Assessment 06 - Introduction to ggplot2

Phaneendra Valaboju - Harvard Data Science Professional

ggplot2 basics

Start by loading the dplyr and ggplot2 libraries as well as the `murders` data.

```
library(dplyr)
library(ggplot2)
library(dslabs)
data(murders)
```

Note that you can load both dplyr and ggplot2, as well as other packages, by installing and loading the tidyverse package.

With ggplot2 plots can be saved as objects. For example we can associate a dataset with a plot object like this

```
p <- ggplot(data = murders)
```

Because `data` is the first argument we don't need to spell it out. So we can write this instead:

```
p <- ggplot(murders)
```

or, if we load `dplyr`, we can use the pipe:

```
p <- murders %>% ggplot()
```

Remember the pipe sends the object on the left of `%>%` to be the first argument for the function the right of `%>%`.

Now let's get an introduction to `ggplot`. What

is the class of the object p?

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats': ##
##     filter, lag
```

```
## The following objects are masked from 'package:base': ##
##     intersect, setdiff, setequal, union
```

```
library(ggplot2)
library(dslabs)
data(heights)
data(murders)
p <- ggplot(murders)
class(p)
```

```
## [1] "gg"     "ggplot"
```

Printing

Remember that to print an object you can use the command `print` or simply type the object. For example, instead of

```
x <-2
print(x)
```

```
## [1] 2
```

you can simply type

```
x <-2
x
```

```
## [1] 2
```

Print the object `p` defined in exercise one

```
p <- ggplot(murders)
```

and describe what you see.

Possible Answers

- Nothing happens
- A blank slate plot [X]
- A scatter plot
- A histogram

Pipes

Now we are going to review the use of pipes by seeing how they can be used with `ggplot`.

Instructions

Using the pipe > %, %create an object `p` associated with the `heights` dataset instead of with the `murders` dataset as in previous exercises.

```
data(heights)
# define ggplot object called p like in the previous exercise but using a pipe
p <-heights  %>% ggplot ()
```

Layers

Now we are going to add layers and the corresponding aesthetic mappings. For the murders data, we plotted total murders versus population sizes in the videos.

Explore the `murders` data frame to remind yourself of the names for the two variables (total murders and population size) we want to plot and select the correct answer.

Possible Answers

- `state` and `abb`
- `total_murders` and `population_size`
- `total` and `population` [X]
- `murders` and `size`

geom_point 1

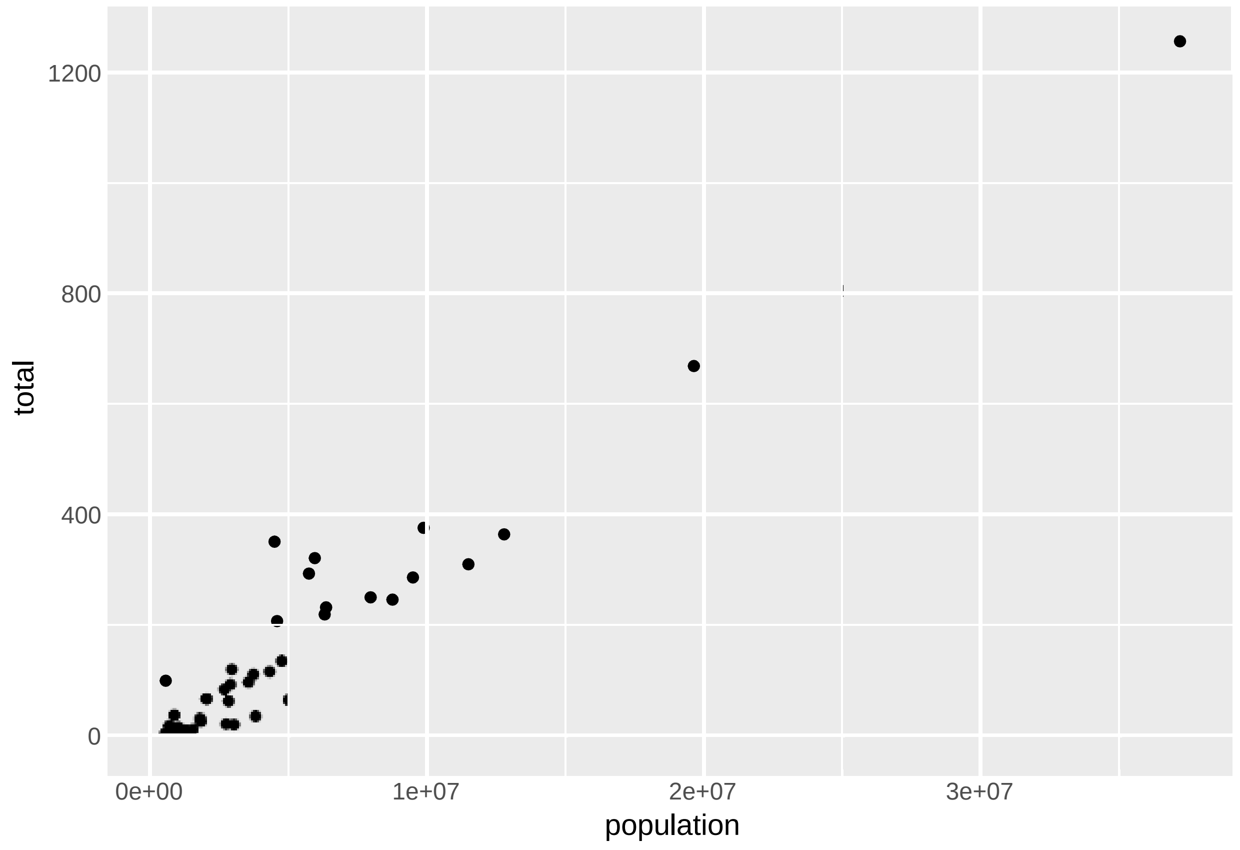
To create a scatter plot, we add a layer with the function geom_point. The aesthetic mappings require us to define the x-axis and y-axis variables respectively. So the code looks like this:

```
murders %>% ggplot(aes(x = , y =
  )) +geom_point()
```

except we have to fill in the blanks to define the two variables x and y. Instructions

Fill out the sample code with the correct variable names to plot total murders versus population size.

```
murders %>% ggplot (aes(x =population, y =total))       +
  geom_point()
```



geom_point 2

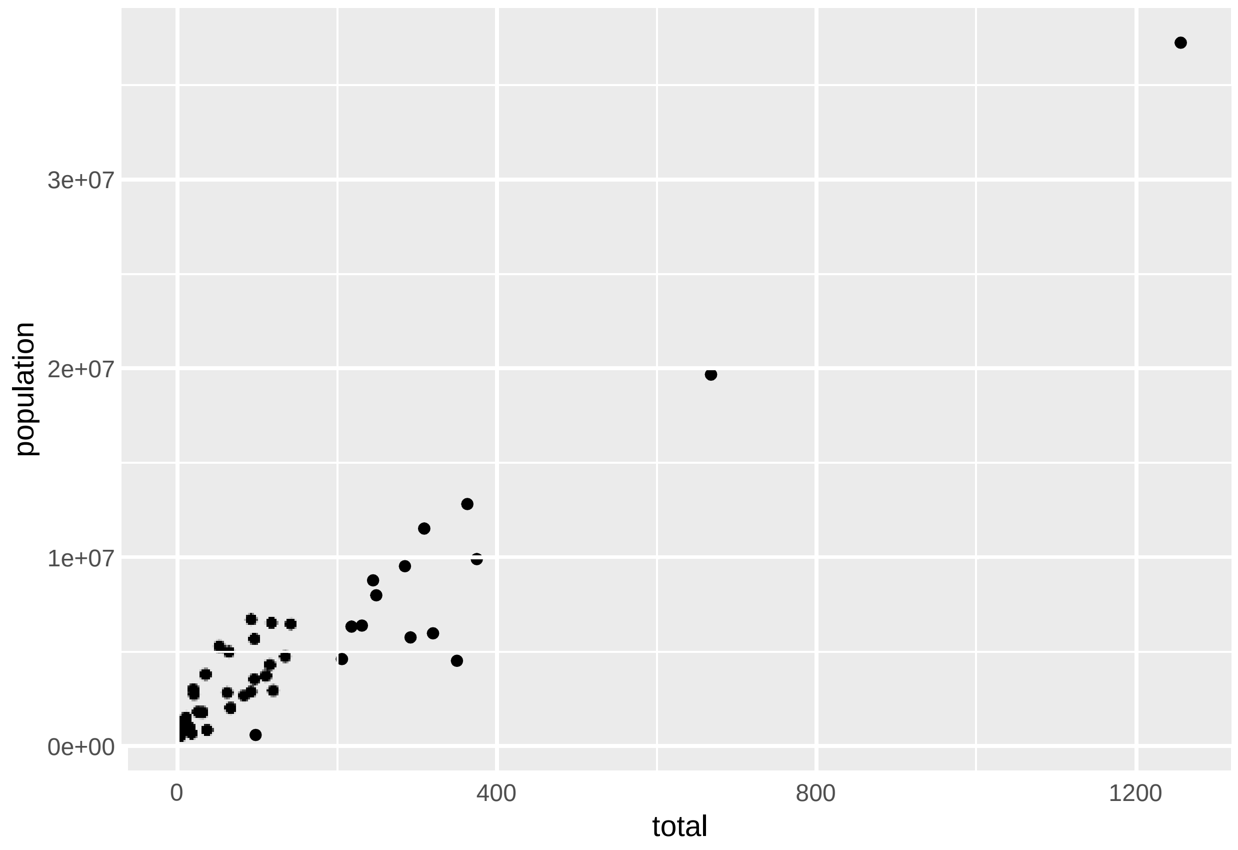
Note that if we don't use argument names, we can obtain the same plot by making sure we enter the variable names in the desired order:

```
murders %>% ggplot(aes(population,
  total)) +geom_point()
```

Instructions

Remake the plot but flip the axes so that `total` is on the x-axis and `population` is on the y-axis.

```
murders %>% ggplot (aes(x =total, y =population))       +
  geom_point()
```



geom_point text

If instead of points we want to add text, we can use the `geom_text()` or `geom_label()` geometries. However, note that the following code

```
murders %>% ggplot(aes(population,
  total)) geom_label()
```

will give us the error message: `Error: geom_label requires the following missing aesthetics: label`

Why is this?

Possible Answers

- We need to map a character to each point through the label argument in aes [X]
- We need to let `geom_label` know what character to use in the plot
- The `geom_label` geometry does not require x-axis and y-axis values.
- `geom_label` is not a ggplot2command

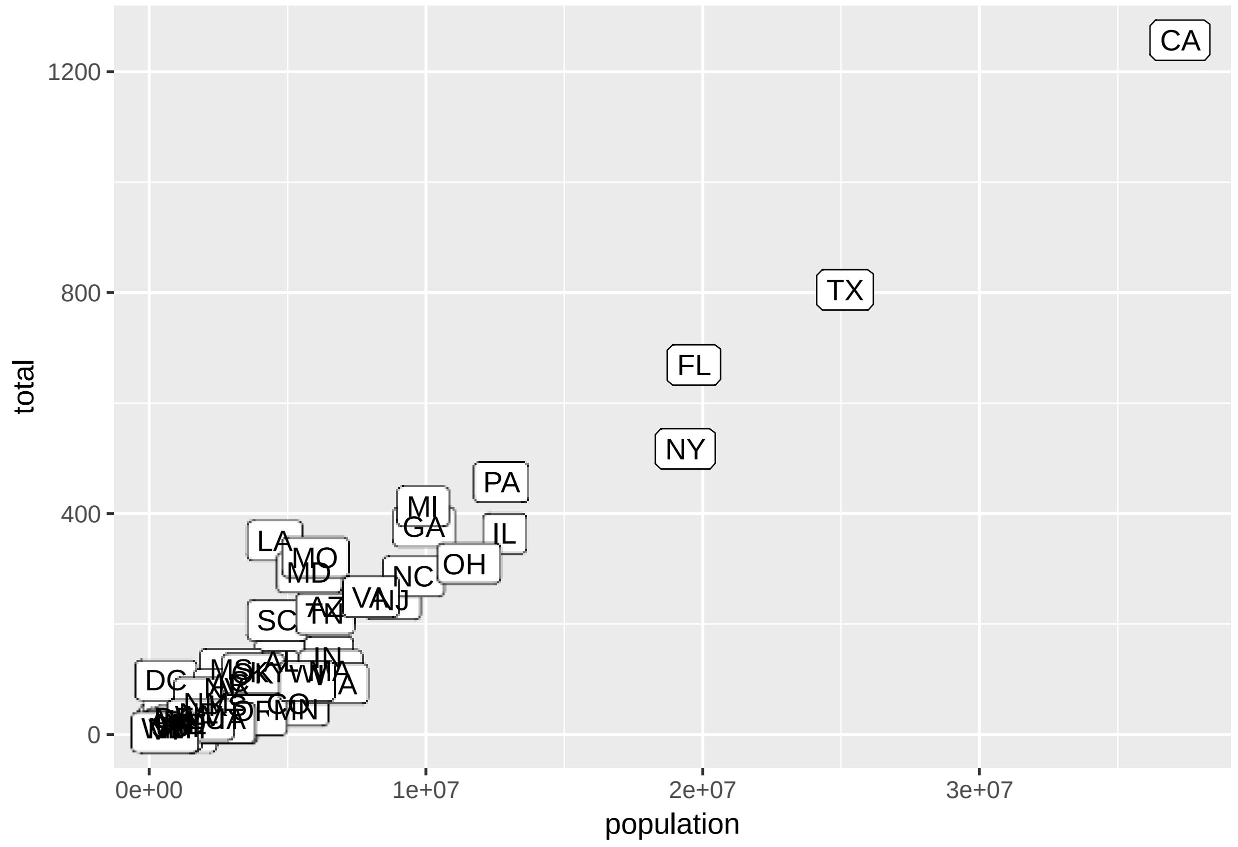
geom_point text

You can also add labels to the points on a plot.

Instructions

Rewrite the code from the previous exercise to add the state abbreviation as the label through `aes`.

```
library(dplyr)
library(ggplot2)
library(dslabs)
data(heights)
data(murders)
murders %>% ggplot (aes(population, total, label =abb))     +
  geom_label()
```



geom_point colors

Now let's change the color of the labels to blue. How can we do this? Possible

Answers

- By adding a column called `blue` to `murders`
- By mapping the colors through `aes` because each label needs a different color
- By using the `color` argument in`ggplot`
- By using the `color` argument in `geom_label` because we want all colors to be blue so we do not need to map colors [X]

geom_point colors 2

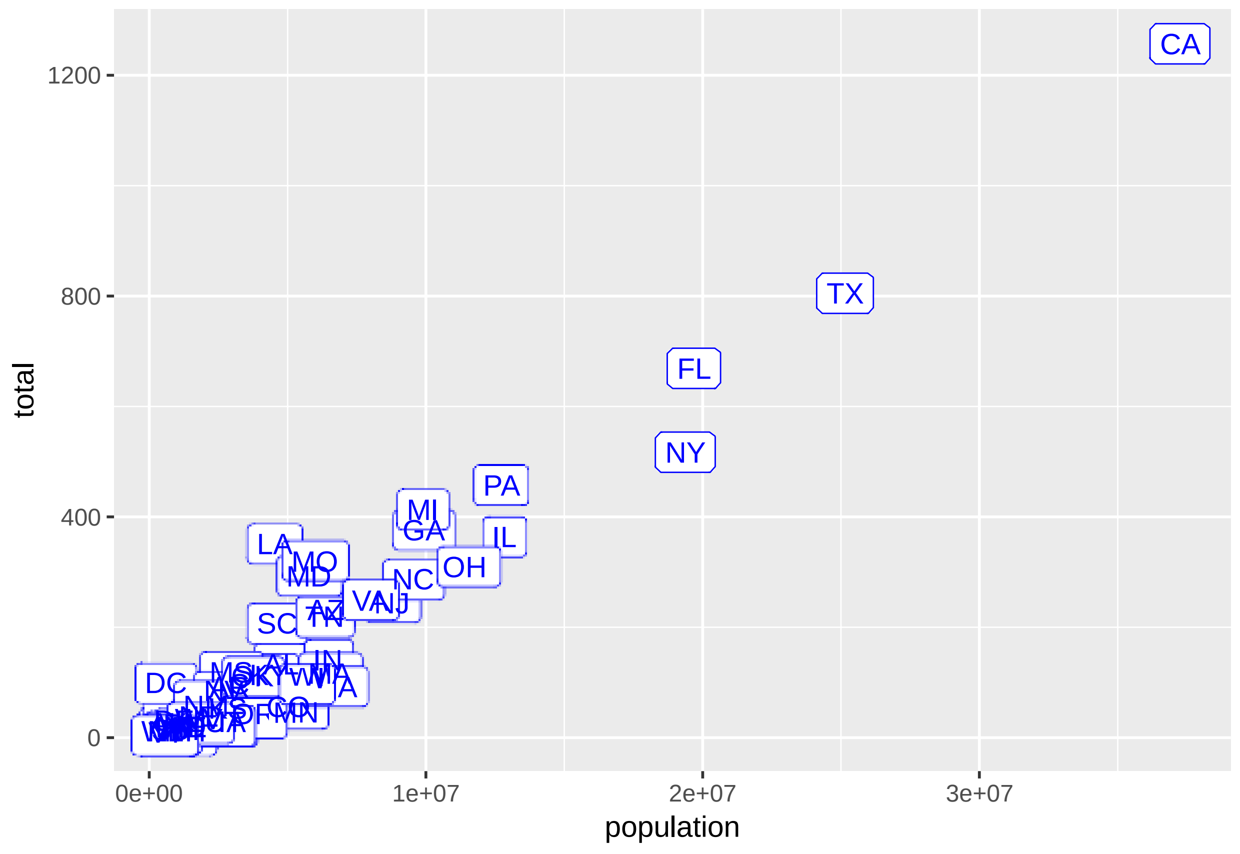
Now let's go ahead and make the labels blue. We previously wrote this code to add labels to our plot:

```
murders %>% ggplot(aes(population, total, label=
  abb)) geom_label()
Now we will edit this code.
```

Instructions

Rewrite the code above to make the labels blue by adding an argument to `geom_label`.

```
murders %>% ggplot (aes(population, total,label=abb))   +
  geom_label(color='blue')
```



geom_labels by region

Now suppose we want to use color to represent the different regions. So the states from the West will be one color, states from the Northeast another, and so on. In this case, which of the following is most appropriate:

Possible Answers

- Adding a column called `color` to `murders` with the color we want to use
- Mapping the colors through the color argument of `aes` because each label needs a different color
- Using the `color` argument in `ggplot`
- Using the `color` argument in `geom_label` because we want all colors to be blue so we do not need to map colors

geom_label colors

We previously used this code to make a plot using the state abbreviations as labels:

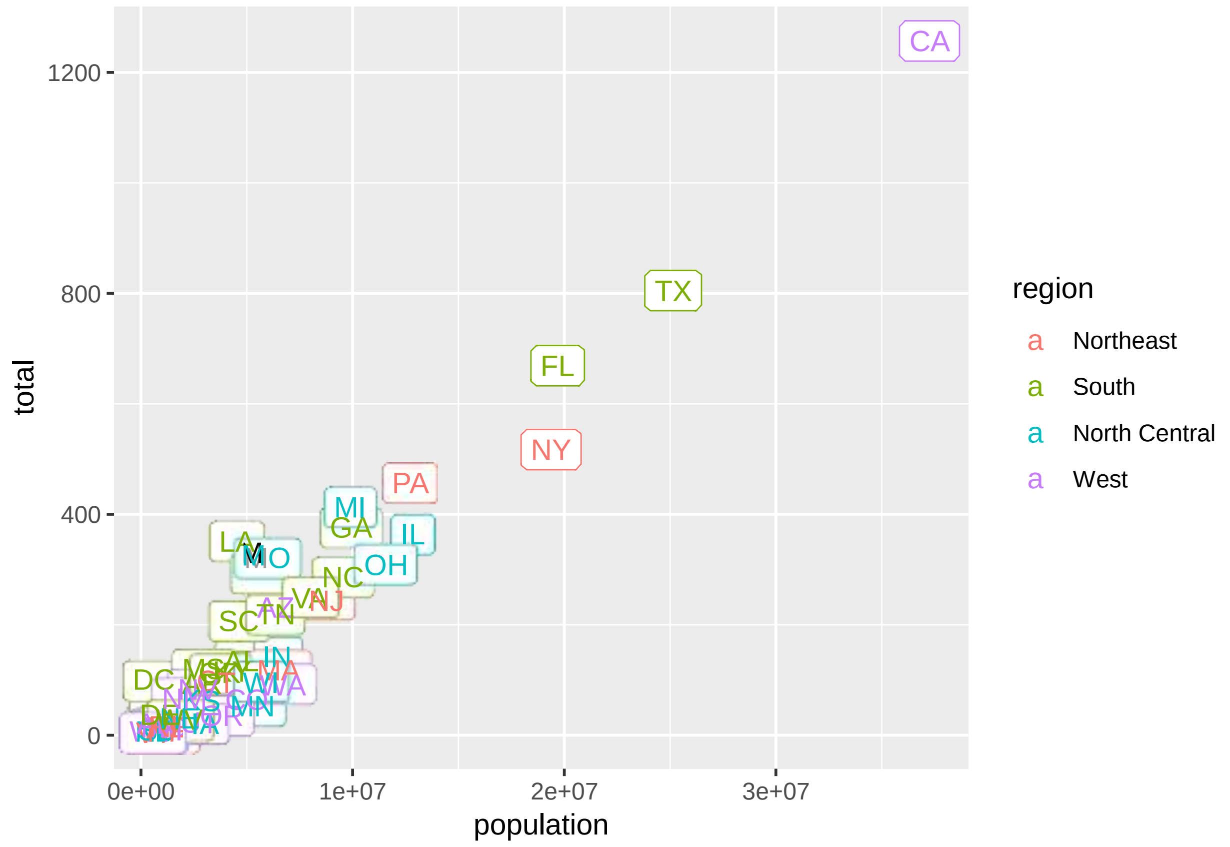
```
murders %>% ggplot(aes(population, total, label =
  abb)) +geom_label()
```

We are now going to add color to represent the region.

Instructions

Rewrite the code above to make the label color correspond to the state's region. Because this is a mapping, you will have to do this through the `aes` function.

```
## edit this code
murders %>% ggplot (aes(population, total, label =abb, color=region))      +
  geom_label()
```



Log-scale

Now we are going to change the axes to log scales to account for the fact that the population distribution is skewed. Let's start by defining an object p that holds the plot we have made up to now:

```
p <- murders %>% ggplot(aes(population, total, label = abb, color =
  region)) geom_label()
```

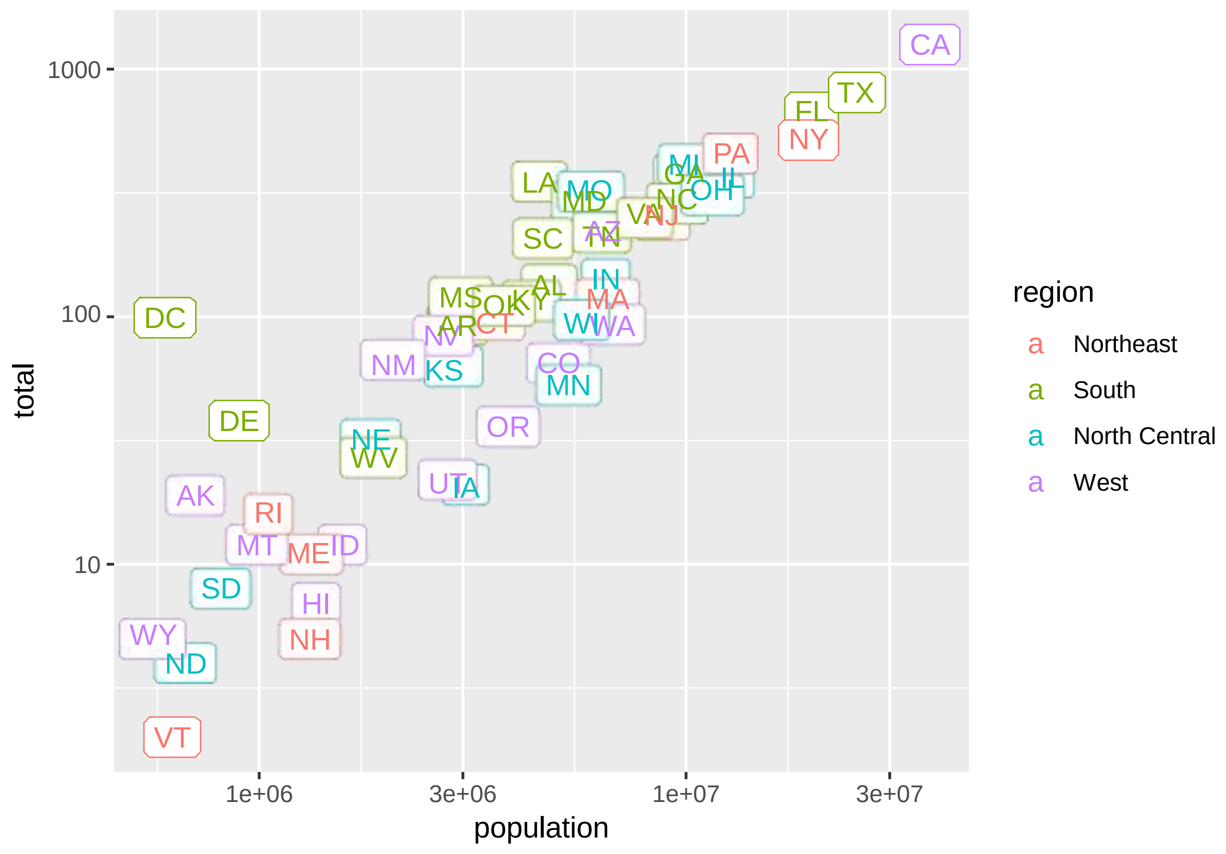
To change the x-axis to a log scale we learned about the `scale_x_log10()` function. We can change the axis by adding this layer to the object p to change the scale and render the plot using the following code:

```
p + scale_x_log10()
```

Instructions

Change both axes to be in the log scale. Make sure you do not redefine p - just add the appropriate layers.

```
p <-murders  %>% ggplot (aes(population, total, label =abb, color =region))      +geom_label ()
## add layers to p here
p +scale_x_log10 () +scale_y_log10 ()
```



Titles

In the previous exercises we created a plot using the following code:

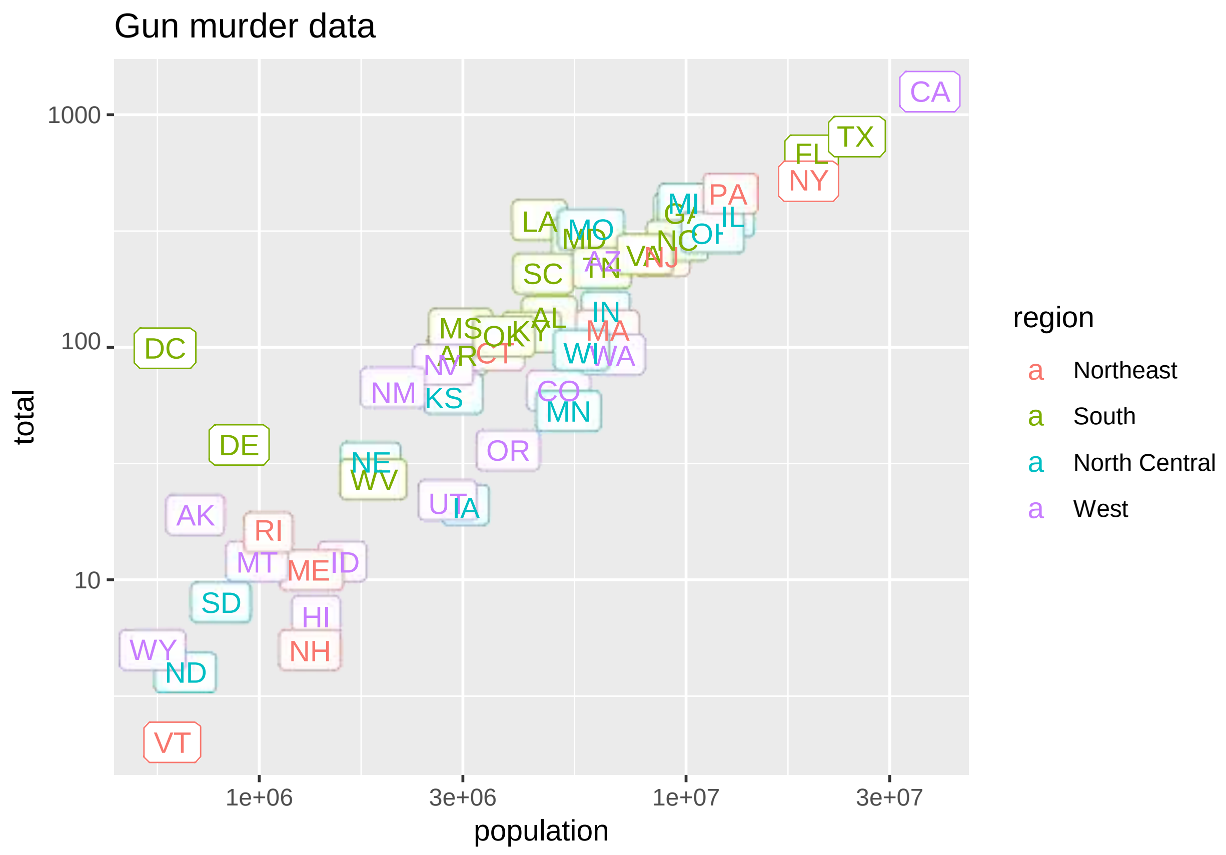
```
library(dplyr)
library(ggplot2)
library(dslabs)
data(murders)
p<- murders %>% ggplot(aes(population, total, label = abb, color =
  region)) geom_label()
p + scale_x_log10() + scale_y_log10()
```

We are now going to add a title to this plot. We will do this by adding yet another layer, this time with the function `ggtitle`.

Instructions

Edit the code above to add the title "Gun murder data" to the plot.

```
p <-murders  %>% ggplot (aes(population, total, label =abb, color =region))        +
  geom_label()
# add a layer to add title to the next line
p +scale_x_log10 () +
    scale_y_log10() + ggtitle('Gun
    murder data')
```



Histograms

We are going to shift our focus from the `murders` dataset to explore the `heights` dataset.

We use the `geom_histogram` function to make a histogram of the heights in the `heights` data frame. When reading the documentation for this function we see that it requires just one mapping, the values to be used for the histogram.

What is the variable containing the heights in inches in the `heights` data frame? Possible

Answers

- `sex`
- `heights`
- `height` [X]
- `heights$height`

A second example

We are now going to make a histogram of the `heights` so we will load the heights dataset. The following code has been pre-run for you to load the heights dataset:

```
library(dplyr)
library(ggplot2)
library(dslabs)
data(heights)
```

Instruction

- Create a ggplot object called p using the pipe to assign the heights data to a ggplot object.
- Assign height to the x values through the aes function.

```
# define p here
p <-heights  %>% ggplot () +aes (x =height)
```

Histograms 2

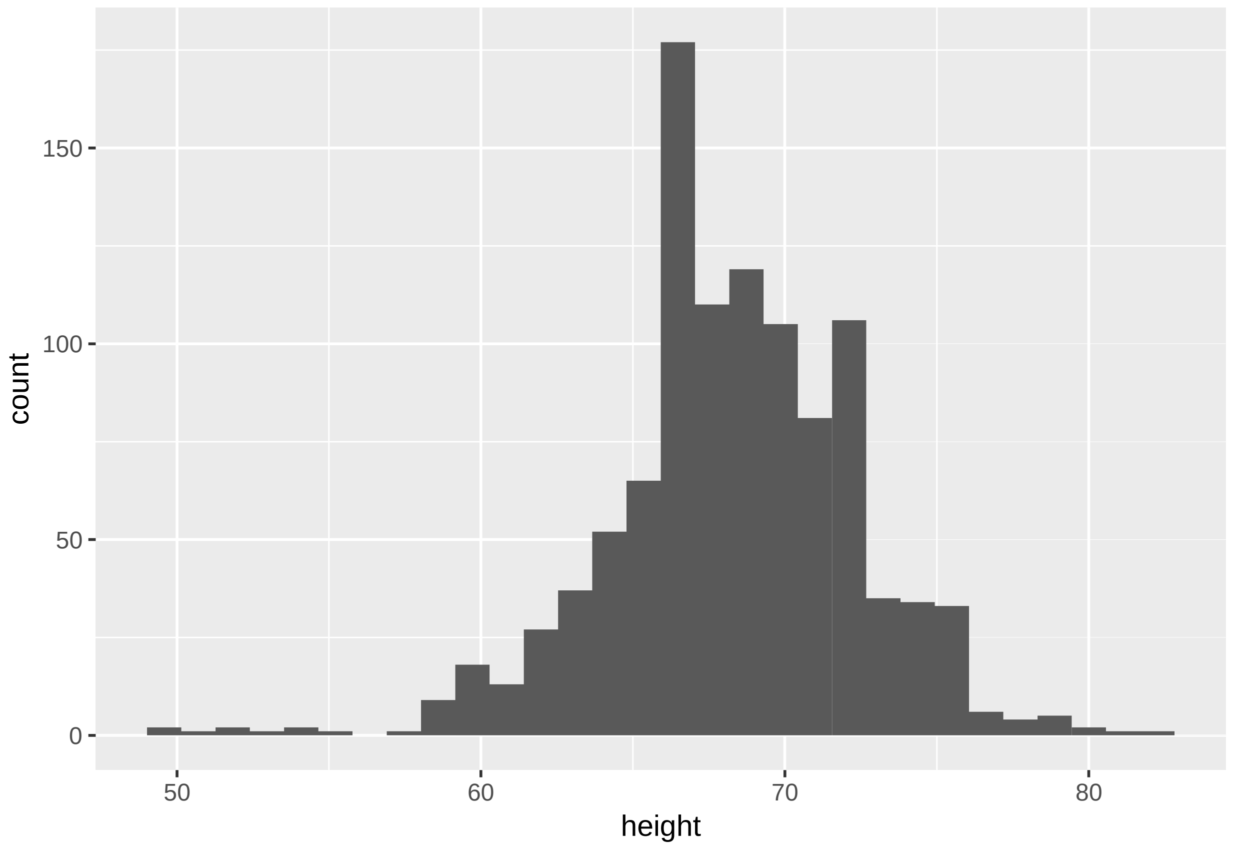
Now we are ready to add a layer to actually make the histogram.

Instructions

Add a layer to the object p(created in the previous exercise) using the geom_histogram function to make the histogram.

```
p <-heights  %>%
  ggplot(aes(height))
## add a layer to p
p +geom_histogram ()
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Histogram binwidth

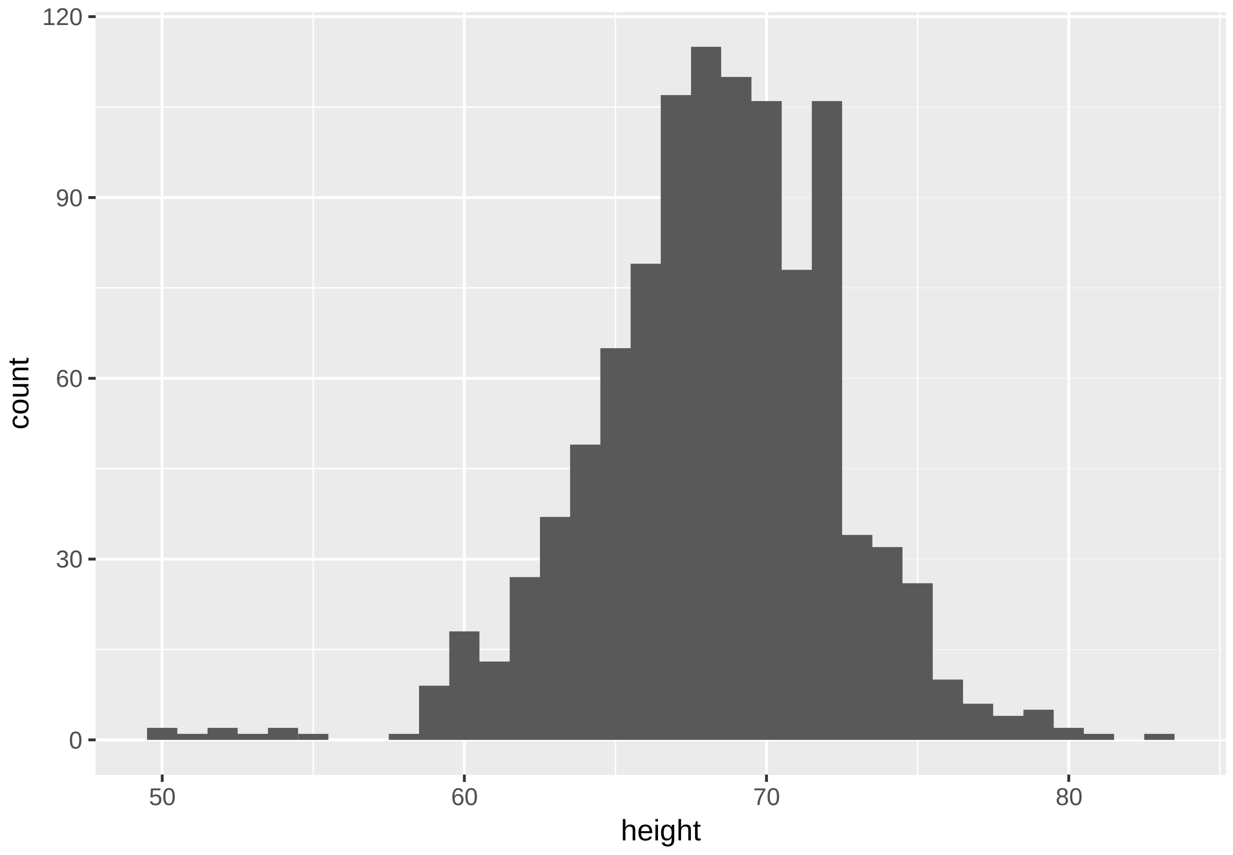
Note that when we run the code from the previous exercise we get the following warning:

```
stat_bin() using bins = 30. Pick better value with binwidth.
```

Instructions

Use the `binwidth` argument to change the histogram made in the previous exercise to use bins of size 1 inch.

```
p <-heights  %>%
  ggplot(aes(height))
## add the geom_histogram layer but with the requested argument
p +geom_histogram (binwidth=1)
```



Smooth density plot

Now instead of a histogram we are going to make a smooth density plot. In this case, we will not make an object `p`. Instead we will render the plot using a single line of code. In the previous exercise, we could have created a histogram using one line of code like this:

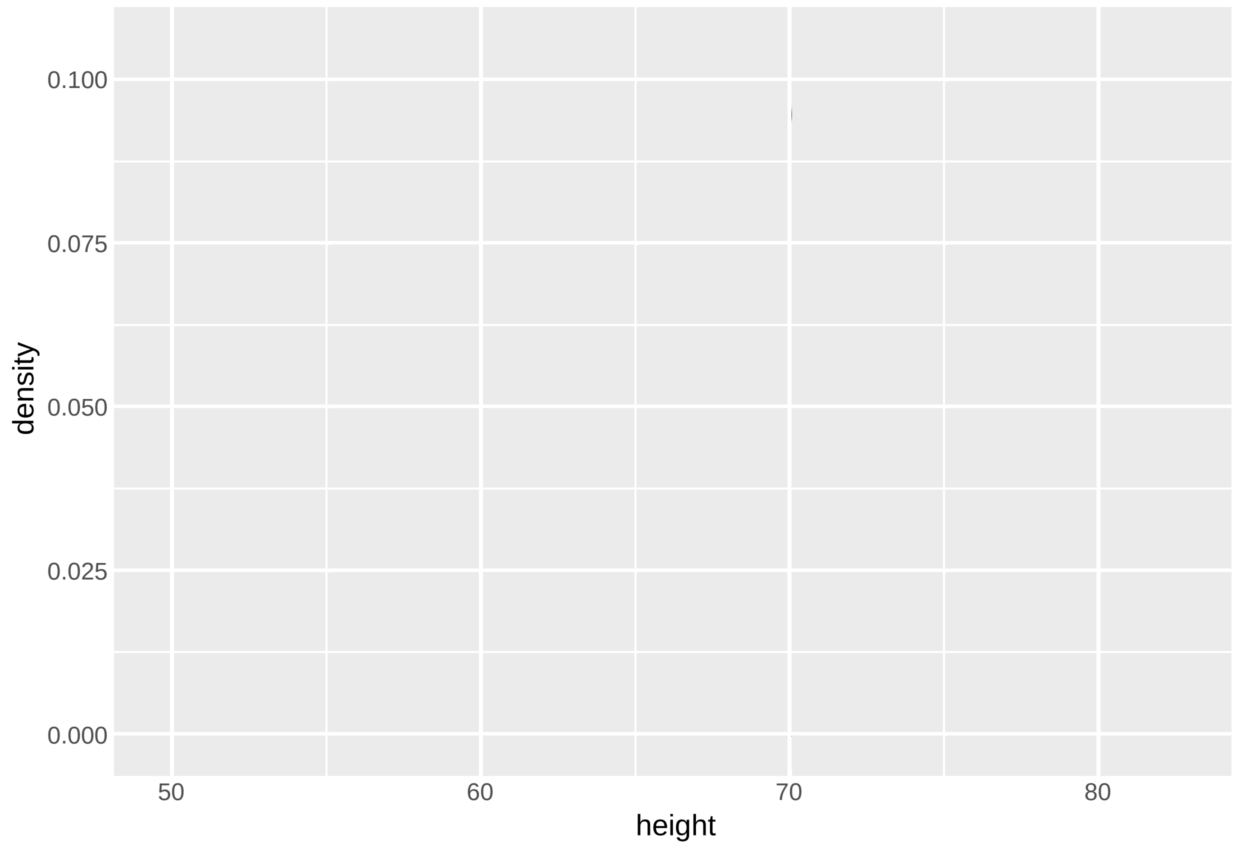
```
heights %>%
  ggplot(aes(height)) +
  geom_histogram()
```

Now instead of `geom_histogram` we will use `geom_density` to create a smooth density plot.

Instructions

Add the appropriate layer to create a smooth density plot of heights.

```
## add the correct layer using +
heights %>%
  ggplot(aes(height)) +geom_density ()
```



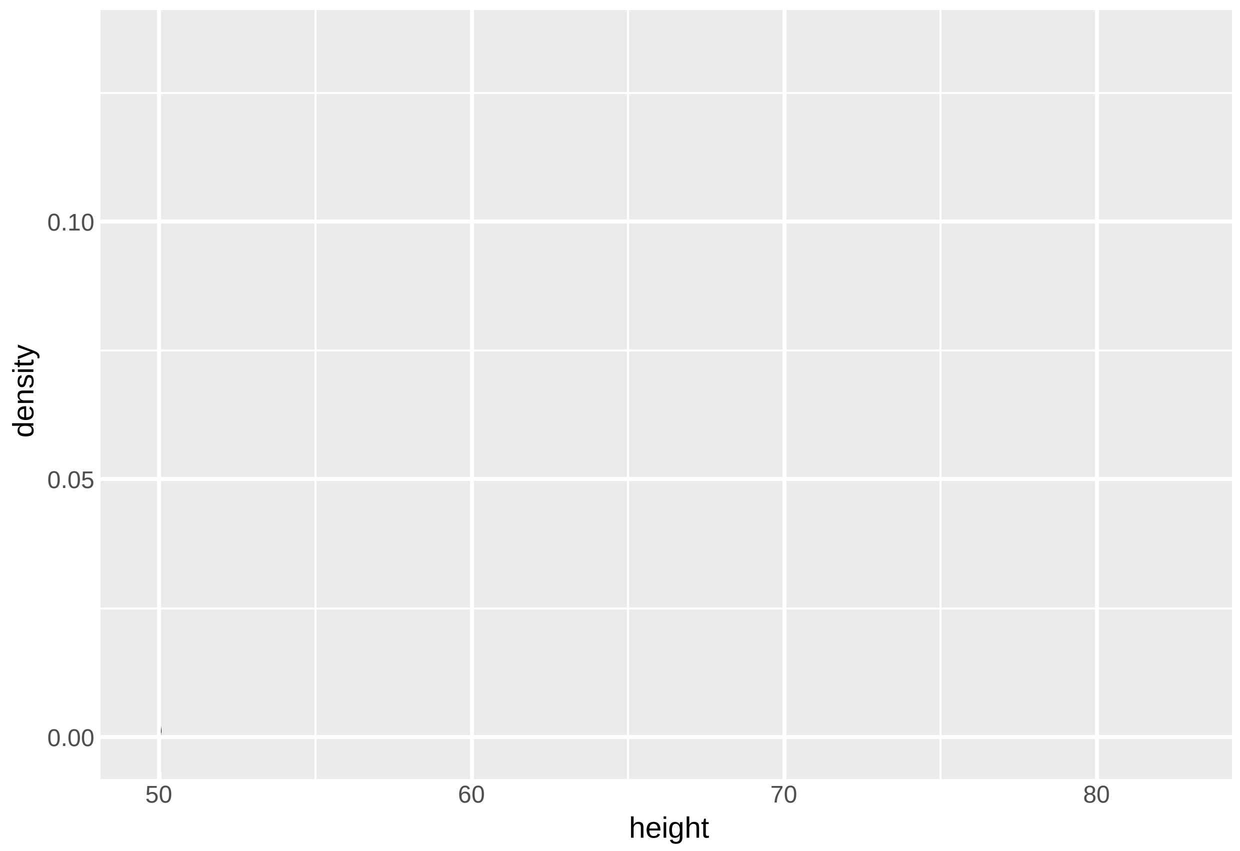
Two smooth density plots

Now we are going to make density plots for males and females separately. We can do this using the `group` argument within the `aes` mapping. Because each point will be assigned to a different density depending on a variable from the dataset, we need to map within `aes`.

Instructions

Create separte smooth density plots for males and females by defining `group` by sex.

```
## add the group argument then a layer with +
heights %>%
  ggplot(aes(height, group=sex))    +geom_density ()
```



Two smooth density plots 2

In the previous exercise we made the two density plots, one for each sex, using:

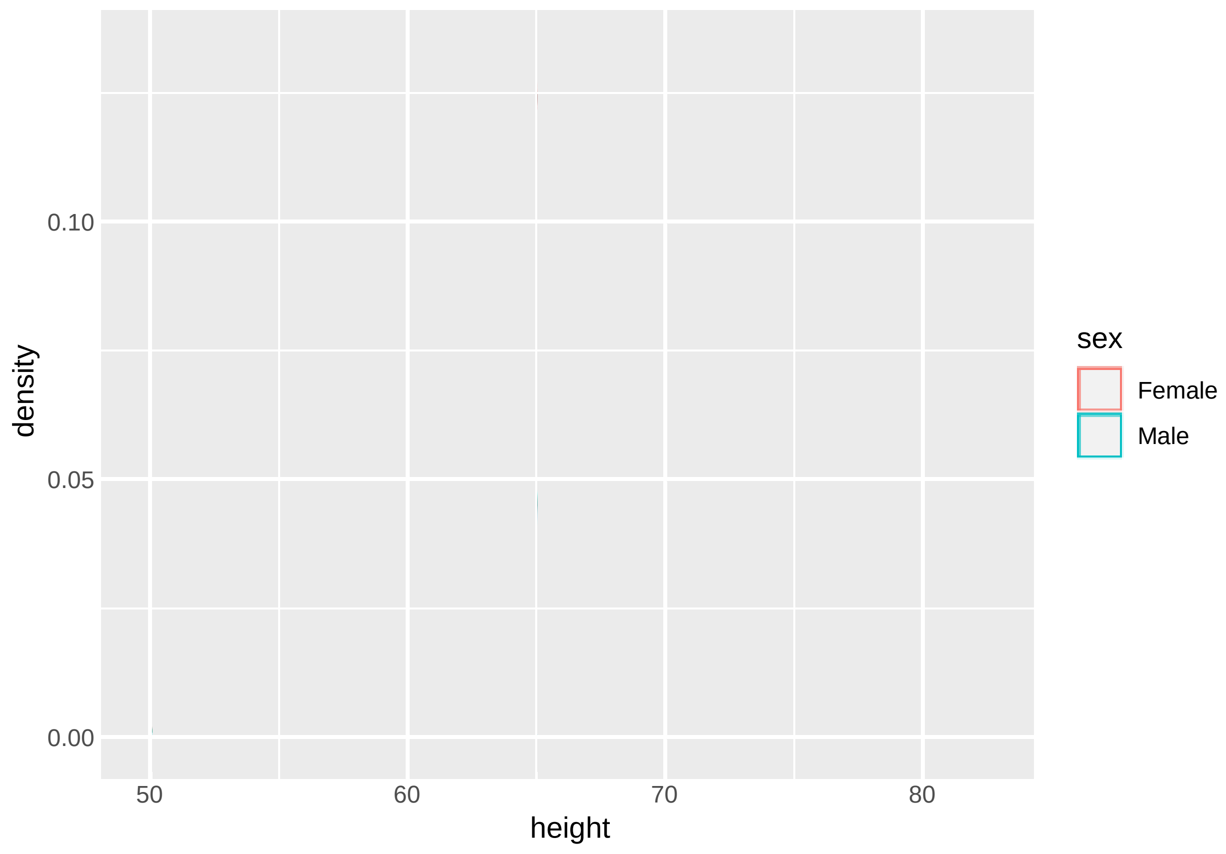
```
heights %>%
  ggplot(aes(height, group = sex)) +
  geom_density()
```

We can also assign groups through the `color` or `fill` argument. For example, if you type `color = sex` ggplot knows you want a different color for each sex. So two densities must be drawn. You can therefore skip the `group = sex` mapping. Using `color` has the added benefit that it uses color to distinguish the groups.

Instructions

Change the density plots from the previous exercise to add color.

```
## edit the next line to use color instead of group then add a density layer
heights %>%
  ggplot(aes(height,group =sex,color=sex))      +geom_density ()
```



Two smooth density plots 3

We can also assign groups using the `fill` argument. When using the `geom_density` geometry, `color` creates a colored line for the smooth density plot while `fill` colors in the area under the curve.

We can see what this looks like by running the following code:

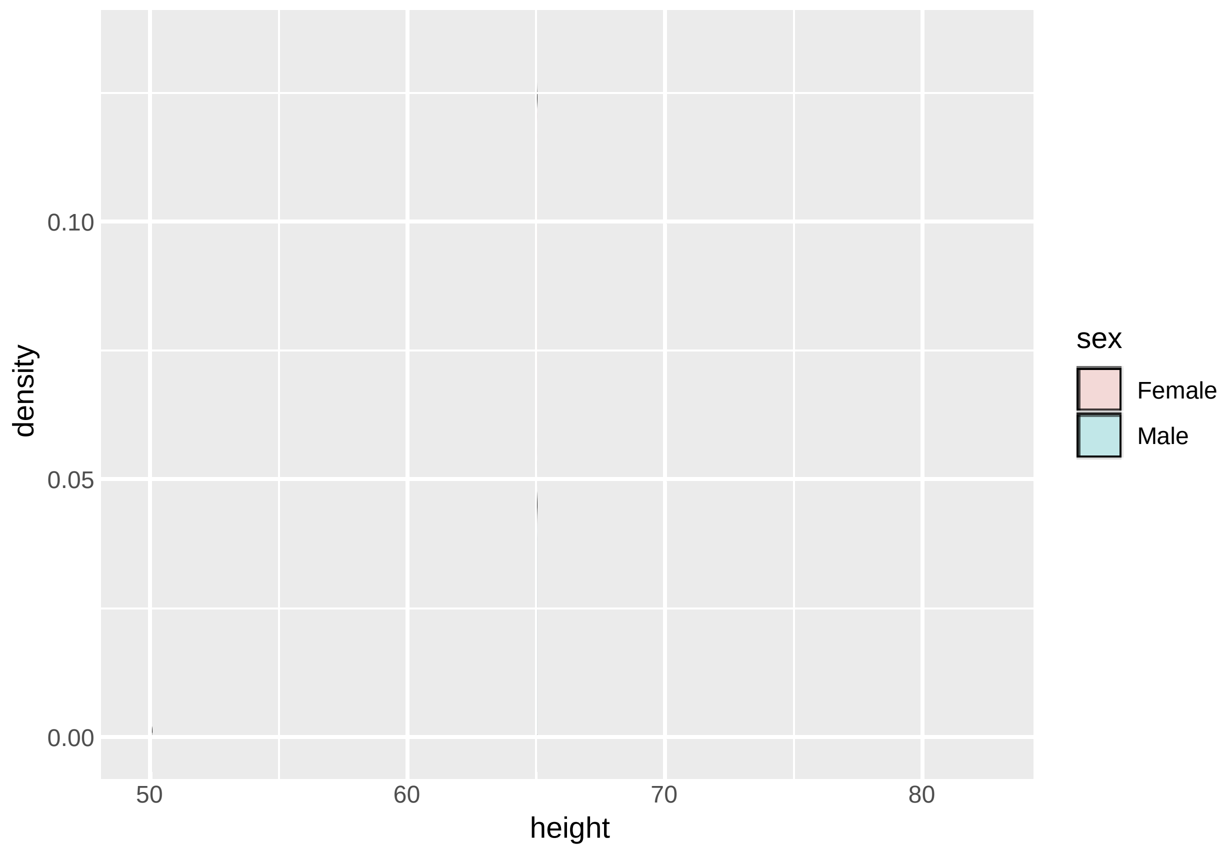
```
heights %>%
  ggplot(aes(height, fill = sex)) +
  geom_density()
```

However, here the second density is drawn over the other. We can change this by using something called alpha blending.

Instructions

Set the alpha parameter to 0.2 in the `geom_density` function to make this change.

```
heights %>%
  ggplot(aes(height,fill =sex))     +
  geom_density(alpha =0.2)
```



density
0.10
0.05
0.00
50
60
70
80
height
sex
Female
Male