

Week-7: Visualizing data using `ggplot`

NM2207: Computational Media Literacy

Narayani Vedam, Ph.D.

Department of Communications and New Media



**Faculty of Arts
& Social Sciences**

This week

Table of contents

I. All about `ggplot2` (click here)

II. Visualizing numeric variables (click here)

III. Visualizing categoric variables (click here)

IV. Visualizing variables of varied types (click here)

I. All about `ggplot2` package

ggplot2

- It is the `tidyverse` library's visualization package
- `gg` in `ggplot` stands for **G**rammar of **G**raphics
- Enables us to describe different components of a graphic
- Structure of the code for plots can be summarized as

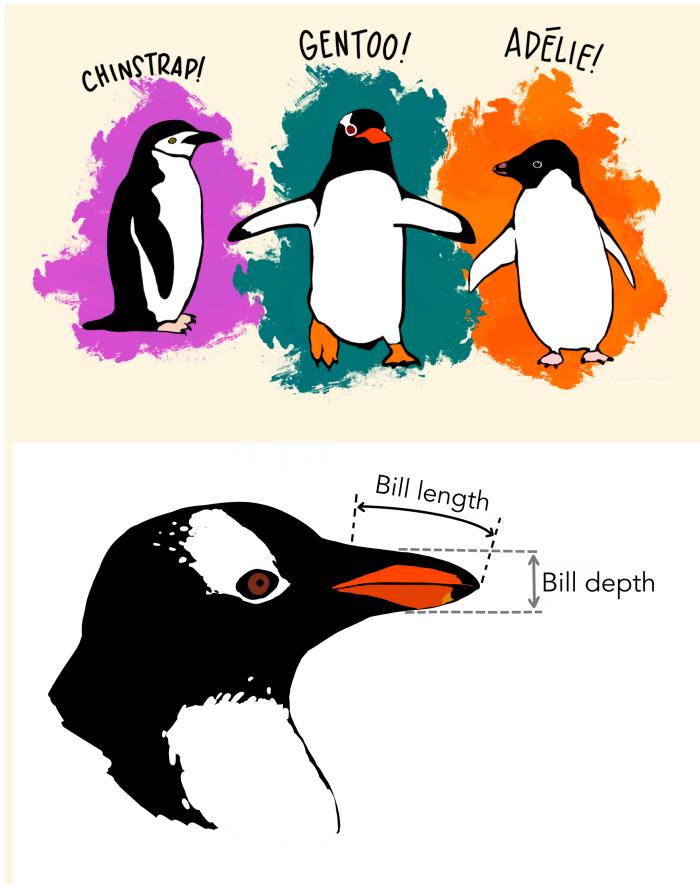
```
ggplot(data = [dataset],
       mapping = aes(x = [x-variable],
                     y = [y-variable])) +
  geom_xxx() +
  other options
```



Figure: Different layers of ggplot

Data: Palmer Penguins

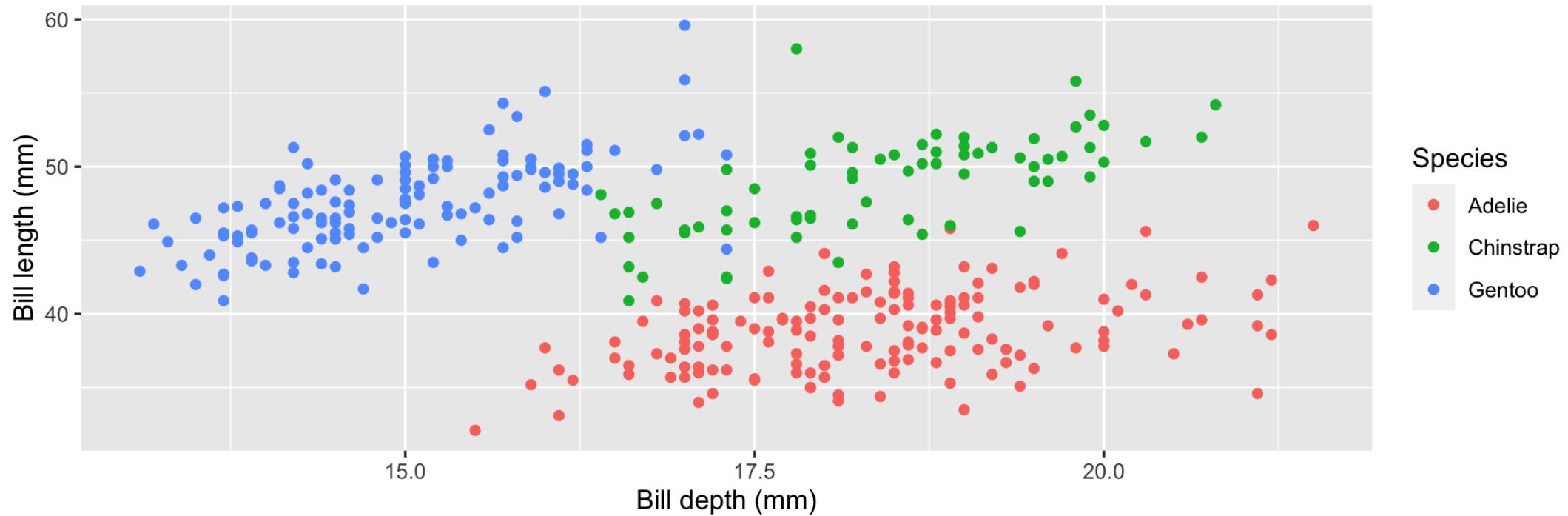
Measurements for penguin species, island in Palmer Archipelago, size (flipper length, body mass, bill dimensions), and gender.



```
library(tidyverse)
library(palmerpenguins)
glimpse(penguins) penguins dataset is part of palmer penguins data set
```

Plots: Palmer Penguins

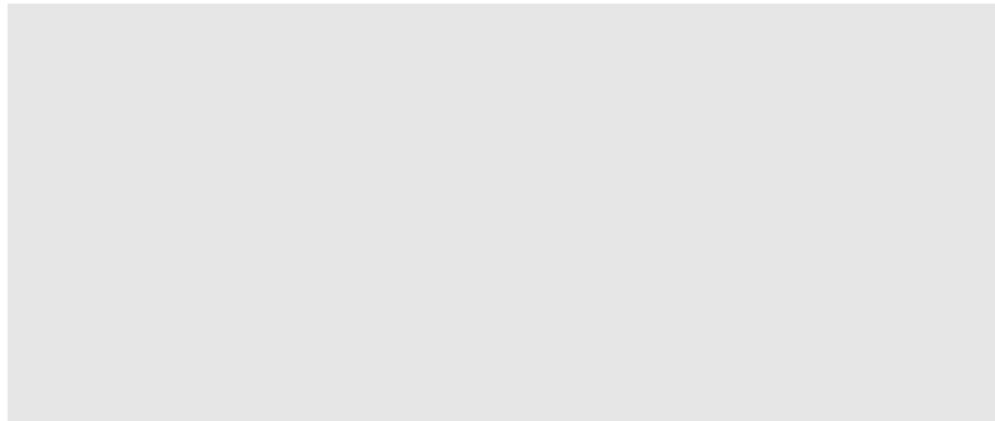
Bill depth and length
Dimensions for Adelie, Chinstrap, and Gentoo Penguins



Palmer Penguins: Plot recreation

a. Start with the `penguins` data frame

```
ggplot(data = penguins)      passing penguin dataset
```

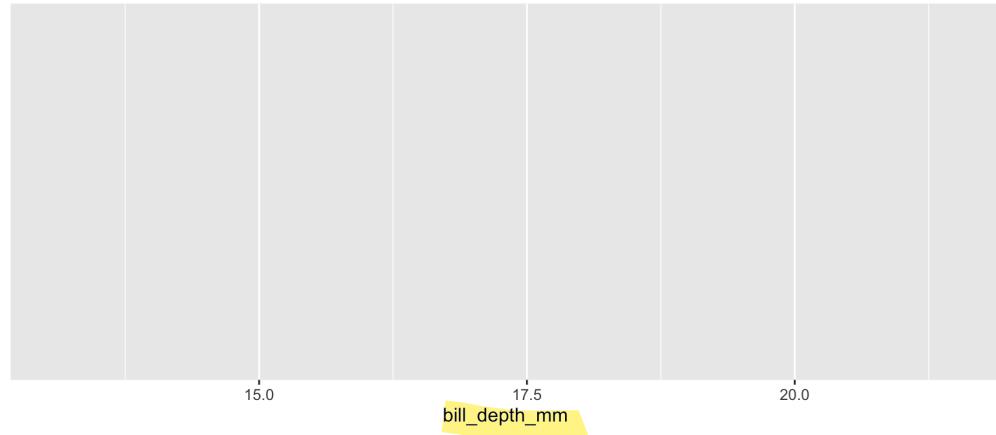


Palmer Penguins: Plot recreation

b. Map **bill depth** to the x-axis

```
ggplot(data = penguins,  
       mapping = aes(x = bill_depth_mm))
```

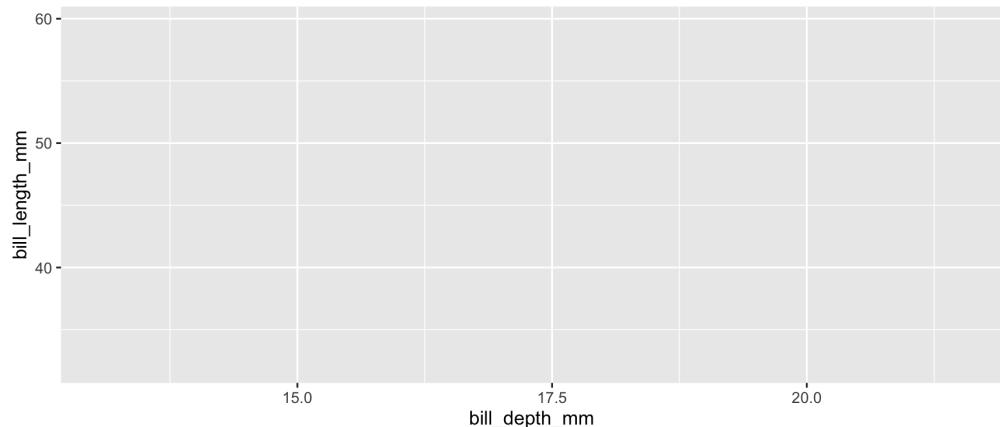
name of var has to be = to that of dataset



Palmer Penguins: Plot recreation

c. Map **bill length** to the y-axis

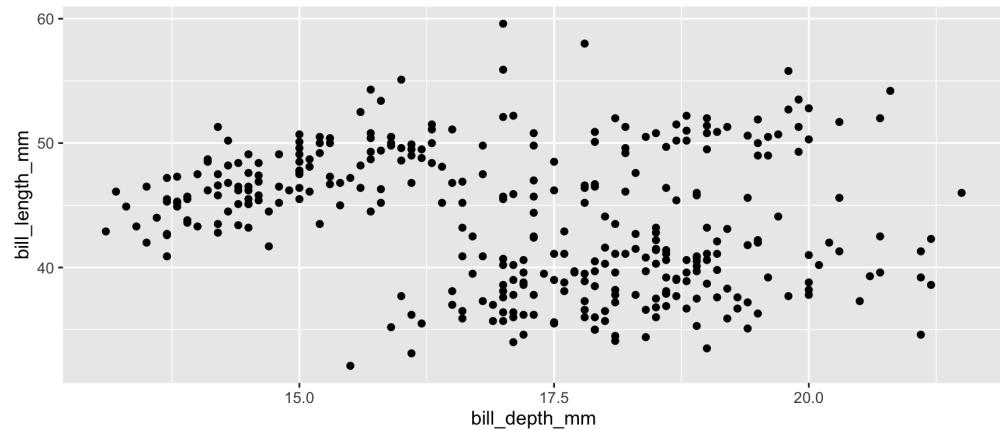
```
ggplot(data = penguins,  
       mapping = aes(x = bill_depth_mm,  
                      y = bill_length_mm))
```



Palmer Penguins: Plot recreation

d. Represent each observation with a point

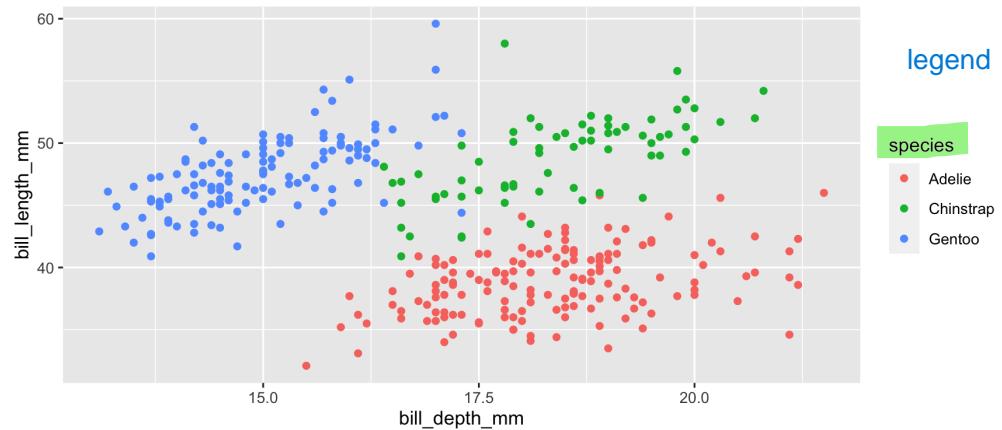
```
ggplot(data = penguins,  
       mapping = aes(x = bill_depth_mm,  
                      y = bill_length_mm)) +  
  geom_point()
```



Palmer Penguins: Plot recreation

e. Map species to the colour of each point

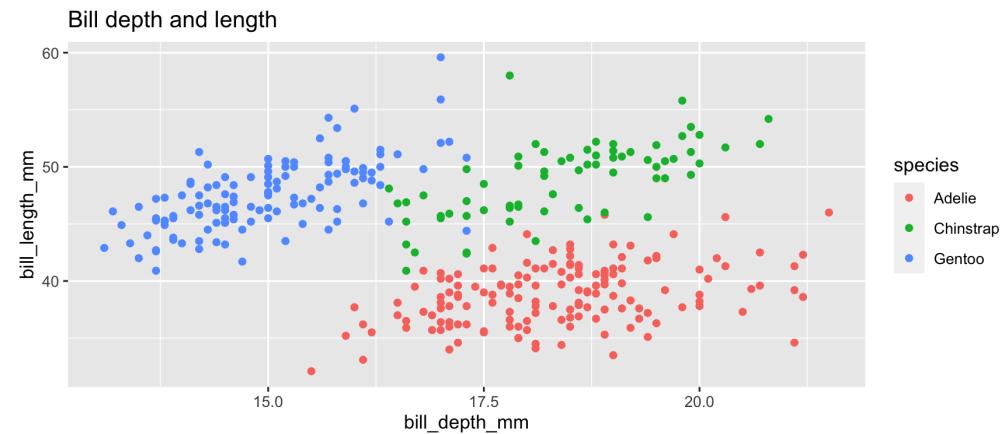
```
ggplot(data = penguins,
       mapping = aes(x = bill_depth_mm,
                      y = bill_length_mm,
                      colour = species)) +
  geom_point()
```



Palmer Penguins: Plot recreation

f. Title the plot "Bill depth and length"

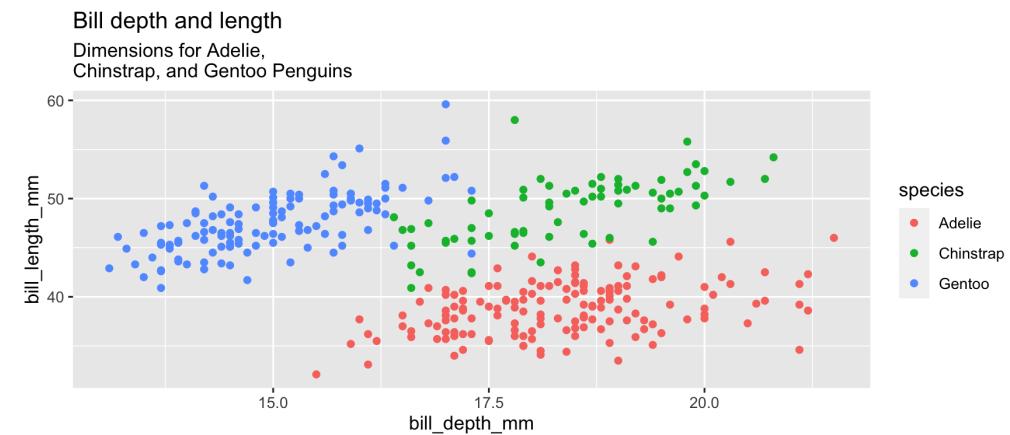
```
ggplot(data = penguins,
       mapping = aes(x = bill_depth_mm,
                      y = bill_length_mm,
                      colour = species)) +
  geom_point() +
  labs(title = "Bill depth and length")
```



Palmer Penguins: Plot recreation

g. Add the subtitle "Dimensions for Adelie, Chinstrap, and Gentoo Penguins"

```
ggplot(data = penguins,
       mapping = aes(x = bill_depth_mm,
                      y = bill_length_mm,
                      colour = species)) +
  geom_point() +
  labs(title = "Bill depth and length",
       subtitle = "Dimensions for Adelie,
                  Chinstrap, and Gentoo Penguins")
```



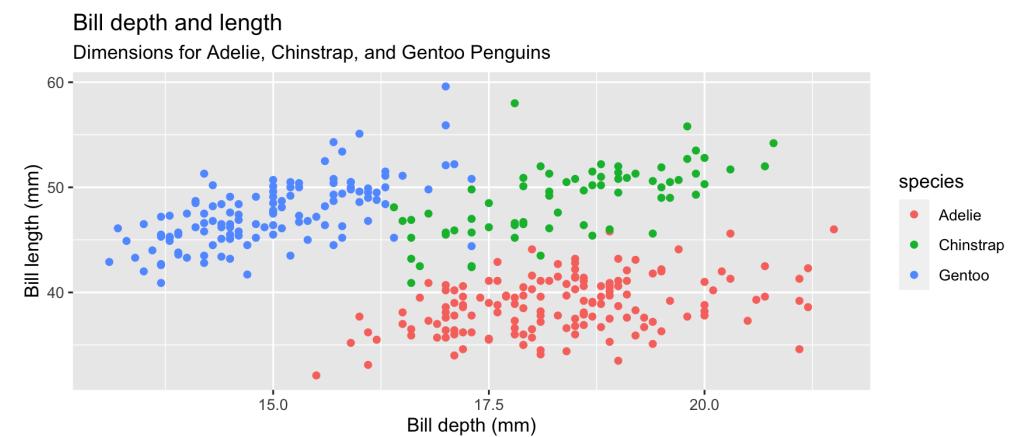
Precise title
descriptive subtitle

Palmer Penguins: Plot recreation

h. Label the x and y axes as "Bill depth (mm)" and "Bill length (mm)", respectively

dw default labelling -> change via this way

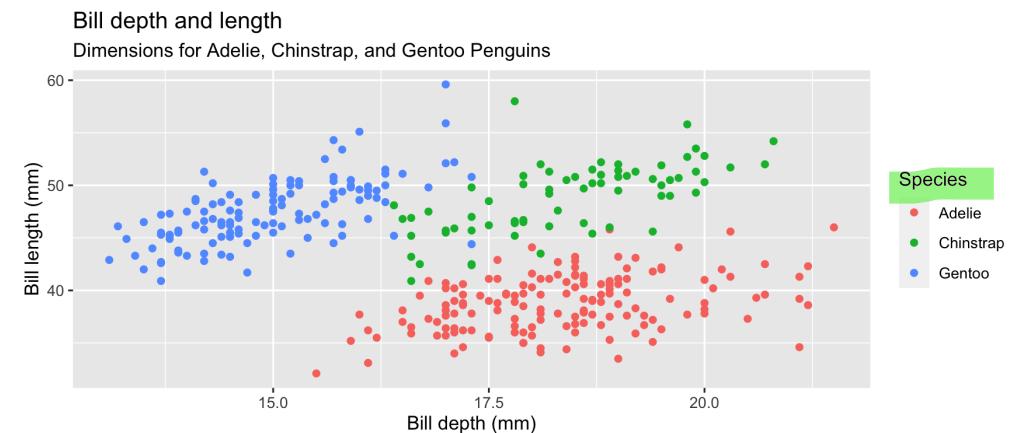
```
ggplot(data = penguins,
       mapping = aes(x = bill_depth_mm,
                     y = bill_length_mm,
                     colour = species)) +
  geom_point() +
  labs(title = "Bill depth and length",
       subtitle = "Dimensions for Adelie, Chinstrap",
       x = "Bill depth (mm)",
       y = "Bill length (mm)")
```



Palmer Penguins: Plot recreation

- Label the legend "Species"

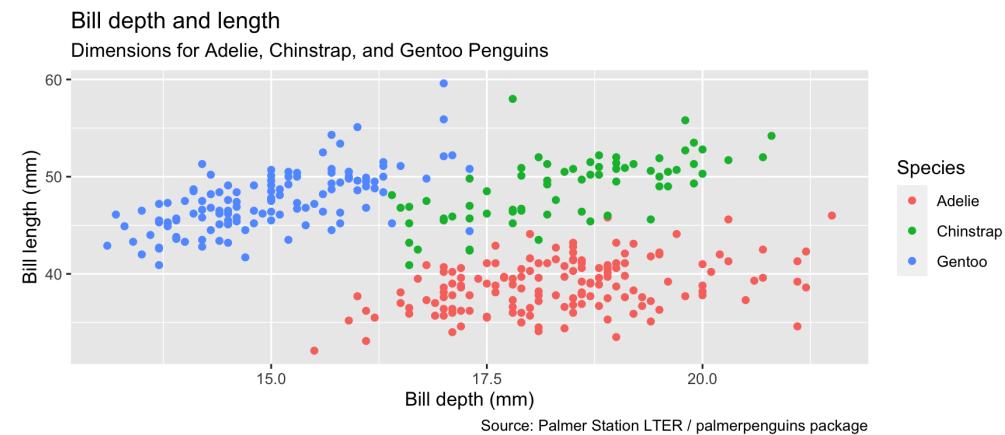
```
ggplot(data = penguins,
       mapping = aes(x = bill_depth_mm,
                     y = bill_length_mm,
                     colour = species)) +
  geom_point() +
  labs(title = "Bill depth and length",
       subtitle = "Dimensions for Adelie, Chinstrap, and Gentoo Penguins",
       x = "Bill depth (mm)", y = "Bill length (mm)",
       colour = "Species")
```



Palmer Penguins: Plot recreation

j. Add a **caption** for the data source

```
ggplot(data = penguins,
       mapping = aes(x = bill_depth_mm,
                     y = bill_length_mm,
                     colour = species)) +
  geom_point() +
  labs(title = "Bill depth and length",
       subtitle = "Dimensions for Adelie, Chinstrap, and Gentoo Penguins",
       x = "Bill depth (mm)", y = "Bill length (mm)",
       colour = "Species",
       caption = "Source: Palmer Station LTER")
```



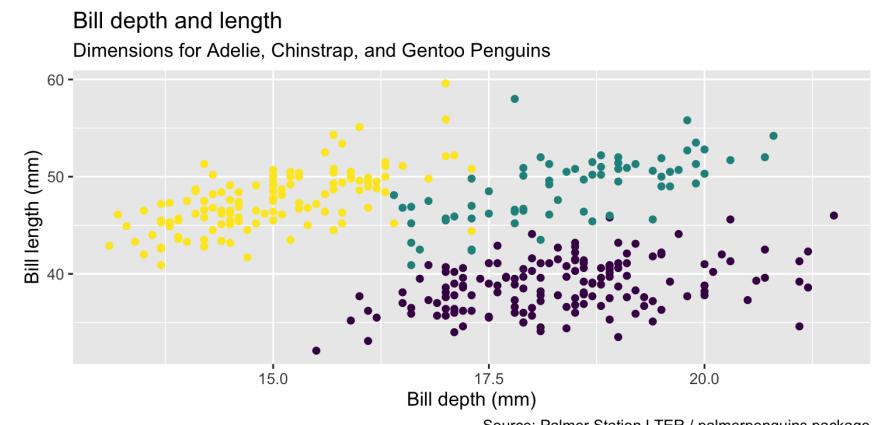
Palmer Penguins: Plot recreation

k. Finally, use a discrete colour scale that is designed to be perceived by viewers with common forms of colour blindness.

```
ggplot(data = penguins,
       mapping = aes(x = bill_depth_mm,
                     y = bill_length_mm,
                     colour = species)) +
  geom_point() +
  labs(title = "Bill depth and length",
       subtitle = "Dimensions for Adelie, Chinstrap, and Gentoo Penguins",
       x = "Bill depth (mm)", y = "Bill length (mm)",
       colour = "Species",
       caption = "Source: Palmer Station LTER",
       scale_colour_viridis_d()
```



plus symbol needed



Palmer Penguins: Recreated plot

a. Start with the `penguins` data frame,

b. Map bill depth to the x-axis

c. Map bill length to the y-axis

d. Represent each observation with a point

e. Map species to the colour of each point

f. Title the plot "Bill depth and length"

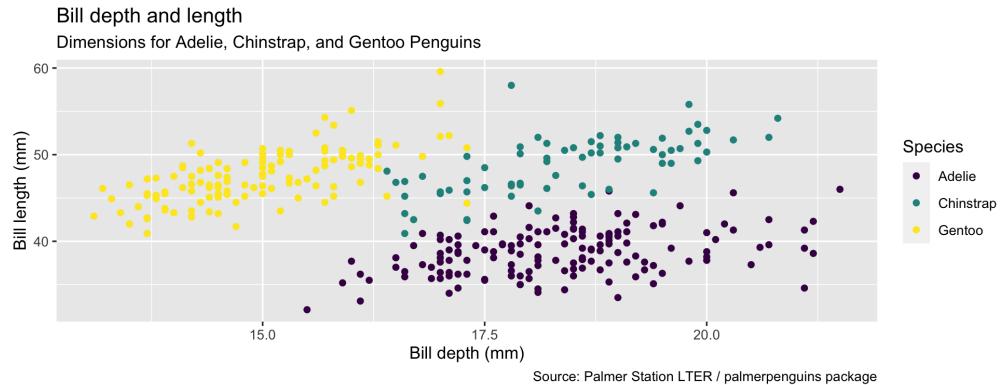
g. Add the subtitle "Dimensions for Adelie, Chinstrap, and Gentoo Penguins"

h. Label the x and y axes as "Bill depth (mm)" and "Bill length (mm)", respectively

i. Label the legend "Species"

j. Add a caption for the data source

k. Finally, use a discrete colour scale that is designed to be



Palmer Penguins: Argument names

alt ways of writing the same code as above

You can omit the names of first two arguments when building plots with `ggplot()`

```
ggplot(data = penguins,
       mapping = aes(x = bill_depth_mm,
                      y = bill_length_mm,
                      colour = species)) +
  geom_point() +
  scale_colour_viridis_d()
```



```
ggplot(penguins,
       aes(x = bill_depth_mm,
           y = bill_length_mm,
           colour = species)) +
  geom_point() +
  scale_colour_viridis_d()
```

```
ggplot(penguins) + # Data layer
  aes(x = bill_depth_mm,
      y = bill_length_mm,
      colour = species) + # Aesthetics layer
  geom_point() + # Geometric layer
  scale_colour_viridis_d()
```

aes layer outside ggplot function

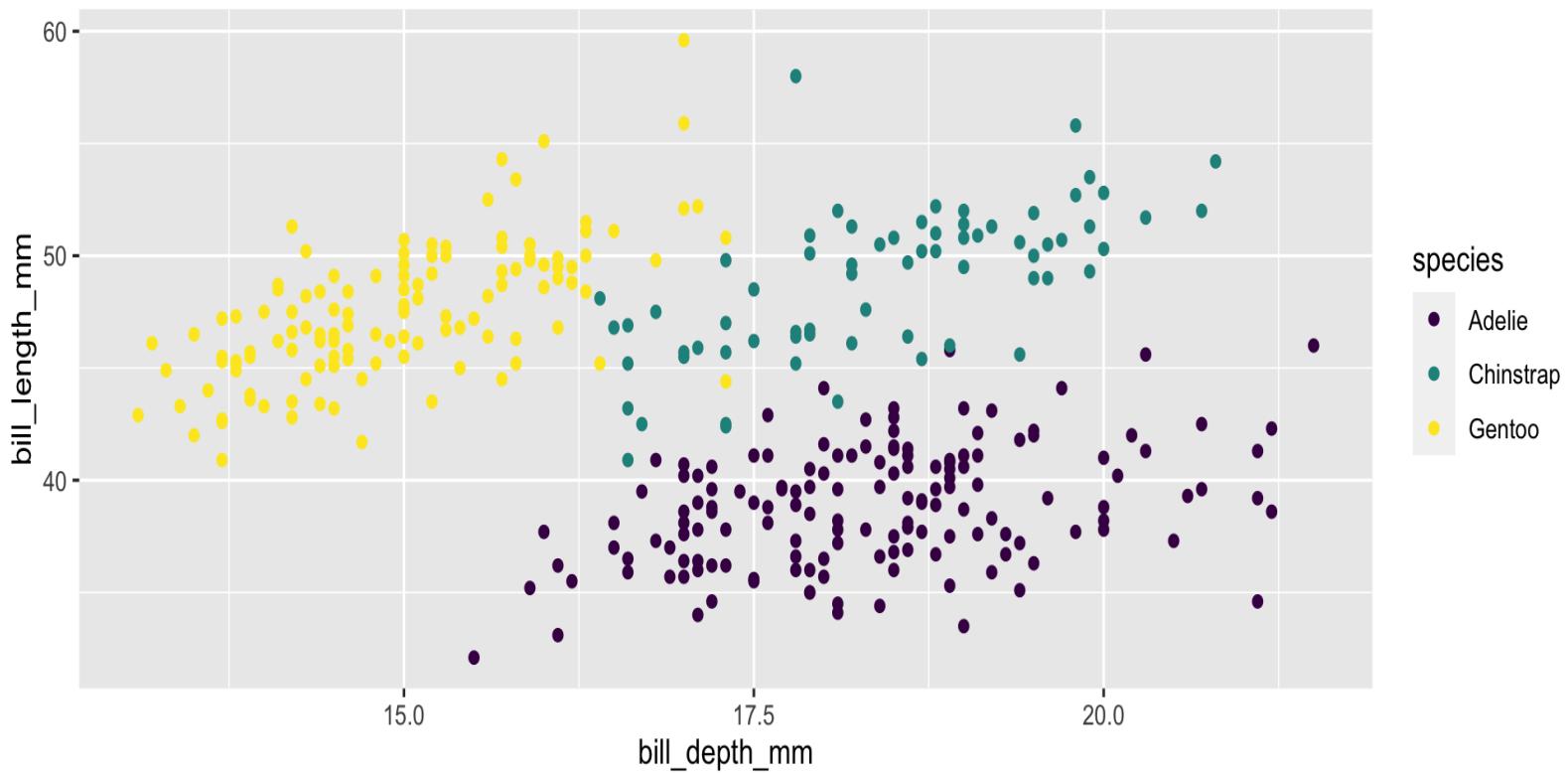
Aesthetics options

Commonly used features of `ggplot` that can be **mapped to a specific variable** in the data are,

- `colour`
- `shape`
- `size`
- `alpha` (transparency)

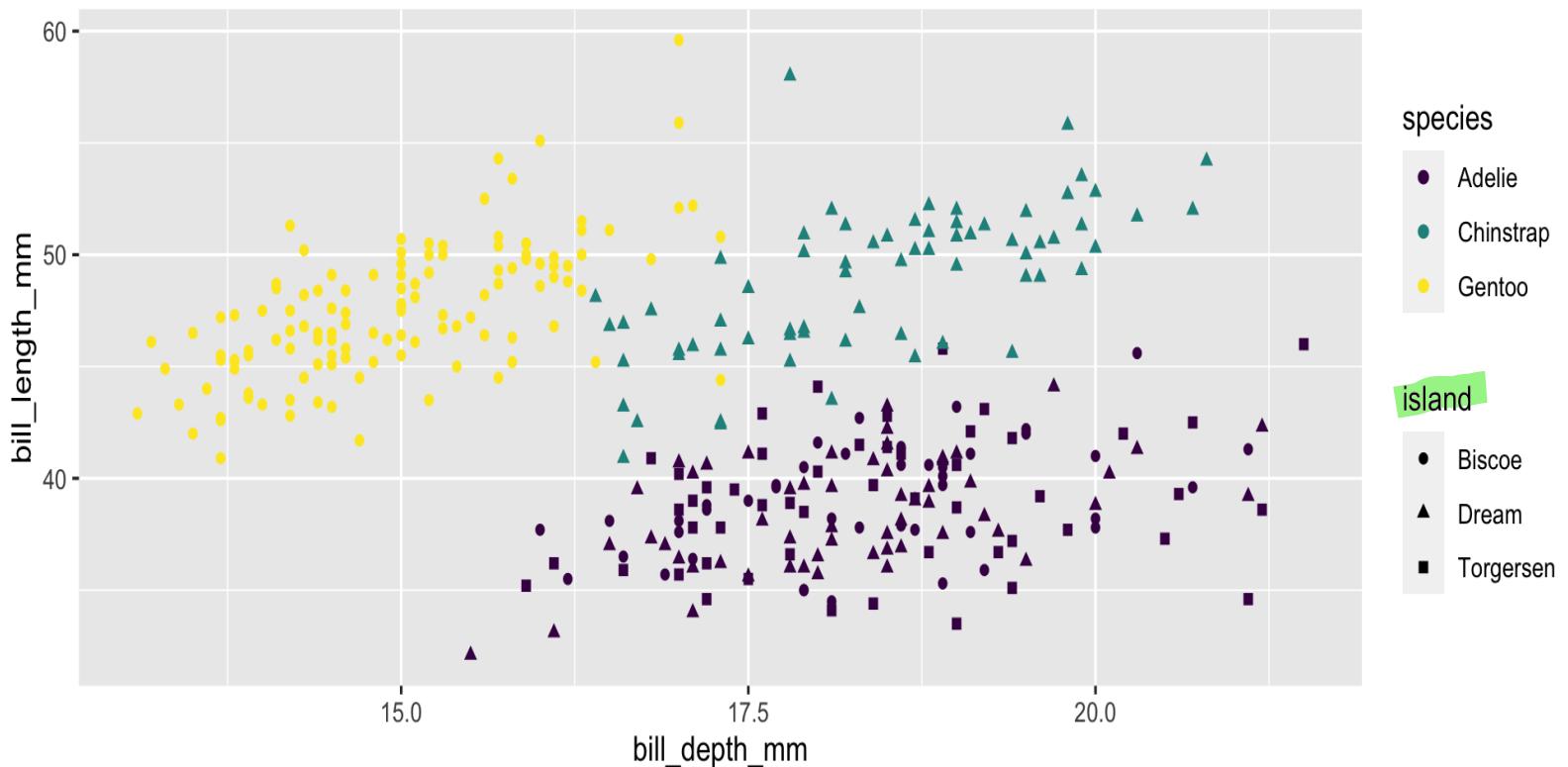
Palmer Penguins: Colour

```
ggplot(penguins) + aes(x = bill_depth_mm, y = bill_length_mm,  
colour = species) +  
geom_point() + scale_colour_viridis_d()
```



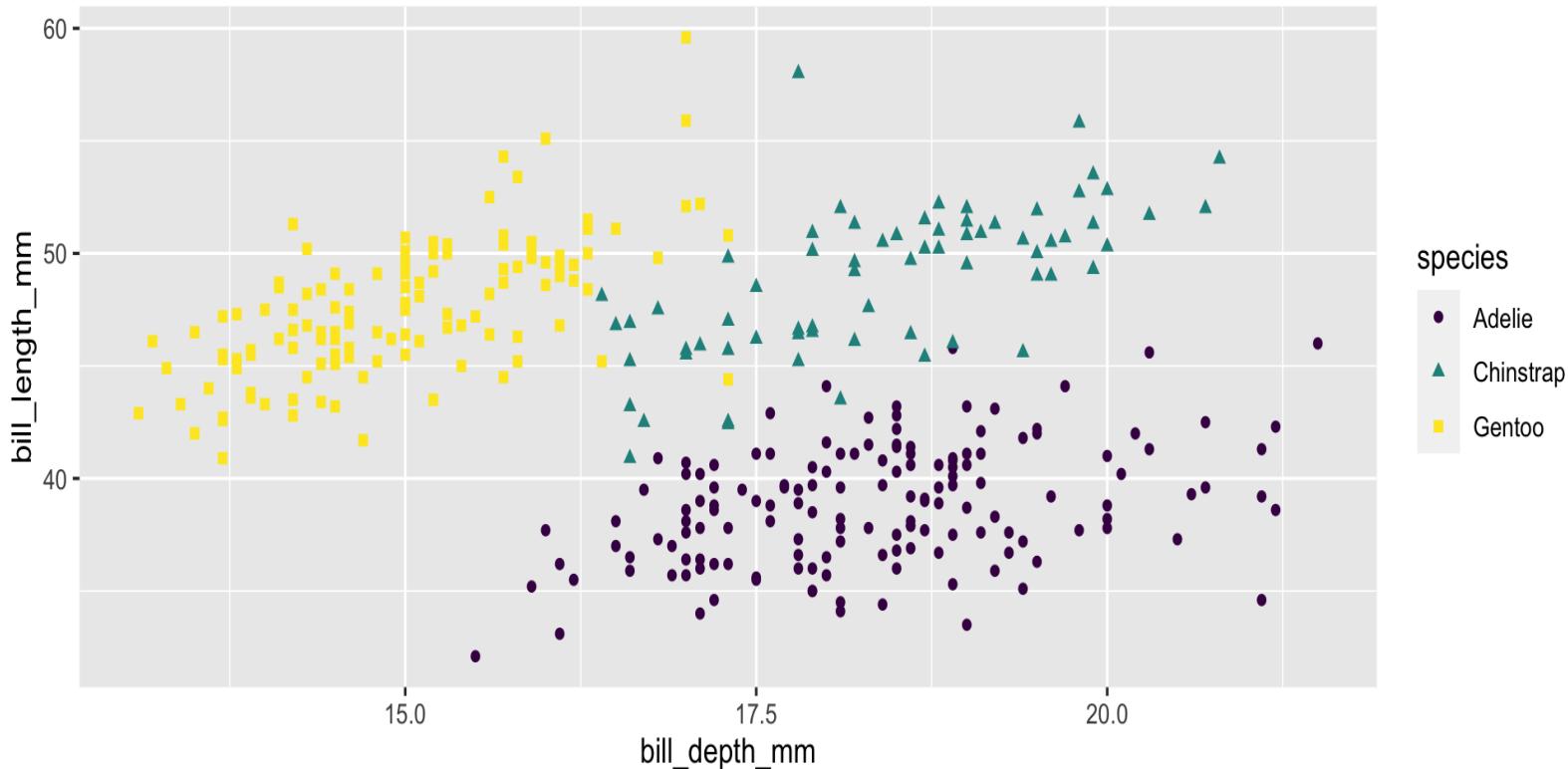
Palmer Penguins: Shape

```
ggplot(penguins, aes(x = bill_depth_mm, y = bill_length_mm, colour = species,
                     shape = island)) +
  geom_point() + scale_colour_viridis_d()
```



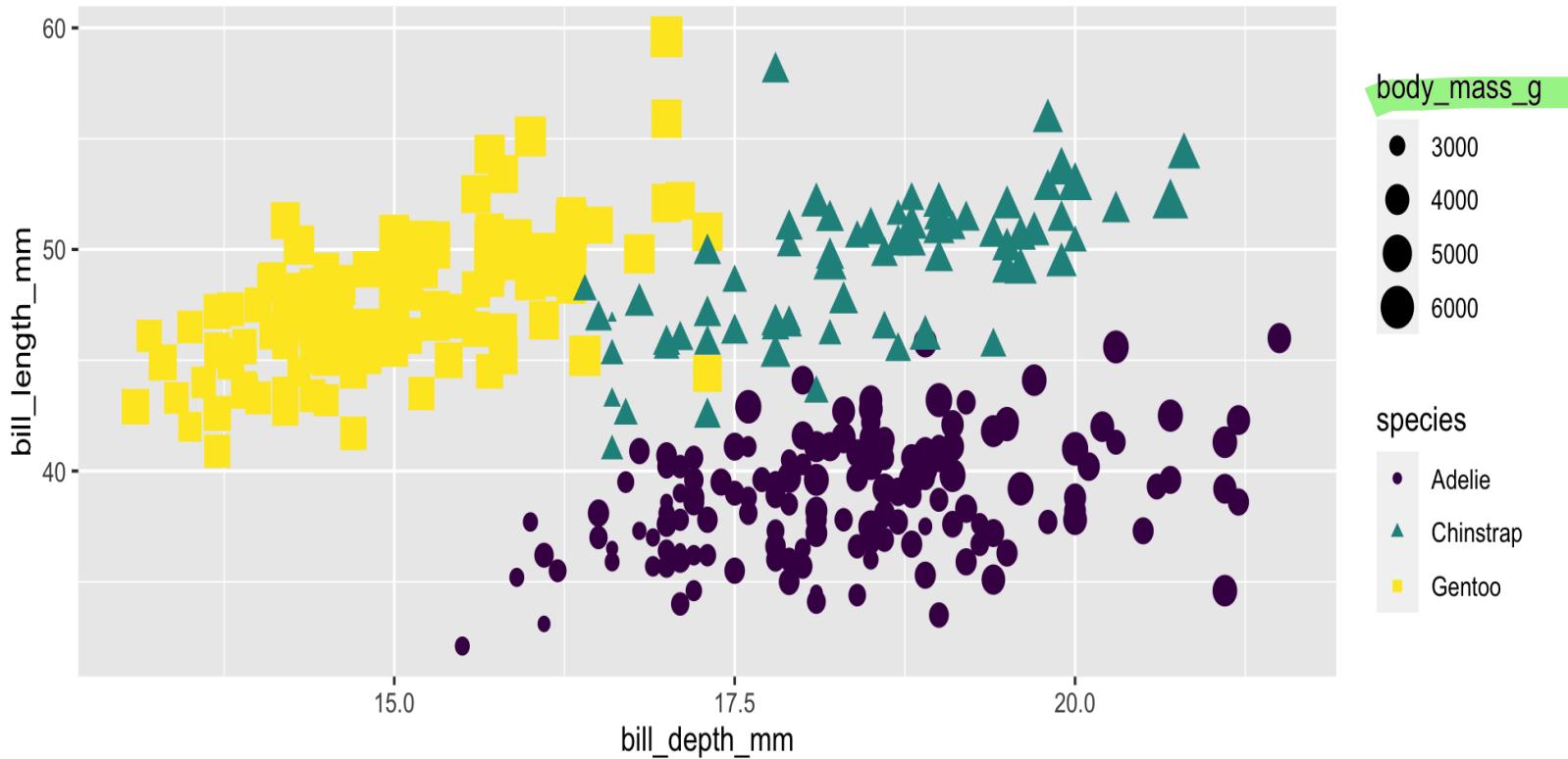
Palmer Penguins: Shape

```
ggplot(penguins, aes(x = bill_depth_mm, y = bill_length_mm, colour = species,  
                     shape = species)) + redundant  
  geom_point() + scale_colour_viridis_d()
```



Palmer Penguins: Size

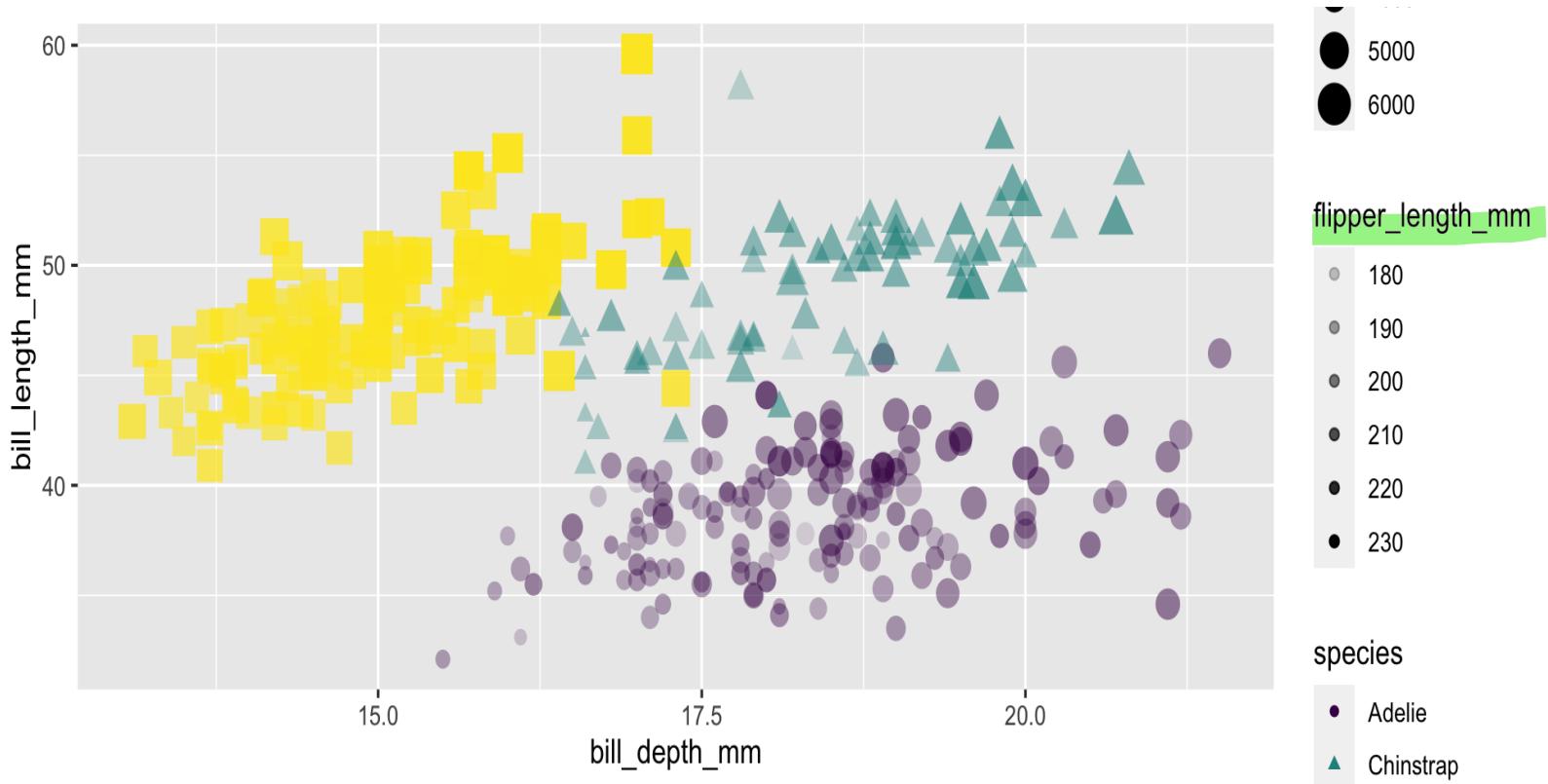
```
ggplot(penguins, aes(x = bill_depth_mm, y = bill_length_mm, colour = species, shape = species,  
size = body_mass_g)) +  
  geom_point() + scale_colour_viridis_d()
```



Alpha

transparency; want smaller flipper lengths to appear less obv bc less impt

```
ggplot(penguins, aes(x = bill_depth_mm, y = bill_length_mm, colour = species,
                     shape = species, size = body_mass_g, alpha = flipper_length_mm)) +
  geom_point() + scale_colour_viridis_d()
```



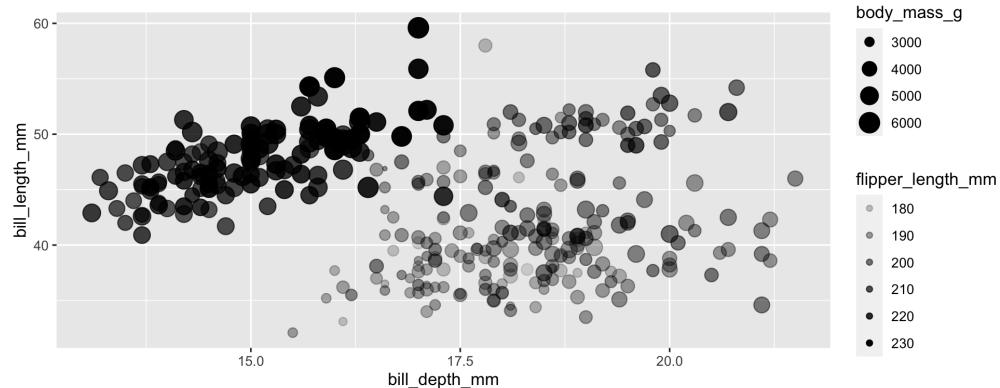
Mapping vs. Setting

- **Mapping:** Determine the size, alpha, etc. of points based on the values of a variable in the data
 - goes into `aes()`
- **Setting:** Determine the size, alpha, etc. of points **not** based on the values of a variable in the data
 - goes into `geom_*` (this was `geom_point()` in the previous example, but we'll learn about other geoms soon!)

Mapping vs. Setting

Mapping

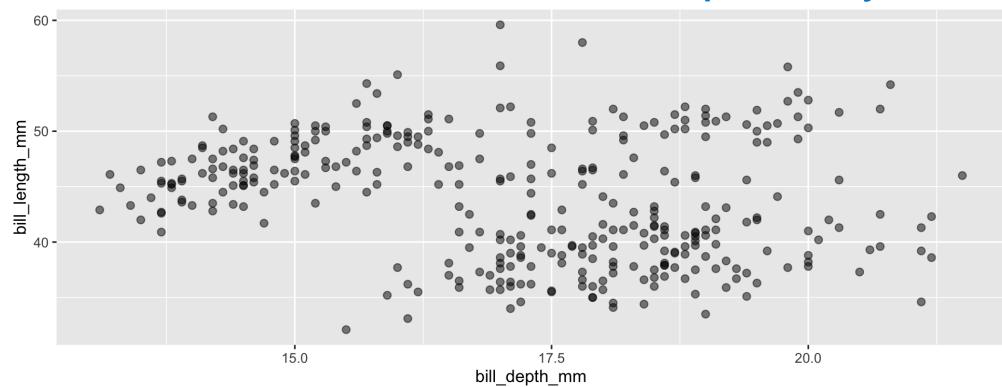
```
ggplot(penguins) +
  aes(x = bill_depth_mm,
      y = bill_length_mm,
      size = body_mass_g,
      alpha = flipper_length_mm) +
  geom_point()
```



Setting

```
ggplot(penguins) +
  aes(x = bill_depth_mm,
      y = bill_length_mm) +
  geom_point(size = 2, alpha = 0.5)
```

all have same size and transparency

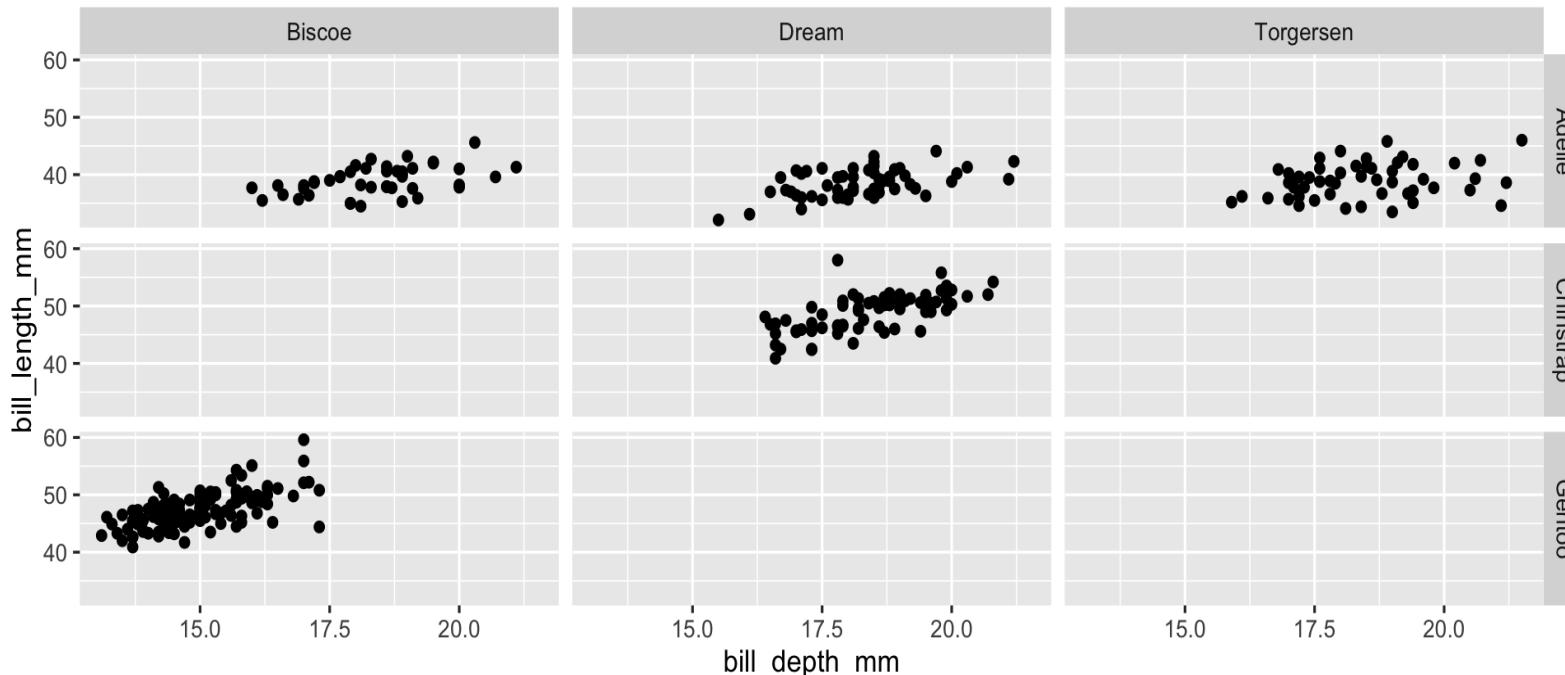


Faceting

- Smaller plots that display different subsets of the data
- Useful for exploring conditional relationships and large data

```
ggplot(penguins) +
  aes(x = bill_depth_mm,
      y = bill_length_mm) +
  geom_point() +
  facet_grid(species ~ island)
```

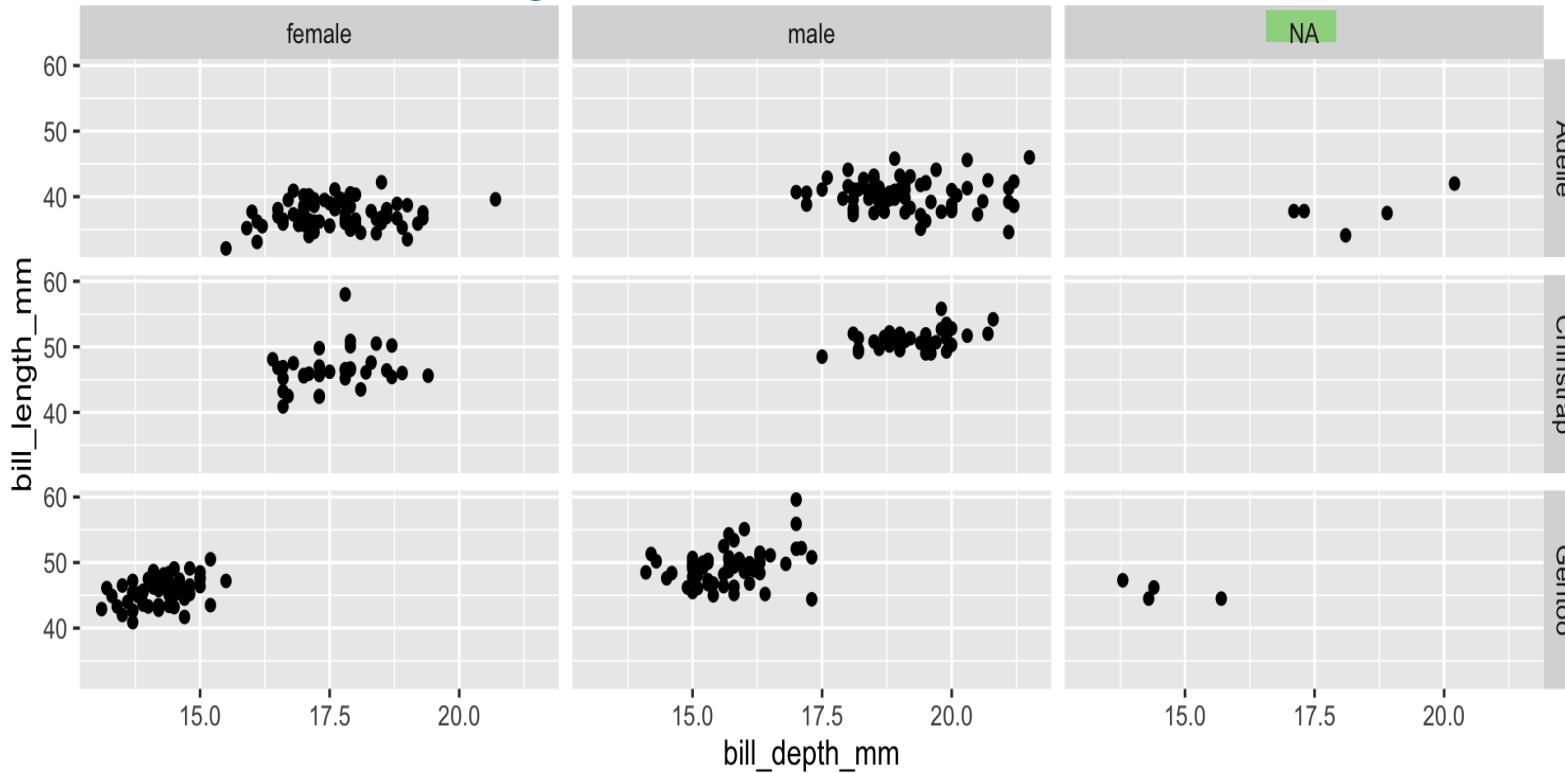
rows ~ columns



Facet 2

```
ggplot(penguins, aes(x = bill_depth_mm, y = bill_length_mm)) + geom_point() +  
  facet_grid(species ~ sex)
```

columns = gender



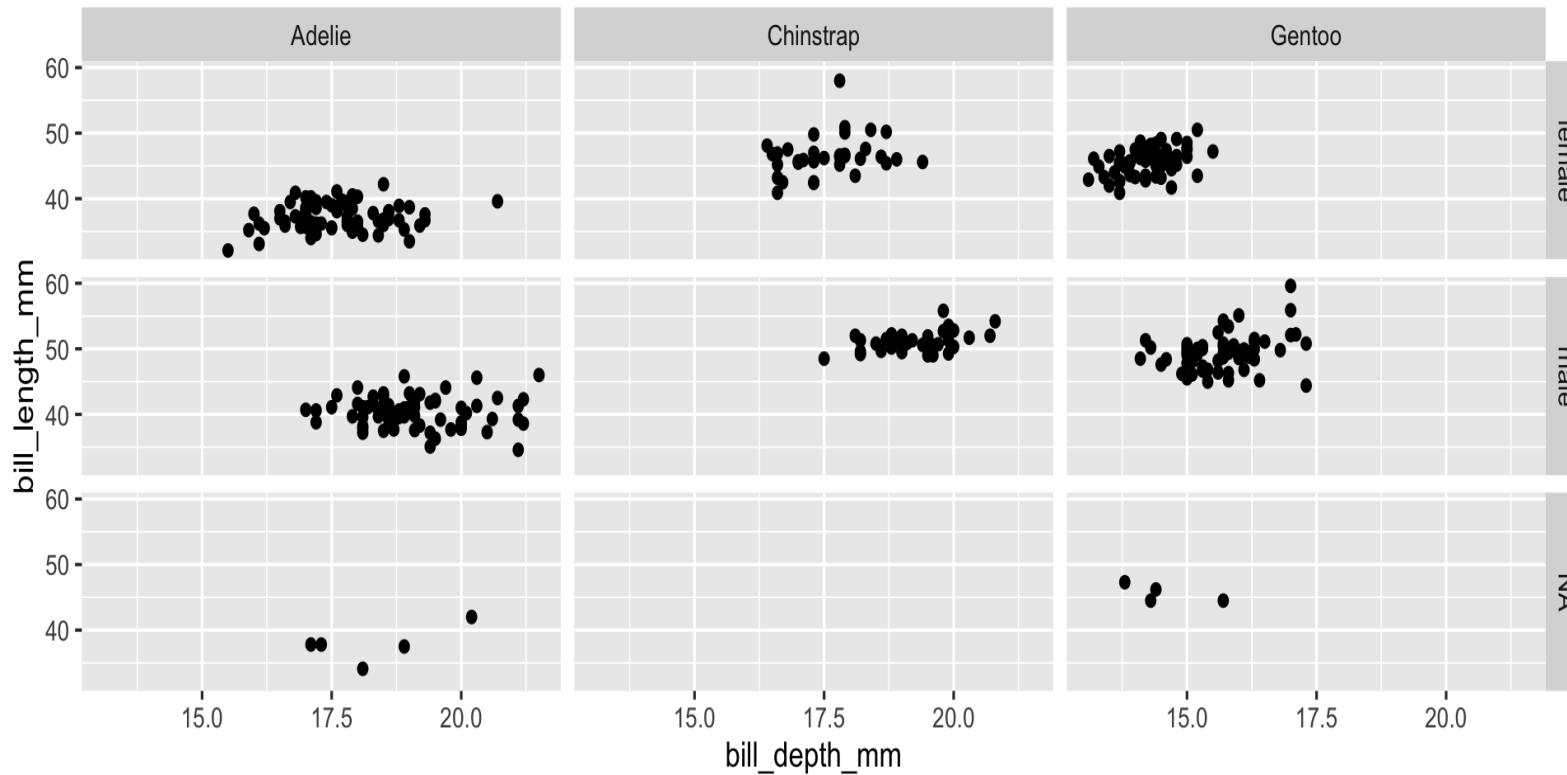
gender not tabulated = NA

rows = species

Facet 3

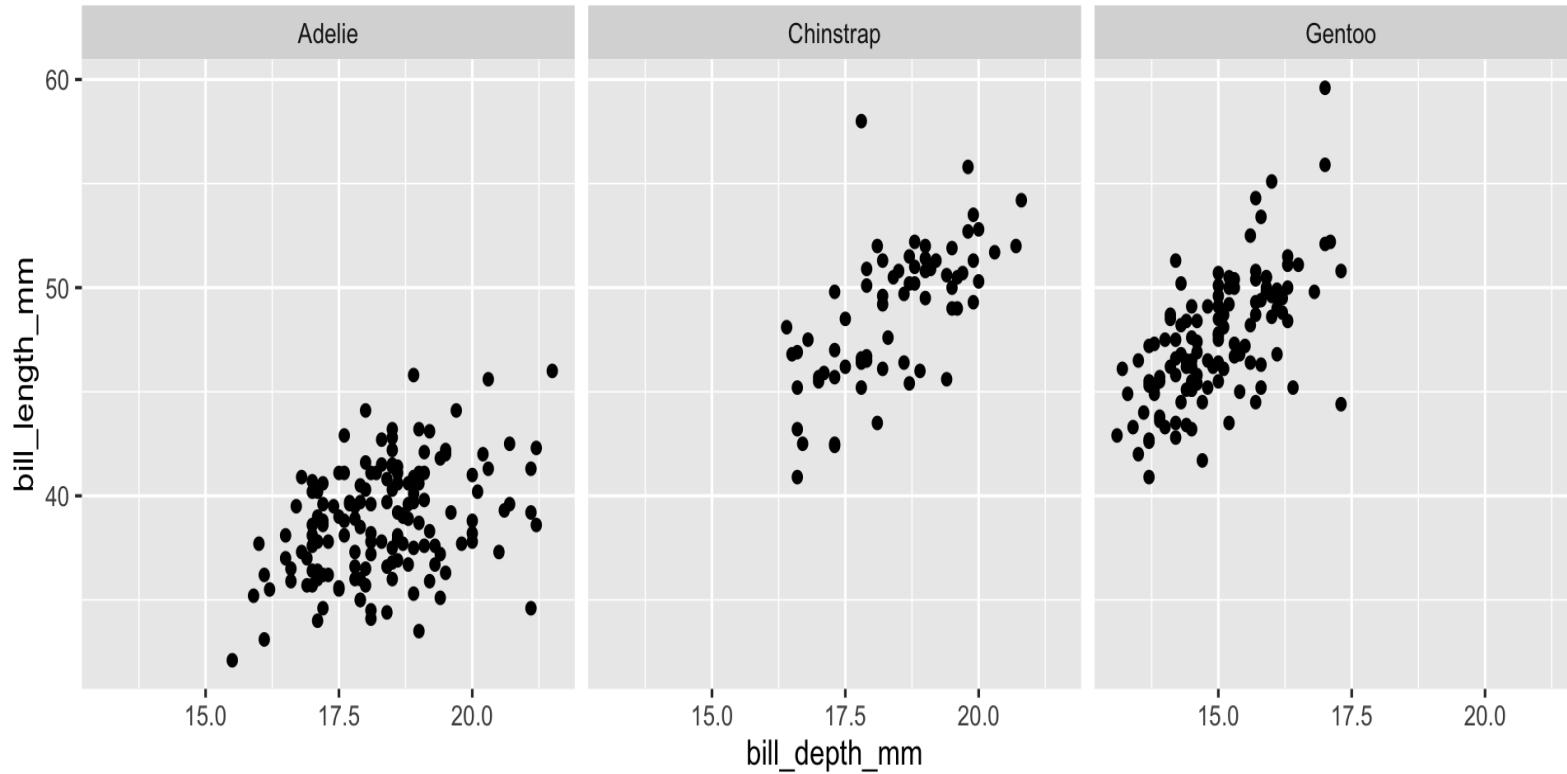
```
ggplot(penguins, aes(x = bill_depth_mm, y = bill_length_mm)) + geom_point() +  
  facet_grid(sex ~ species)
```

opposite look at order in which its typed



Facet 4

```
ggplot(penguins, aes(x = bill_depth_mm, y = bill_length_mm)) + geom_point() +  
  facet_wrap(~ species)
```

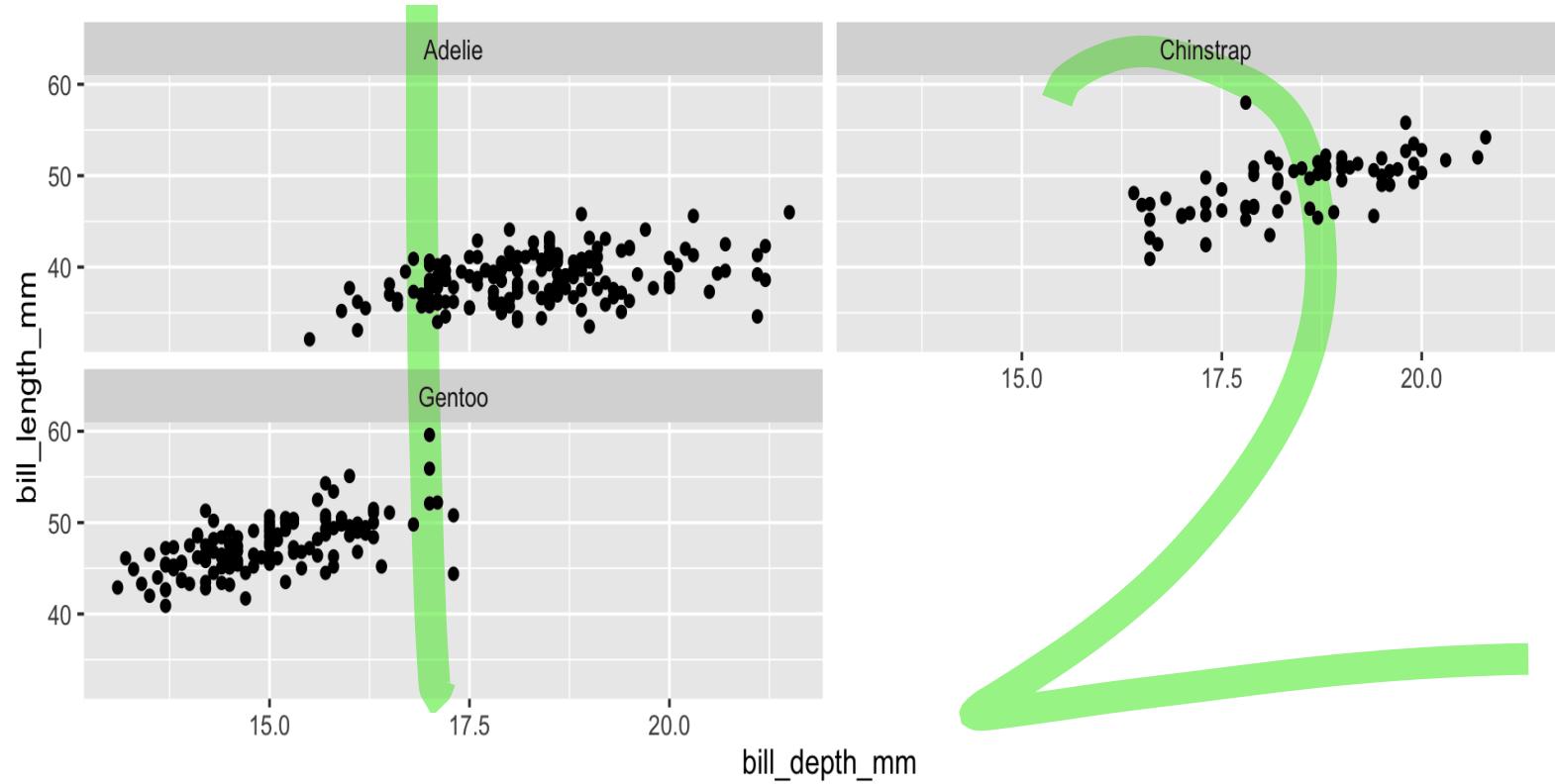


only have 1 var uw to sort the data points by instead of 2 like before

Facet 5

```
ggplot(penguins, aes(x = bill_depth_mm, y = bill_length_mm)) + geom_point() +  
  facet_wrap(~ species, ncol = 2)
```

arranging into 2 columns

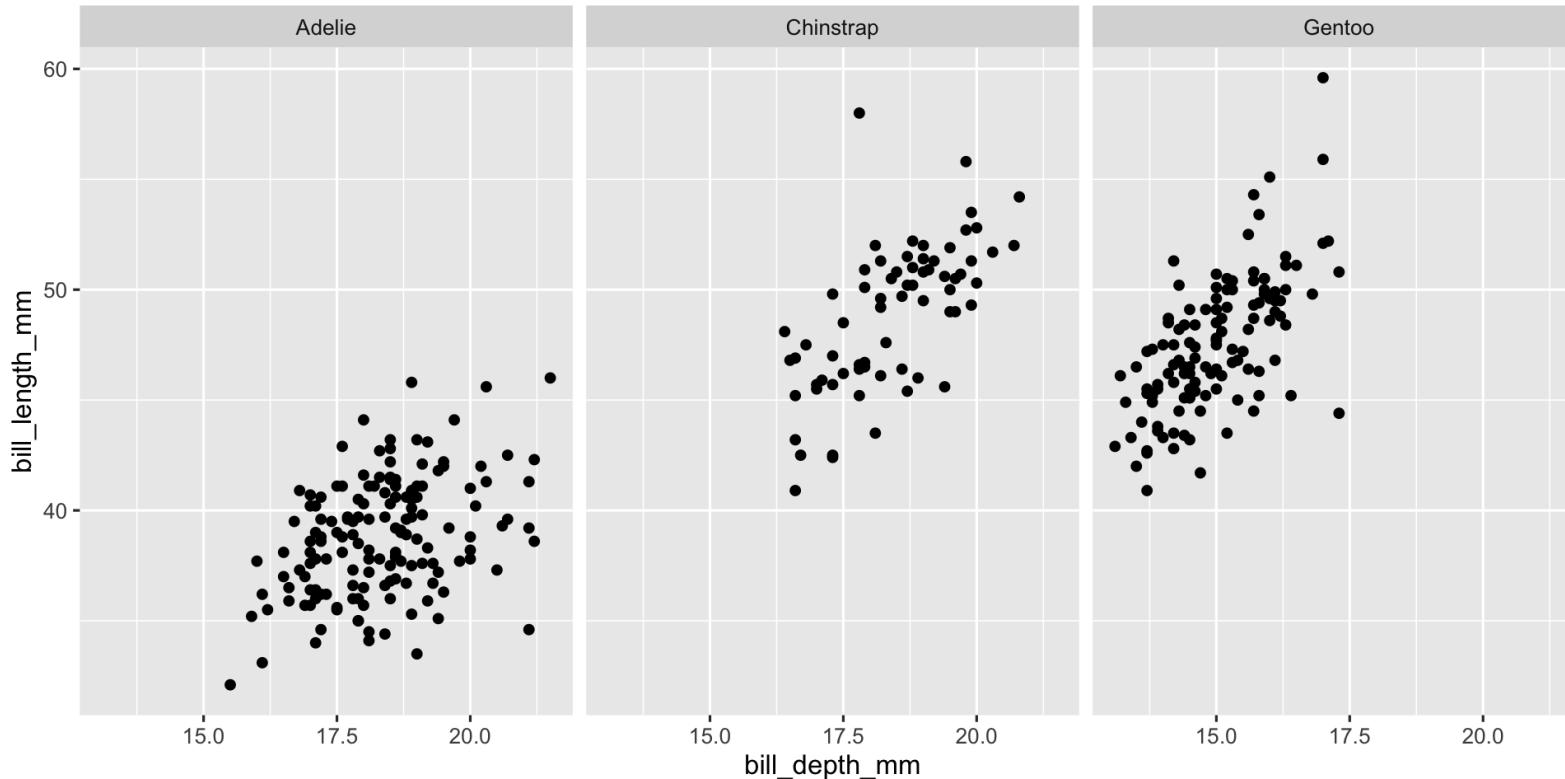


Facet 6

above is wrong, revised: facet_grid(. ~ species) -> note that u need to replace the var with a fullstop when uw only 1 var

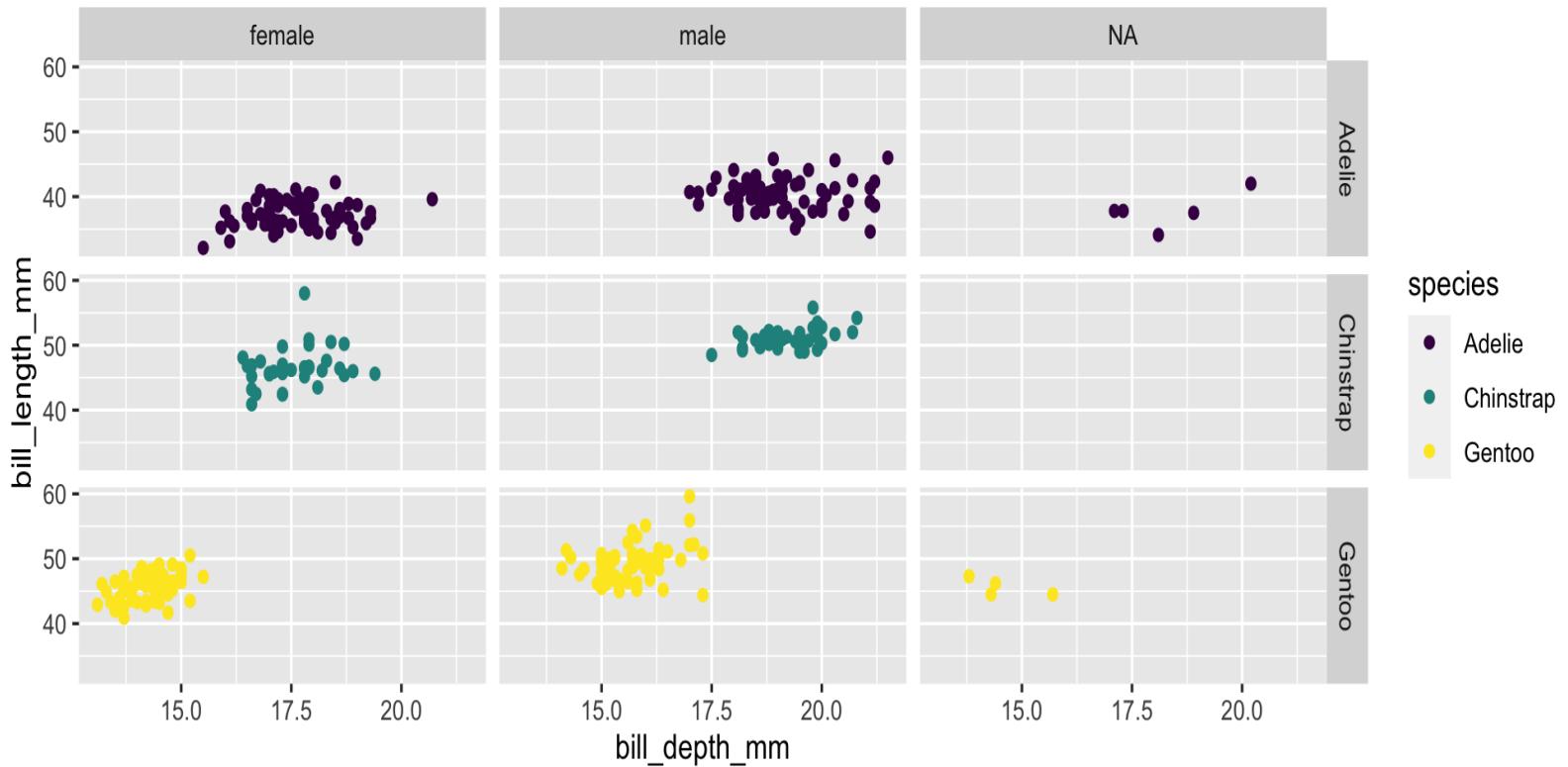
```
ggplot(penguins, aes(x = bill_depth_mm, y = bill_length_mm)) + geom_point() +
  facet_grid(. ~ species)
```

can change order also



Facet and color

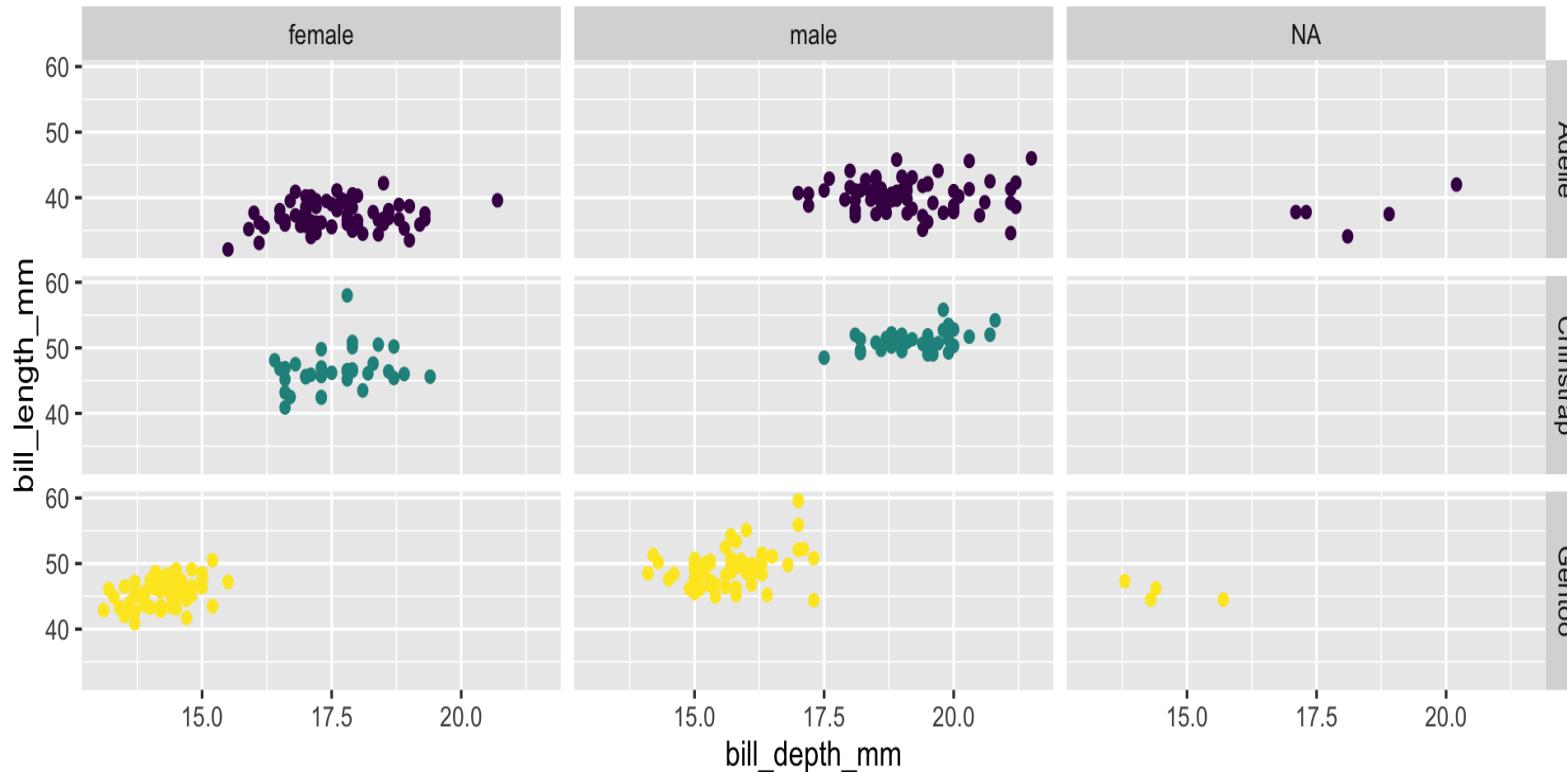
```
ggplot(penguins, aes(x = bill_depth_mm, y = bill_length_mm, color = species)) +
  geom_point() + facet_grid(species ~ sex) + scale_color_viridis_d()
```



Face and color, no legend

if udw a legend do this

```
ggplot(penguins, aes(x = bill_depth_mm, y = bill_length_mm, color = species)) +
  geom_point() + facet_grid(species ~ sex) + scale_color_viridis_d() +
  guides(color = "none")
```



II. Visualizing numeric variables

Dataset 2: Lending Club

- Thousands of loans made through the Lending Club, which is a platform that allows individuals to lend to other individuals
- Not all loans are created equal -- ease of getting a loan depends on (apparent) ability to pay back the loan
- Data includes loans *made*, these are not loan applications





Take a peek at data

```
library(openintro)  
glimpse(loans_full_schema)
```

```

## Rows: 10,000
## Columns: 55
## $ emp_title
## $ emp_length
## $ state
## $ homeownership
## $ annual_income
## $ verified_income
## $ debt_to_income
## $ annual_income_joint
## $ verification_income_joint
## $ debt_to_income_joint
## $ delinq_2y
## $ months_since_last_delinq
## $ earliest_credit_line
## $ inquiries_last_12m
## $ total_credit_lines
## $ open_credit_lines
## $ total_credit_limit
## $ total_credit_utilized

```

Selected variables

```
loans <- loans_full_schema %>%
  select(loan_amount, interest_rate, term, grade,
         state, annual_income, homeownership, debt_to_income)
glimpse(loans)
```

```
## Rows: 10,000
## Columns: 8
## $ loan_amount <int> 28000, 5000, 2000, 21600, 23000, 5000, 24000, 20000, 20...
## $ interest_rate <dbl> 14.07, 12.61, 17.09, 6.72, 14.07, 6.72, 13.59, 11.99, 1...
## $ term <dbl> 60, 36, 36, 36, 36, 36, 60, 60, 36, 36, 60, 60, 36, 60, ...
## $ grade <fct> C, C, D, A, C, A, C, B, C, A, C, B, C, B, D, D, D, F, E...
## $ state <fct> NJ, HI, WI, PA, CA, KY, MI, AZ, NV, IL, IL, FL, SC, CO, ...
## $ annual_income <dbl> 90000, 40000, 40000, 30000, 35000, 34000, 35000, 110000...
## $ homeownership <fct> MORTGAGE, RENT, RENT, RENT, RENT, OWN, MORTGAGE, MORTGA...
## $ debt_to_income <dbl> 18.01, 5.04, 21.15, 10.16, 57.96, 6.46, 23.66, 16.19, 3...
```

Selected variables

variable	description
loan_amount	Amount of the loan received, in US dollars
interest_rate	Interest rate on the loan, in an annual percentage
term	The length of the loan, which is always set as a whole number of months
grade	Loan grade, which takes values A through G and represents the quality of the loan and its likelihood of being repaid
state	US state where the borrower resides
annual_income	Borrower's annual income, including any second income, in US dollars
homeownership	Indicates whether the person owns, owns but has a mortgage, or rents
debt_to_income	Debt-to-income ratio

Variable types

numerical : discrete, cont

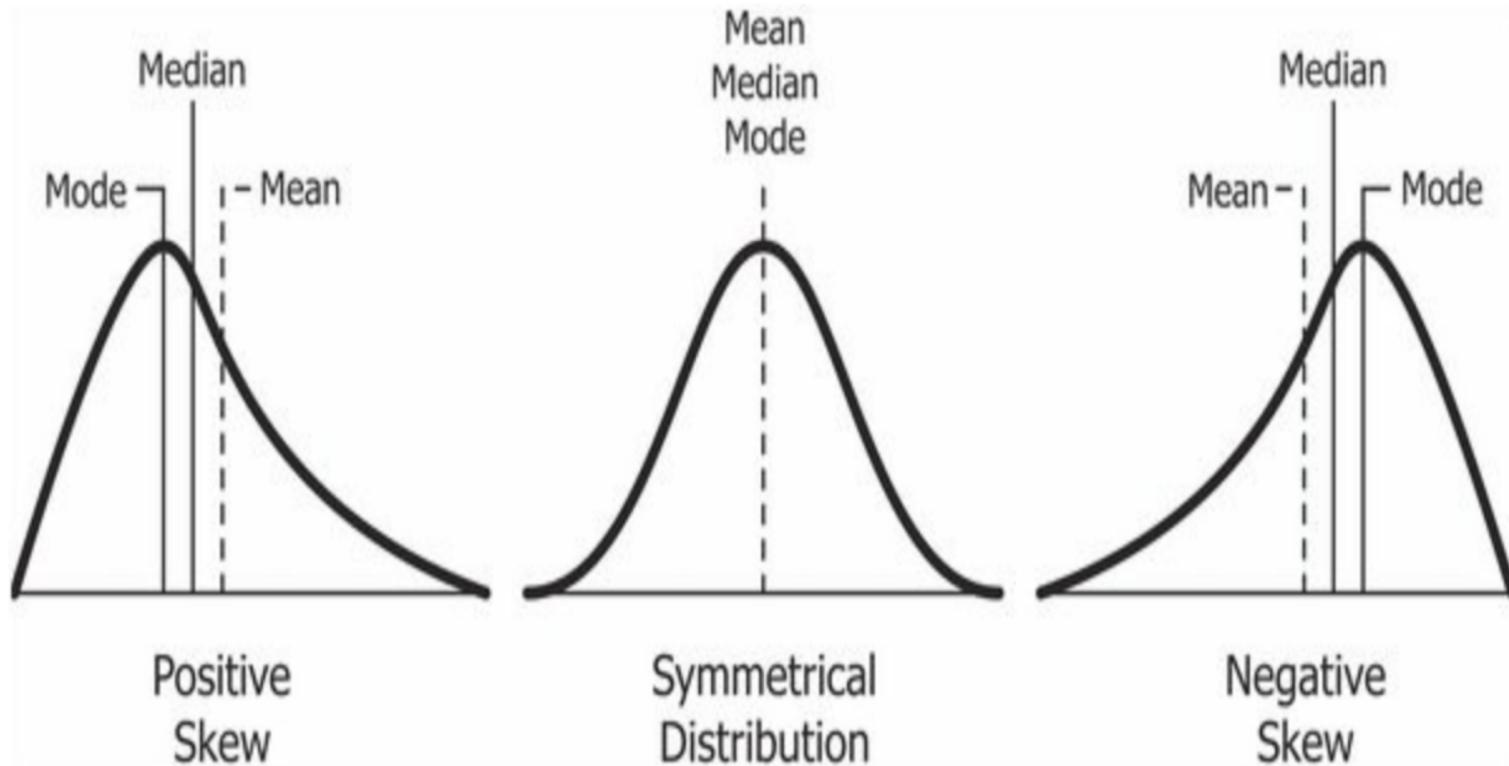
categorical: ordinal, nominal

variable	type
loan_amount	numerical, continuous
interest_rate	numerical, continuous
term	numerical, discrete
grade	categorical, ordinal
state	categorical, not ordinal
annual_income	numerical, continuous
homeownership	categorical, not ordinal
debt_to_income	numerical, continuous

Numerical distributions

- **Shape** biased/conc at certain values
 - Skewness: right-skewed, left-skewed, symmetric
 - Modality: unimodal, bimodal, multimodal, uniform telling u abt peaks of dist
- **Center** mean (`mean`), median (`median`), mode (not always useful)
- **Spread** range (`range`), standard deviation (`sd`), inter-quartile range (`IQR`)
- unusual observations

Numerical distributions: Skewness



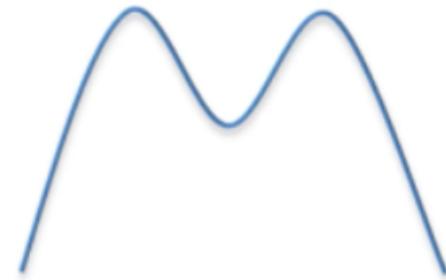
Source: Wikipedia

Numerical distributions: Modality

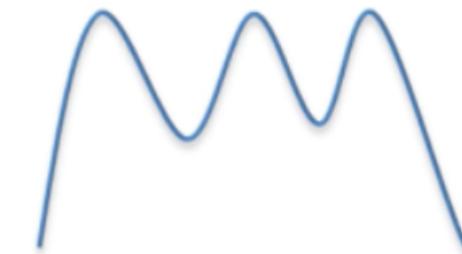
Unimodal



Bimodal



Multimodal



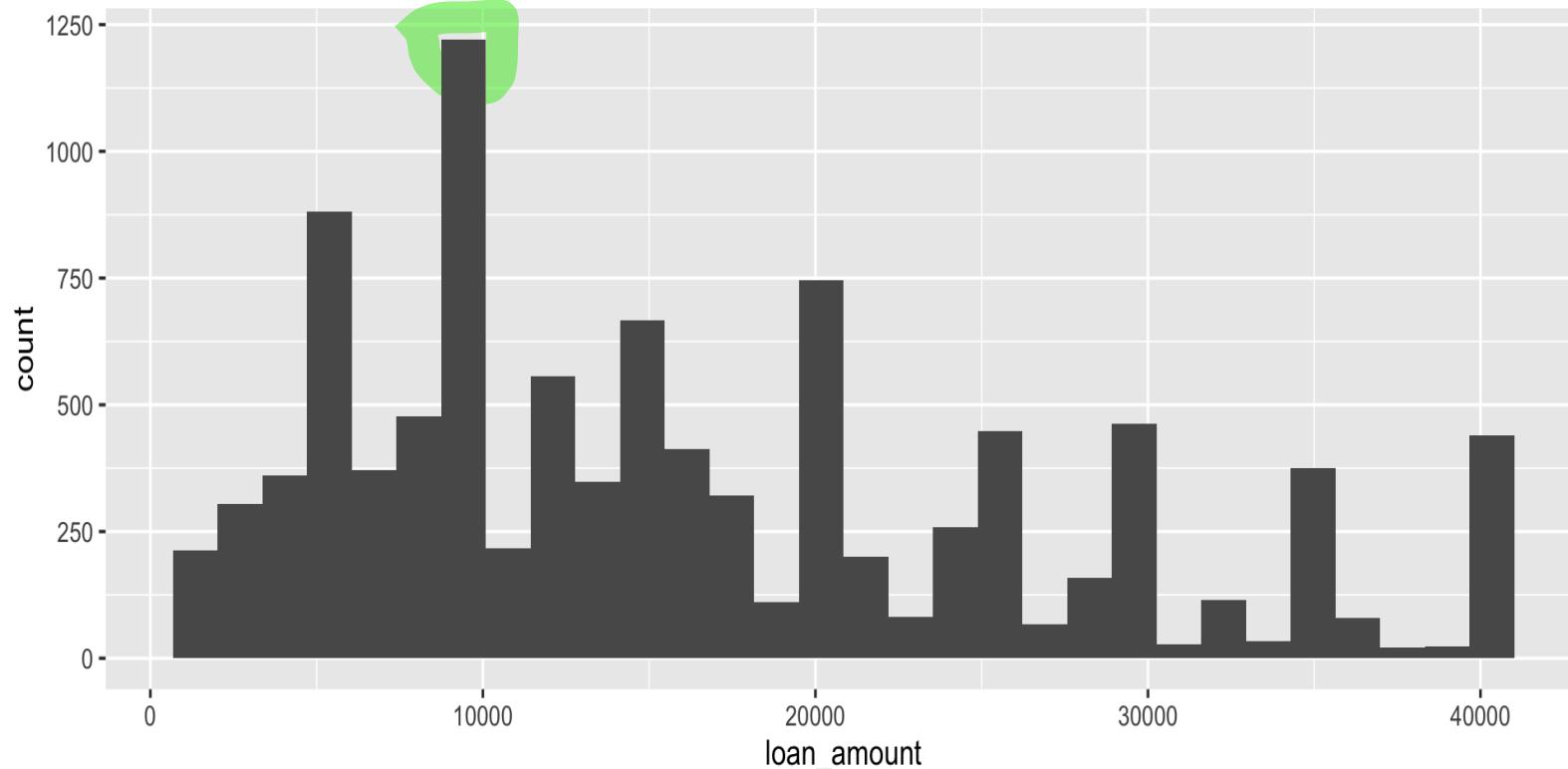
Source: <https://builtin.com/data-science/intro-descriptive-statistics>

Histogram

how many times a certain value appears in a dist/var -> use histograms

dots -> scatterplots -> rs btw 2 var

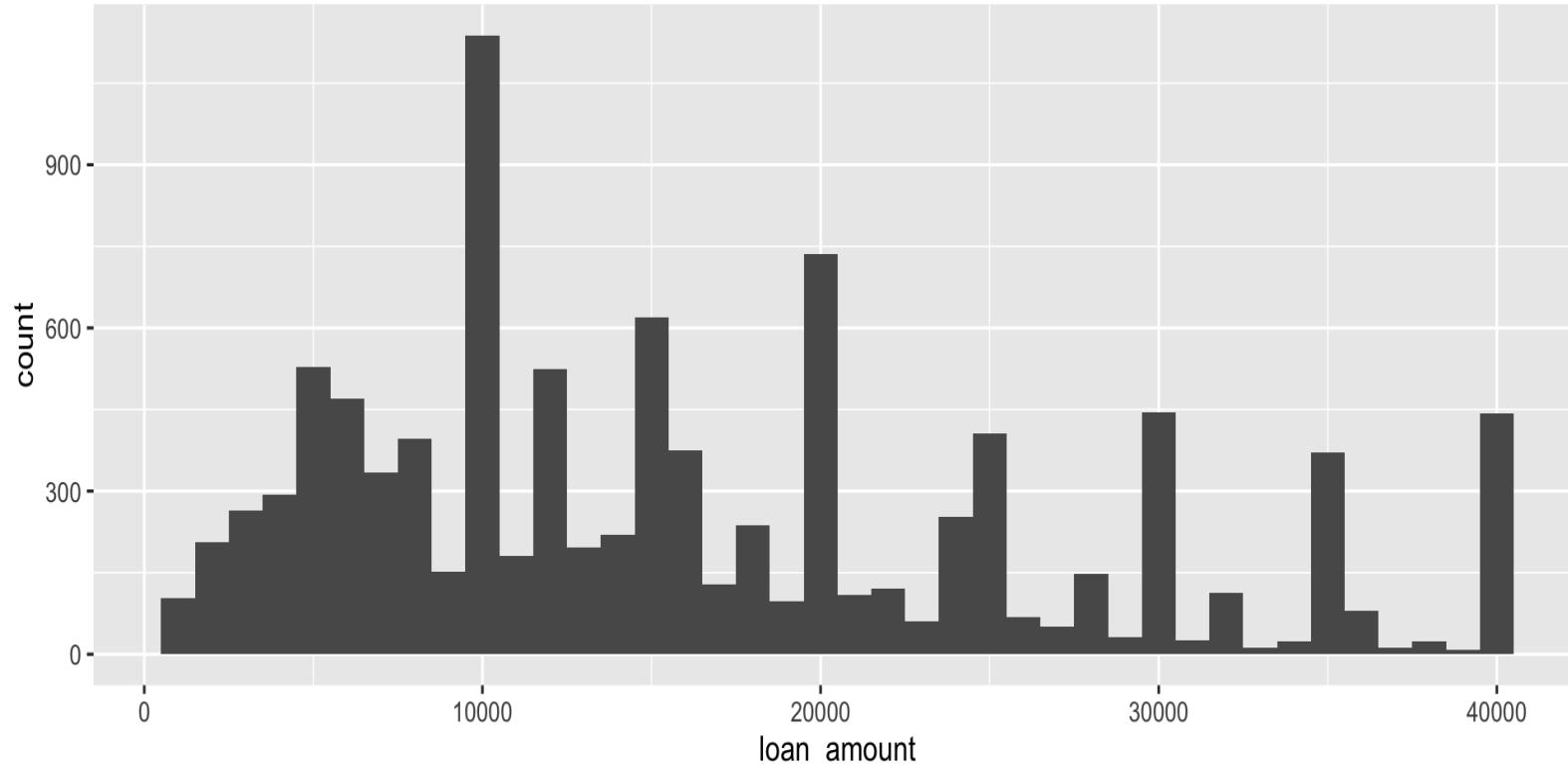
```
ggplot(loans) + aes(x = loan_amount) +  
  geom_histogram()
```



Histograms and `binwidth=1000`

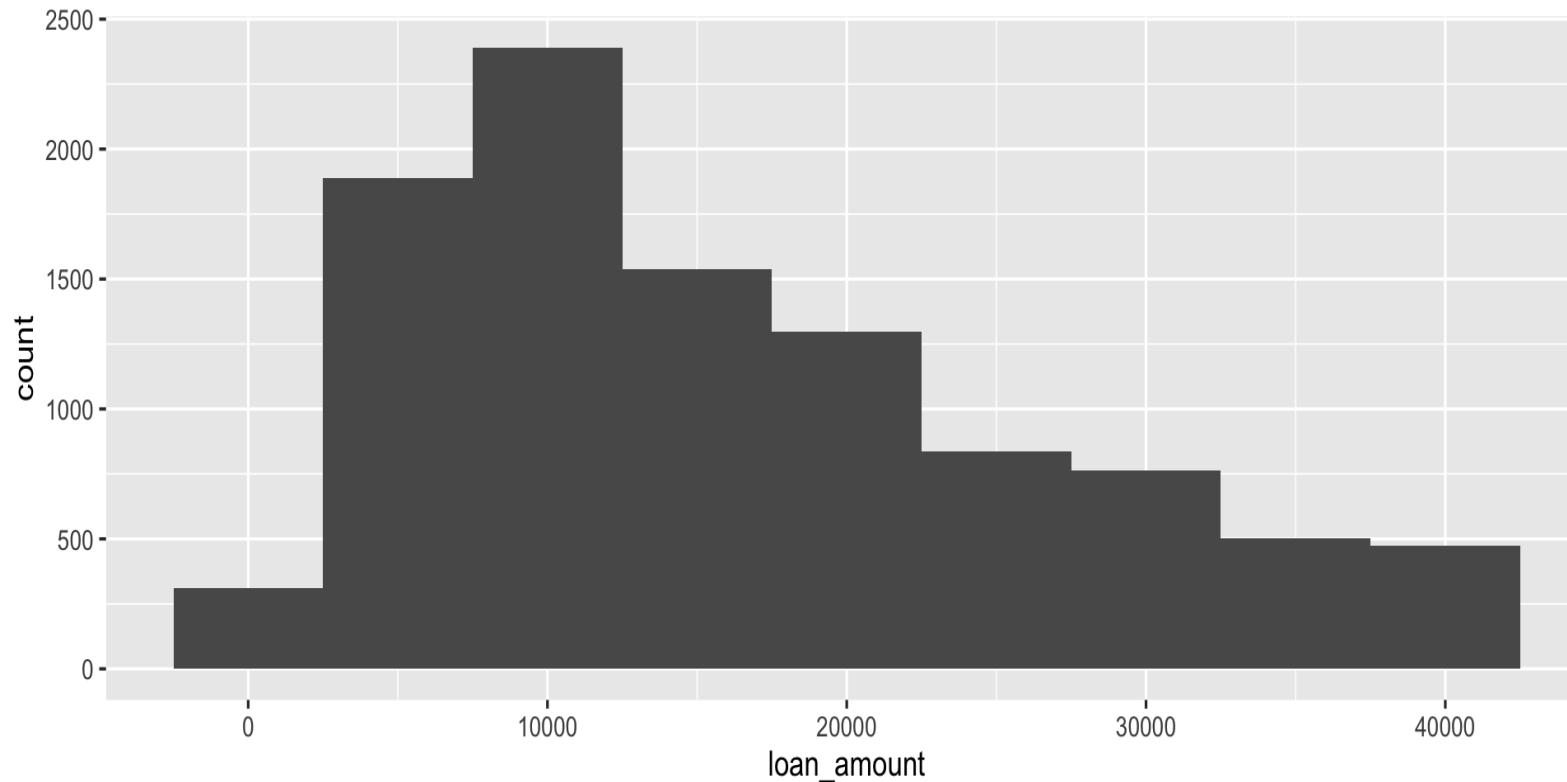
settings go into geom layer

```
# binwidth = 1000
ggplot(loans, aes(x = loan_amount)) +
  geom_histogram(binwidth = 1000)
```



Histograms and **binwidth = 5000**

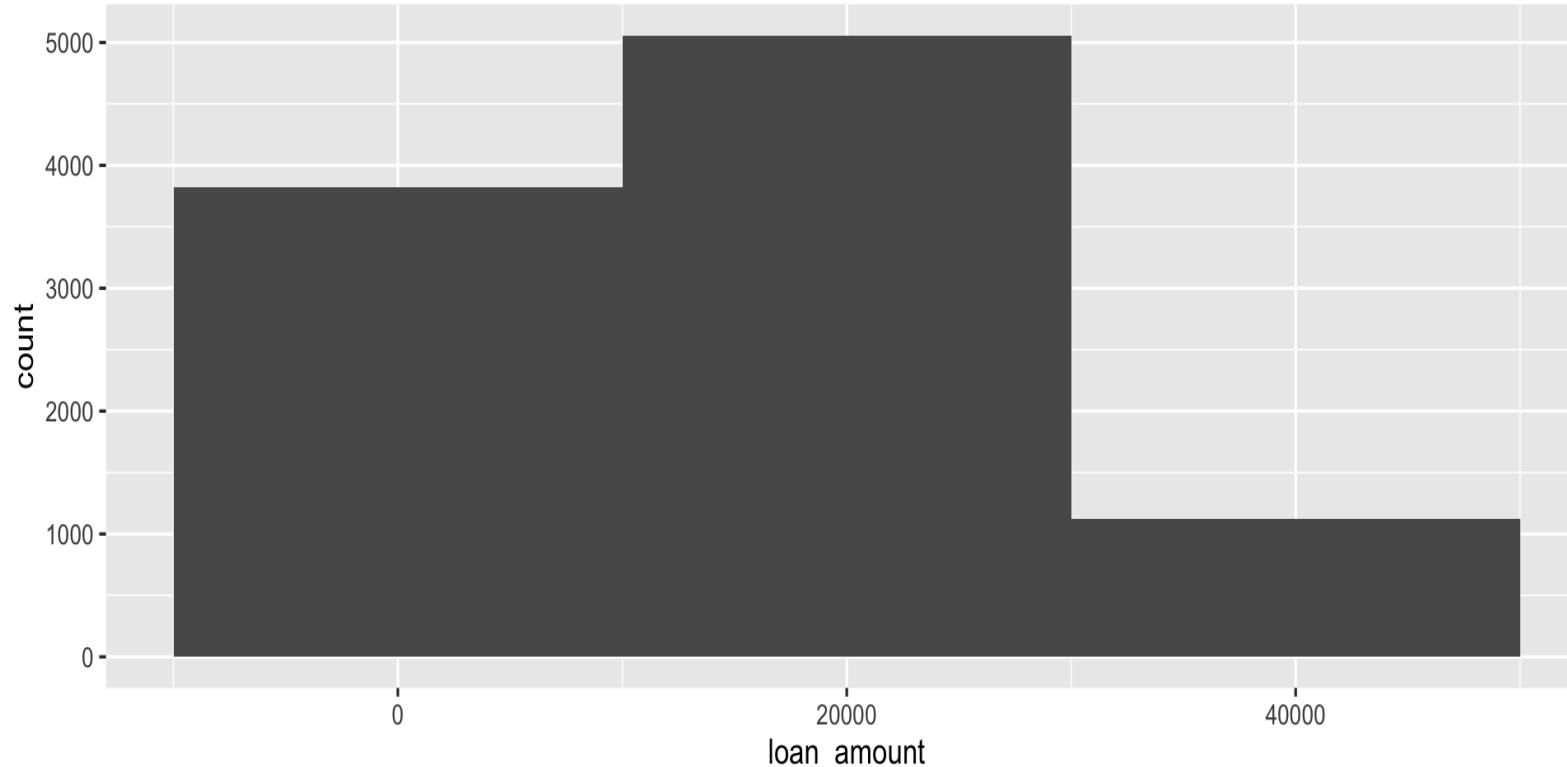
```
# binwidth = 5000
ggplot(loans, aes(x = loan_amount)) +
  geom_histogram(binwidth = 5000)
```



Histograms and binwidth=20000

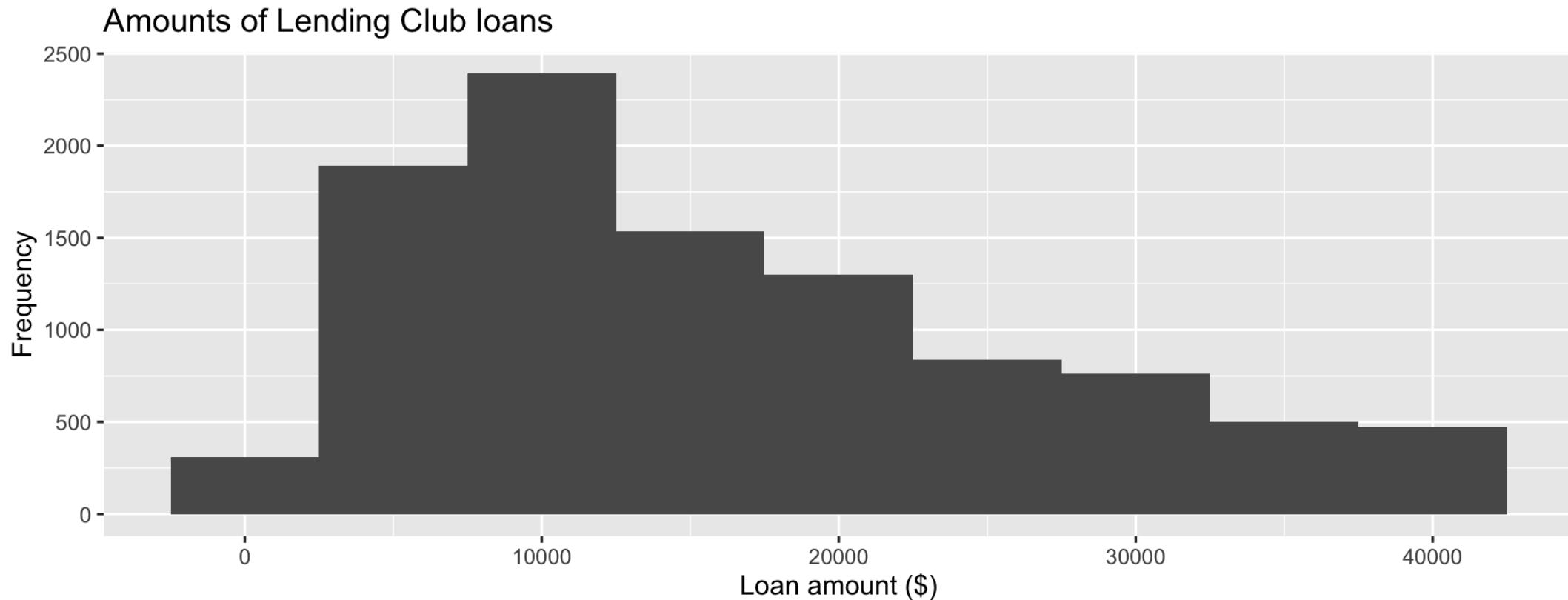
increase binwidth -> lowers resolution -> harder to make inf

```
# binwidth = 20000
ggplot(loans, aes(x = loan_amount)) +
  geom_histogram(binwidth = 20000)
```



Customizing histograms

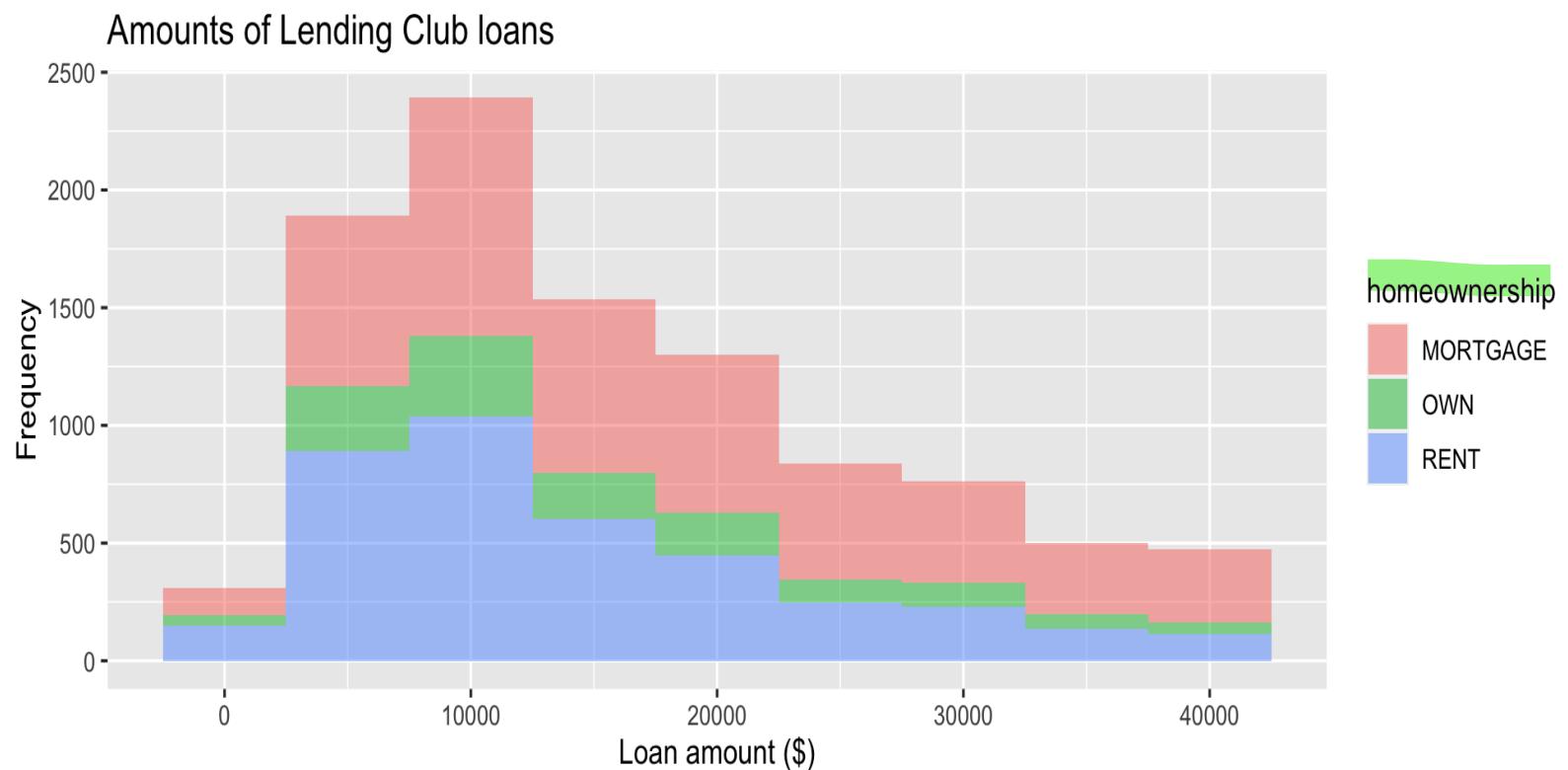
```
ggplot(loans, aes(x = loan_amount)) + geom_histogram(binwidth = 5000) +  
  labs(x = "Loan amount ($)", y = "Frequency", title = "Amounts of Lending Club loans" )
```



Fill with a categorical variable

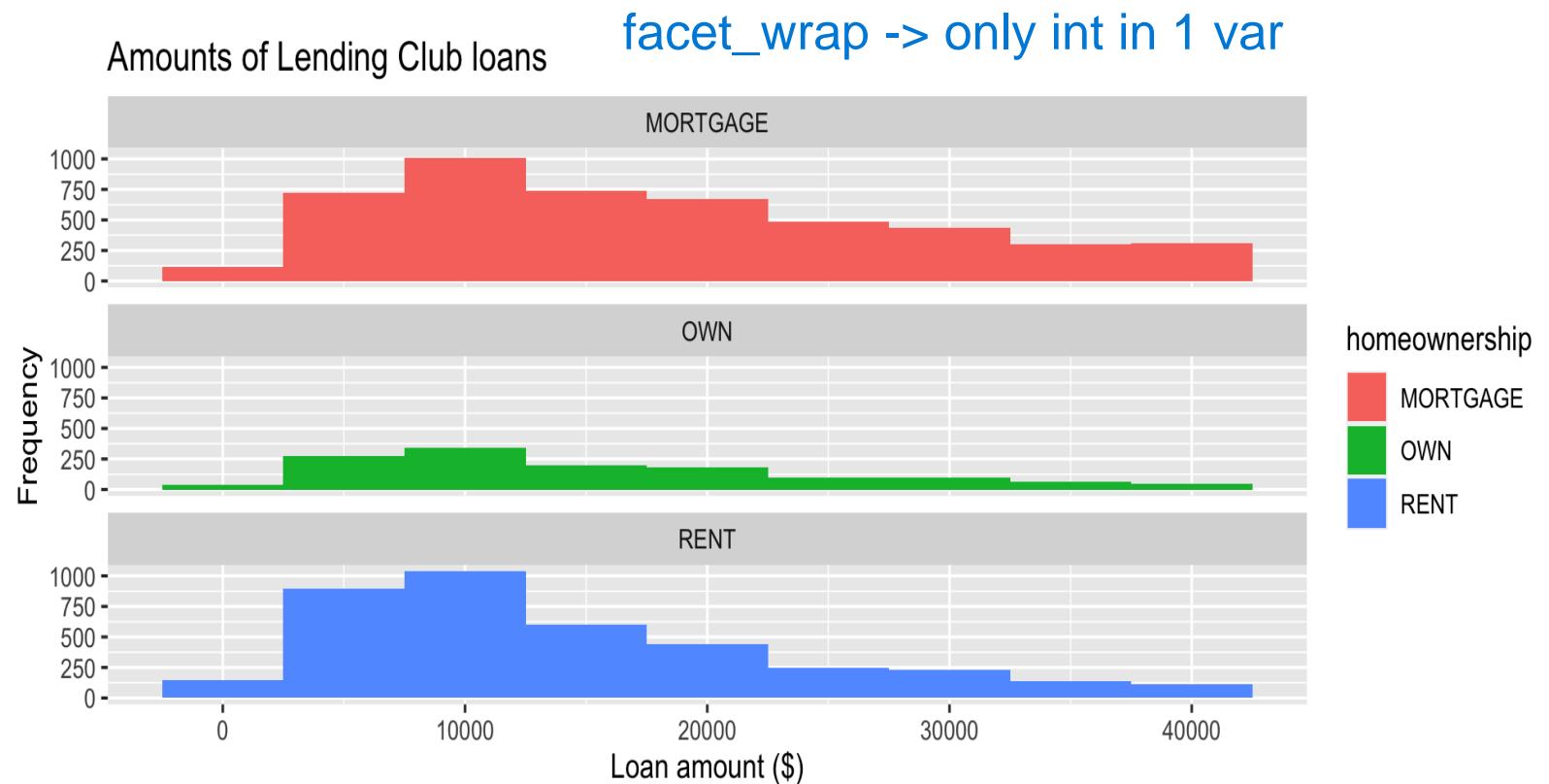
chg transparency also

```
ggplot(loans, aes(x = loan_amount, fill = homeownership)) +
  geom_histogram(binwidth = 5000, alpha = 0.5) +
  labs(x = "Loan amount ($)", y = "Frequency", title = "Amounts of Lending Club loans")
```



Facet with a categorical variable

```
ggplot(loans, aes(x = loan_amount, fill = homeownership)) + geom_histogram(binwidth = 5000) +
  labs(x = "Loan amount ($)", y = "Frequency", title = "Amounts of Lending Club loans") +
  facet_wrap(~ homeownership, nrow = 3)
```

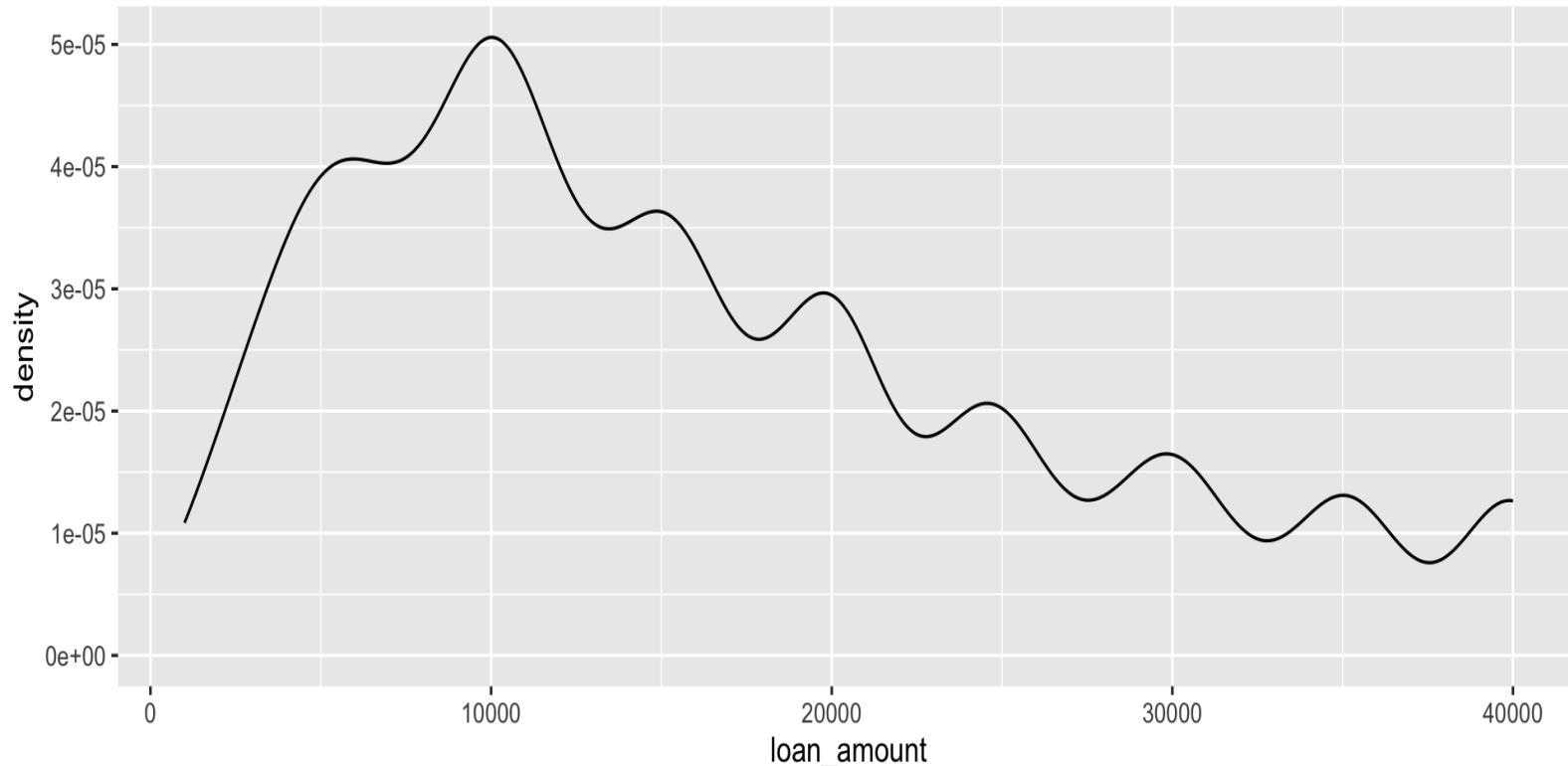


Density plot

like histograms but rep w smooth curves instead of bars
 instead of count/freq -> gvs u **prob density** of dat within ea range (no. of time values in a certain range occur/total no. of values in the var)

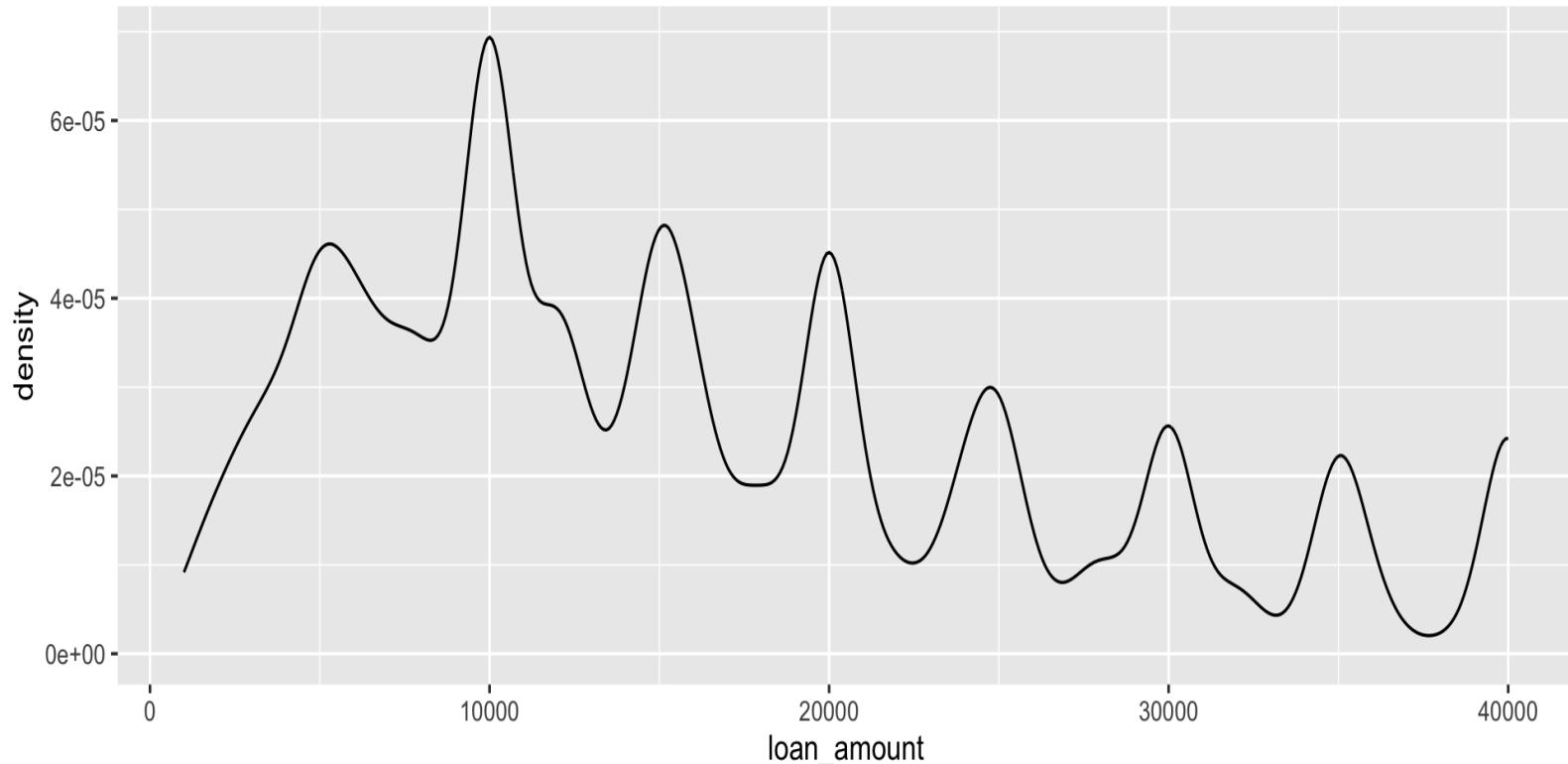
```
ggplot(loans, aes(x = loan_amount)) +  

  geom_density()
```



Density plots and adjusting bandwidth

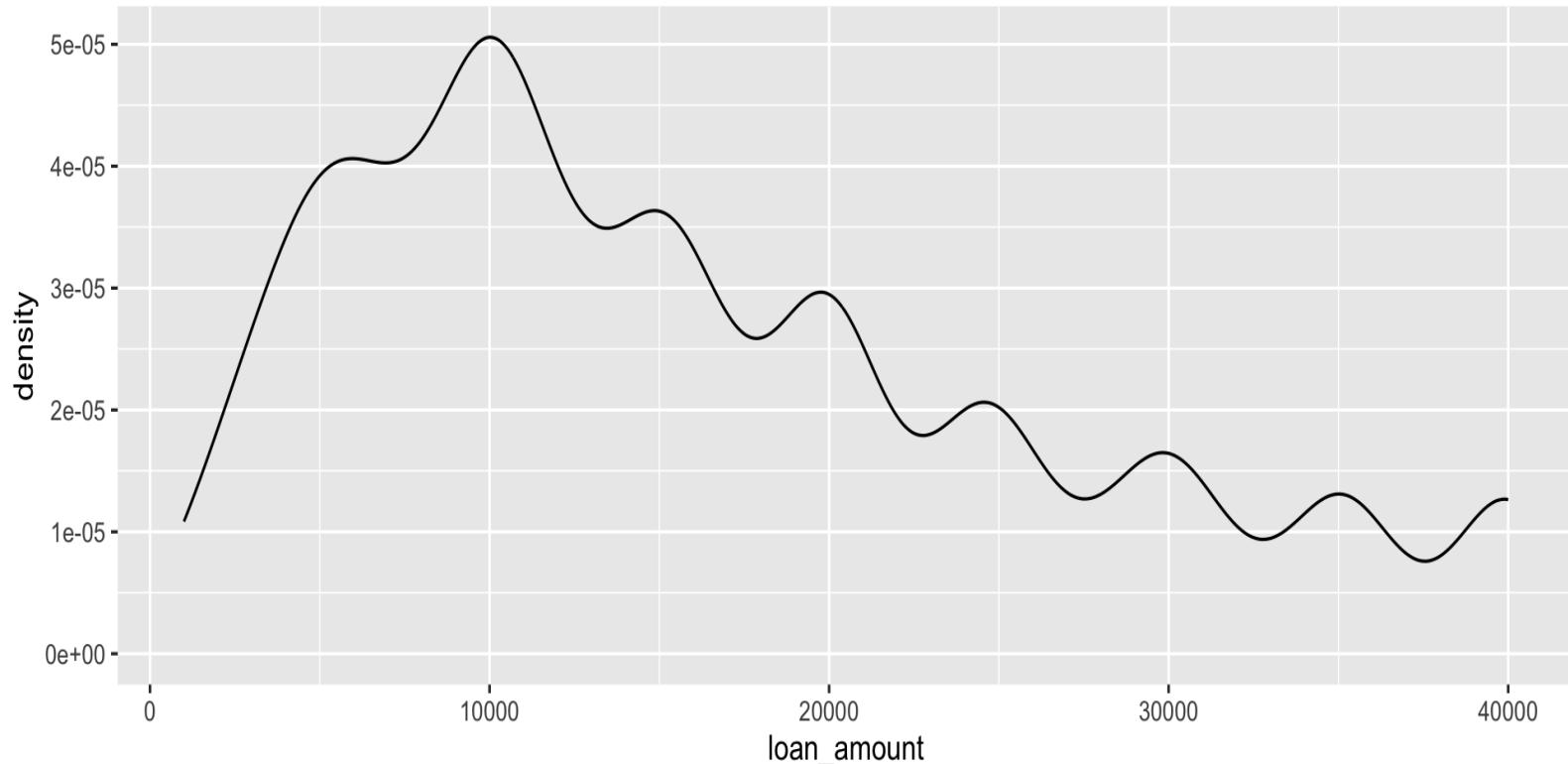
```
ggplot(loans, aes(x = loan_amount)) +  
  geom_density(adjust = 0.5)
```



Density plots and adjusting bandwidth

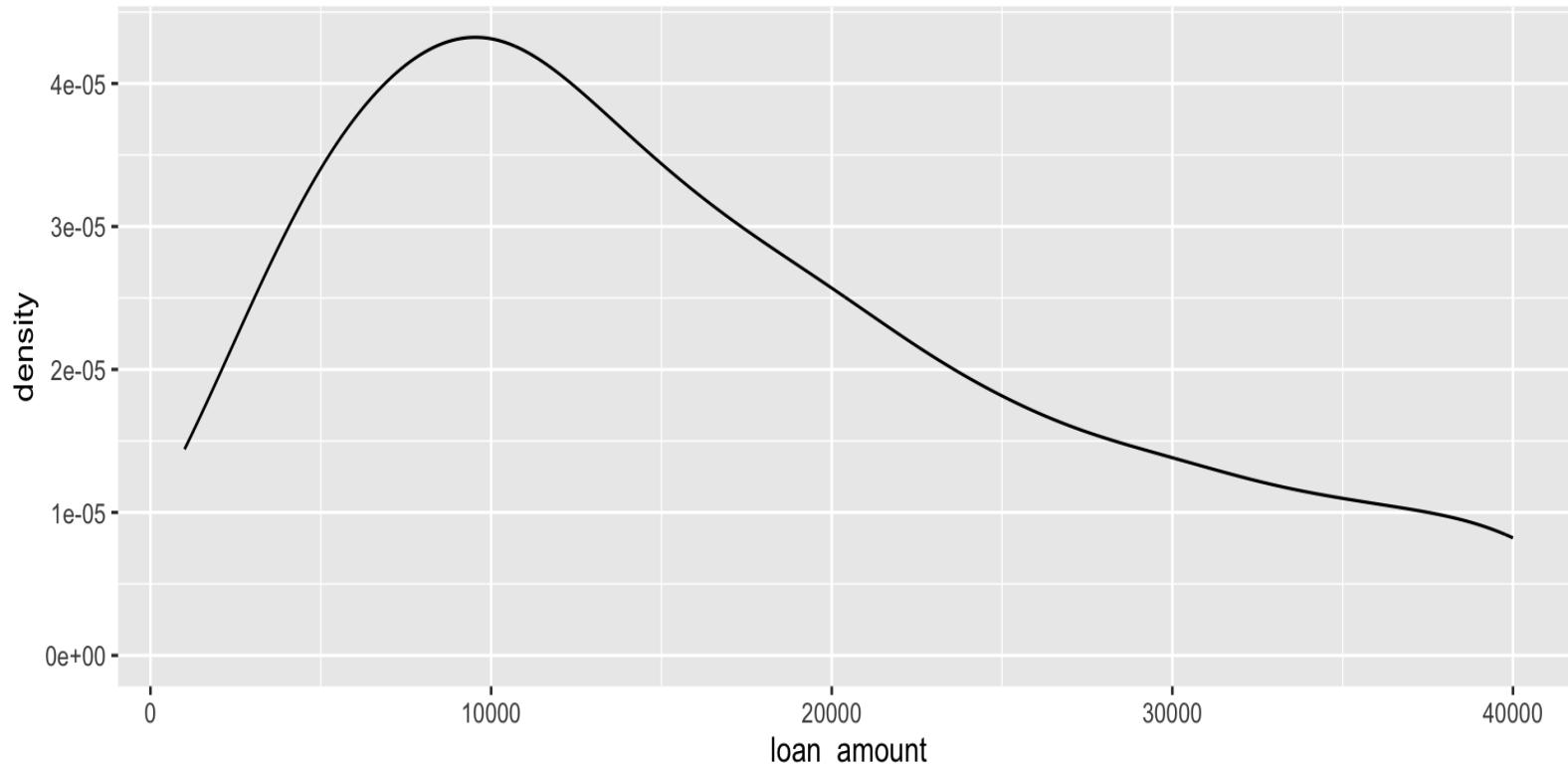
```
ggplot(loans, aes(x = loan_amount)) +  
  geom_density(adjust = 1) # default bandwidth
```

lowering resolution of plot



Density plots and adjusting bandwidth

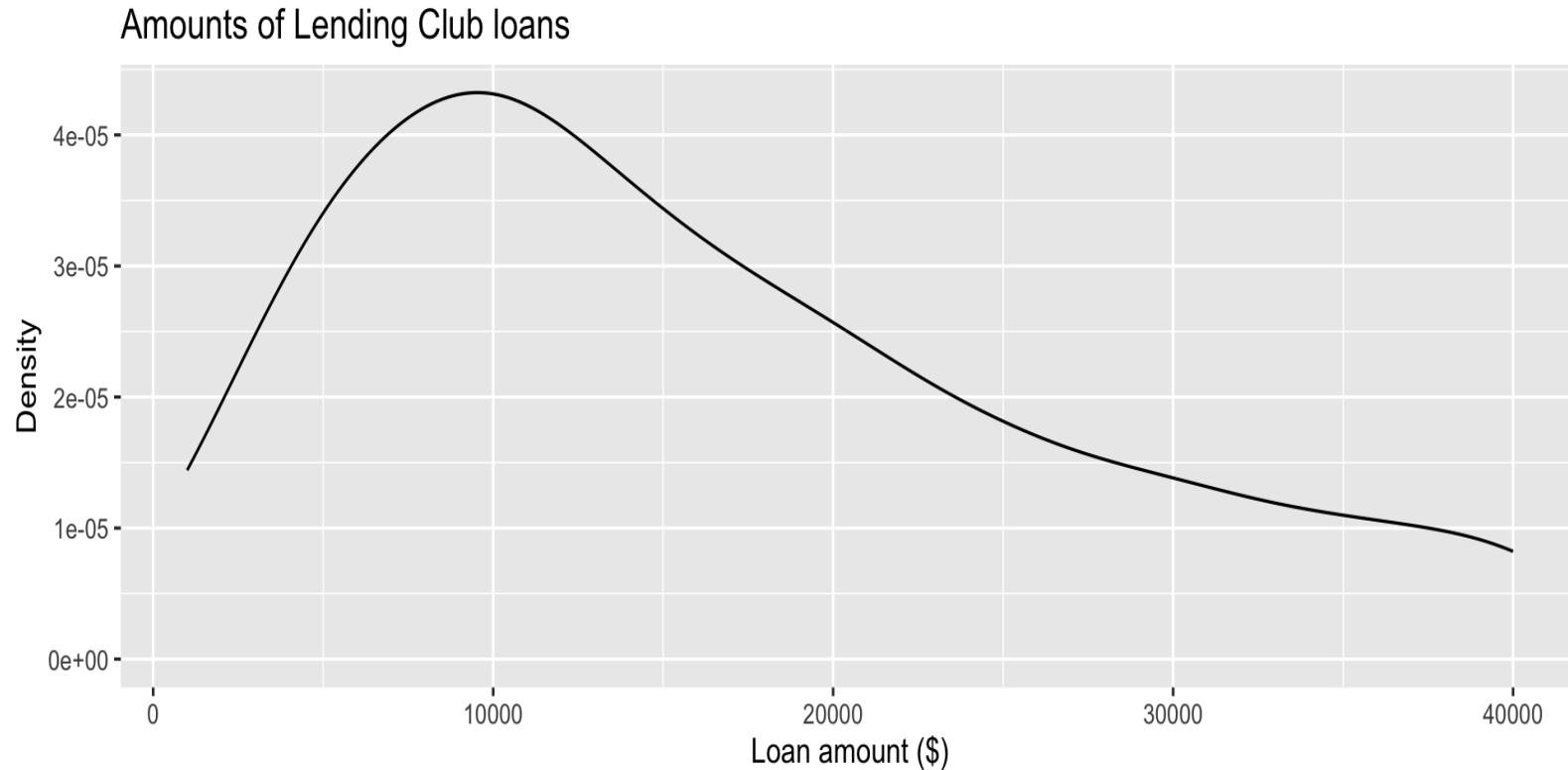
```
ggplot(loans, aes(x = loan_amount)) +  
  geom_density(adjust = 2)
```



Customizing density plots

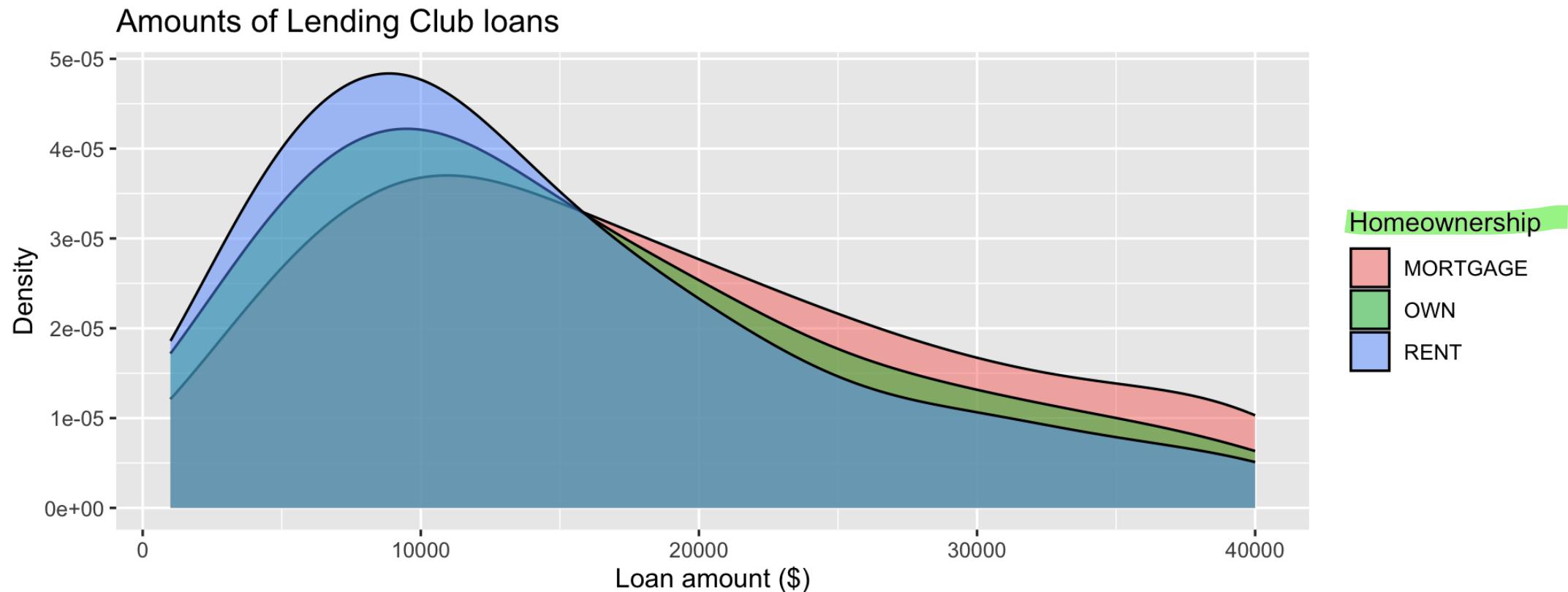
adding labels

```
ggplot(loans, aes(x = loan_amount)) +  
  geom_density(adjust = 2) +  
  labs( x = "Loan amount ($)", y = "Density", title = "Amounts of Lending Club loans" )
```



Adding a categorical variable

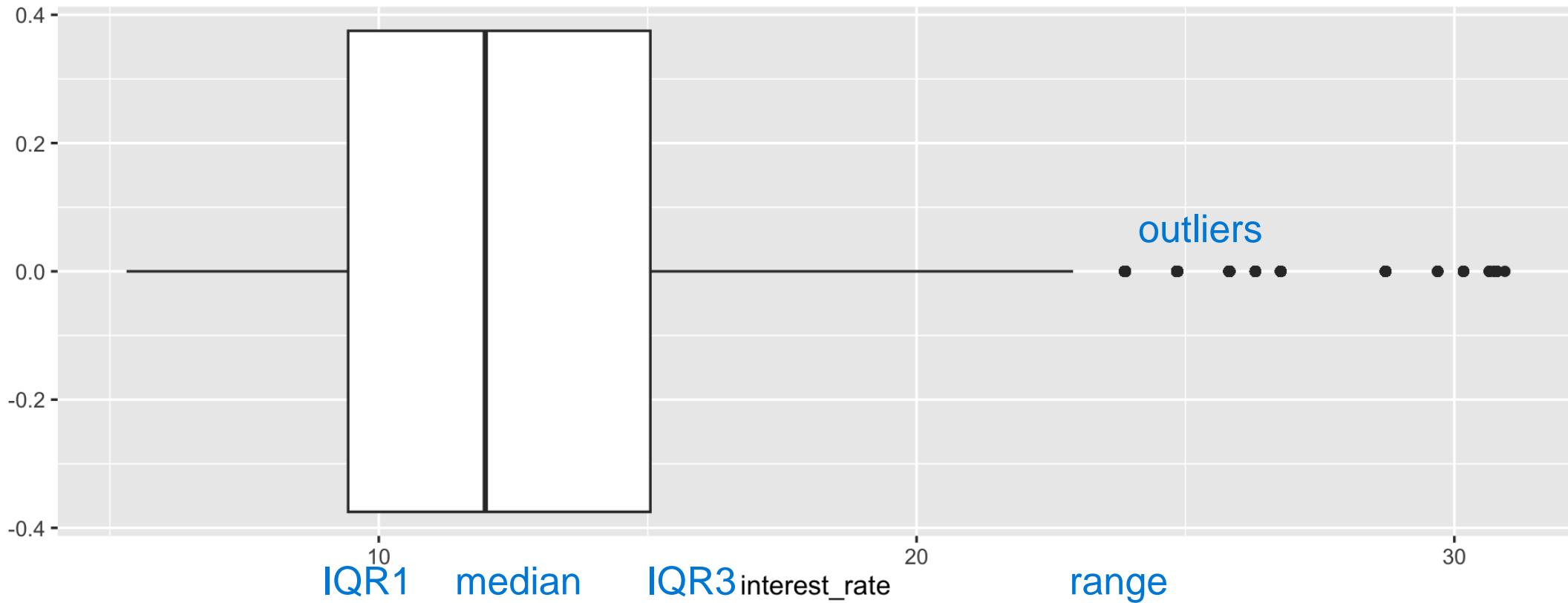
```
ggplot(loans, aes(x = loan_amount, fill = homeownership)) +  
  geom_density(adjust = 2, alpha = 0.5) +  
  labs(x = "Loan amount ($)", y = "Density", title = "Amounts of Lending Club loans", fill = "Homeow
```



Box plot

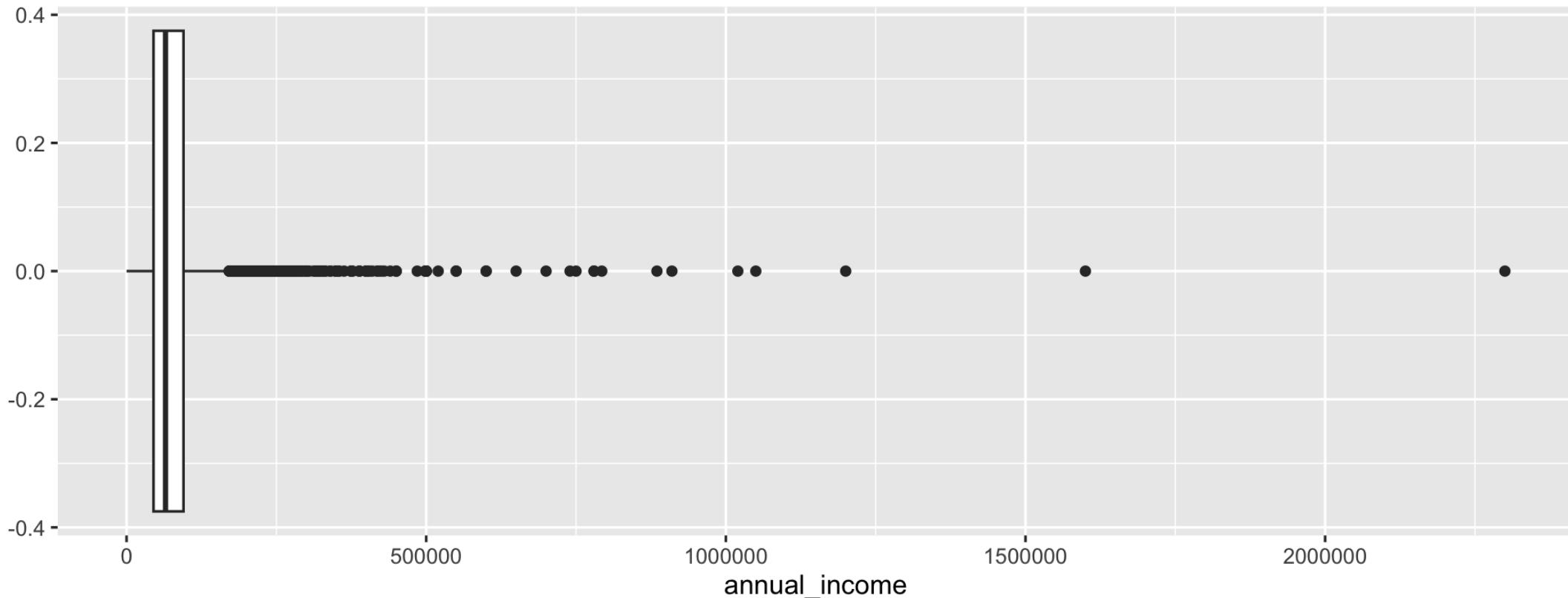
alt to histograms (freq) and density plots (prob density of proportion)

```
ggplot(loans, aes(x = interest_rate)) +  
  geom_boxplot()
```



Box plot and outliers

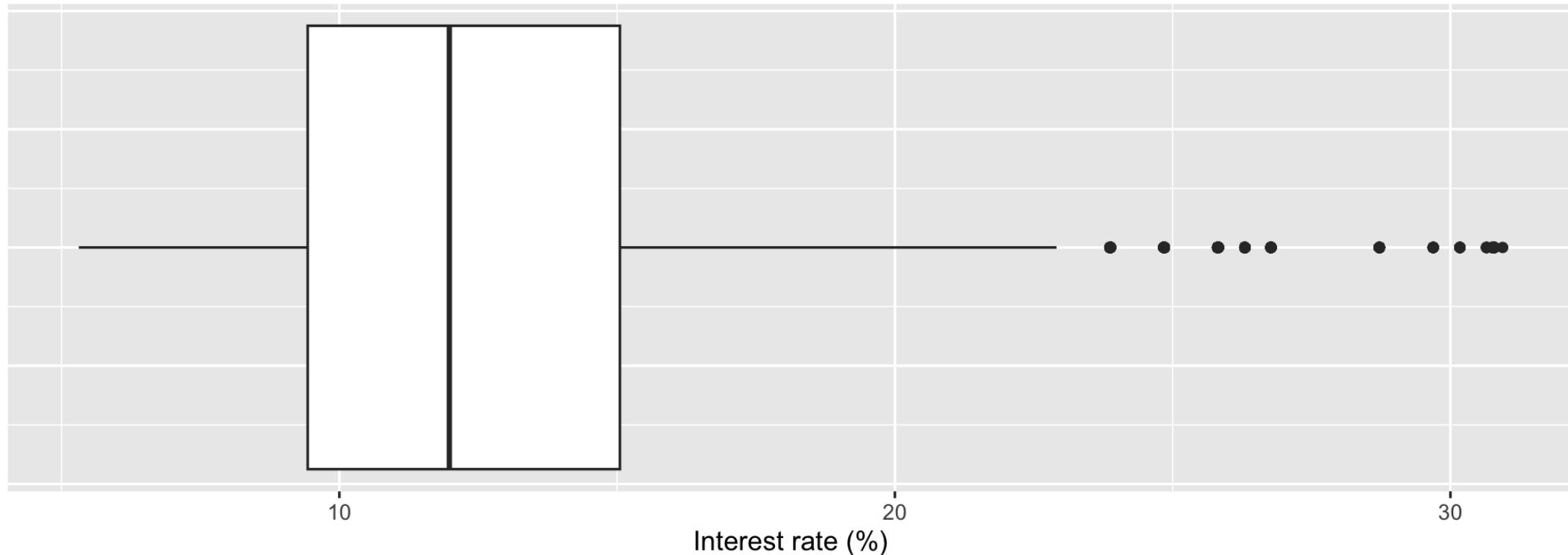
```
ggplot(loans, aes(x = annual_income)) +  
  geom_boxplot()
```



Customizing box plots

```
ggplot(loans, aes(x = interest_rate)) +geom_boxplot() +labs(x = "Interest rate (%)",y = NULL,  
title = "Interest rates of Lending Club loans") +  
theme( axis.ticks.y = element_blank(), axis.text.y = element_blank() )
```

Interest rates of Lending Club loans

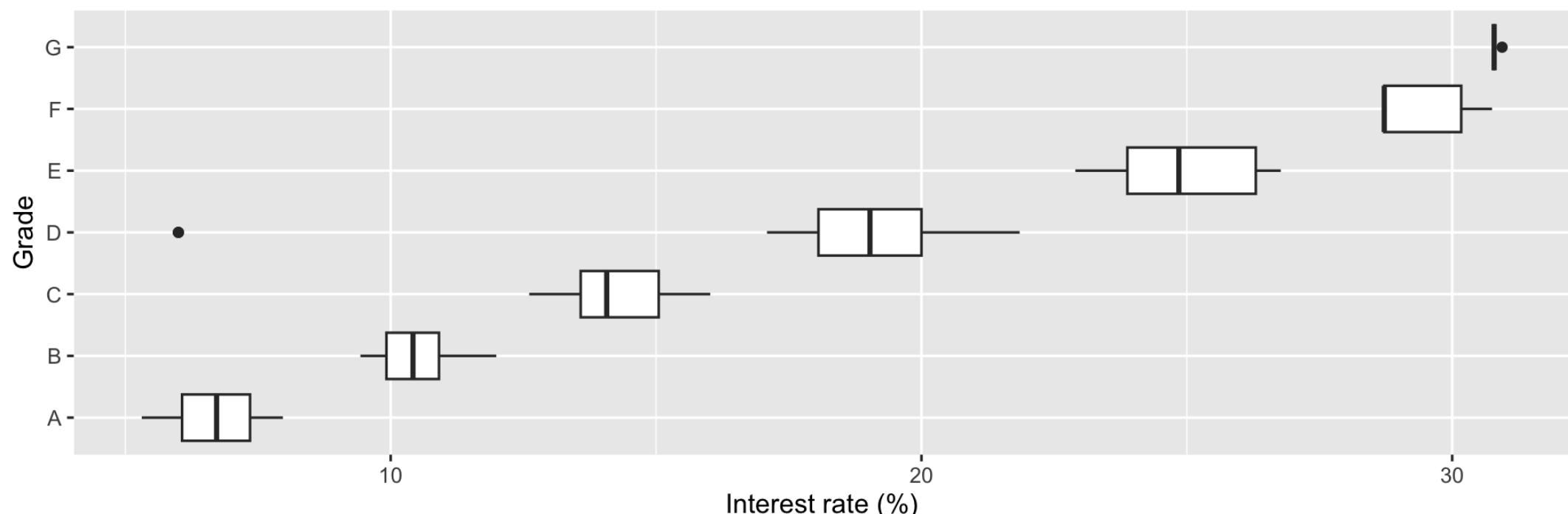


Adding a categoric variable

sort "int rate" by "grade of app"

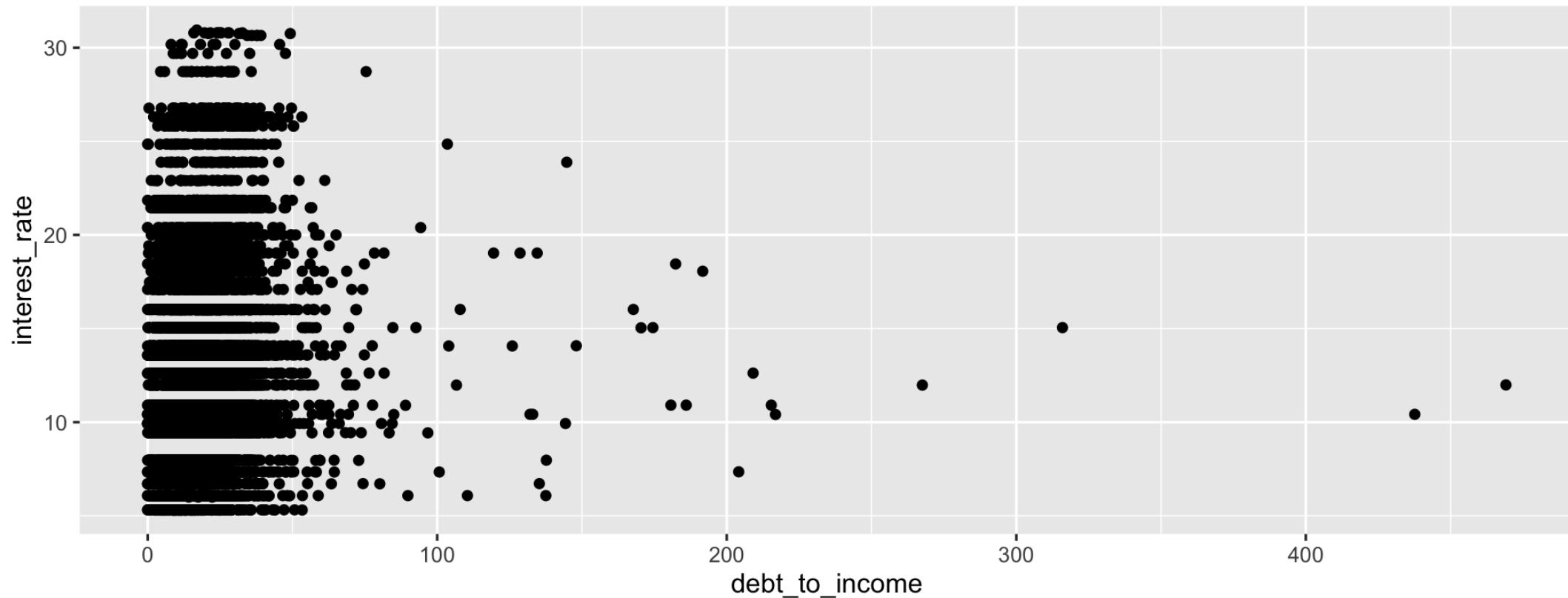
```
ggplot(loans, aes(x = interest_rate,  
                   y = grade)) +  
  geom_boxplot() +  
  labs(x = "Interest rate (%)", y = "Grade", title = "Interest rates of Lending Club loans", subtitle
```

Interest rates of Lending Club loans
by grade of loan



Scatterplot

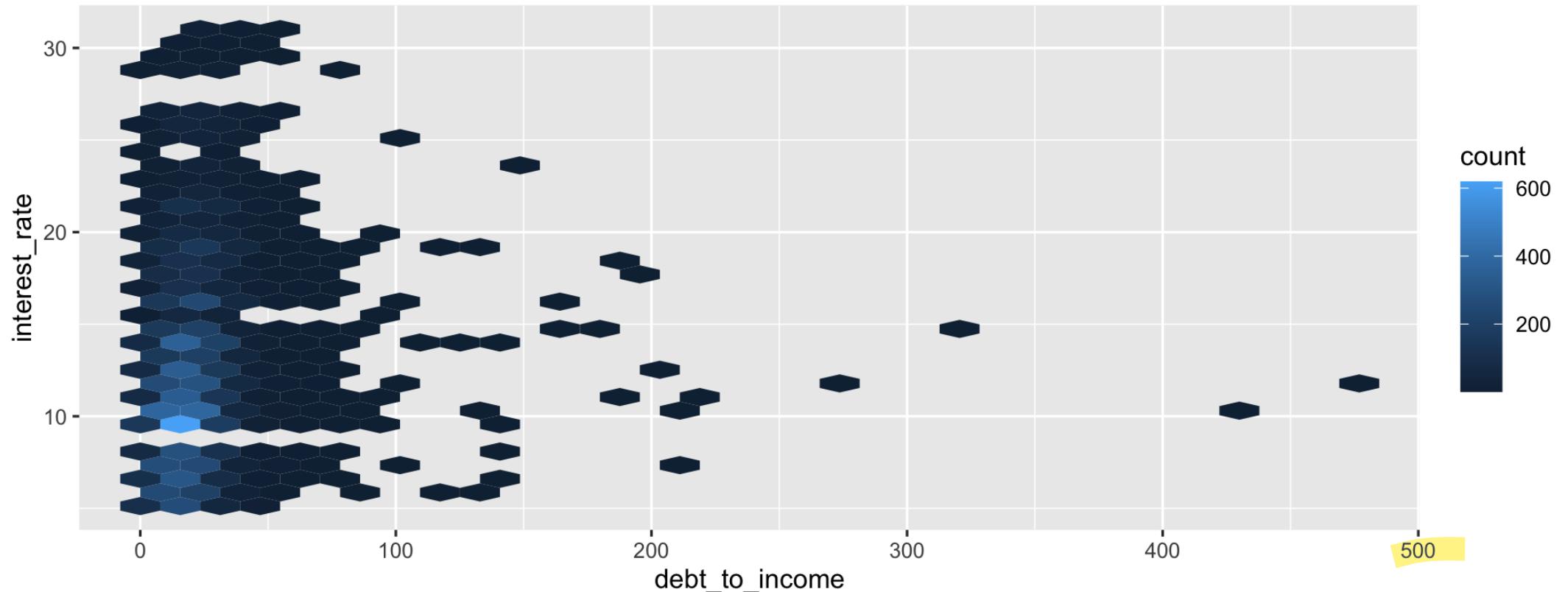
```
ggplot(loans, aes(x = debt_to_income, y = interest_rate)) +  
  geom_point()
```



Hex plot

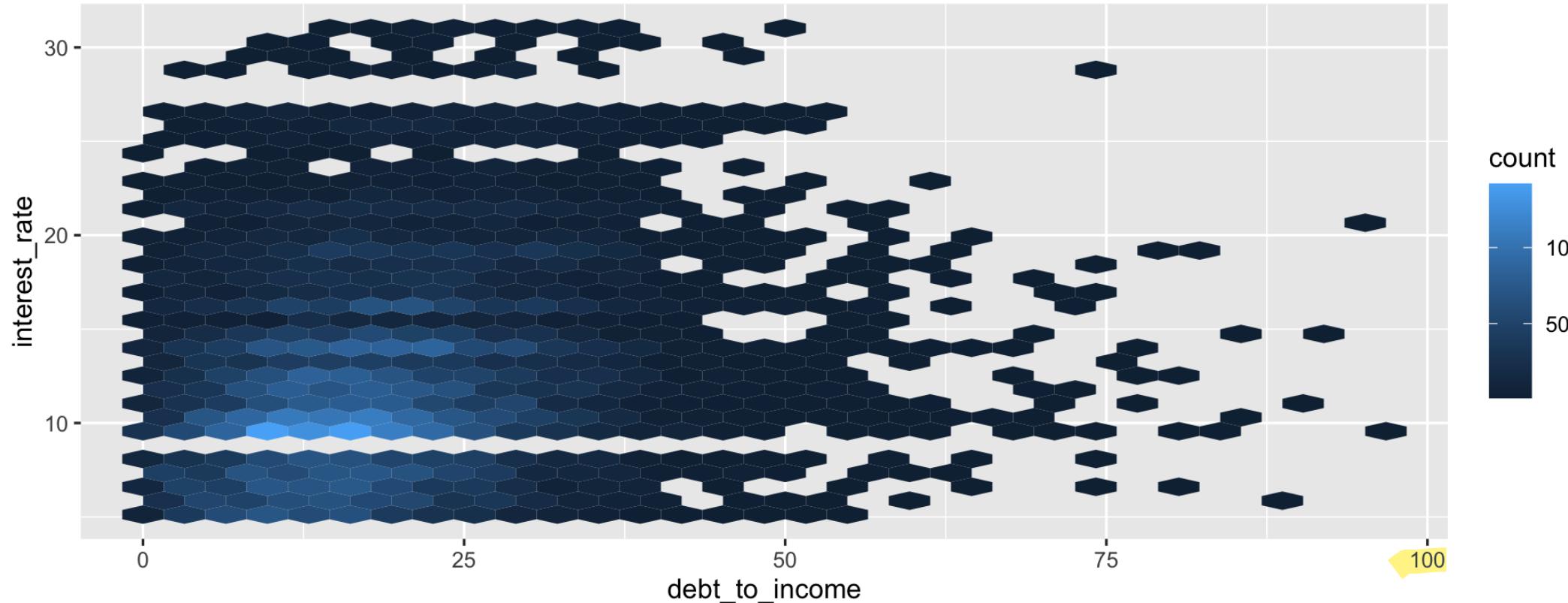
similar to histograms; instead of dots, colour coded hexagons are being used

```
ggplot(loans, aes(x = debt_to_income, y = interest_rate)) +  
  geom_hex()
```



Hex plot

```
ggplot(loans %>% filter(debt_to_income < 100),
       aes(x = debt_to_income, y = interest_rate)) +
  geom_hex()
```

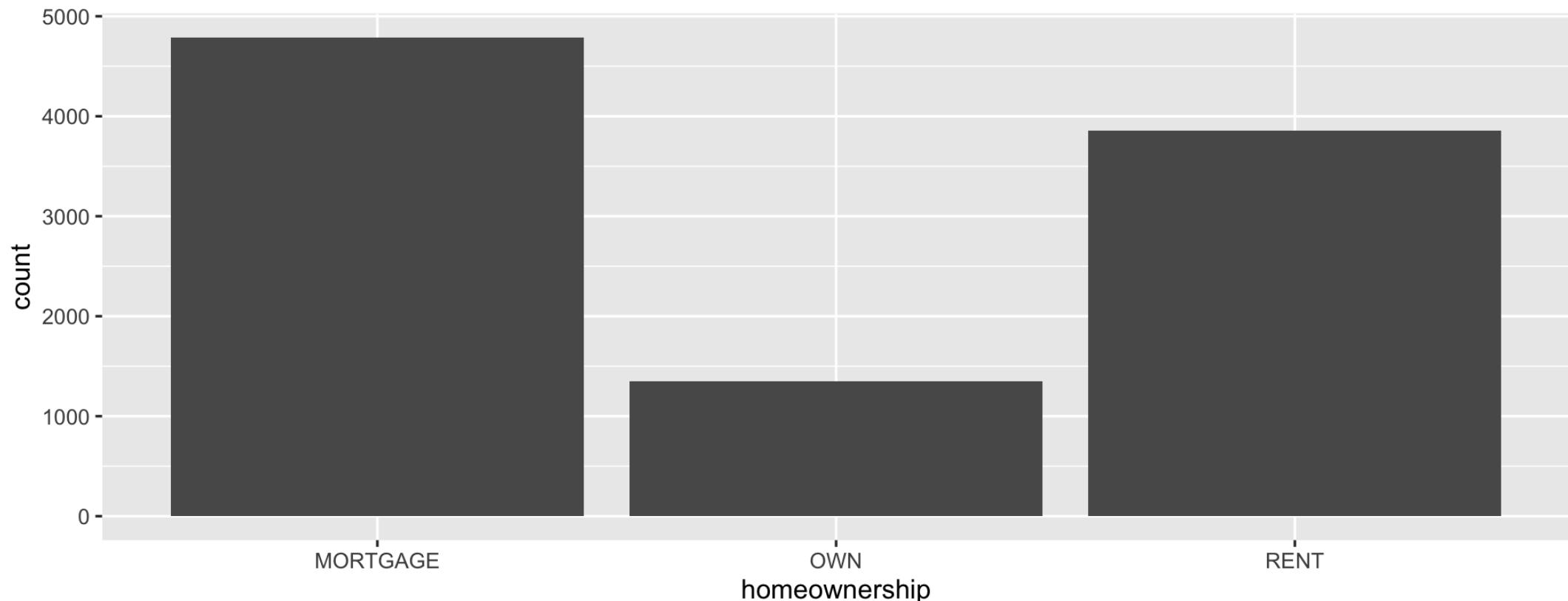


III. Visualizing categoric variables

Bar plot

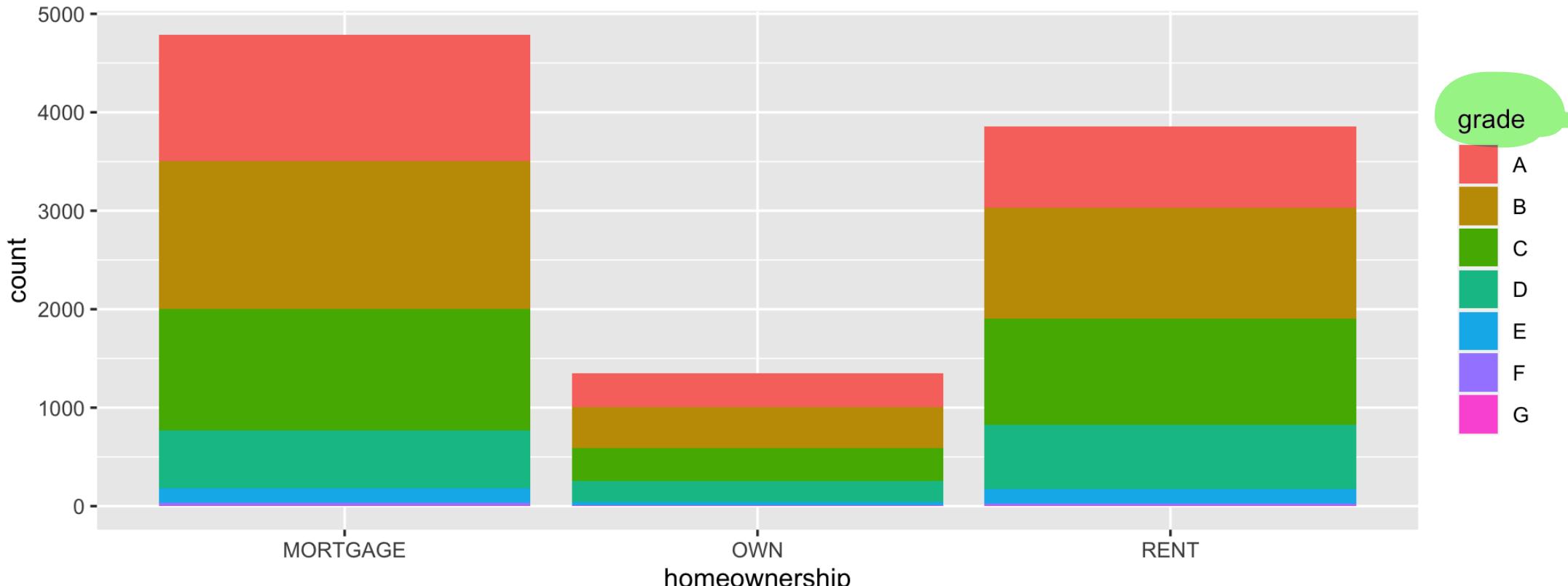
name of dataset cat nominal var

```
ggplot(loans, aes(x = homeownership)) +  
  geom_bar()
```



Segmented bar plot

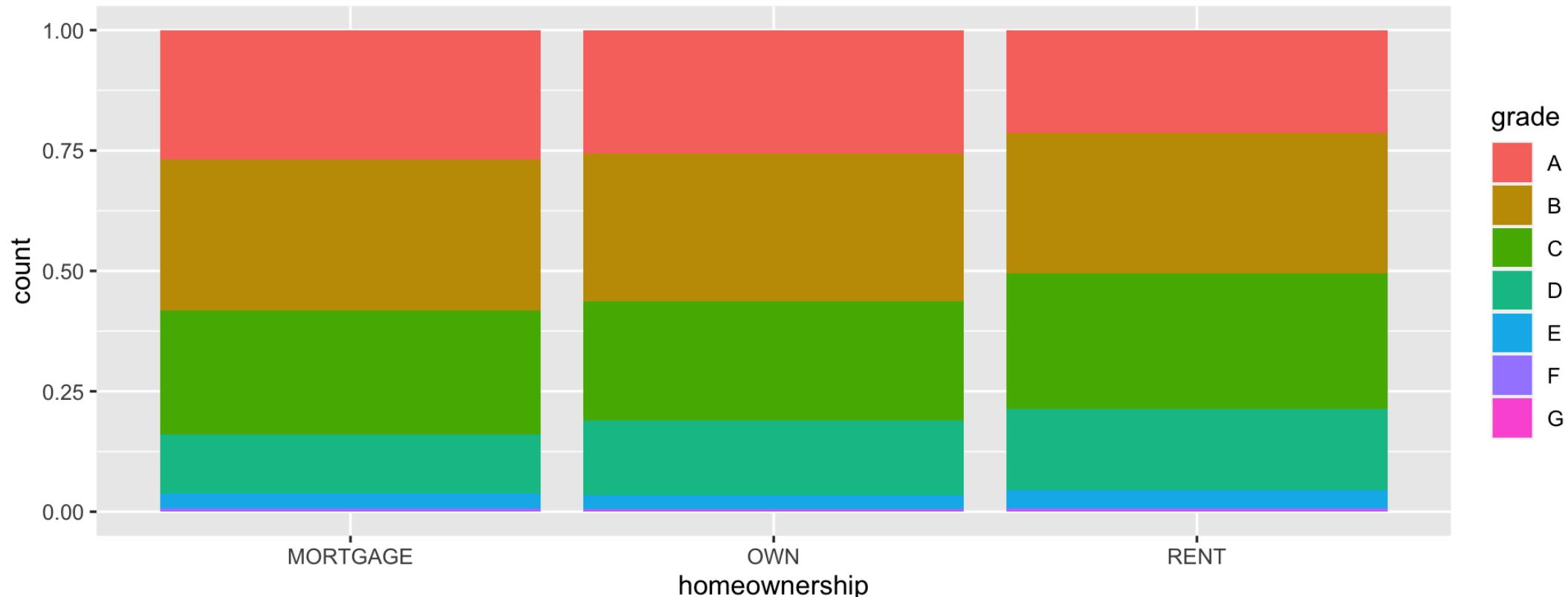
```
ggplot(loans, aes(x = homeownership,  
                   fill = grade)) +  
  geom_bar()
```



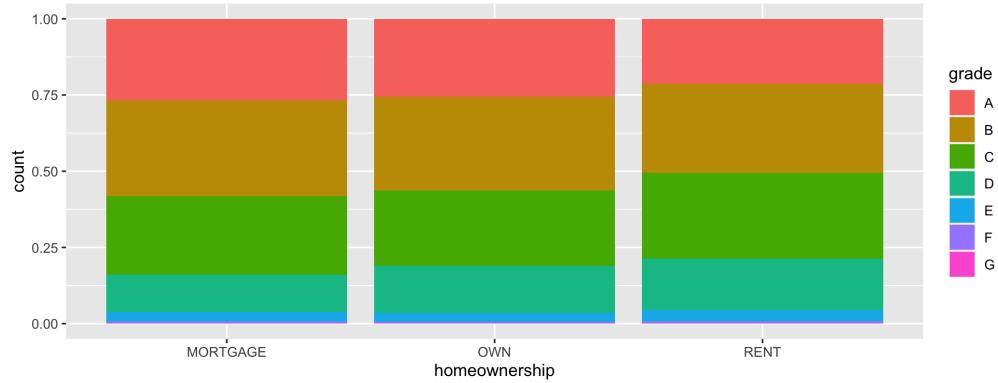
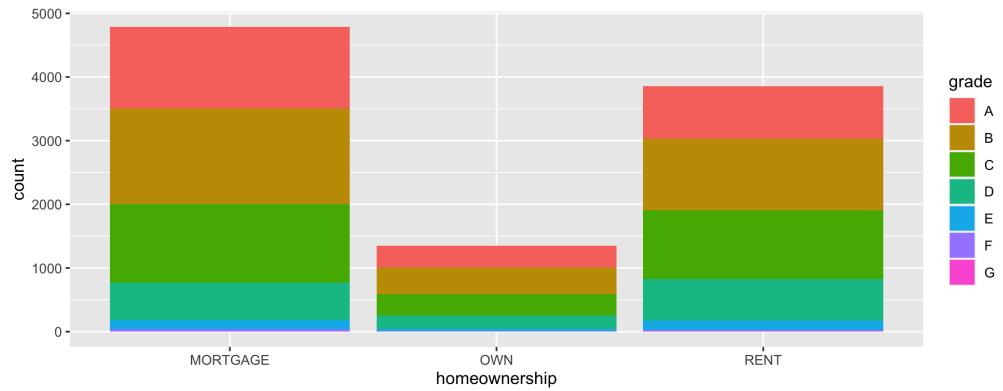
Segmented bar plot

making all bars of the same height for ezs comparison
count: not freq, it's proportion now

```
ggplot(loans, aes(x = homeownership, fill = grade)) +  
  geom_bar(position = "fill") note double quotes
```



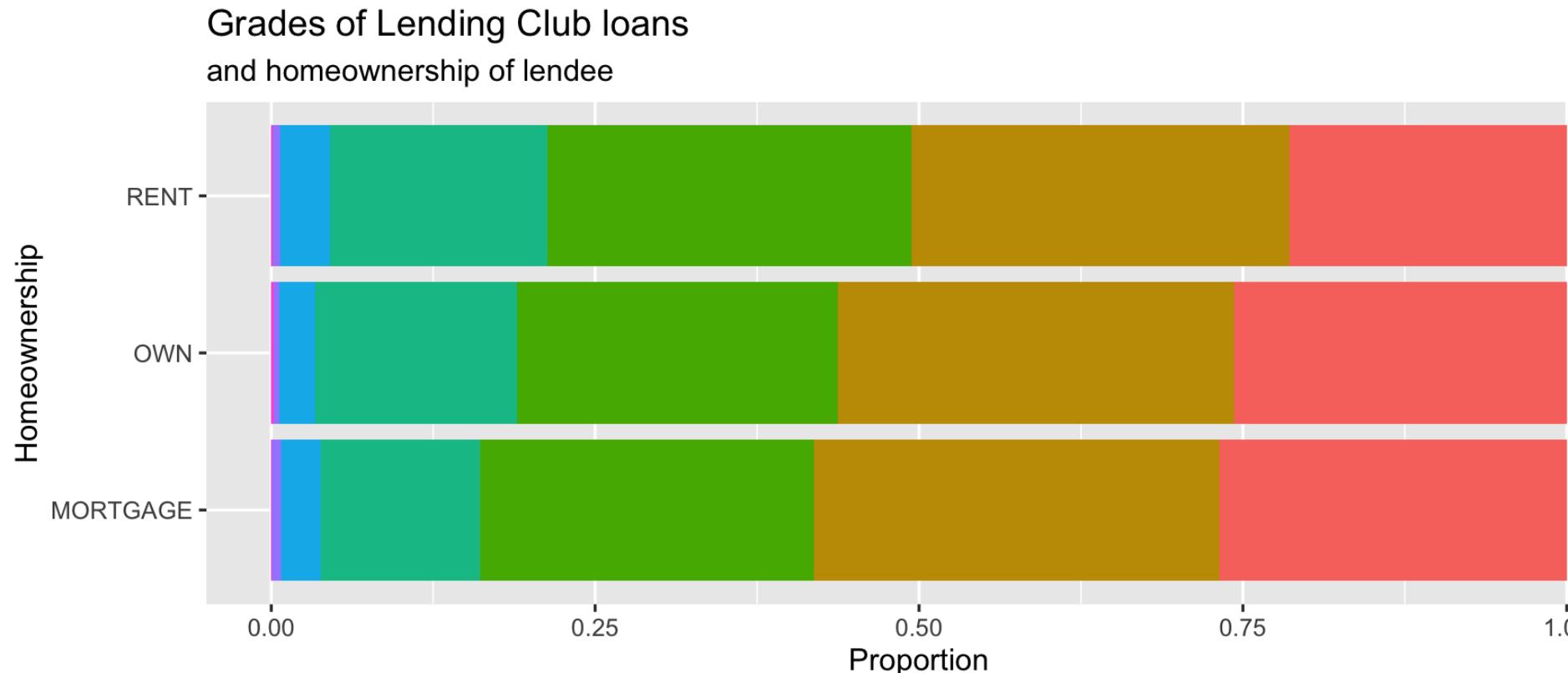
Comparison



Customizing bar plots

swapped rows w columns + adding labels

```
ggplot(loans, aes(y = homeownership, fill = grade)) + geom_bar(position = "fill") +
  labs( x = "Proportion", y = "Homeownership", fill = "Grade", title = "Grades of Lending Club loans"
```

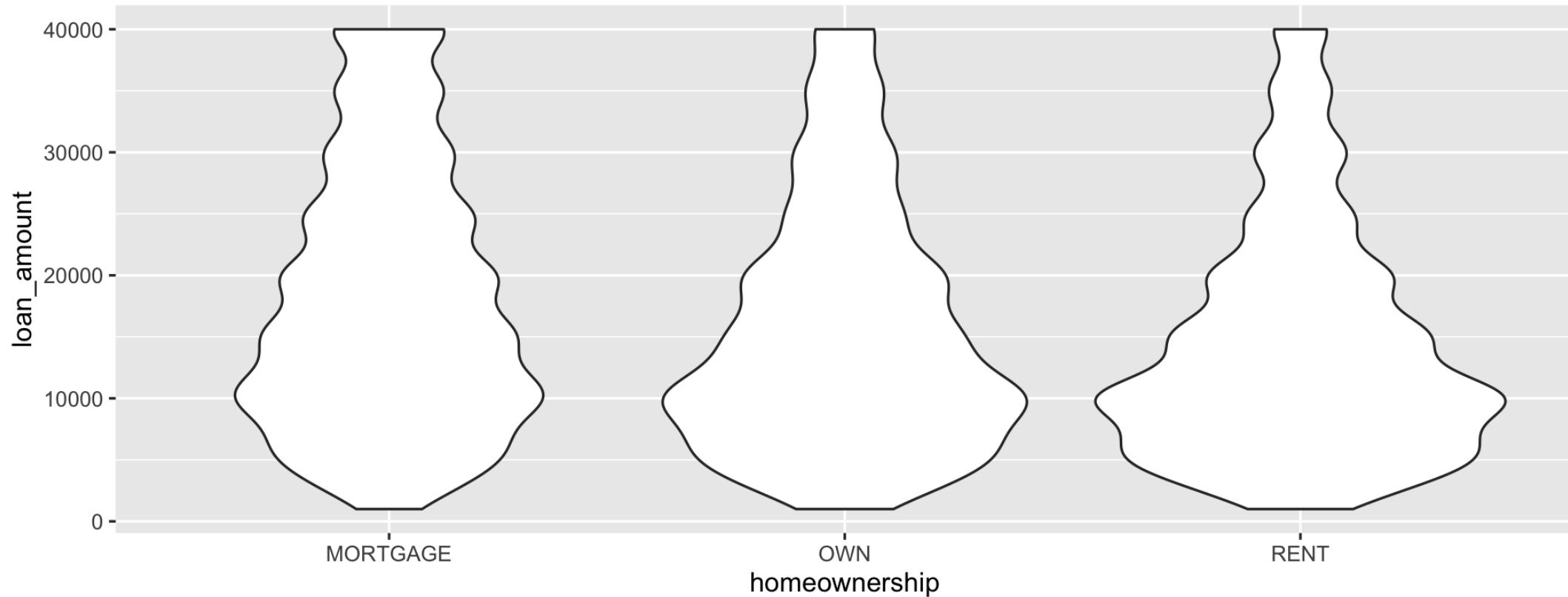


IV. Visualizing variables of varied types

Violin plots

similar to density plot but -> helps visualise rs btw var of diff types (eg numeric & cat)

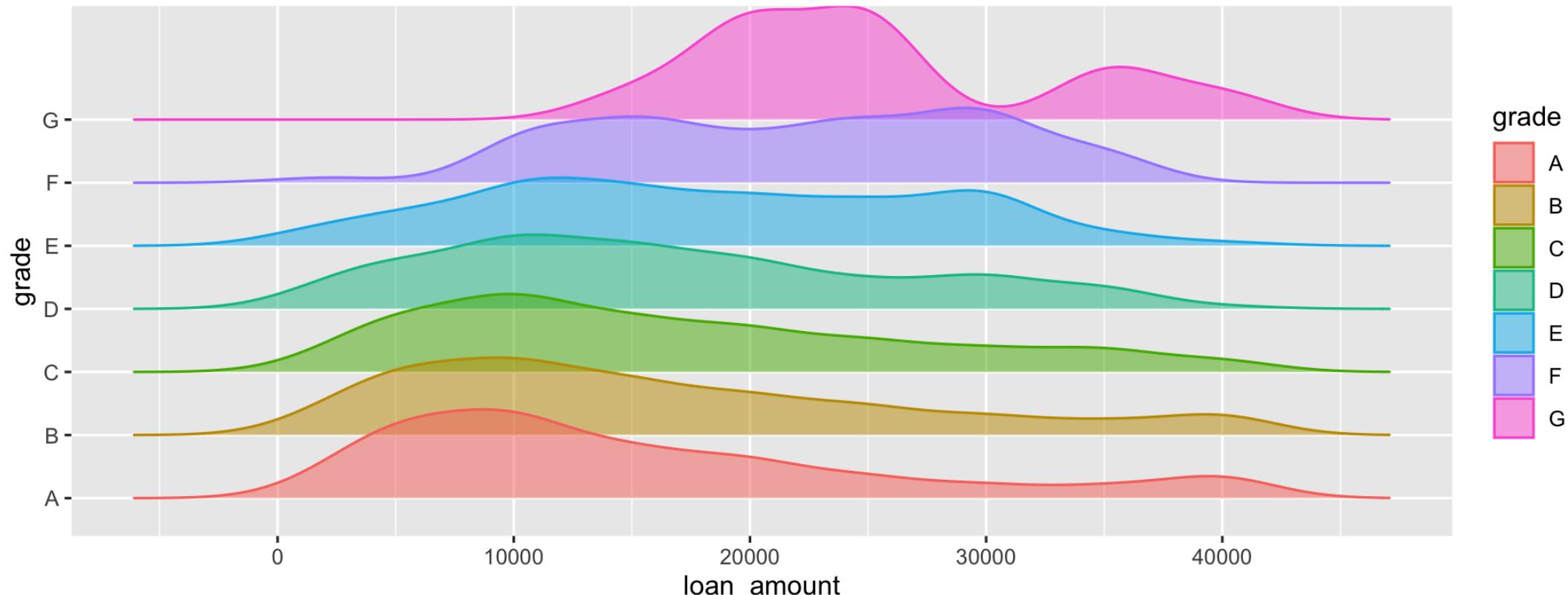
```
ggplot(loans, aes(x = homeownership, y = loan_amount)) +  
  geom_violin()
```



Ridge plots

density plots for diff grades; instead of superimposing them, now place them above one another

```
library(ggridges)
ggplot(loans, aes(x = loan_amount, y = grade, fill = grade, color = grade)) +
  geom_density_ridges(alpha = 0.5)
```



Thanks!

Slides created via the  packages:

xaringan
gadenbuie/xaringanthemer.



Faculty of Arts
& Social Sciences