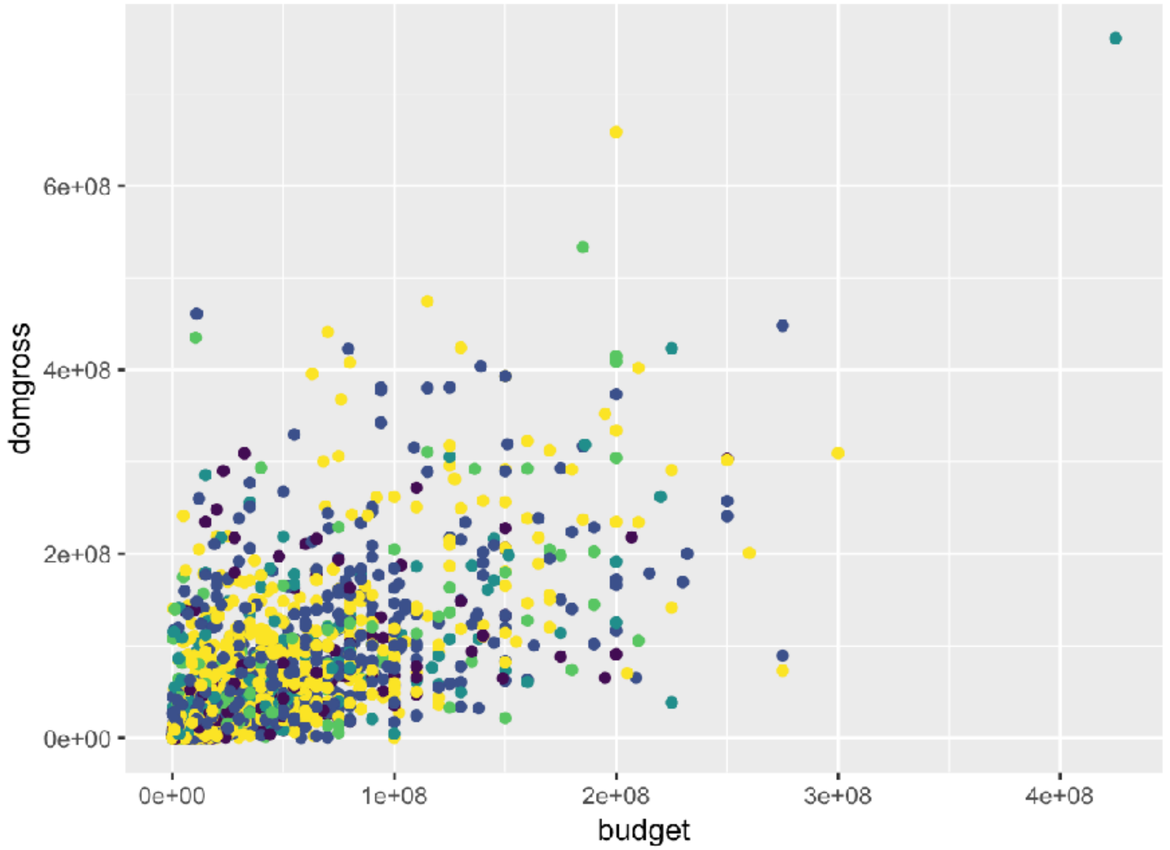
color		clean_test
Purple	⟷	nowomen
Blue	⟷	notalk
Teal	⟷	men
Lime	⟷	dubious
Yellow	⟷	ok

Aesthetic property

Variable to map it to

```
ggplot(data = bechdel) +
  geom_point(mapping = aes(x = budget, y = domgross, color = clean_test))
```

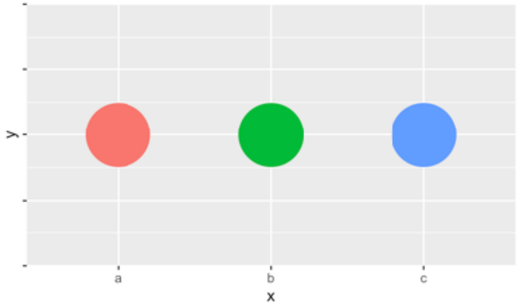
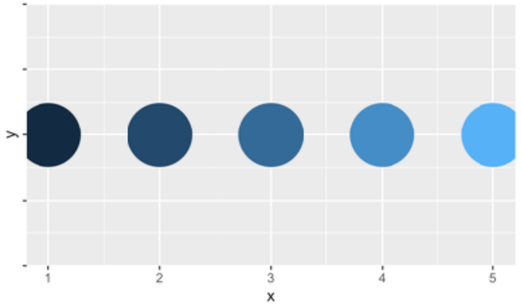
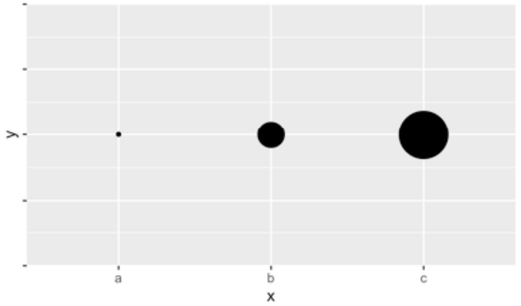
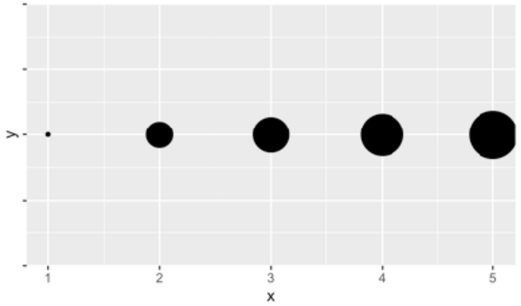
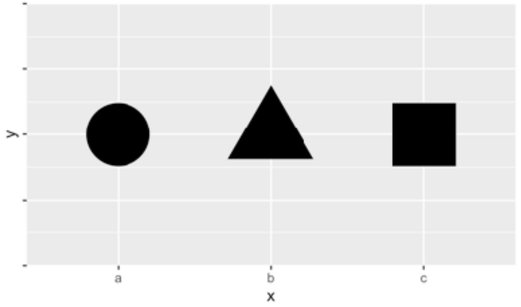
```
ggplot(data = bechdel) +
  geom_point(mapping = aes(x = budget, y = domgross, color = clean_test))
```



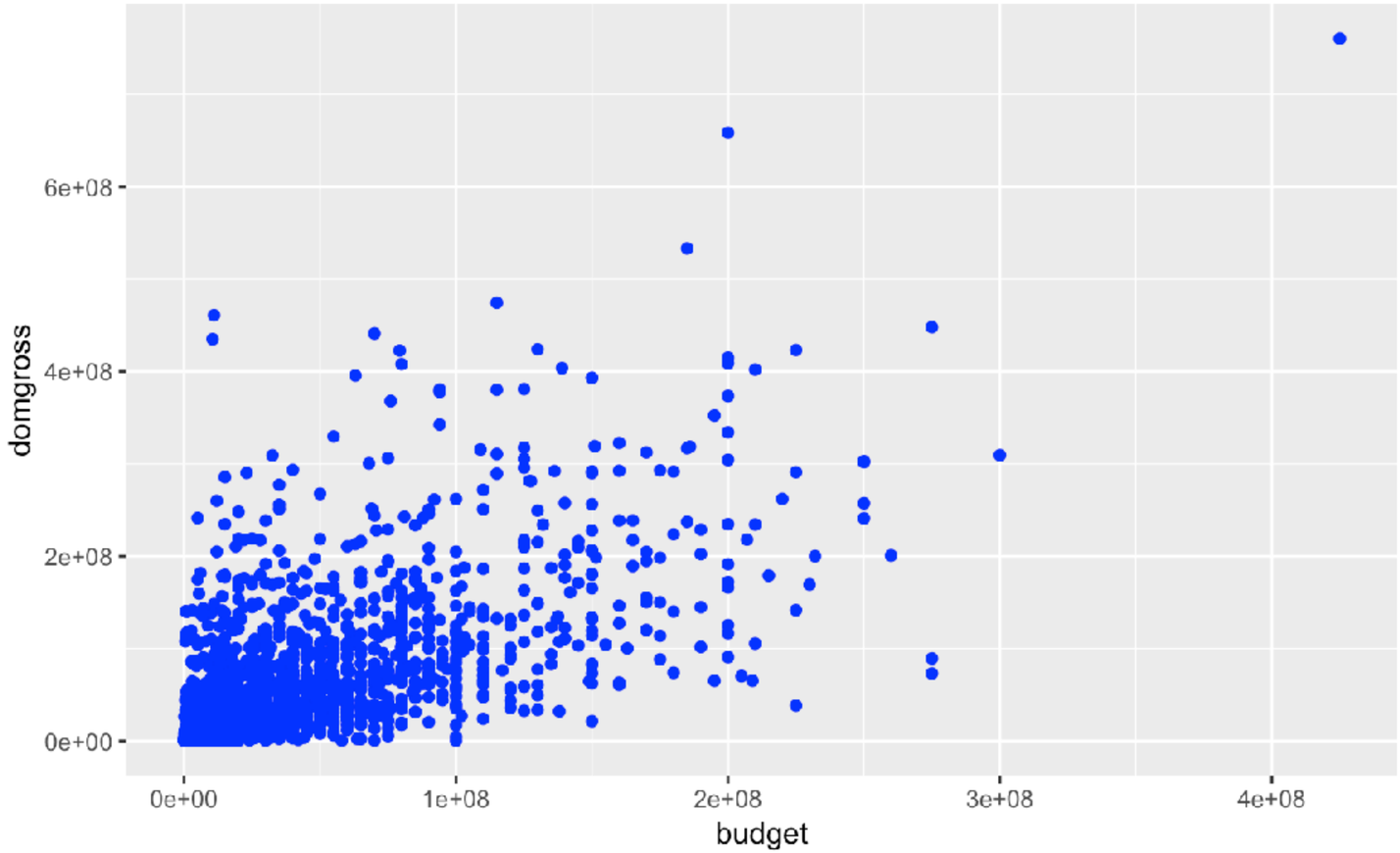
Legend added automatically

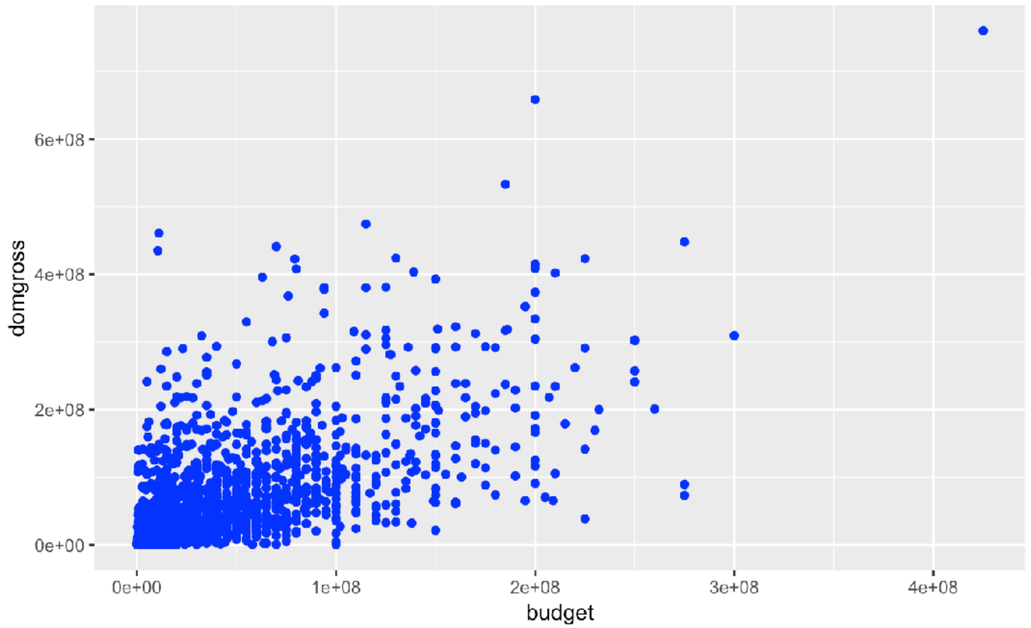
Your Turn 2

- Experiment with adding color, size, alpha, and shape aesthetics to your graph
- How do aesthetics behave different when mapped to discrete and continuous variables?
- What happens when you use more than one aesthetic?

	Discrete	Continuous
Color		
Size		
Shape		

Set vs. map

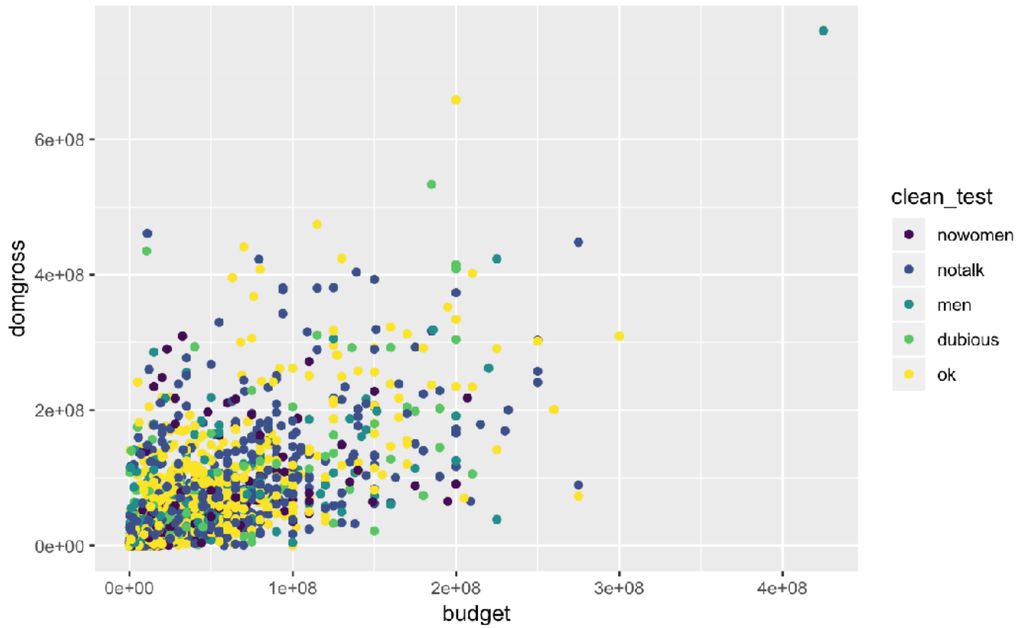


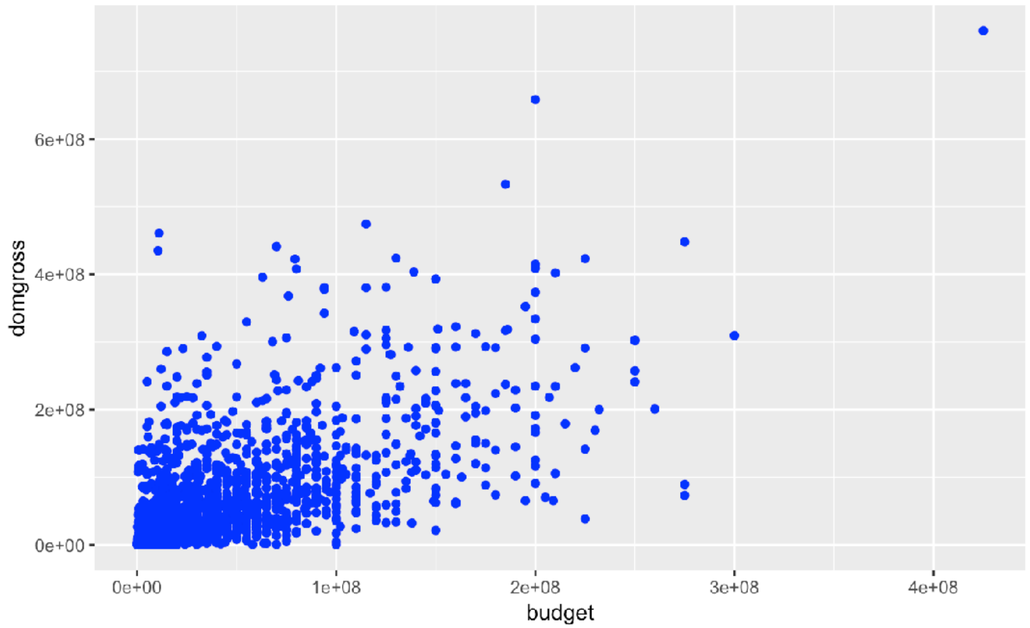


Outside of aes(): sets an aesthetic to a value

```
ggplot(data = bechdel) +
  geom_point(mapping = aes(x = budget, y = domgross, color = clean_test))
```

```
ggplot(data = bechdel) +
  geom_point(mapping = aes(x = budget, y = domgross), color = "blue")
```

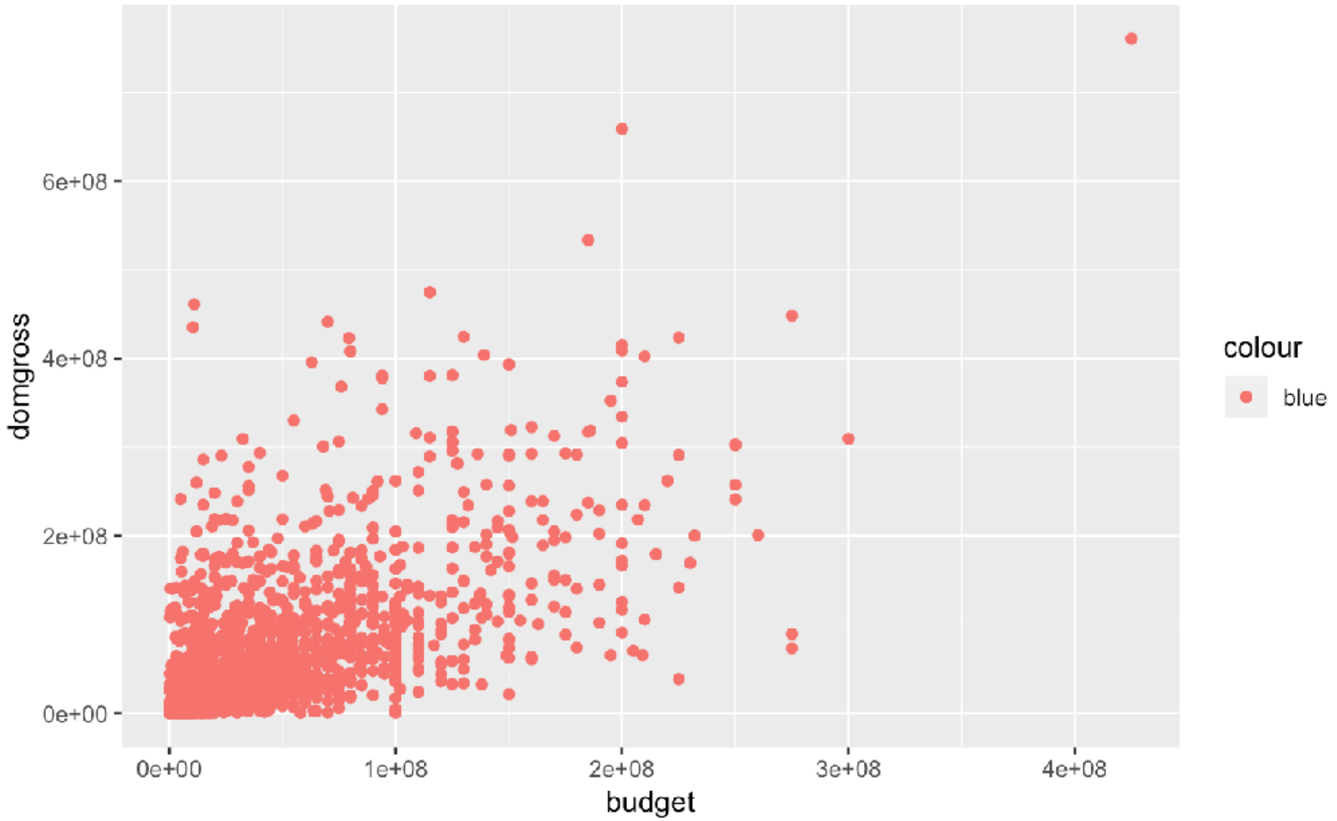




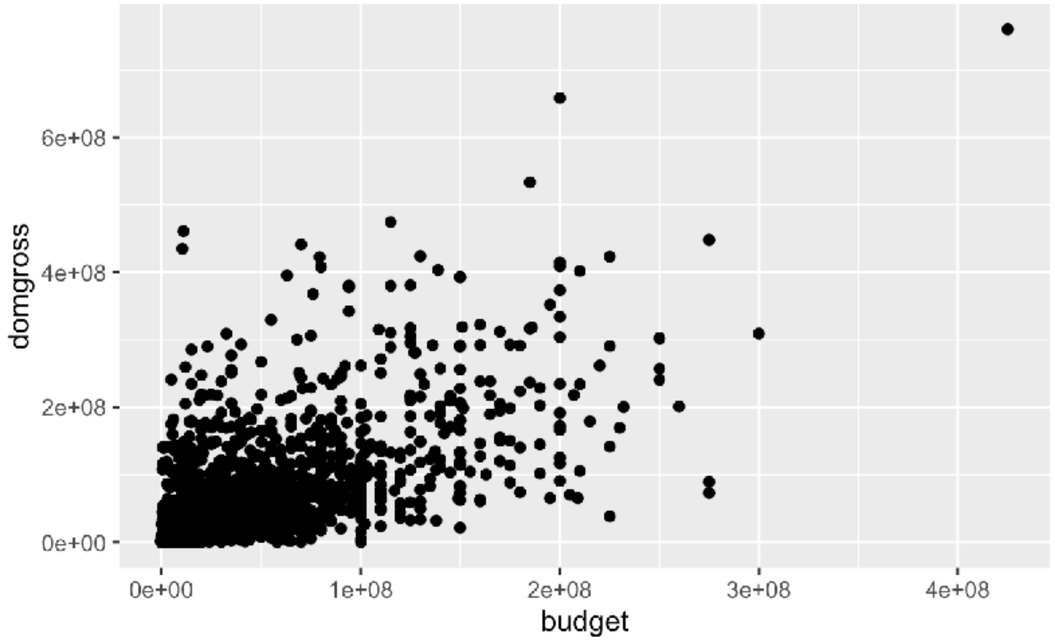

```
ggplot(data = bechdel) +
  geom_point(mapping = aes(x = budget, y = domgross, color = clean_test))
```

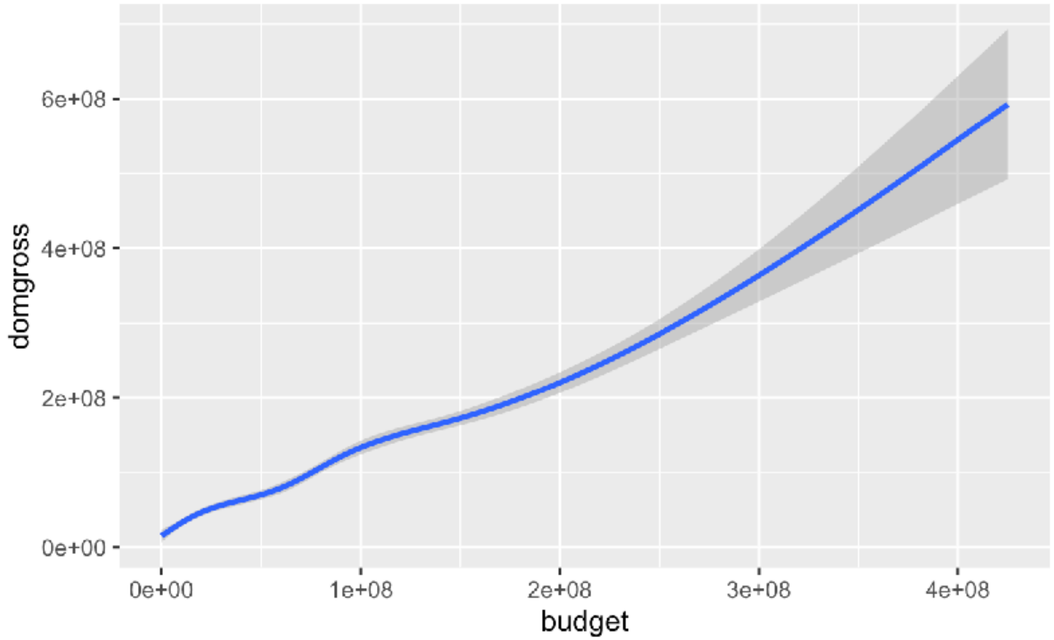
```
ggplot(data = bechdel) +
  geom_point(mapping = aes(x = budget, y = domgross), color = "blue")
```

```
ggplot(data = bechdel) +
  geom_point(mapping = aes(x = budget, y = domgross, color = "blue"))
```

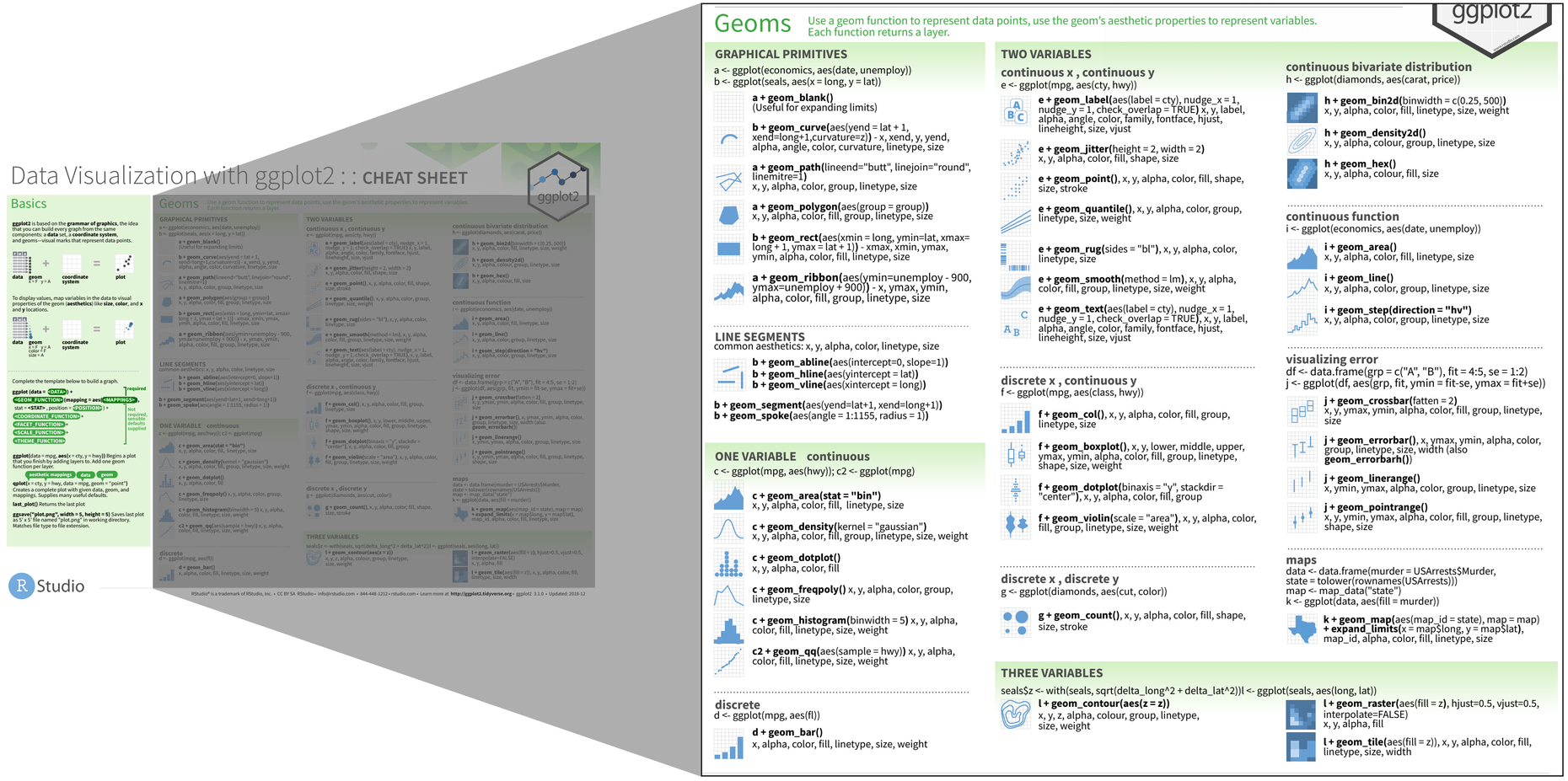


Geoms





```
ggplot(data = <DATA>) +
  <GEOM_FUNCTION>(mapping = aes(<MAPPINGS>))
```



Data Visualization with ggplot2 : : **CHEAT SHEET**

Basics

ggplot2 is based on the **grammar of graphics**, the idea that you can build every graph from the same components: a **data** set, a **coordinate system**, and geoms—visual marks that represent data points.

data + **geom** (x = F, y = A) + **coordinate system** = **plot**

To display values, map variables in the data to visual properties of the geom (**aesthetics**) like **size**, **color**, and **x** and **y** locations.

data + **geom** (x = F, y = A, color = F, size = A) + **coordinate system** = **plot**

Complete the template below to build a graph.

```
ggplot (data = <DATA>) +
  <GEOM_FUNCTION> (mapping = aes(<MAPPINGS>),
    stat = <STAT>, position = <POSITION>) +
  <COORDINATE_FUNCTION> +
  <FACET_FUNCTION> +
  <SCALE_FUNCTION> +
  <THEME_FUNCTION>
```

required

Not required, sensible defaults supplied

ggplot(data = mpg, **aes(**x = cty, y = hwy**))** Begins a plot that you finish by adding layers to. Add one geom function per layer.

aesthetic mappings · data · geom

qplot(x = cty, y = hwy, data = mpg, geom = "point" **)** Creates a complete plot with given data, geom, and mappings. Supplies many useful defaults.

last_plot() Returns the last plot

ggsave("plot.png", width = 5, height = 5) Saves last plot as 5' x 5' file named "plot.png" in working directory. Matches file type to file extension.

RStudio

Geoms

Use a geom function to represent data points, use the geom's aesthetic properties to represent variables. Each function returns a layer.

GRAPHICAL PRIMITIVES

```
a <- ggplot(economics, aes(date, unemploy))
b <- ggplot(seals, aes(x = long, y = lat))
```

a + geom_blank()
(Useful for expanding limits)

b + geom_curve(aes(yend = lat + 1, xend=long+1,curvature=z**))** - x, xend, y, yend, alpha, angle, color, curvature, linetype, size

a + geom_path(lineend="butt", linejoin="round", linemitre=1**)**
x, y, alpha, color, group, linetype, size

a + geom_polygon(aes(group = group)**)**
x, y, alpha, color, fill, group, linetype, size

b + geom_rect(aes(xmin = long, ymin=lat, xmax= long + 1, ymax = lat + 1**))** - xmax, xmin, ymax, ymin, alpha, color, fill, linetype, size

a + geom_ribbon(aes(ymin=unemploy - 900, ymax=unemploy + 900**))** - x, ymax, ymin, alpha, color, fill, group, linetype, size

LINE SEGMENTS

common aesthetics: x, y, alpha, color, linetype, size

b + geom_abline(aes(intercept=0, slope=1**))**
b + geom_hline(aes(yintercept = lat**))**
b + geom_vline(aes(xintercept = long**))**

b + geom_segment(aes(yend=lat+1, xend=long+1**))**
b + geom_spoke(aes(angle = 1:1155, radius = 1**))**

ONE VARIABLE continuous

```
c <- ggplot(mpg, aes(hwy)); c2 <- ggplot(mpg)
```

c + geom_area(stat = "bin")
x, y, alpha, color, fill, linetype, size

c + geom_density(kernel = "gaussian"**)**
x, y, alpha, color, fill, group, linetype, size, weight

c + geom_dotplot()
x, y, alpha, color, fill

c + geom_freqpoly() x, y, alpha, color, group, linetype, size

c + geom_histogram(binwidth = 5**)** x, y, alpha, color, fill, linetype, size, weight

c2 + geom_qq(aes(sample = hwy**))** x, y, alpha, color, fill, linetype, size, weight

TWO VARIABLES

continuous x , continuous y

```
e <- ggplot(mpg, aes(cty, hwy))
```

e + geom_label(aes(label = cty), nudge_x = 1, nudge_y = 1, check_overlap = TRUE**))** x, y, label, alpha, angle, color, family, fontface, hjust, lineheight, size, vjust

e + geom_jitter(height = 2, width = 2**)**
x, y, alpha, color, fill, shape, size

e + geom_point(), x, y, alpha, color, fill, shape, size, stroke

e + geom_quantile(), x, y, alpha, color, group, linetype, size, weight

e + geom_rug(sides = "bl"**)**, x, y, alpha, color, linetype, size

e + geom_smooth(method = lm**)**, x, y, alpha, color, fill, group, linetype, size, weight

e + geom_text(aes(label = cty), nudge_x = 1, nudge_y = 1, check_overlap = TRUE**))** x, y, label, alpha, angle, color, family, fontface, hjust, lineheight, size, vjust

discrete x , continuous y

```
f <- ggplot(mpg, aes(class, hwy))
```

f + geom_col(), x, y, alpha, color, fill, group, linetype, size

f + geom_boxplot(), x, y, lower, middle, upper, ymax, ymin, alpha, color, fill, group, linetype, shape, size, weight

f + geom_dotplot(binaxis = "y", stackdir = "center"**)**, x, y, alpha, color, fill, group

f + geom_violin(scale = "area"**)**, x, y, alpha, color, fill, group, linetype, size, weight

discrete x , discrete y

```
g <- ggplot(diamonds, aes(cut, color))
```

g + geom_count(), x, y, alpha, color, fill, shape, size, stroke

THREE VARIABLES

```
seals$z <- with(seals, sqrt(delta_long^2 + delta_lat^2))
l <- ggplot(seals, aes(long, lat))
```

l + geom_contour(aes(z = z))
x, y, z, alpha, colour, group, linetype, size, weight

l + geom_raster(aes(fill = z**)**, hjust=0.5, vjust=0.5, interpolate=FALSE)
x, y, alpha, fill

l + geom_tile(aes(fill = z**))**, x, y, alpha, color, fill, linetype, size, width

continuous bivariate distribution

```
h <- ggplot(diamonds, aes(carat, price))
```

h + geom_bin2d(binwidth = c(0.25, 500)**)**
x, y, alpha, color, fill, linetype, size, weight

h + geom_density2d()
x, y, alpha, colour, group, linetype, size

h + geom_hex()
x, y, alpha, colour, fill, size

continuous function

```
i <- ggplot(economics, aes(date, unemploy))
```

i + geom_area()
x, y, alpha, color, fill, linetype, size

i + geom_line()
x, y, alpha, color, group, linetype, size

i + geom_step(direction = "hv")
x, y, alpha, color, group, linetype, size

visualizing error

```
df <- data.frame(grp = c("A", "B"), fit = 4:5, se = 1:2)
j <- ggplot(df, aes(grp, fit, ymin = fit-se, ymax = fit+se))
```

j + geom_crossbar(fatten = 2**)**
x, y, ymax, ymin, alpha, color, fill, group, linetype, size

j + geom_errorbar(), x, ymax, ymin, alpha, color, group, linetype, size, width (also **geom_errorbarh()**)

j + geom_linerange()
x, ymin, ymax, alpha, color, group, linetype, size

j + geom_pointrange()
x, y, ymin, ymax, alpha, color, fill, group, linetype, shape, size

maps

```
data <- data.frame(murder = USArrests$Murder,
state = tolower(rownames(USArrests)))
map <- map_data("state")
k <- ggplot(data, aes(fill = murder))
```

k + geom_map(aes(map_id = state), map = map**)**
+ expand_limits(x = map$long, y = map$lat**)**, map_id, alpha, color, fill, linetype, size

RStudio® is a trademark of RStudio, Inc. • CC BY SA RStudio • info@rstudio.com • 844-448-1212 • rstudio.com • Learn more at **http://ggplot2.tidyverse.org** • ggplot2 3.1.0 • Updated: 2018-12

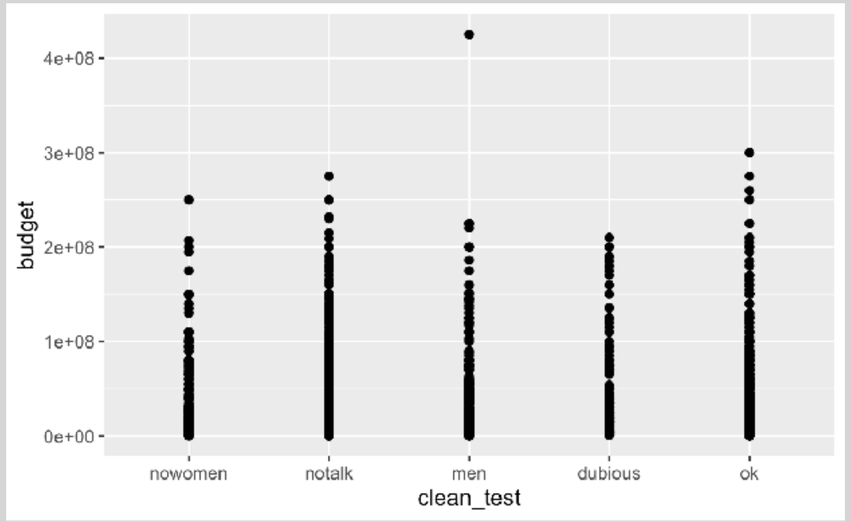
ATLAS

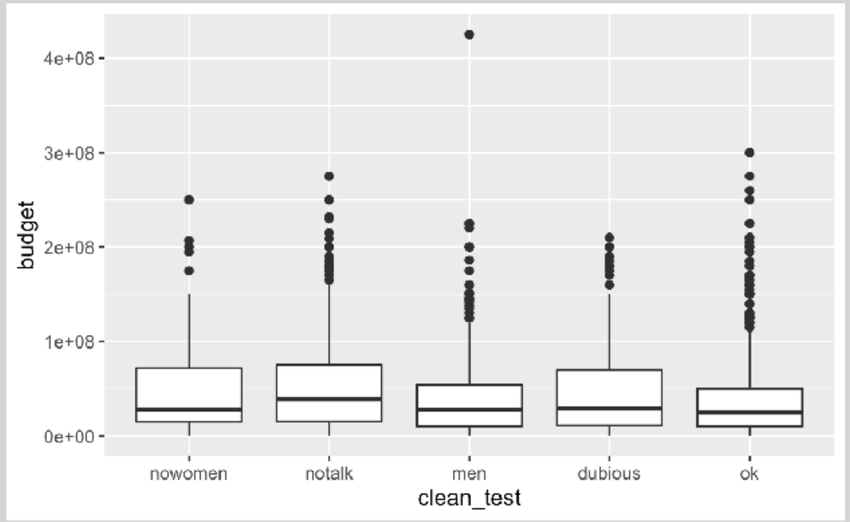
Adapted from Data Science in the tidyverse, CC BY Amelia McNamara

Your Turn 3

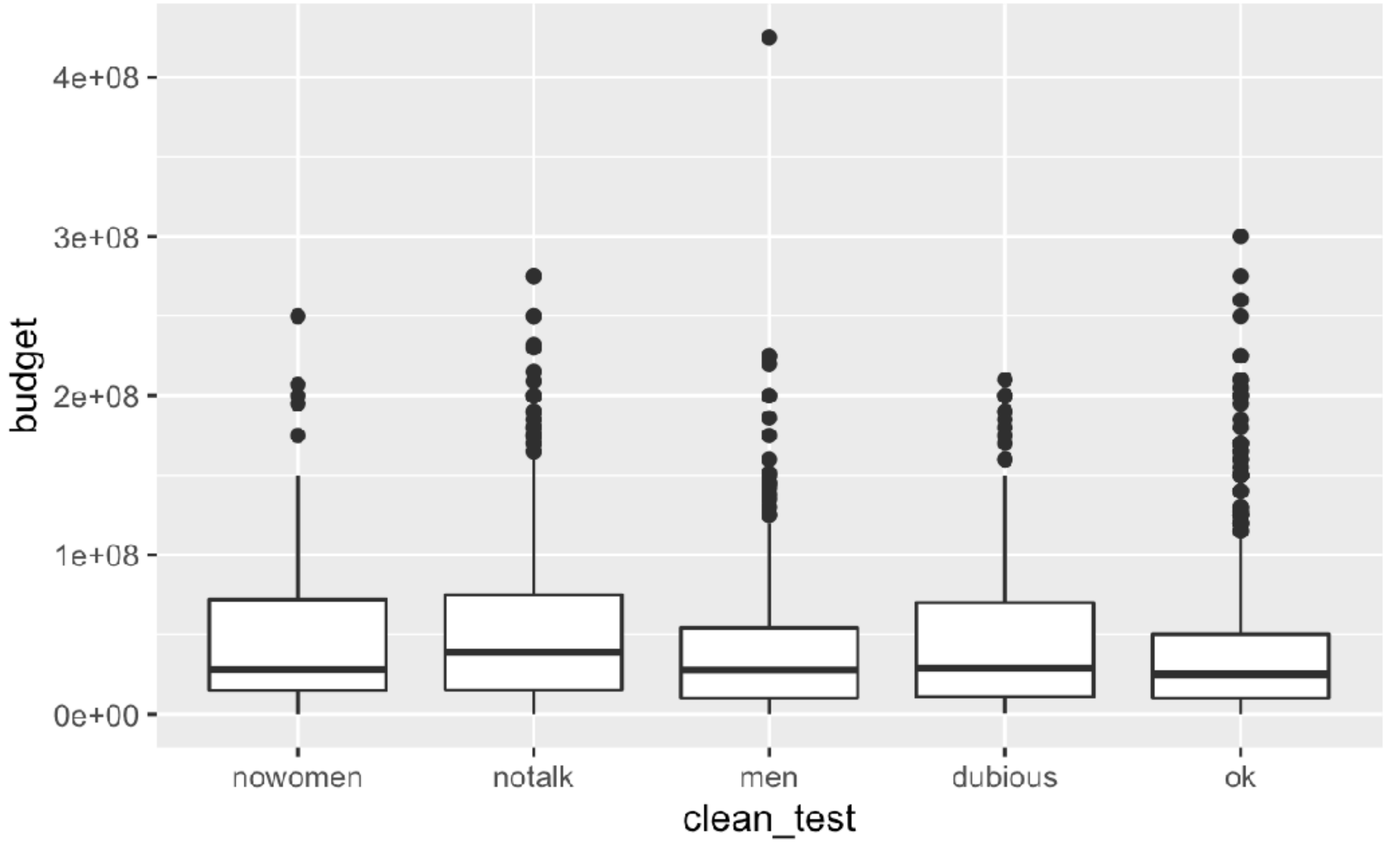
- Replace this scatterplot with one that draws boxplots

```
ggplot(data = bechdel) +
  geom_point(mapping = aes(x = clean_test, y = budget))
```





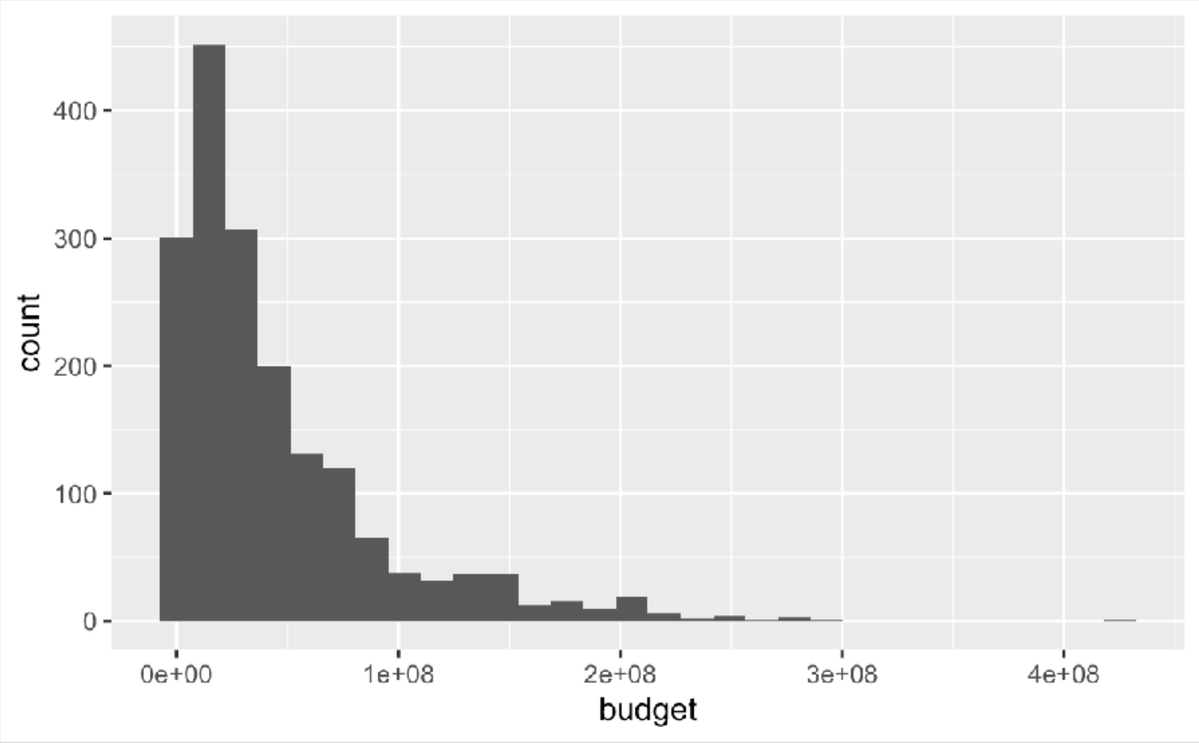
05 : 00



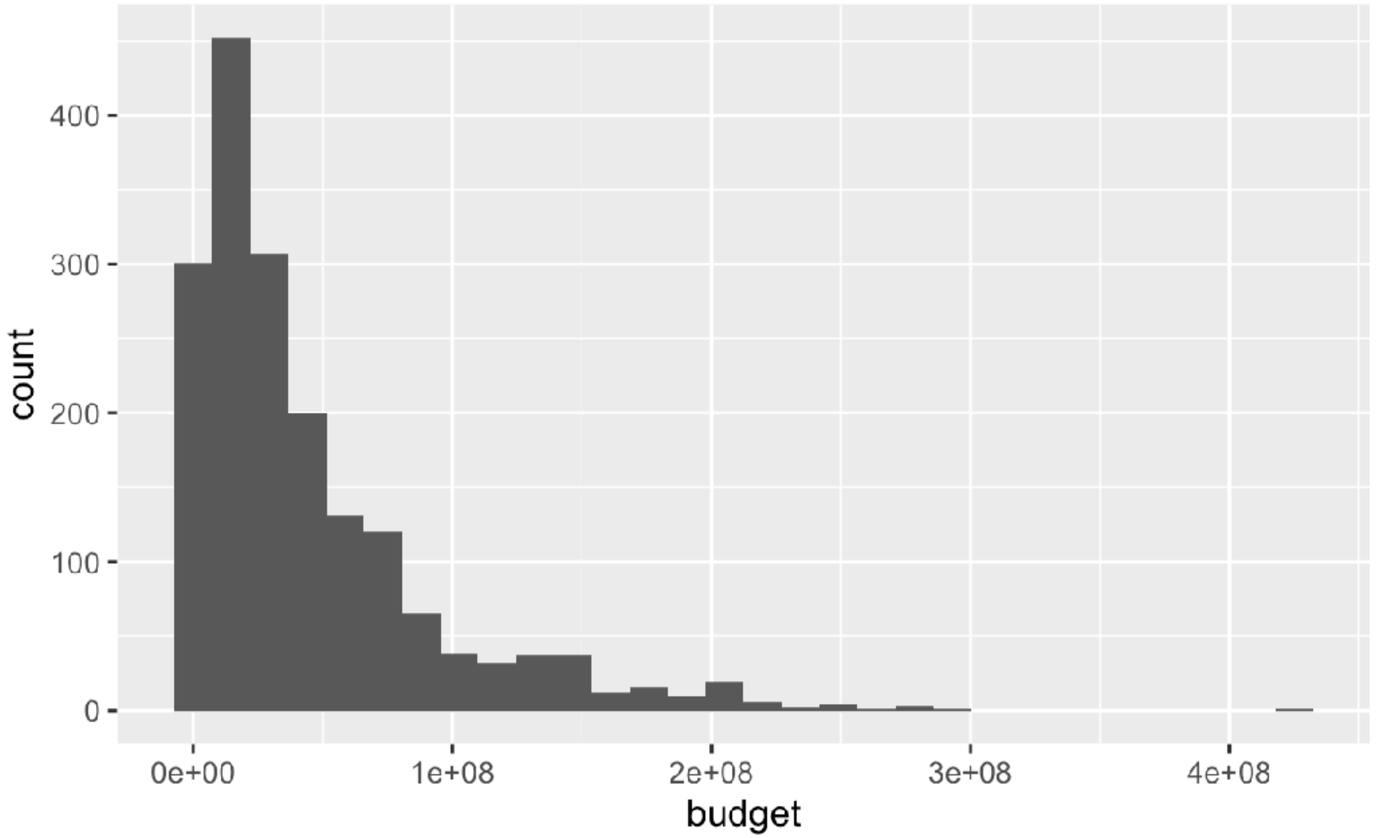

```
ggplot(data = bechdel) +
  geom_boxplot(mapping = aes(x = clean_test, y = budget))
```

Your Turn 4

- Make the histogram of **budget** shown below



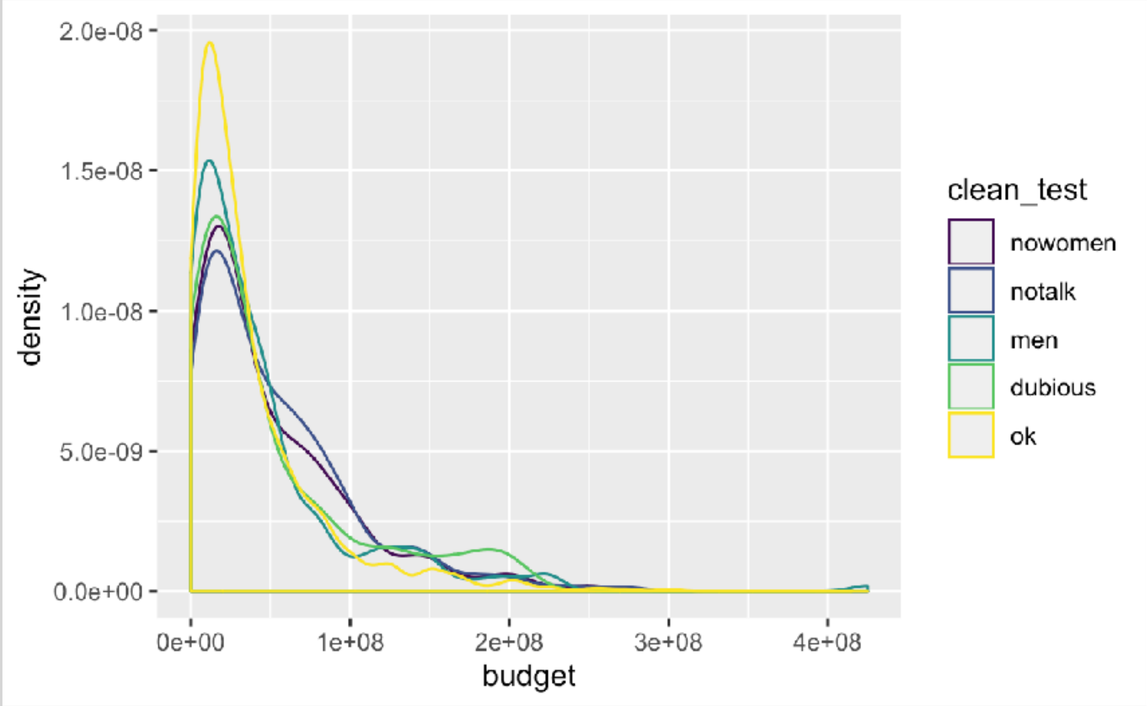
05 : 00



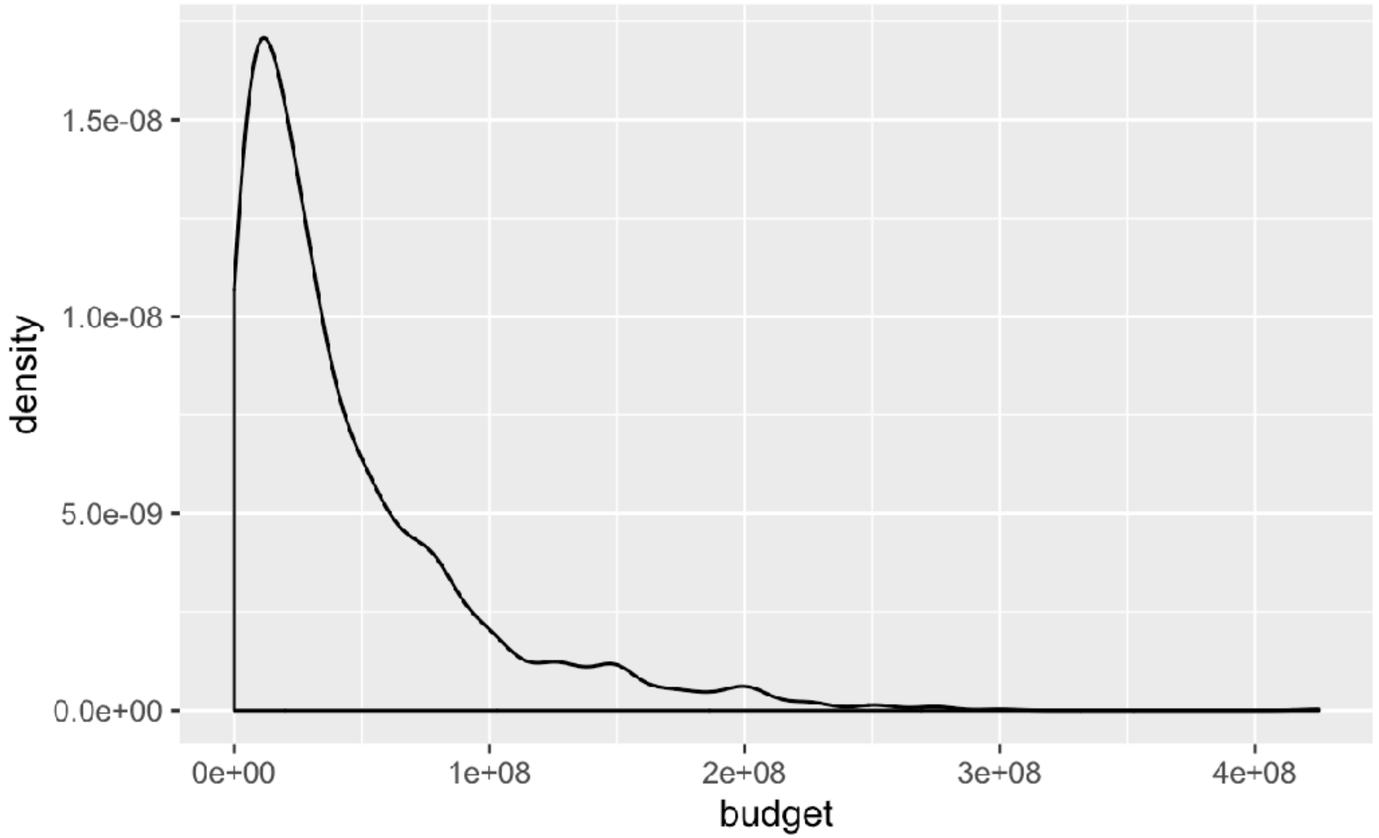

```
ggplot(data = bechdel) +
  geom_histogram(mapping = aes(x = budget))
```

Your Turn 5

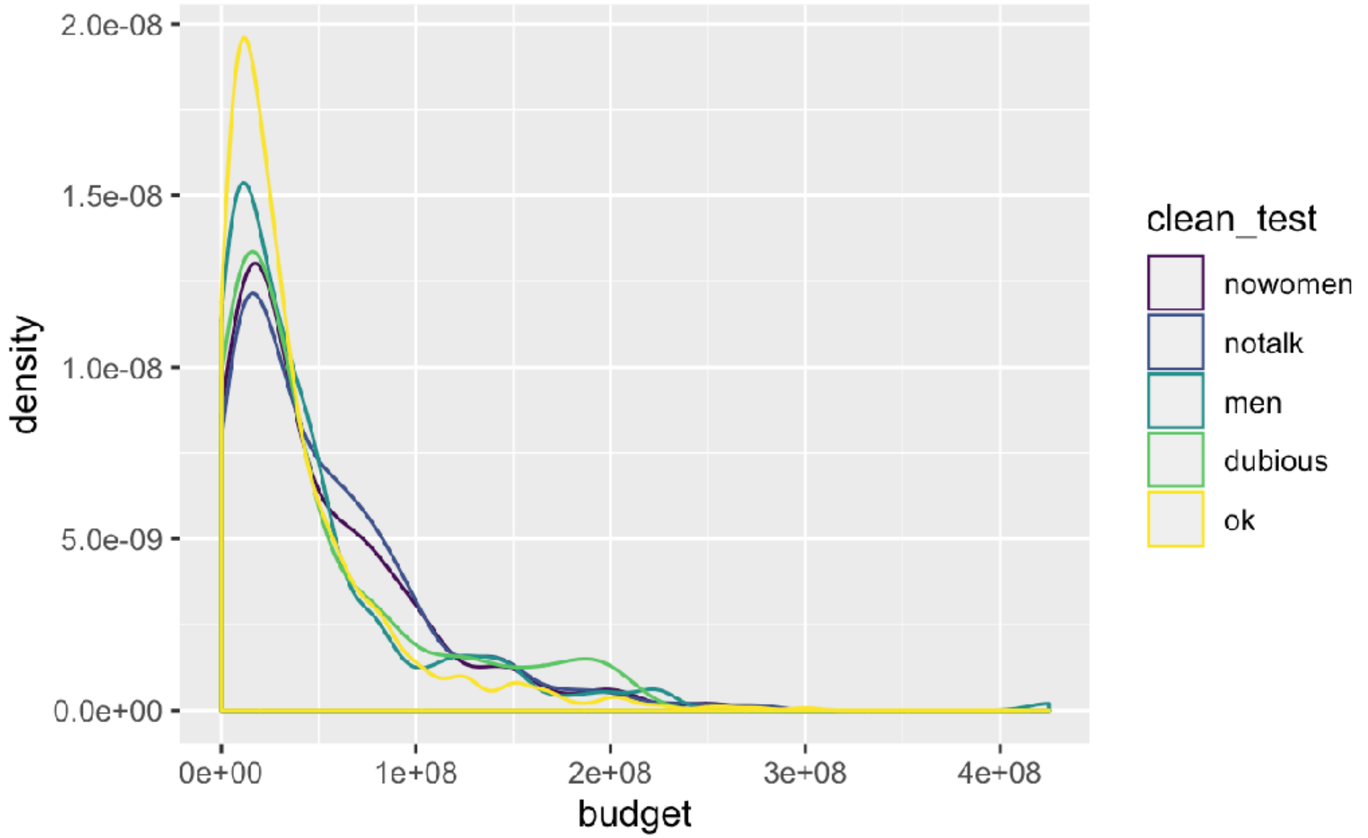
- Make the density plot of **budget** colored by **clean_test** shown below.



05 : 00



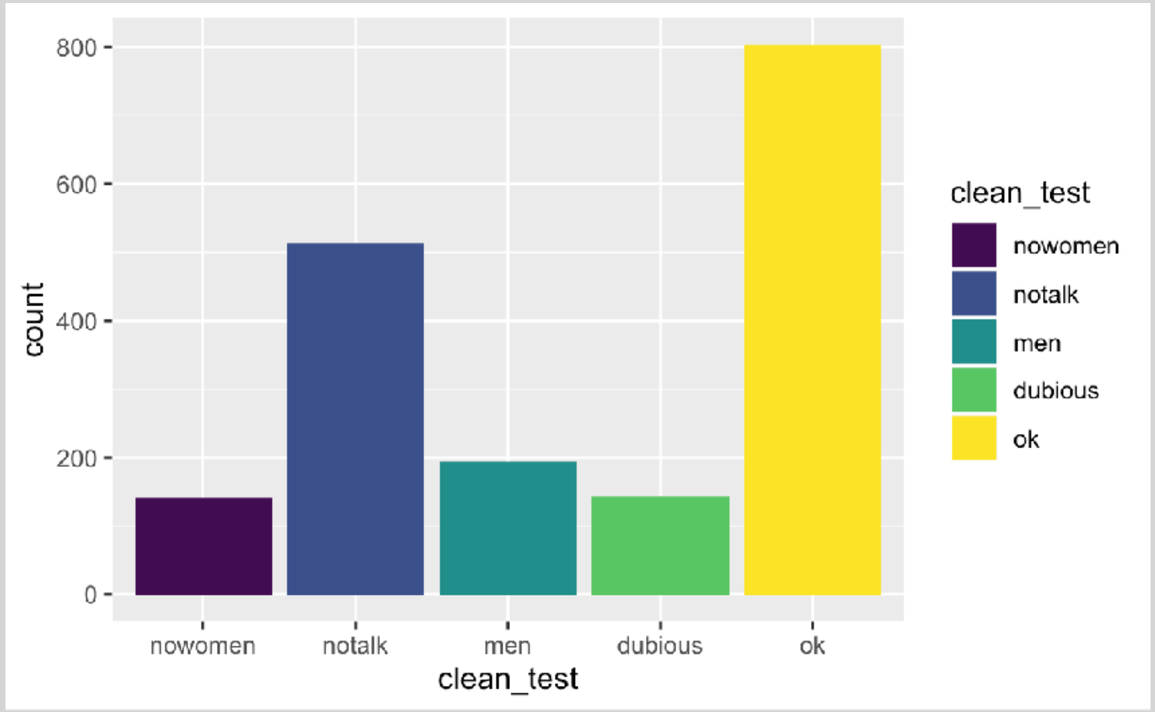

```
ggplot(data = bechdel) +
  geom_density(mapping = aes(x = budget))
```



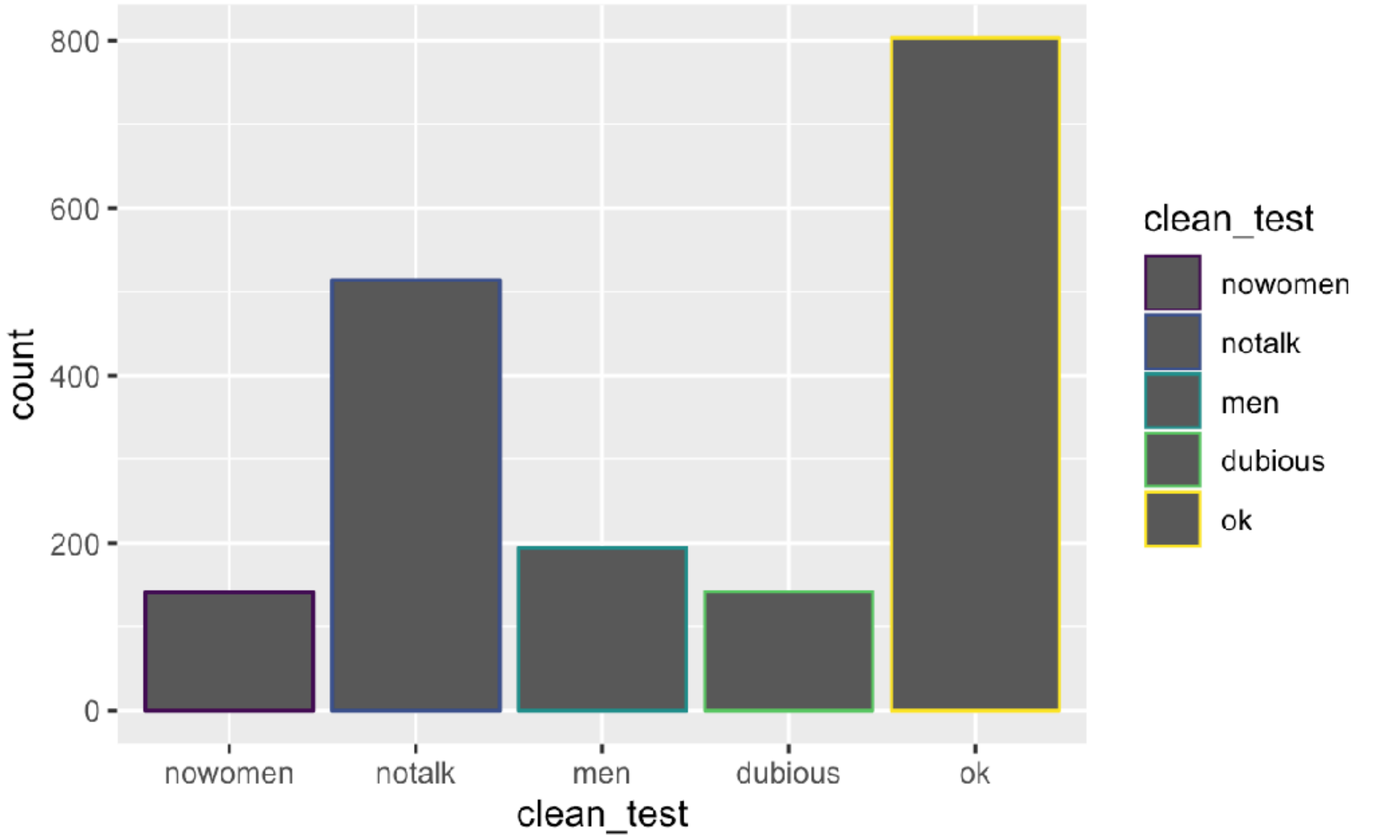

```
ggplot(data = bechdel) +
  geom_density(mapping = aes(x = budget, color = clean_test))
```

Your Turn 6

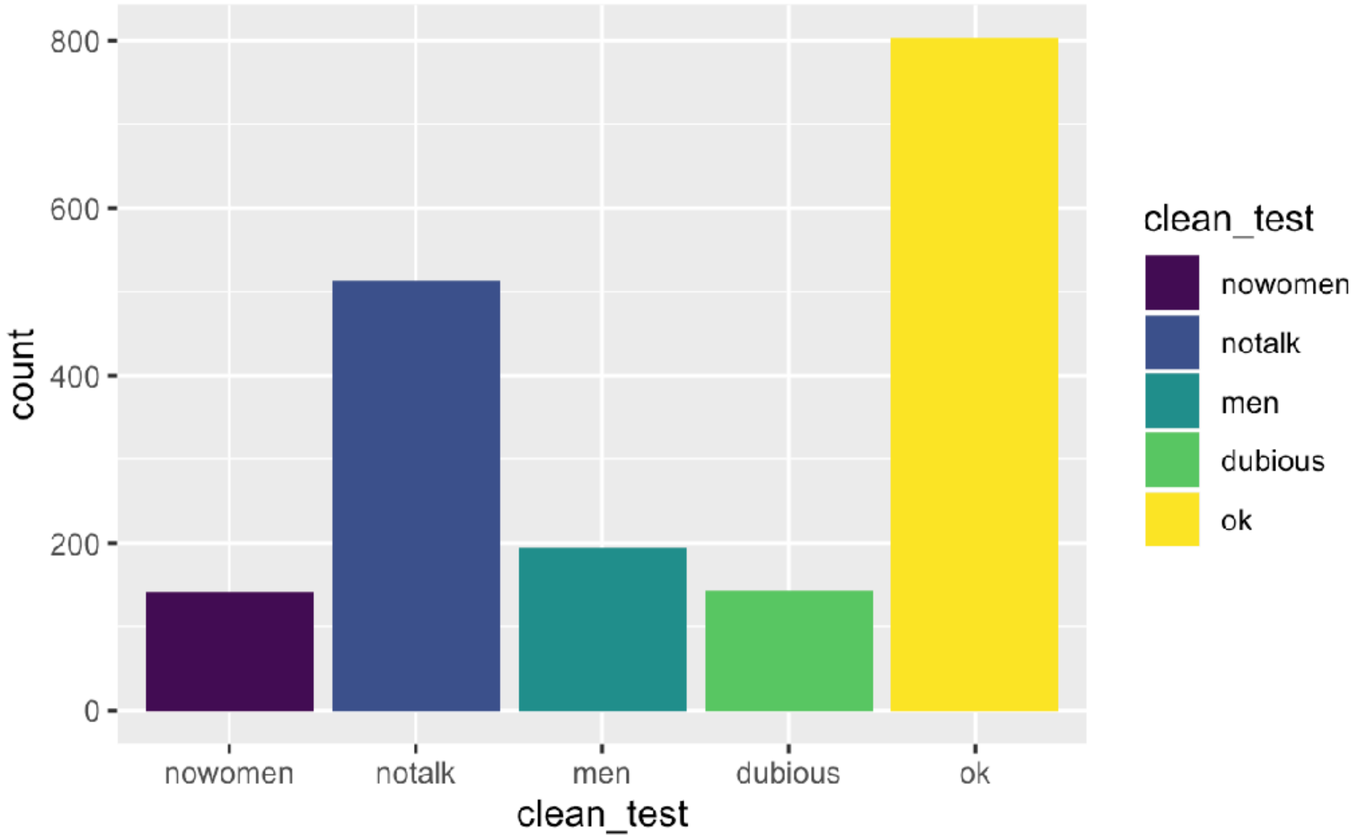
- Make the bar chart of **clean_test** colored by **clean_test** shown below.



05 : 00




```
ggplot(data = bechdel) +
  geom_bar(mapping = aes(x = clean_test, color = clean_test))
```

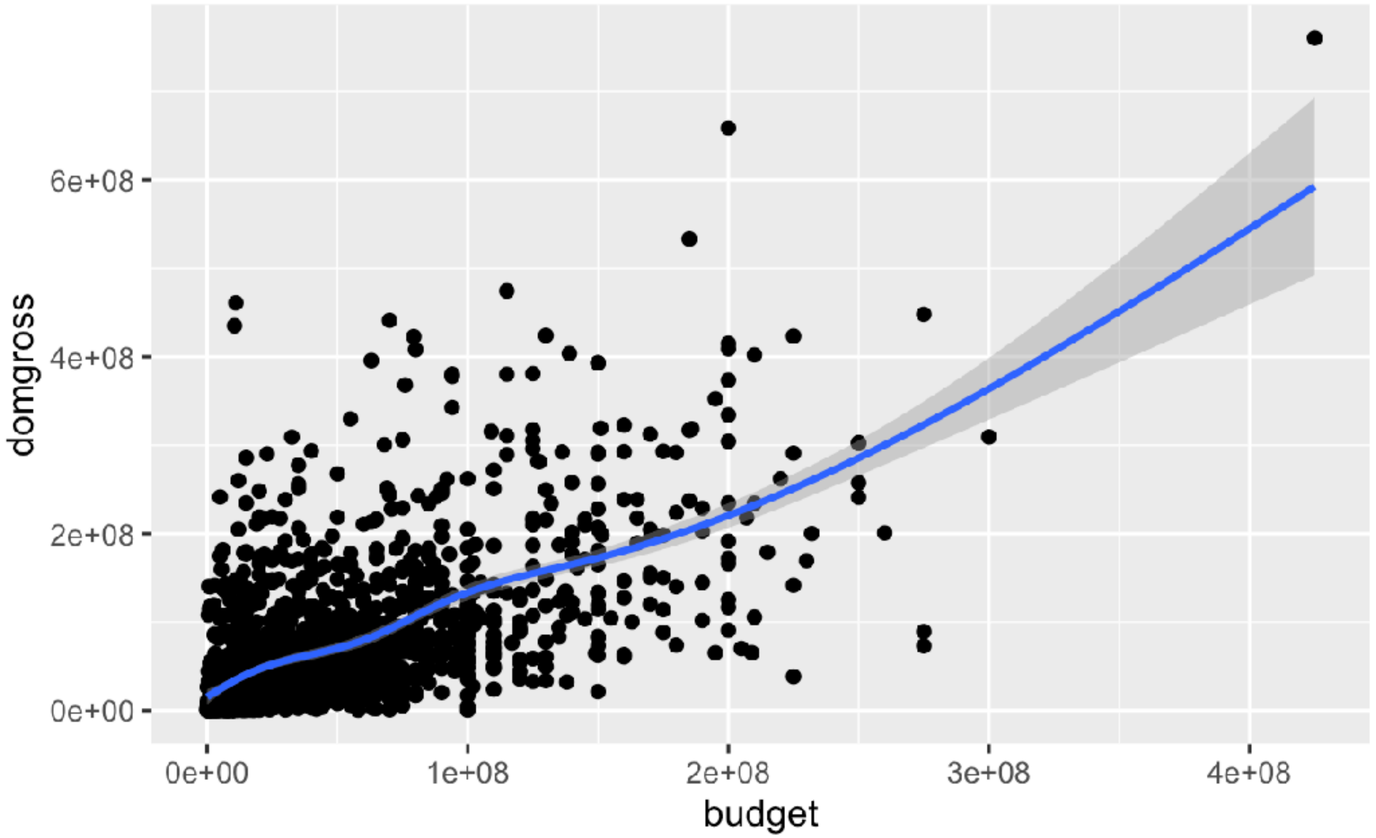



```
ggplot(data = bechdel) +
  geom_bar(mapping = aes(x = clean_test, fill = clean_test))
```

Your Turn 7

- Predict what this code will do.
- Then run it.

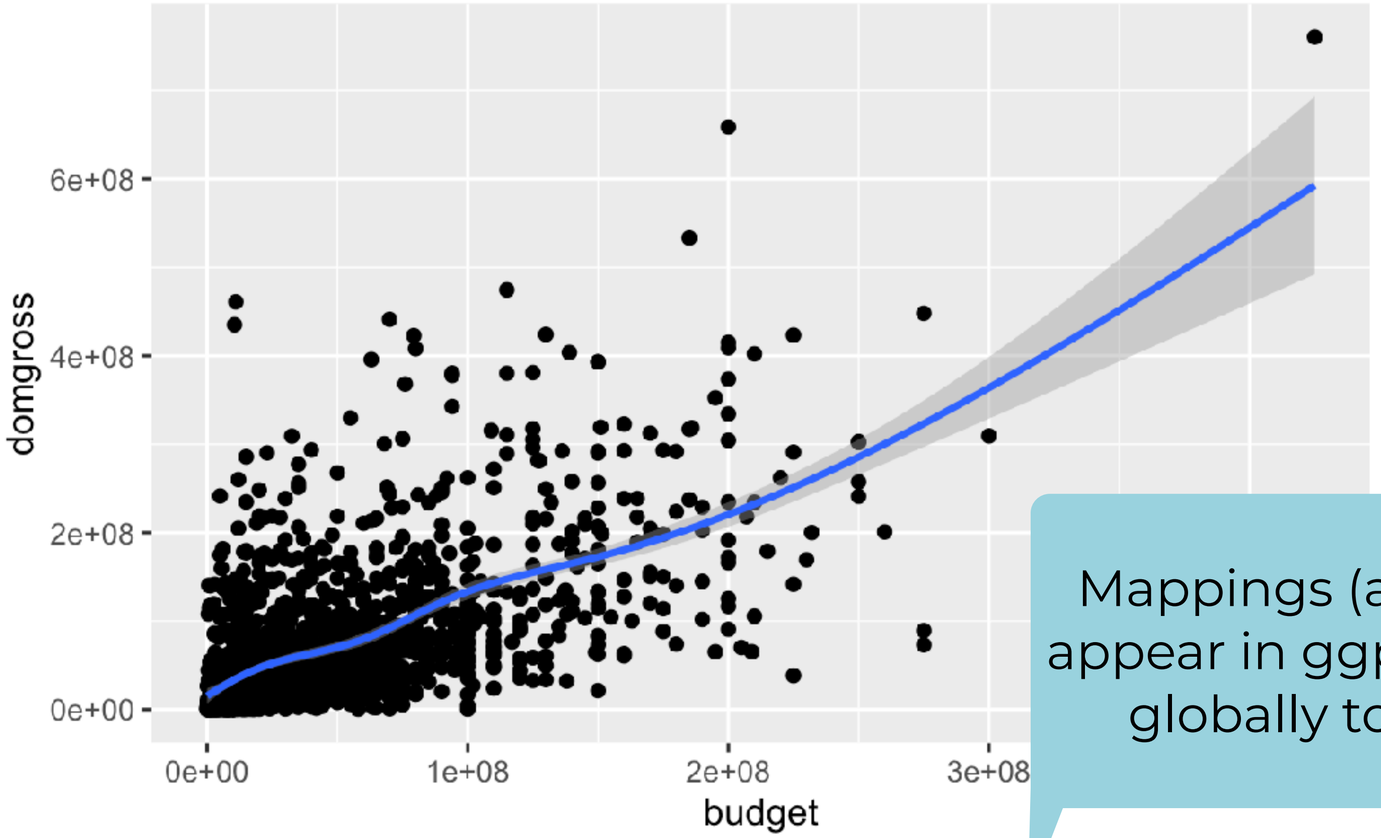
```
ggplot(data = bechdel) +
  geom_point(mapping = aes(x = budget, y = domgross)) +
  geom_smooth(mapping = aes(x = budget, y = domgross))
```



Each new geom adds a new layer

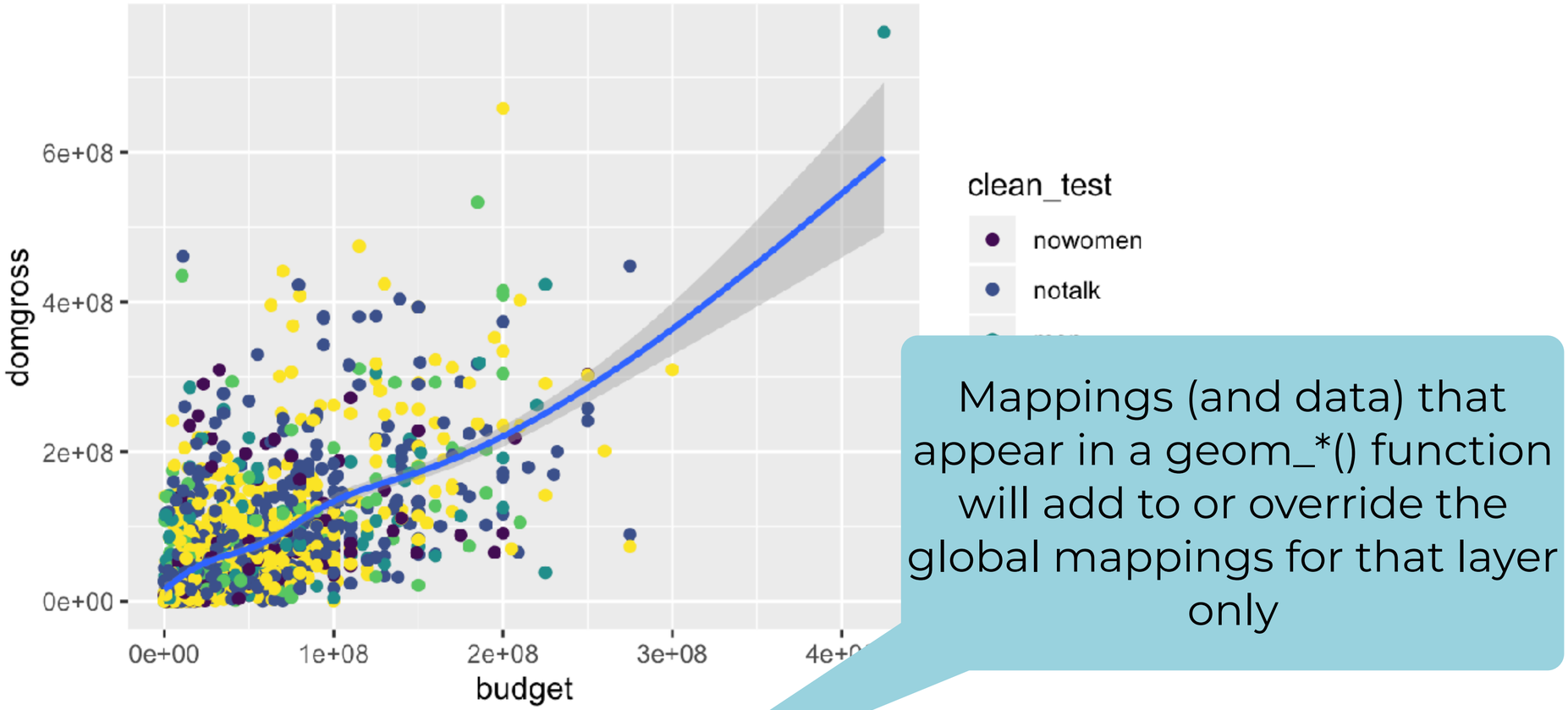
```
ggplot(data = bechdel) +
  geom_point(mapping = aes(x = budget, y = domgross)) +
  geom_smooth(mapping = aes(x = budget, y = domgross))
```

Global vs. Local



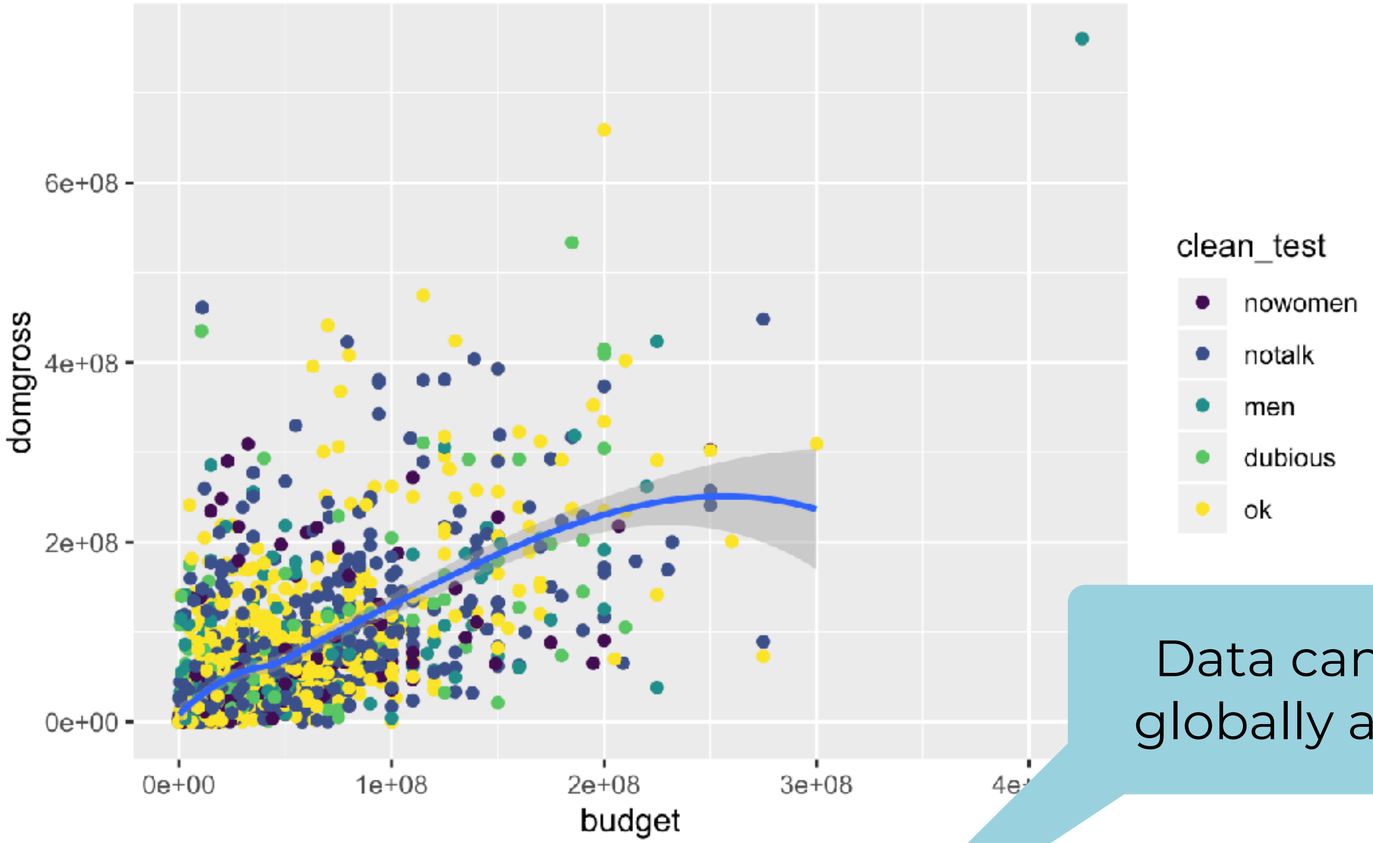
Mappings (and data) that appear in ggplot() will apply globally to every layer

```
ggplot(data = bechdel, mapping = aes(x = budget, y = domgross)) +
  geom_point() +
  geom_smooth()
```



Mappings (and data) that appear in a geom_*() function will add to or override the global mappings for that layer only

```
ggplot(data = bechdel, mapping = aes(x = budget, y = domgross)) +
  geom_point(mapping = aes(color = clean_test)) +
  geom_smooth()
```

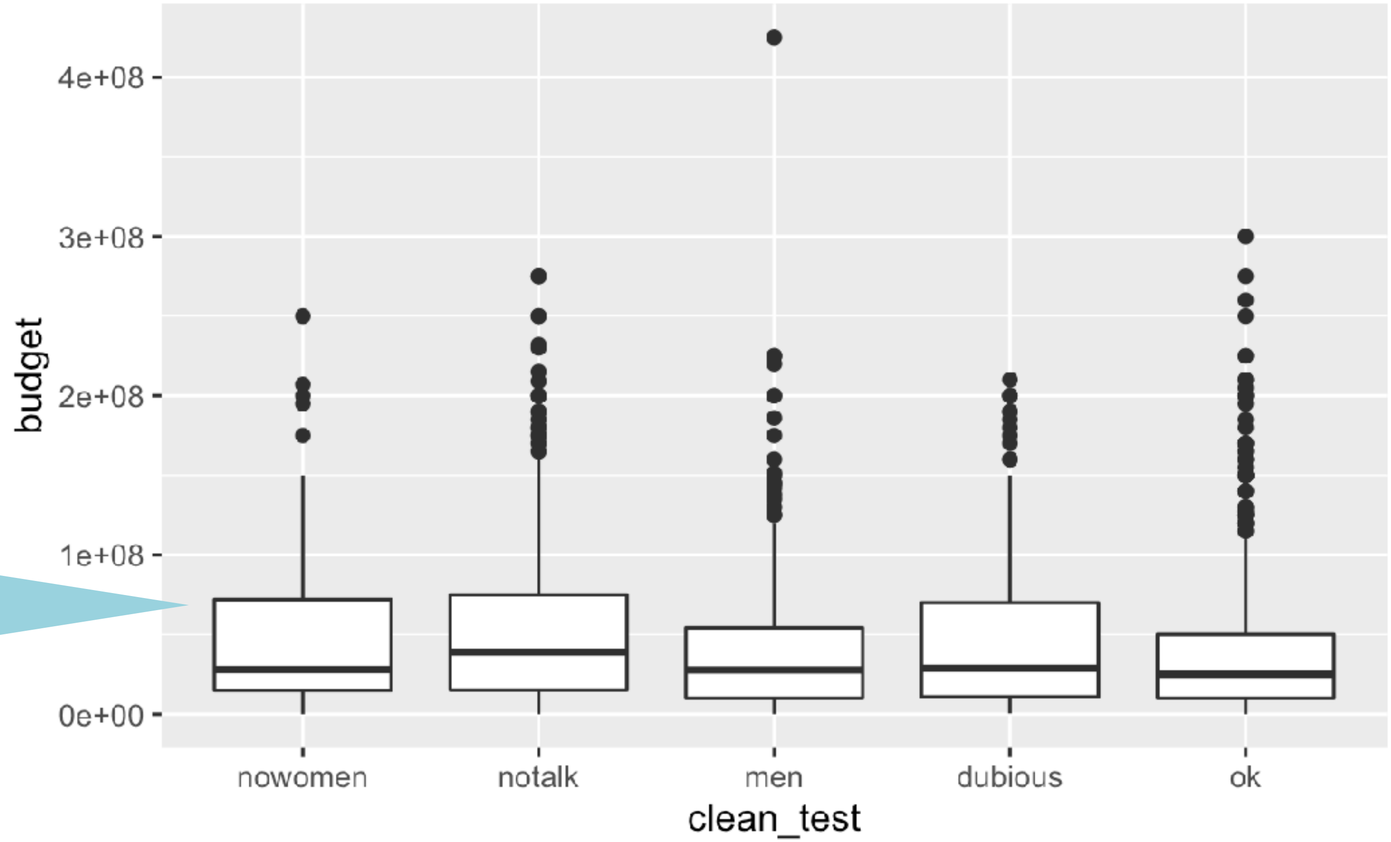


Data can also be set globally and/or locally

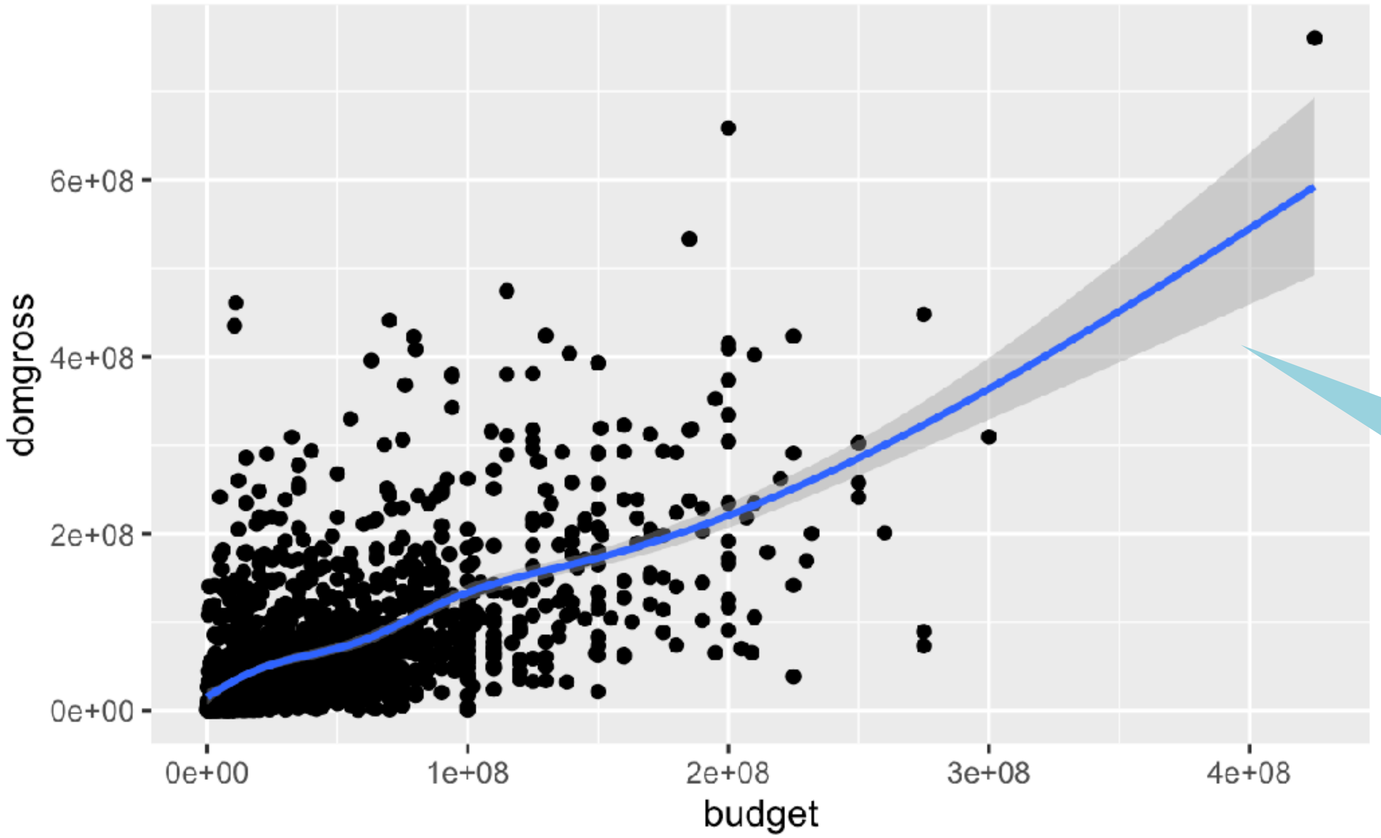
```
ggplot(data = bechdel, mapping = aes(x = budget, y = domgross)) +
  geom_point(mapping = aes(color = clean_test)) +
  geom_smooth(data = filter(bechdel, clean_test == "ok"))
```

Stats

Where do these values come from?

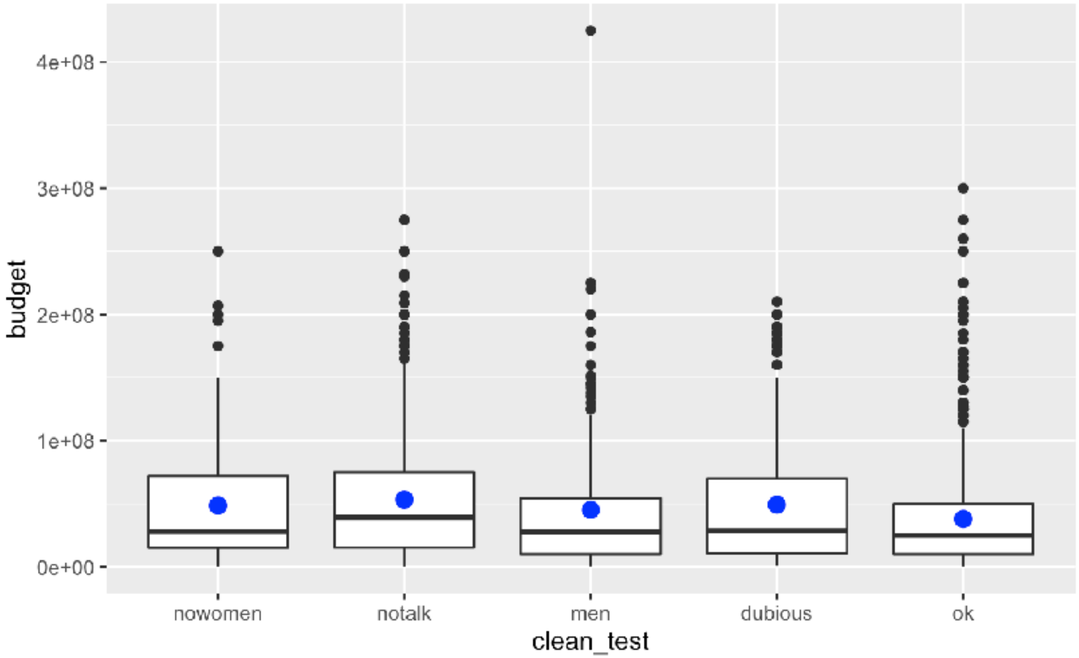



```
ggplot(data = bechdel) +
  geom_boxplot(mapping = aes(x = clean_test, y = budget))
```



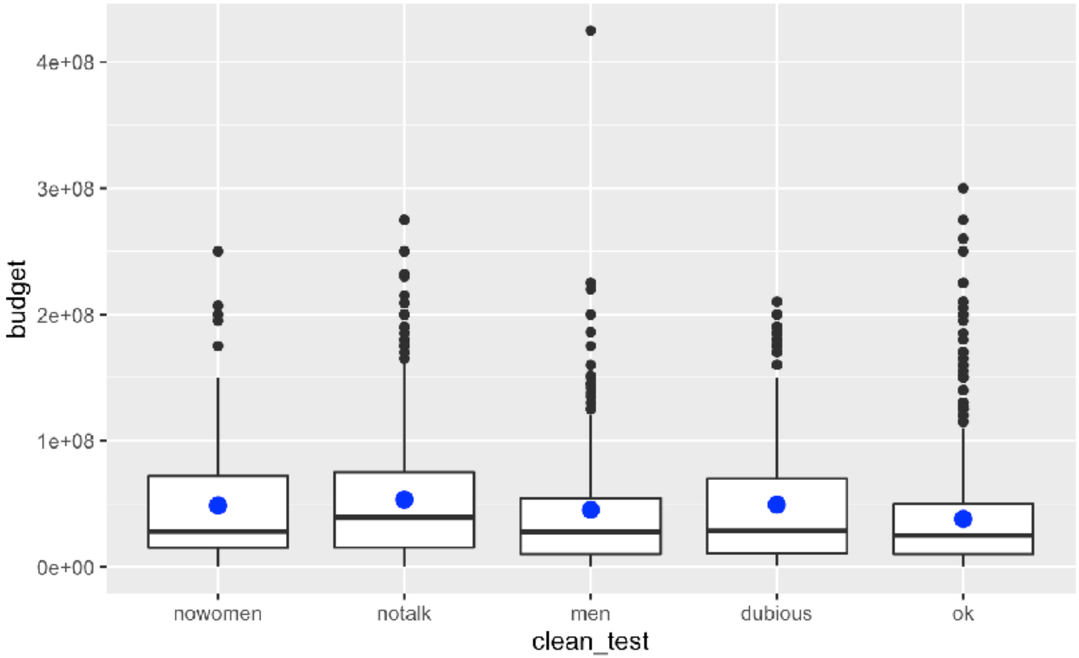

```
ggplot(data = bechdel) +
  geom_point(mapping = aes(x = budget, y = domgross)) +
  geom_smooth(mapping = aes(x = budget, y = domgross))
```

Stats as layers



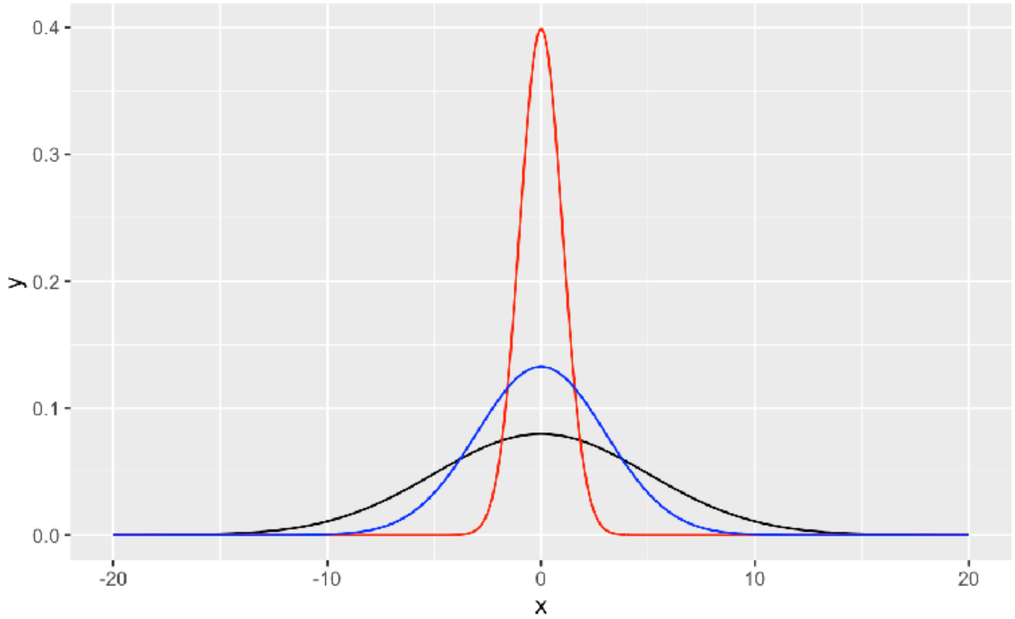

```
ggplot(data = bechdel, mapping = aes(x = clean_test, y = budget)) +
  geom_boxplot() +
  stat_summary(geom = "point", y.fun = "mean", color = "blue", size = 3)
```

Stats as layers



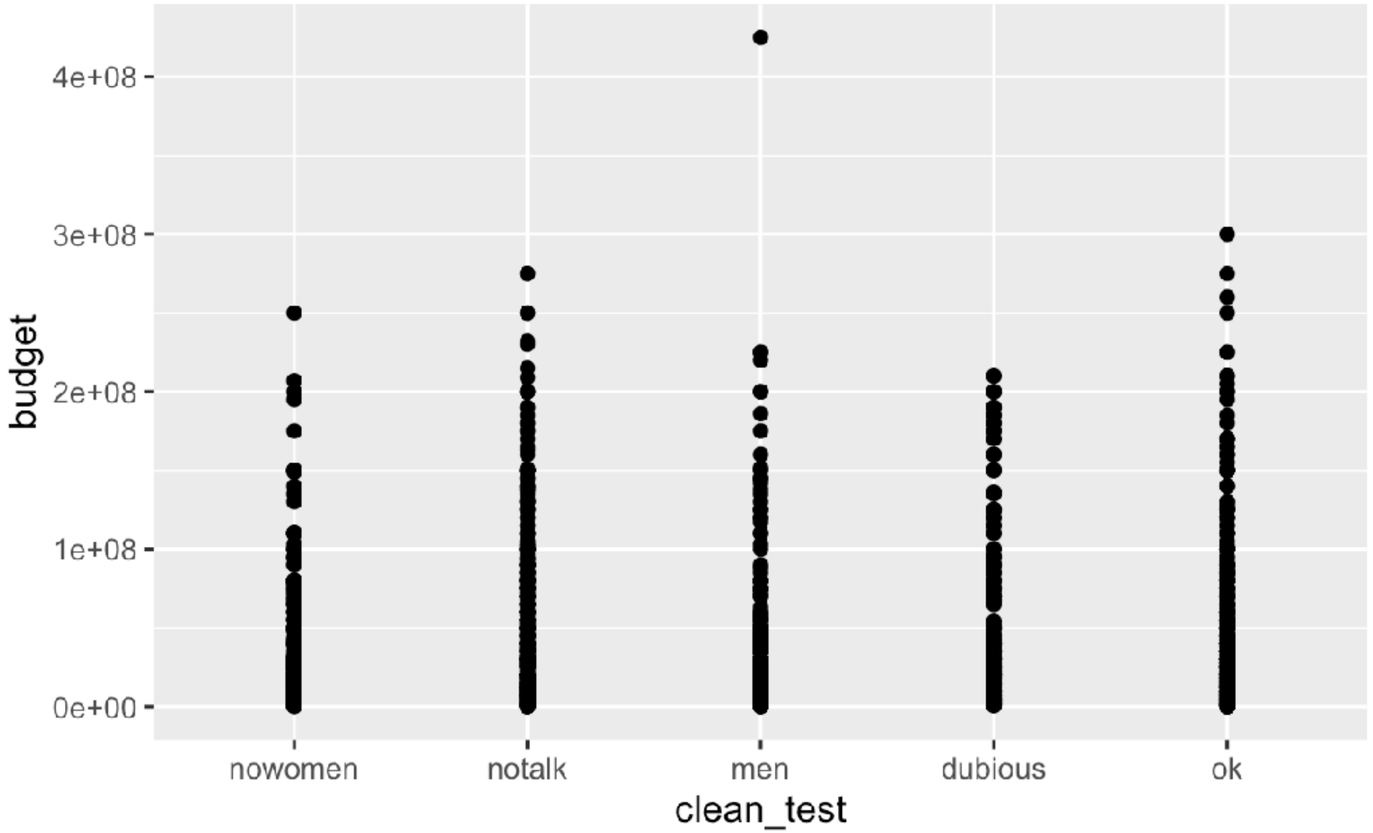

```
ggplot(data = bechdel, mapping = aes(x = clean_test, y = budget)) +
  geom_boxplot() +
  geom_point(stat = "summary", y.fun = "mean", color = "blue", size = 3)
```

Distributions

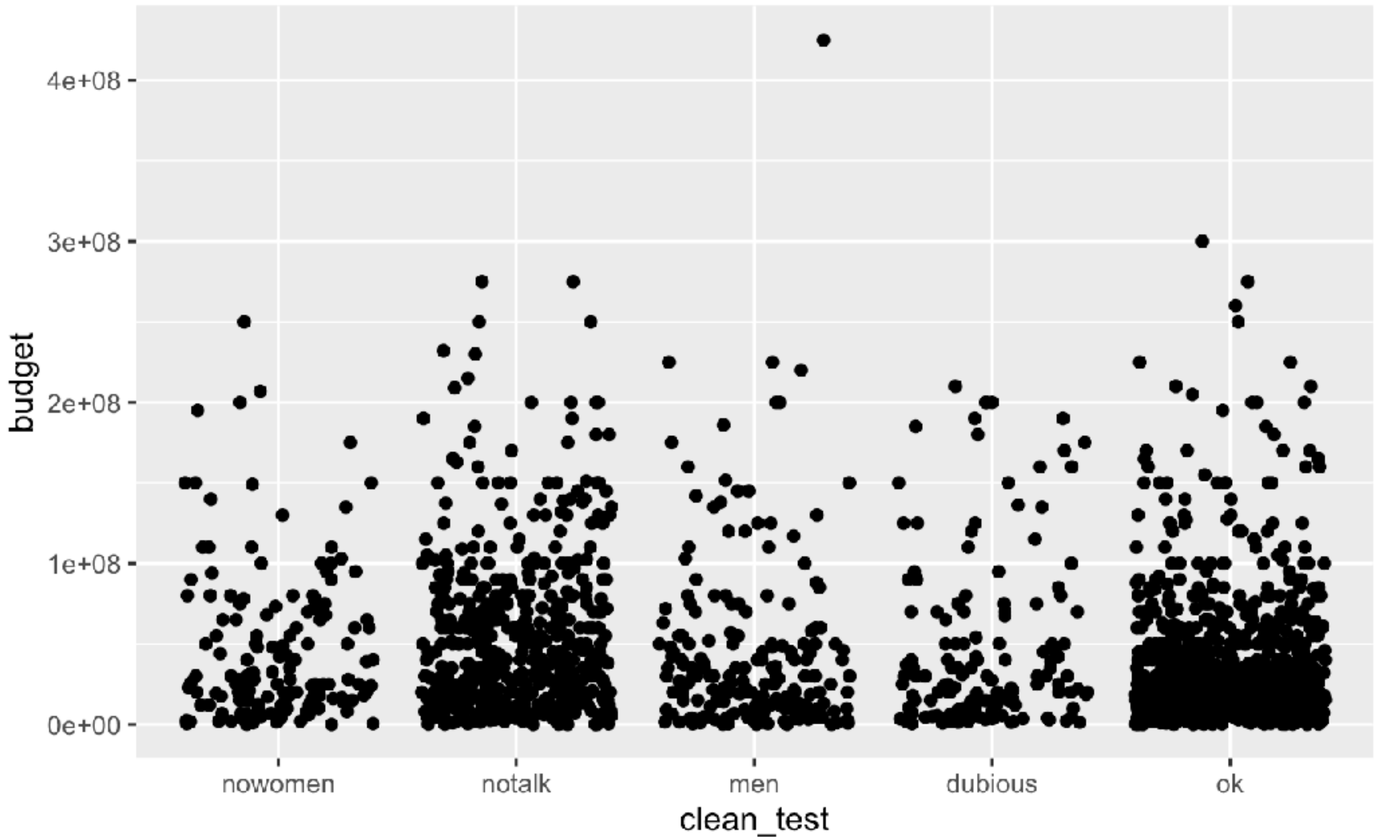



```
ggplot(data = tibble(x = c(-20, 20)), aes(x = x)) +
  stat_function(fun = dnorm, args = list(mean = 0, sd = 5), color = "black") +
  stat_function(fun = dnorm, args = list(mean = 0, sd = 1), color = "red") +
  stat_function(fun = dnorm, args = list(mean = 0, sd = 3), color = "blue")
```

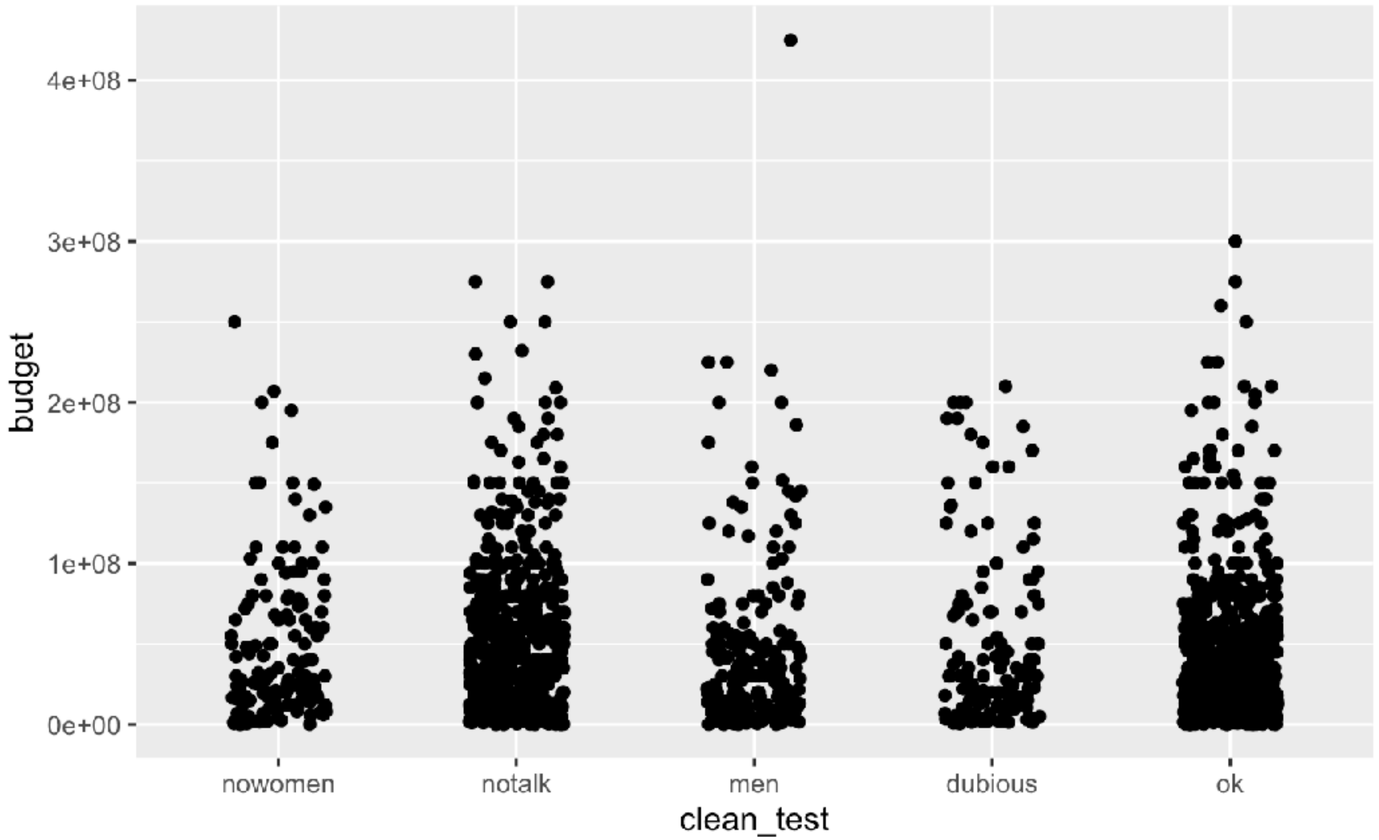
Position




```
ggplot(bechdel, aes(x = clean_test, y = budget)) +
  geom_point()
```



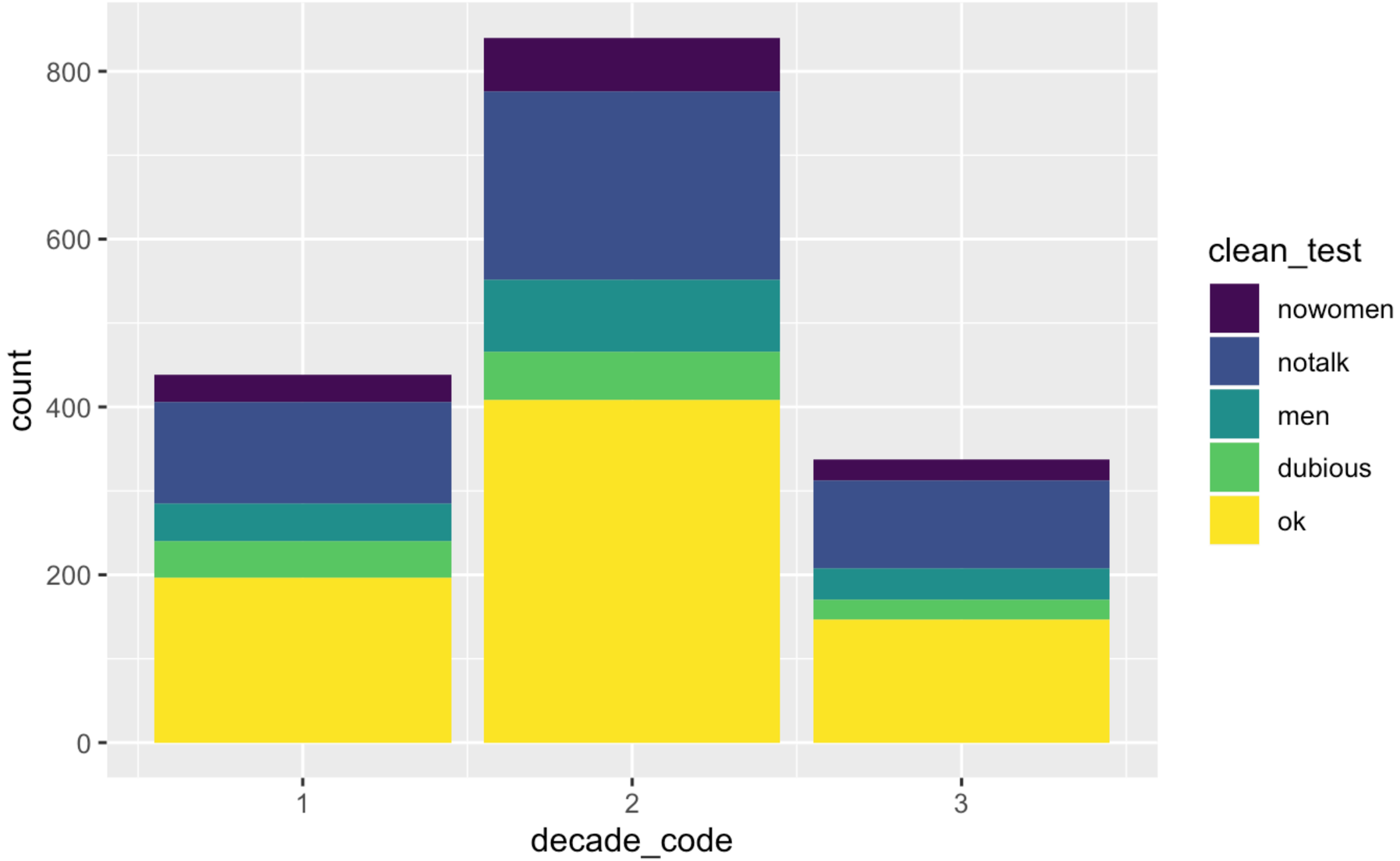

```
ggplot(bechdel, aes(x = clean_test, y = budget)) +
  geom_point(position = "jitter")
```




```
ggplot(bechdel, aes(x = clean_test, y = budget)) +
  geom_point(position = position_jitter(width = 0.2, height = 0))
```

Positions

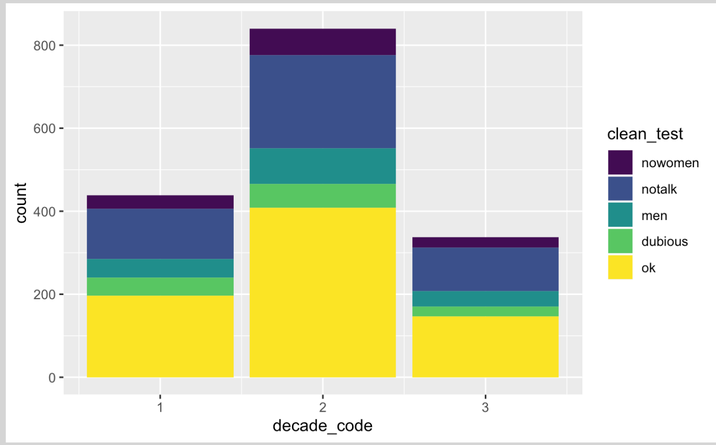
- position_identity()
- position_jitter()
- position_dodge()
- position_fill()
- And more...


800
600
400
200
0
count
1
2
3
decade_code
clean_test
nowomen
notalk
men
dubious
ok

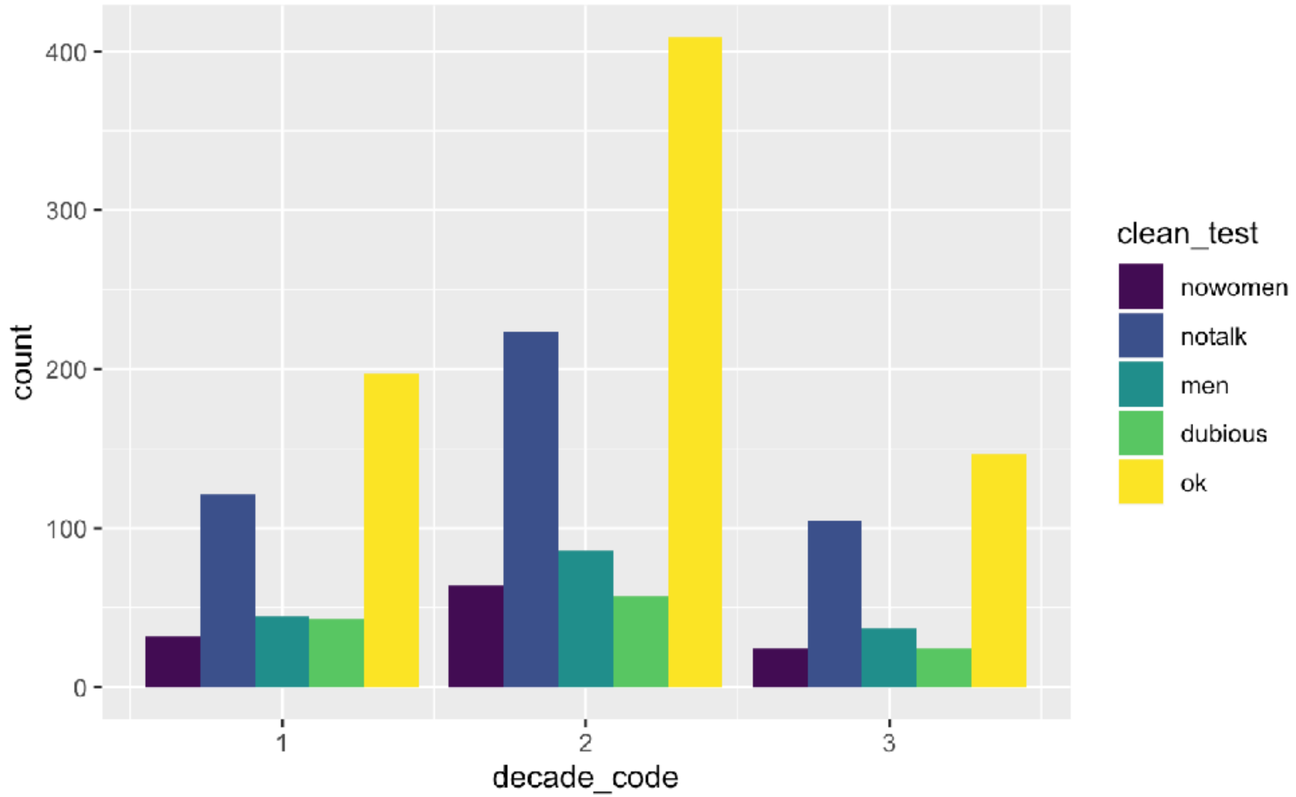
Your Turn 8

- Add a position adjustment to this plot to compare the frequency of test results across decades.

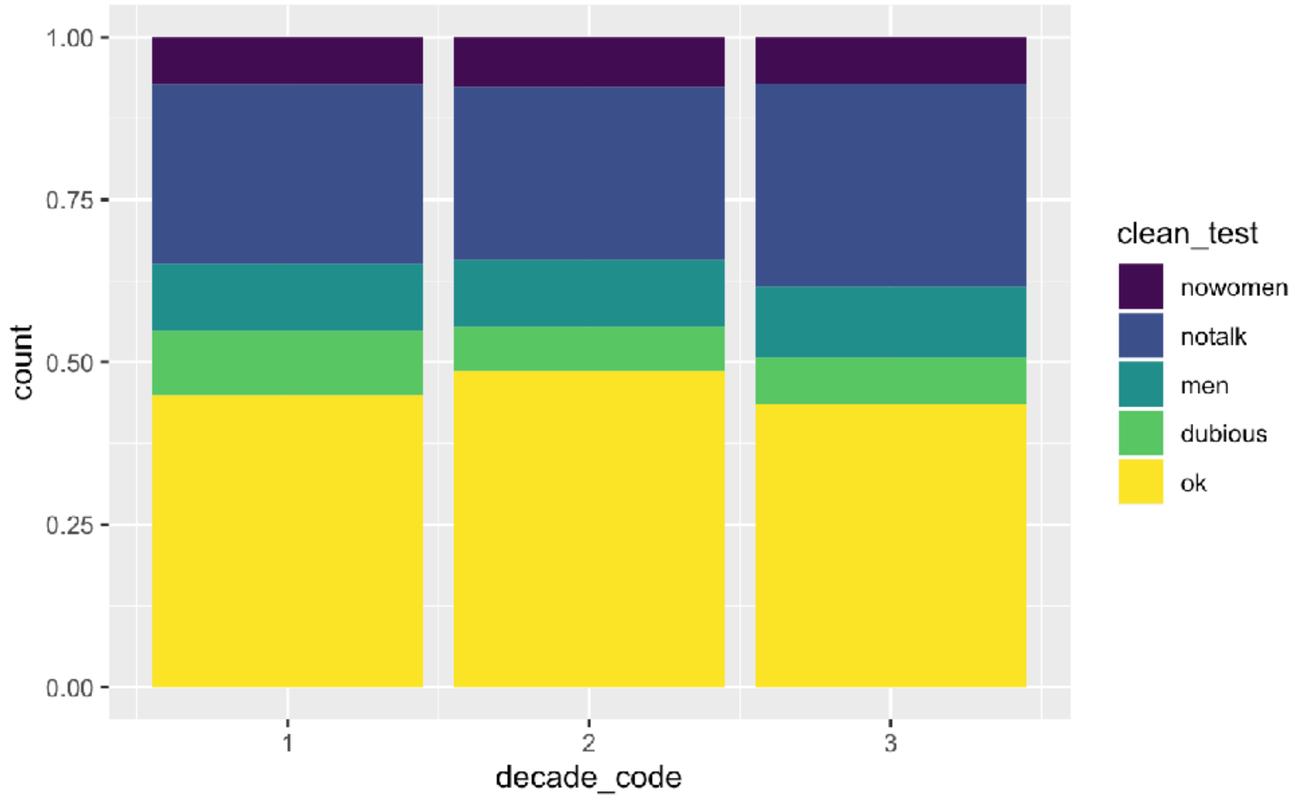
```
ggplot(data = bechdel, mapping = aes(x = decade_code)) +
  geom_bar(mapping = aes(fill = clean_test))
```



05 : 00

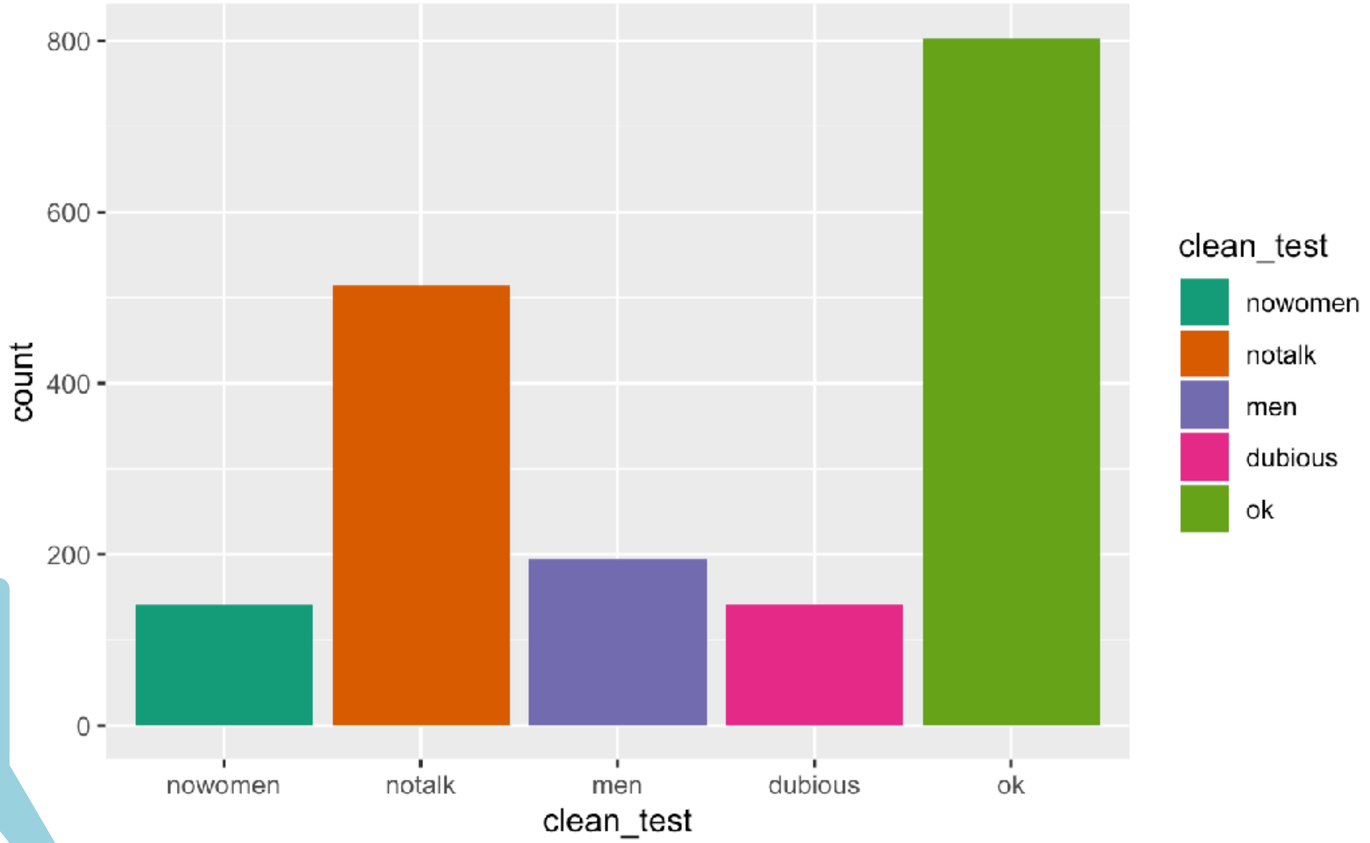



```
ggplot(bechdel, aes(x = decade_code)) +
  geom_bar(aes(fill = clean_test), position = "dodge")
```




```
ggplot(bechdel, aes(x = decade_code)) +
  geom_bar(aes(fill = clean_test), position = "fill")
```

Scales



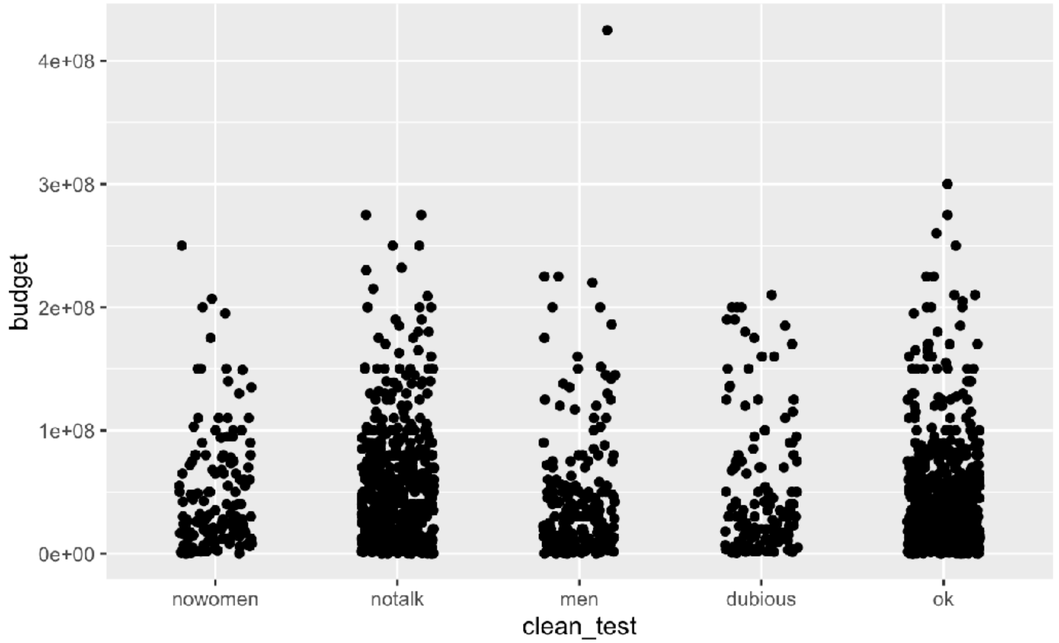
colorbrewer2.org

```
ggplot(data = bechdel) +
  geom_bar(mapping = aes(x = clean_test, fill = clean_test)) +
  scale_fill_brewer(palette = "Dark2")
```

Aesthetic scales

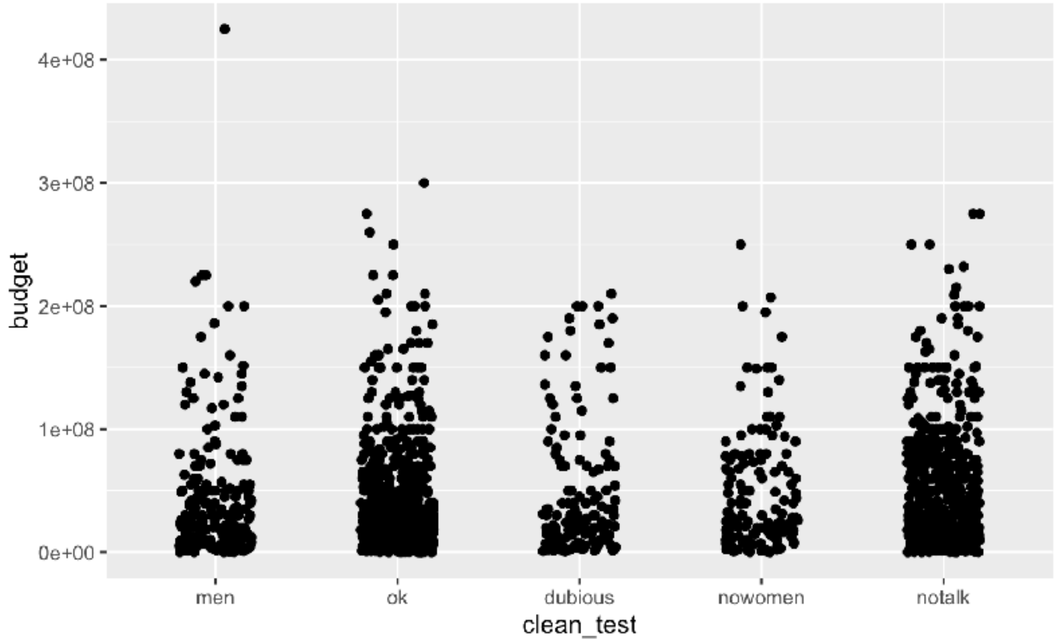
- scale_*_continuous()
- scale_*_discrete()
- scale_*_ordinal()
- scale_*_manual()
- scale_{color/fill}_brewer()
- scale_{color/fill}_distiller()
- scale_{color/fill}_gradient()

Coordinate scales



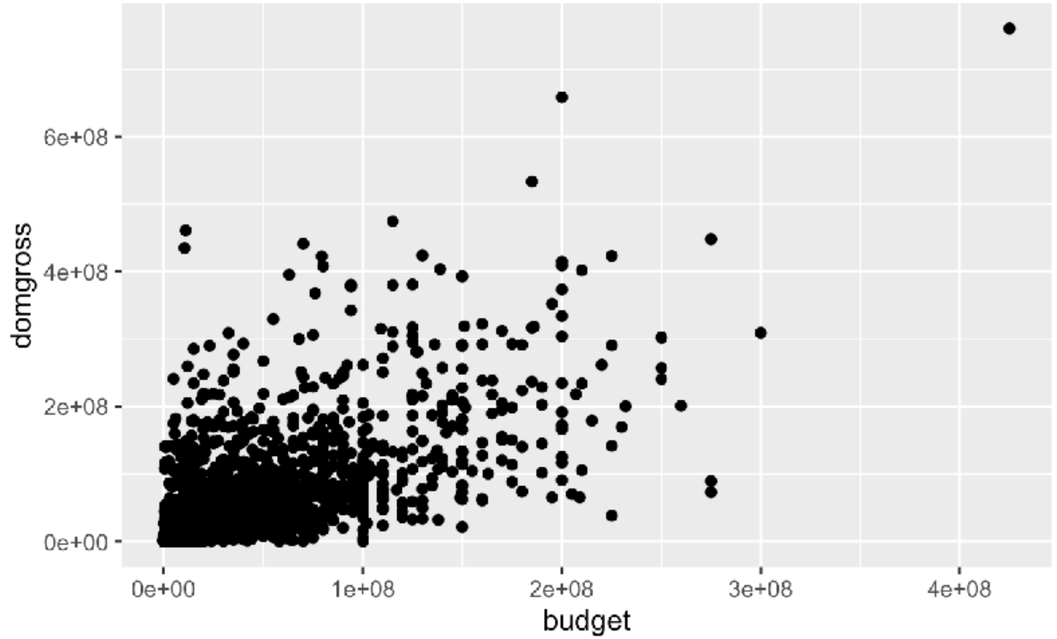

```
ggplot(bechdel, aes(x = clean_test, y = budget)) +
  geom_point(position = position_jitter(width = 0.2, height = 0))
```

Coordinate scales



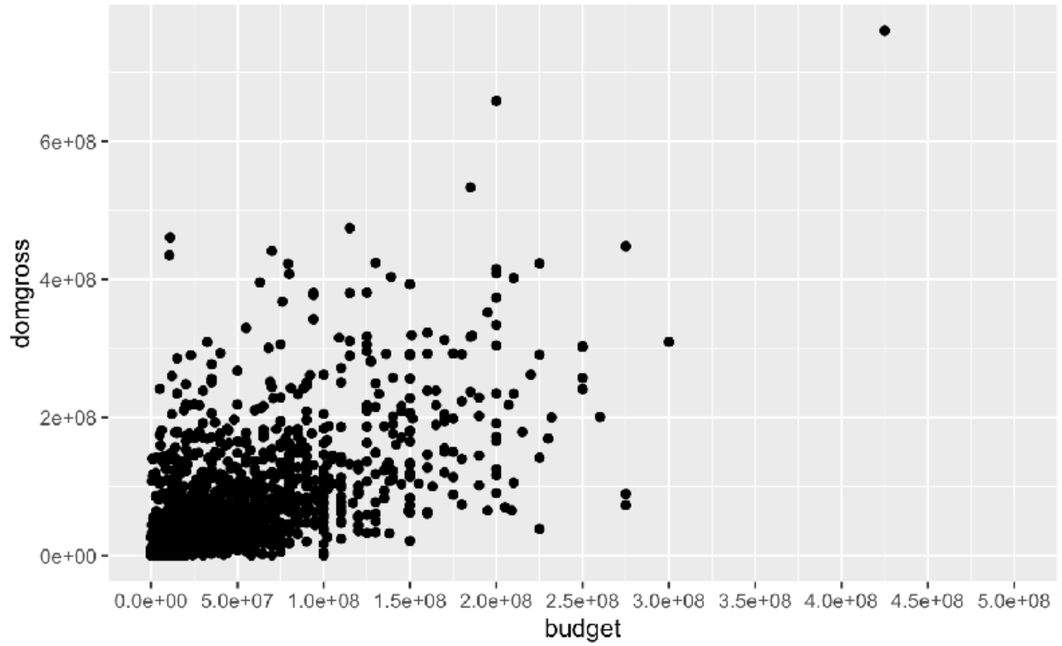

```
ggplot(bechdel, aes(x = clean_test, y = budget)) +
  geom_point(position = position_jitter(width = 0.2, height = 0)) +
  scale_x_discrete(limits = c("men", "ok", "dubious", "nowomen", "notalk"))
```

Coordinate scales



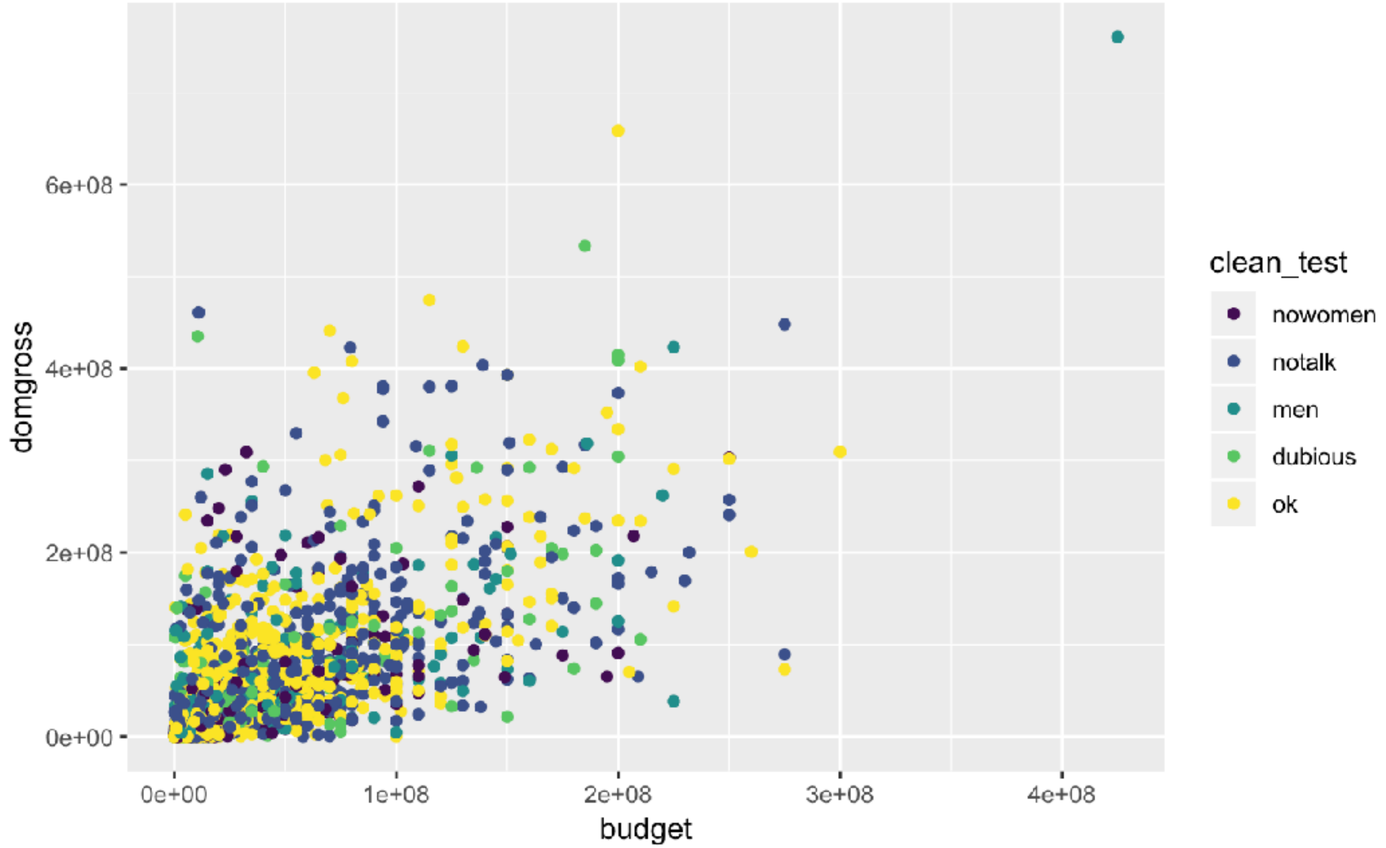

```
ggplot(bechdel, mapping = aes(x = budget, y = domgross)) +
  geom_point()
```

Coordinate scales

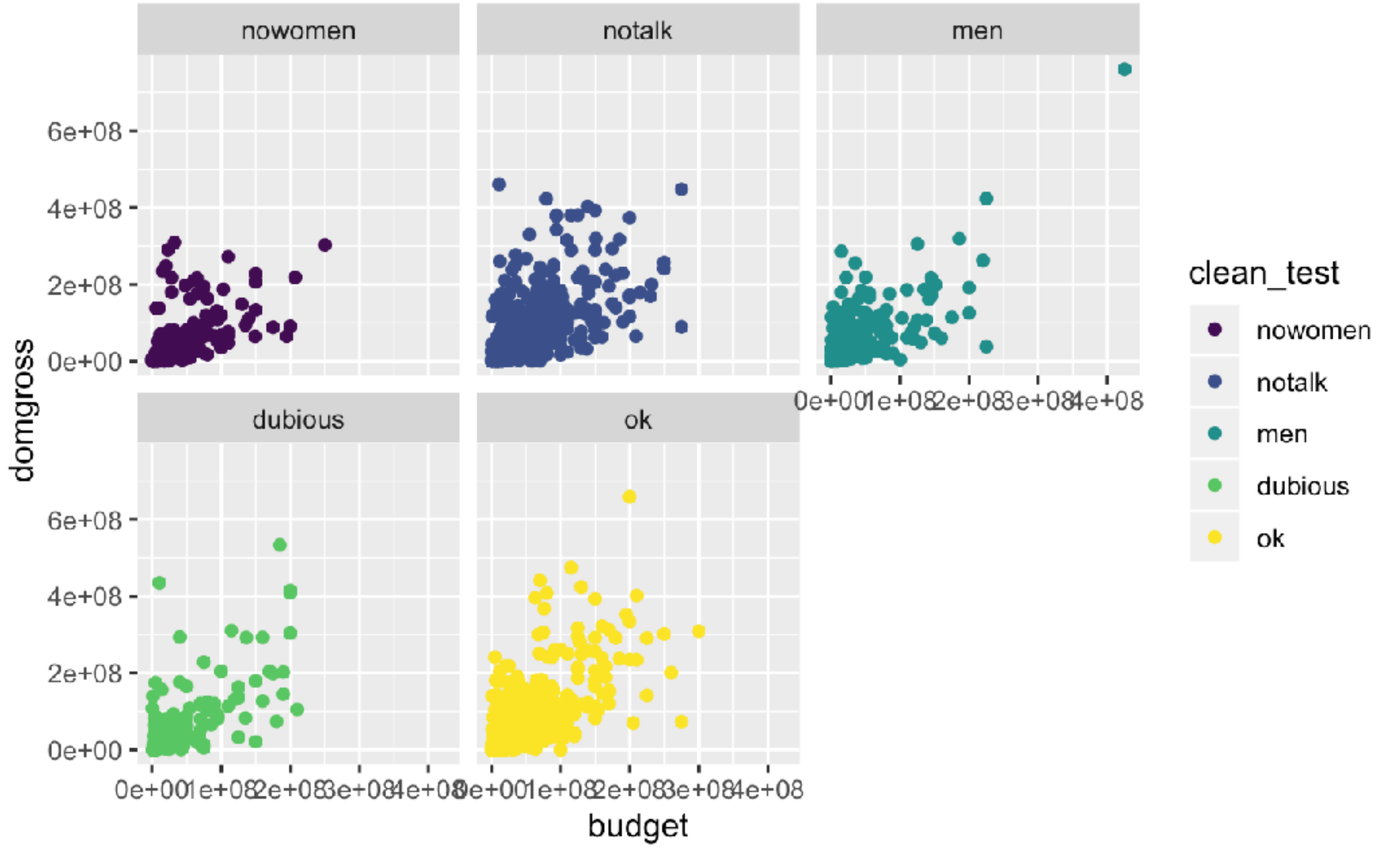



```
ggplot(bechdel, mapping = aes(x = budget, y = domgross)) +
  geom_point() +
  scale_x_continuous(limits = c(0, 5e+08), breaks = seq(0, 5e+08, 5e+07))
```

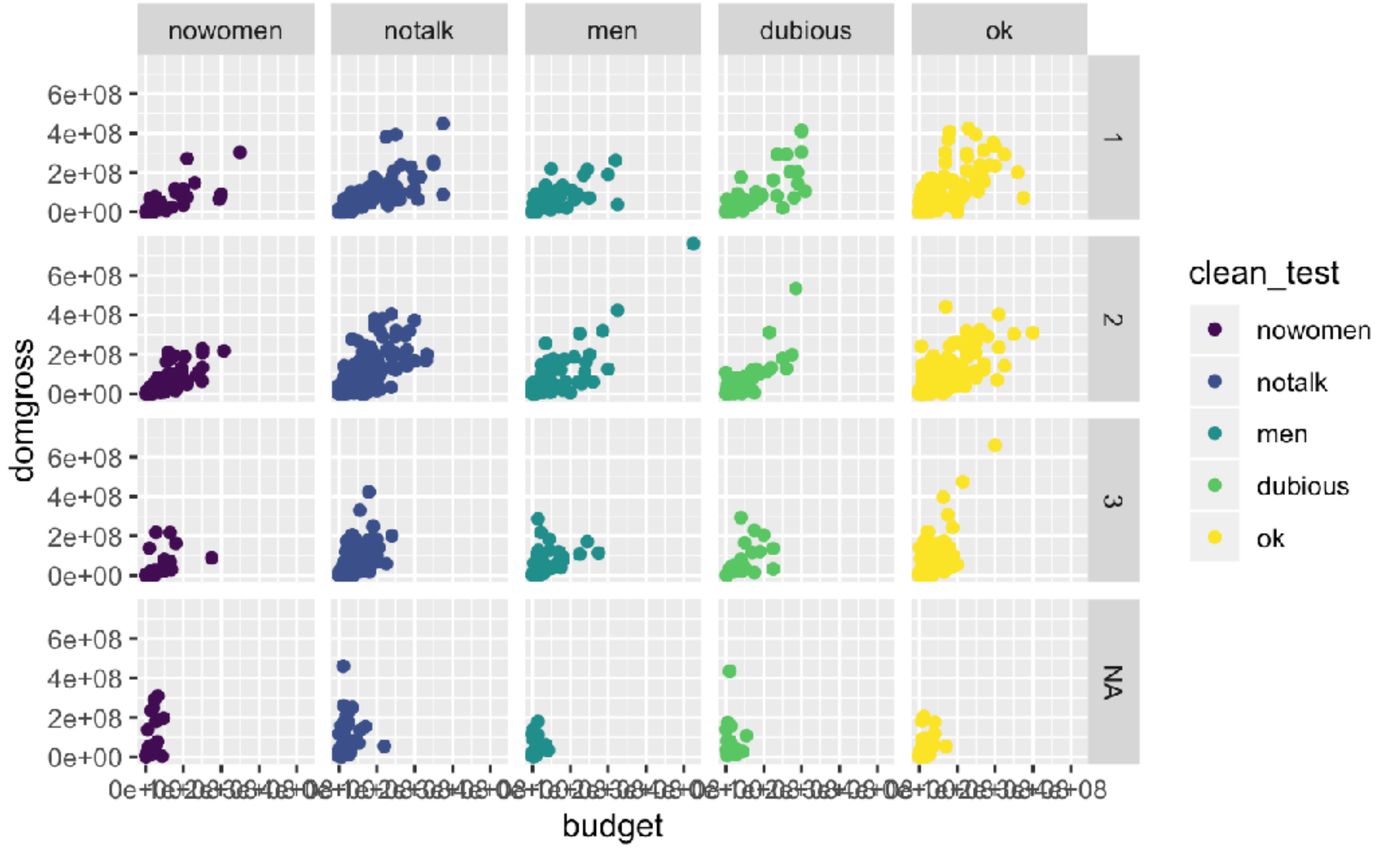
Facets




```
ggplot(data = bechdel) +
  geom_point(mapping = aes(x = budget, y = domgross, color = clean_test))
```

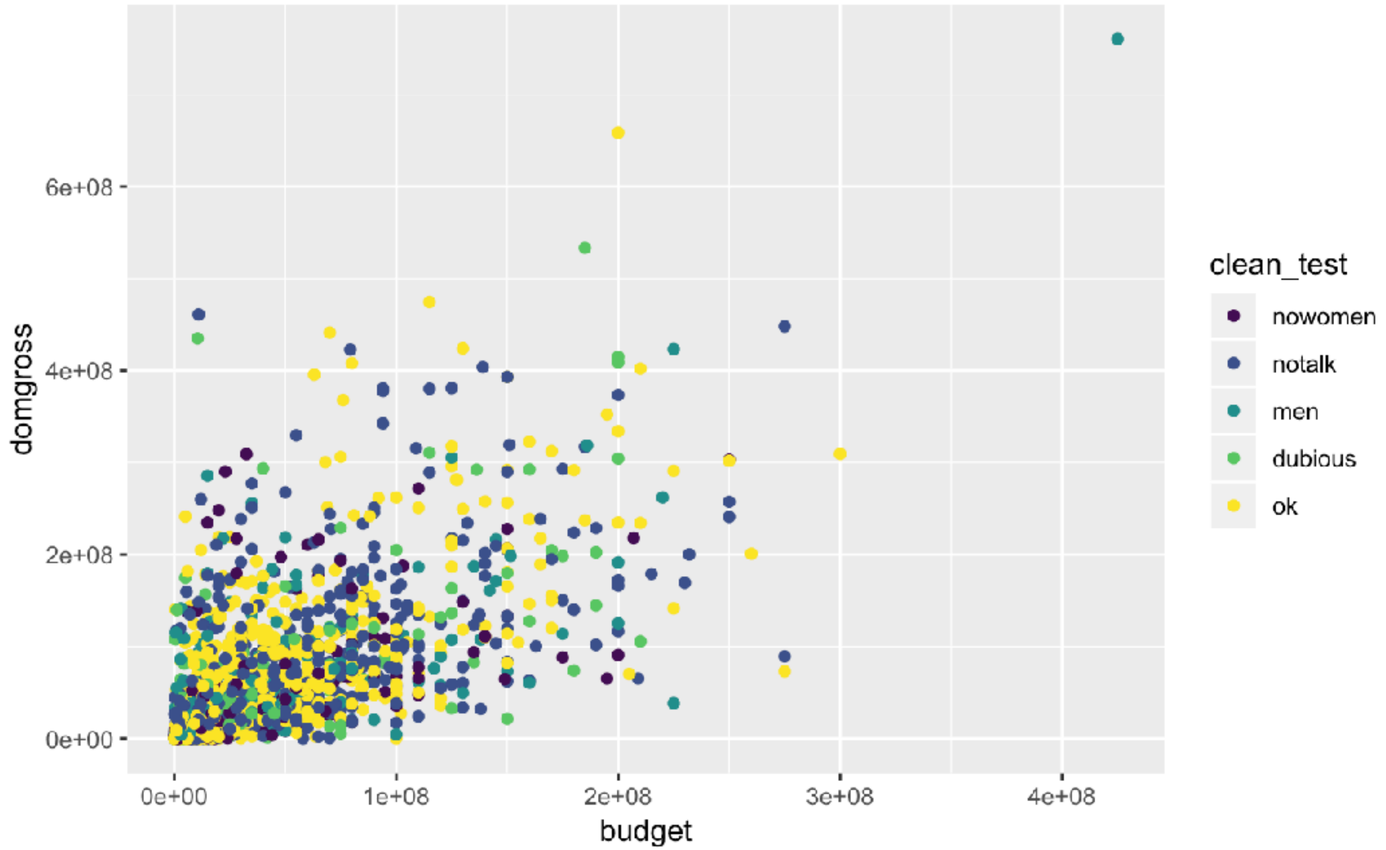



```
ggplot(data = bechdel) +
  geom_point(mapping = aes(x = budget, y = domgross, color = clean_test)) +
  facet_wrap(~ clean_test)
```

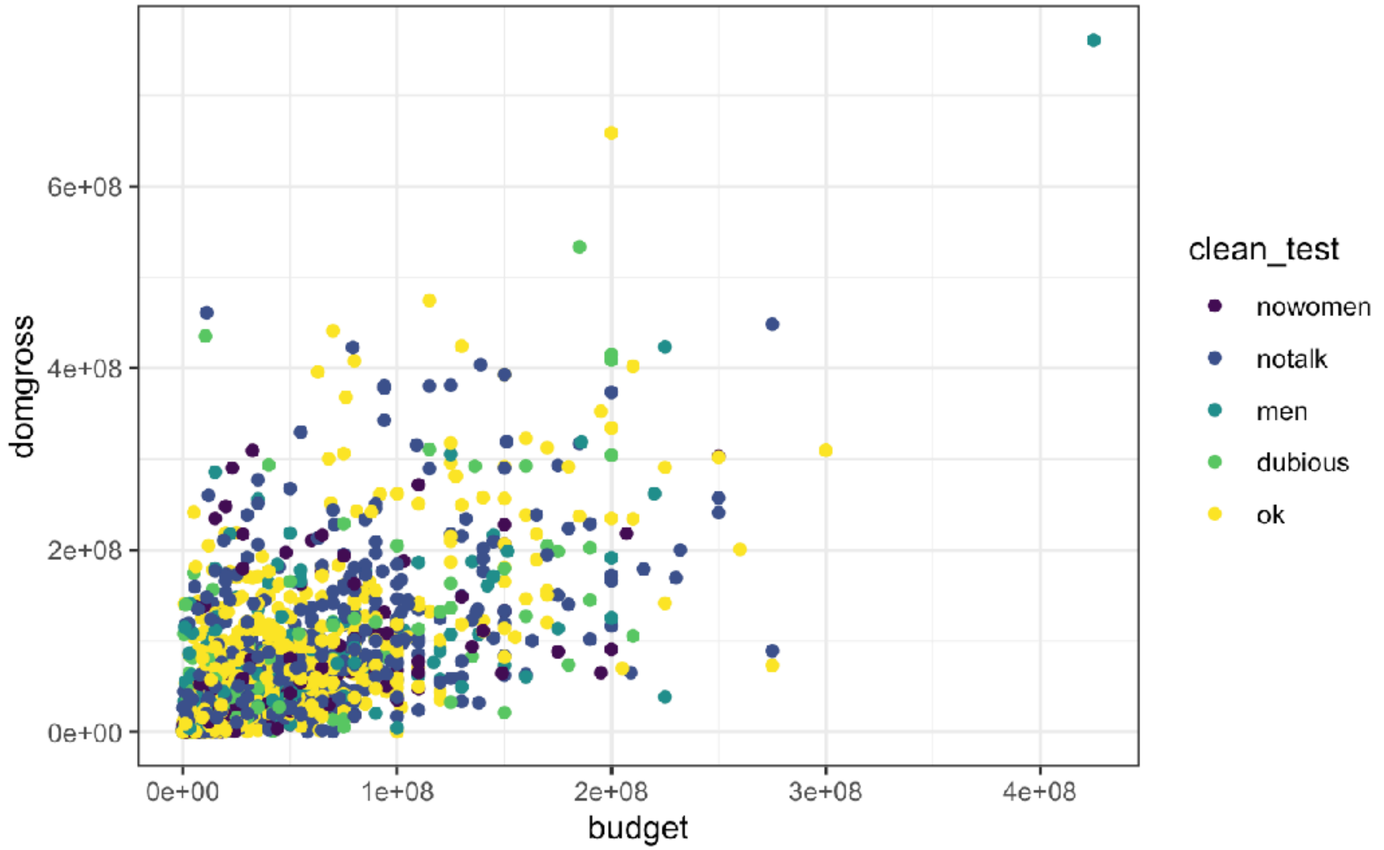



```
ggplot(data = bechdel) +
  geom_point(mapping = aes(x = budget, y = domgross, color = clean_test)) +
  facet_grid(decade_code ~ clean_test)
```

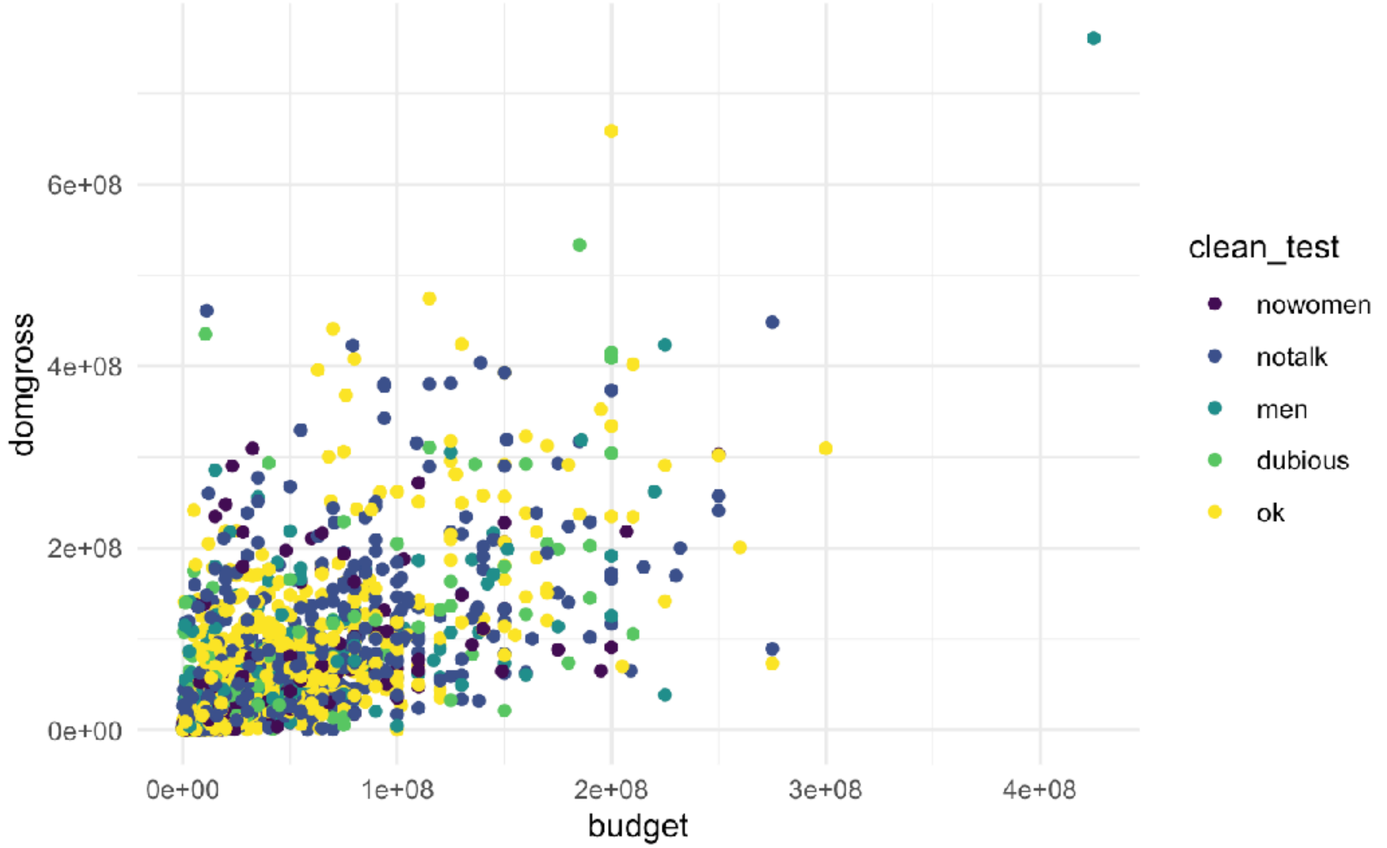
Themes



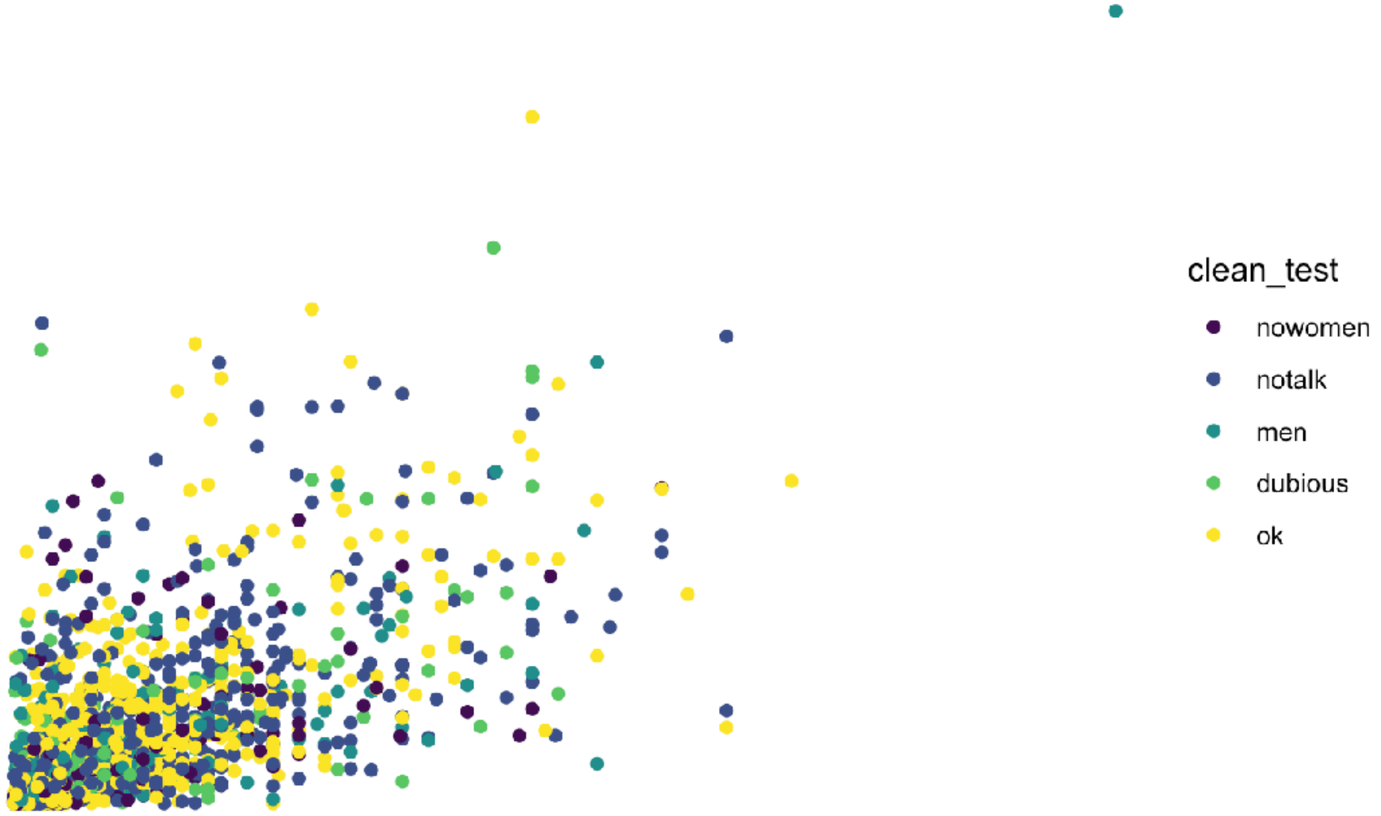

```
ggplot(data = bechdel) +
  geom_point(mapping = aes(x = budget, y = domgross, color = clean_test))
```




```
ggplot(data = bechdel) +
  geom_point(mapping = aes(x = budget, y = domgross, color = clean_test)) +
  theme_bw()
```

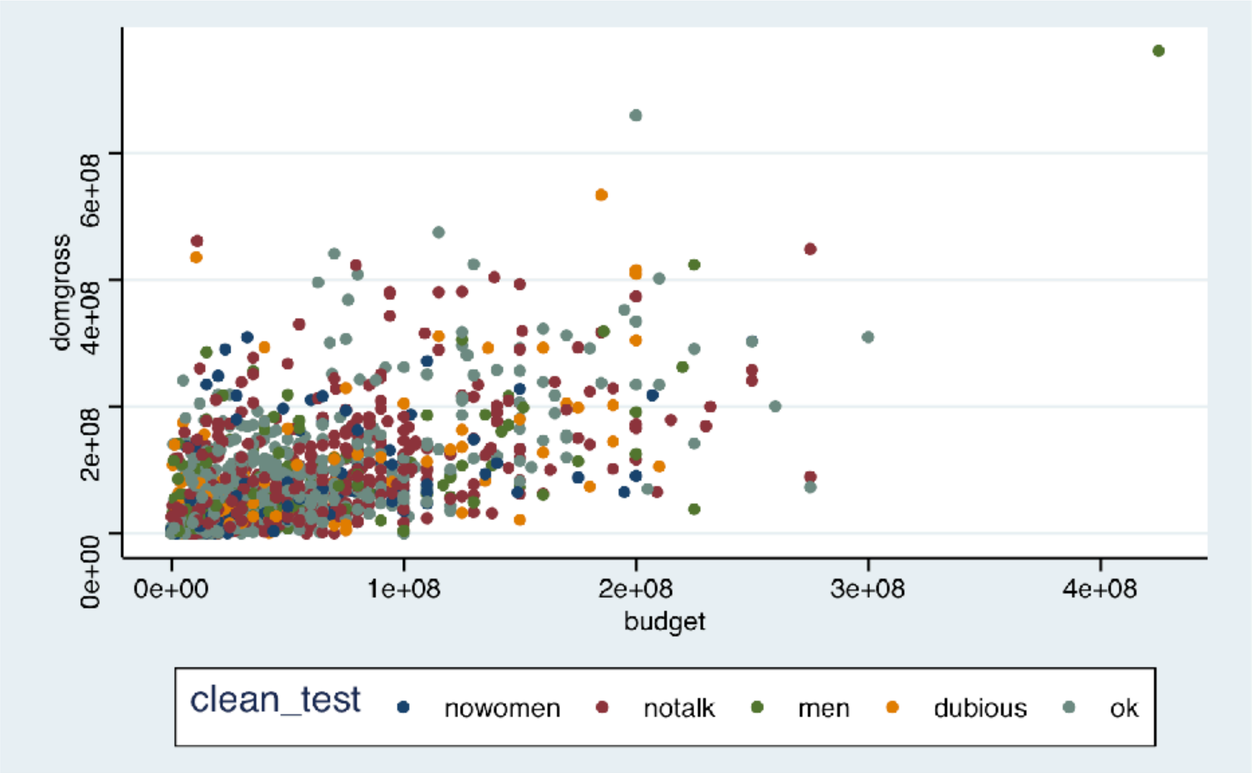



```
ggplot(data = bechdel) +
  geom_point(mapping = aes(x = budget, y = domgross, color = clean_test)) +
  theme_minimal()
```




```
ggplot(data = bechdel) +
  geom_point(mapping = aes(x = budget, y = domgross, color = clean_test)) +
  theme_void()
```

library(ggthemes)




```
ggplot(data = bechdel) +
  geom_point(mapping = aes(x = budget, y = domgross, color = clean_test)) +
  scale_color_stata() +
  theme_stata()
```

ggplot2 template

Required

```
ggplot(data = <DATA>) +
  <GEOM_FUNCTION>(mapping = aes(<MAPPINGS>),
                  stat = <STAT>, position = <POSITION>) +
  <FACET_FUNCTION> +
  <SCALE_FUNCITON> +
  <THEME_FUNCTION>
```

Optional

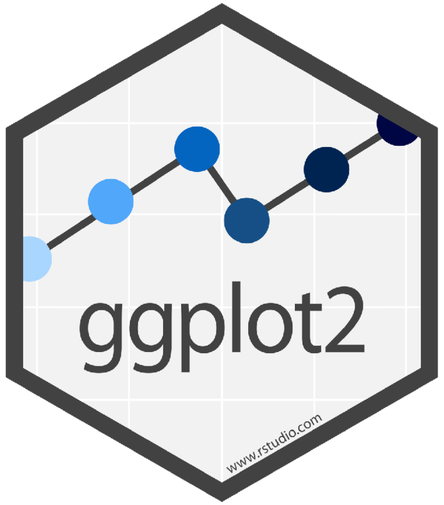
Saving plots

```
color_plot <- ggplot(data = bechdel) +
  geom_point(mapping = aes(x = budget, y = domgross, color = clean_test))

ggsave("filename.png", plot = color_plot, width = 8, height = 6,
       units = "in", dpi = "retina")
```

Will save the last plot created if not specified

Data Visualization



ATLAS
Accessible Teaching, Learning, and Assessment Systems

wjakethompson.com wjakethompson@ku.edu @wjakethompson @wjakethompson