Introduction to Data Visualization with ggplot2

bmRn CSM: An introduction to the grammar of graphics

Martin Frigaard

2020-12-13

Art by Allison Horst

Intro to Data Visualization with R with `ggplot2`

Objectives

1) Introduce the grammar of graphics

2) Identifying graph aesthetics

3) Recognizing and using **`geoms`**

4) Facetting graphs

Resources

Link to these slides

https://mjfrigaard.github.io/data-viz-intro/Index.html

Link to exercises

https://mjfrigaard.github.io/r-meetup-tutorials/gg00_intro-to-data-viz.html

Link to RStudio Project

https://rstudio.cloud/project/1973650

Why use `ggplot2` for data visualization?

1) `ggplot2` provides a comprehensive grammar for creating graphs/figures

2) It works hand-and-hand with the `tidyverse`

3) Better plots = better communication

Why do we create data visualizations?

Clarification

> "*The simple graph has brought more information to the data analyst's mind than any other device.*" - John Tukey

Better decision making

> "*Data visualization is a collection of methods that use visual representations to explore, make sense of, and communicate quantitative data...The ultimate purpose of data visualization, beyond understanding, is to enable better decisions and actions.*" - Stephen Few

How should we start creating data visualizations?

Start with pen and paper

get those first bad ideas out of the way

Import and inspect your data

so you know what to expect

Layered grammar of graphics

> *"appreciating the engineering design behind a sentence – **a linear ordering of phrases which conveys a gnarly network of ideas** — is the key to understanding what you are trying to accomplish when you compose a sentence."* - Stephen Pinker
>
> *"language is a system for making infinite use of finite means."* - Wilhelm von Humboldt

`ggplot2` is a language of *layers*, organized linearly

`ggplot2`'s layers give us a "*linear ordering of phrases*" to build an infinite number of graphs "*which convey a gnarly network of ideas.*"

...infinitely extensible

Let's load some data!

The NHANES package comes with data from the 2014 American National Health and Nutrition Examination surveys. We will load a sample from it below:

```
SmallNhanes <- read_csv("https://bit.ly/nhanes-small")
SmallNhanes
```

ID	Gender	Age	AgeDecade	Race1	HealthGen	Height	BMI
<dbl>	<chr>	<dbl>	<chr>	<chr>	<chr>	<dbl>	<dbl>
51624	male	34	30-39	White	Good	164.7	32.22
51624	male	34	30-39	White	Good	164.7	32.22
51624	male	34	30-39	White	Good	164.7	32.22
51625	male	4	0-9	Other	*NA*	105.4	15.30
51630	female	49	40-49	White	Good	168.4	30.57
51638	male	9	0-9	White	*NA*	133.1	16.82
51646	male	8	0-9	White	*NA*	130.6	20.64
51647	female	45	40-49	White	Vgood	166.7	27.24
51647	female	45	40-49	White	Vgood	166.7	27.24
51647	female	45	40-49	White	Vgood	166.7	27.24

1-10 of 60 rows | 1-8 of 11 columns Previous **1** 2 3 4 5 6 Next

Quick Tip: Column Names

Standardize names

```
SmallNhanes <- SmallNhanes %>% janitor::clean_names()
SmallNhanes
```

id	gender	age	age_decade	race1	health_gen	height	bmi
<dbl>	<chr>	<dbl>	<chr>	<chr>	<chr>	<dbl>	<dbl>
51624	male	34	30-39	White	Good	164.7	32.22
51624	male	34	30-39	White	Good	164.7	32.22
51624	male	34	30-39	White	Good	164.7	32.22
51625	male	4	0-9	Other	*NA*	105.4	15.30
51630	female	49	40-49	White	Good	168.4	30.57
51638	male	9	0-9	White	*NA*	133.1	16.82
51646	male	8	0-9	White	*NA*	130.6	20.64
51647	female	45	40-49	White	Vgood	166.7	27.24
51647	female	45	40-49	White	Vgood	166.7	27.24
51647	female	45	40-49	White	Vgood	166.7	27.24

1-10 of 60 rows | 1-8 of 11 columns Previous **1** 2 3 4 5 6 Next

Quick Tip: Factors

Format factors: We have a `health_gen` variable with the following levels:

`Excellent`, `Vgood`, `Good`, `Fair`, or `Poor`. These are ordered.

```
SmallNhanes <-
  SmallNhanes %>% mutate(
  health_gen = factor(x = health_gen,
                      levels = c("Poor", "Fair",
                                 "Good", "Vgood",
                                 "Excellent"),
                      ordered = TRUE))
```

```
levels(SmallNhanes$health_gen)
```

```
## [1] "Poor"      "Fair"      "Good"      "Vgood"     "Excellent"
```

Layered grammar of graphics

How it works:

- Graphs are *initialized* with *ggplot()*

- Variables are *mapped* to *aesthetics*

- Geoms are linked to *statistics*

Our First Graph

What relationship do you expect to see between **`height`** and **`weight`**?

1. Use data with pipe to initialize graph

```
SmallNhanes %>%
```

2. Map variables to aesthetics

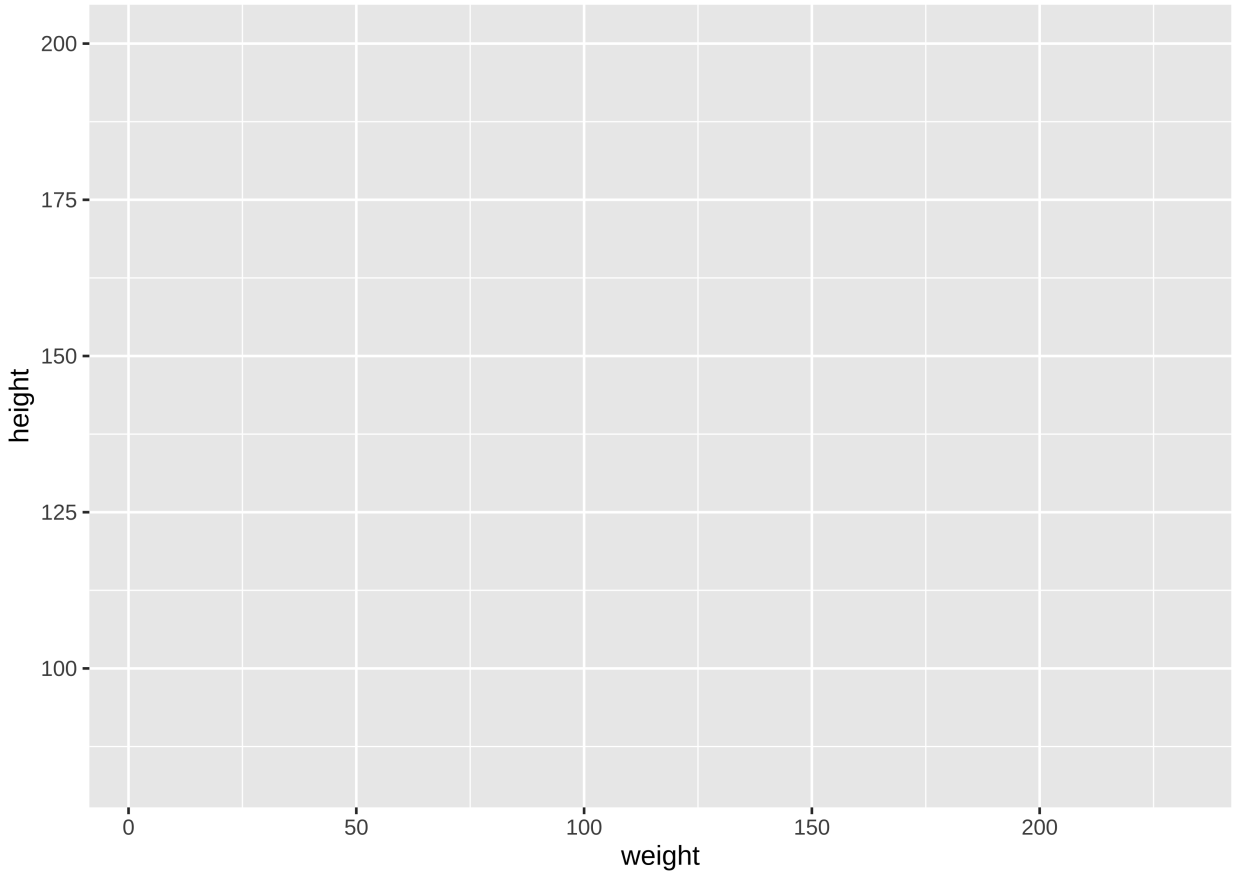
```
SmallNhanes %>%
ggplot(mapping = aes(x = weight, y = height))
```

3. Add geoms and layers

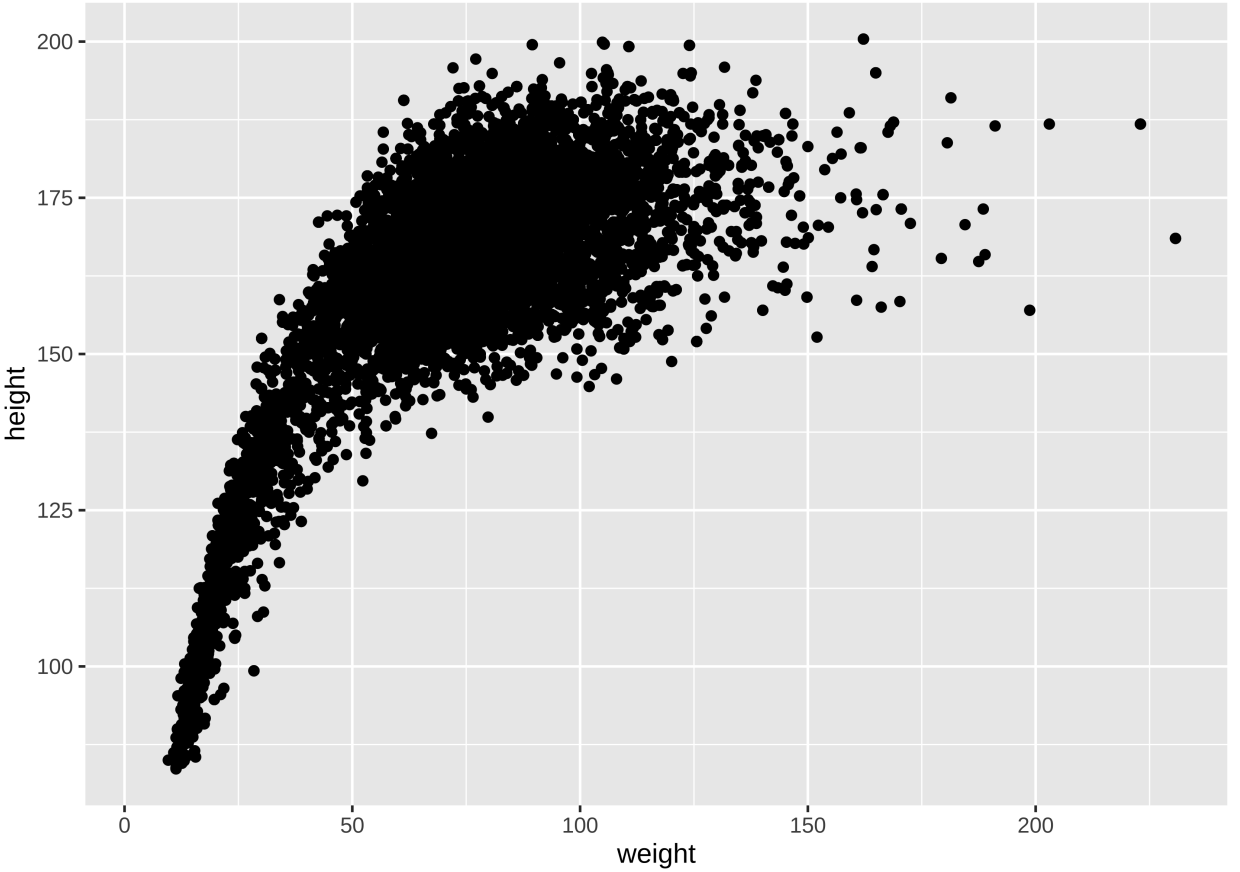
```
SmallNhanes %>%
ggplot(mapping = aes(x = weight, y = height)) +
geom_point()
```

```
SmallNhanes %>%
  ggplot() # initialize
```

```
SmallNhanes %>%
  ggplot(mapping = aes(x = weight, y = height)) # map variables
```



```
SmallNhanes %>%
  ggplot(mapping = aes(x = weight, y = height)) +
  geom_point() # add geoms
```



ggplot2 template


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

We can add more aesthetics *inside* geoms

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

And we can add *more* geoms

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

Graph Aesthetics

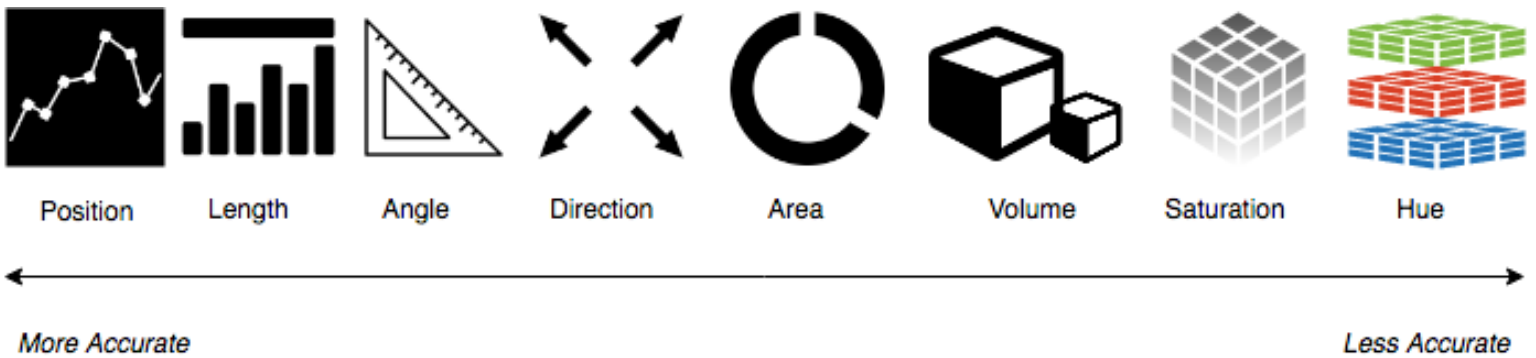
Aesthetics

Is the relationship between `weight` and `height` the same for both `genders`?

We can explore this by mapping the variables to different aesthetics

Aesthetics as graph elements

Examples of aesthetics are `color`, `size`, `shape`, and `alpha`



Mapping (global vs. local)

Global `ggplot2` mapping

inside the `ggplot()` function = setting variables *globally*

Dataset with variables

```
SmallNhanes %>%
```

Initialize plot

Map variables

```
ggplot(mapping = aes(x = weight, y = height)) +
```

Geom function

```
geom_point()
```

Local `ggplot2` mapping

inside the `geom()` function = setting variables *locally*

Dataset with variables

```
SmallNhanes %>%
```

Initialize plot

```
ggplot() +
```

Geom function to map variables to positions

```
geom_point(mapping = aes(x = weight, y = height))
```

`ggplot2` tip

Note the different syntax:

Initializing plots with %>%

```
<DATA> %>% # initialize
```

Adding layers with +

```
<DATA> %>%
  ggplot(mapping = aes(<MAPPINGS>)) + # adding layers
  <GEOM_FUNCTION>()
```

Your Turn

Set local aesthetic mappings

From here...

```
SmallNhanes %>%
  ggplot(mapping = aes(x = weight, y = height)) +
  geom_point() +
  geom_smooth()
```

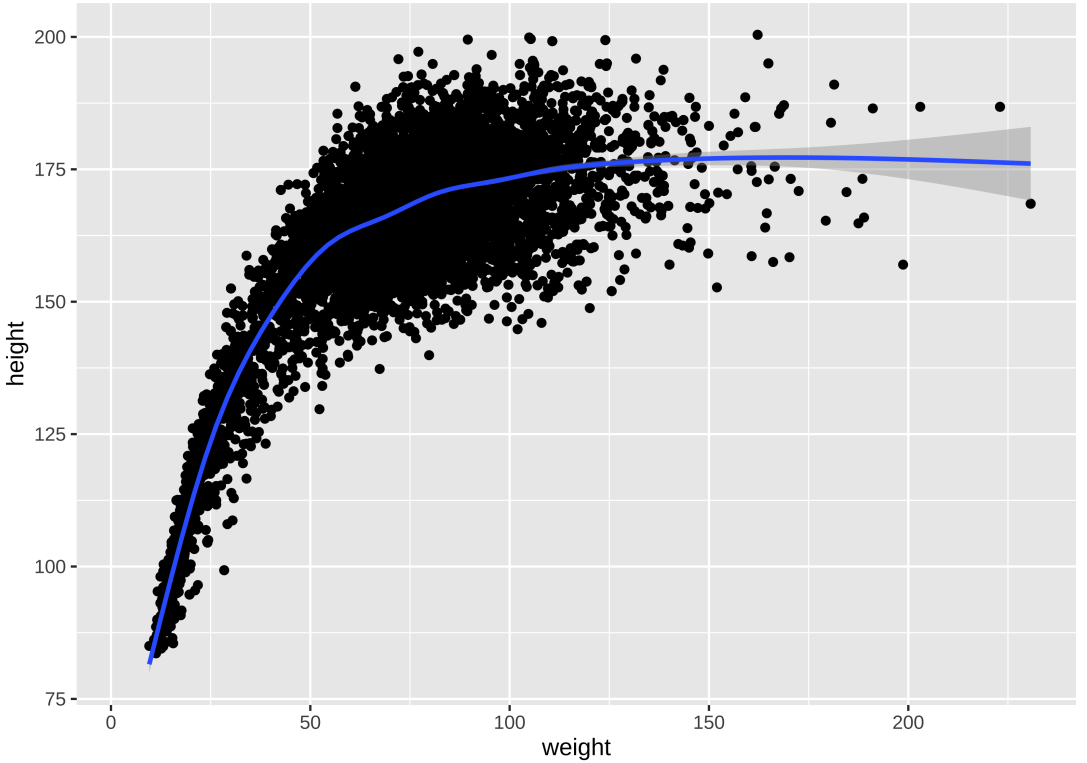
...to here.

```
SmallNhanes %>%
  ggplot() +
  geom_point(mapping = aes(x = weight, y = height))
  geom_smooth(mapping = aes(x = weight, y = height))
```

What do you expect to see?

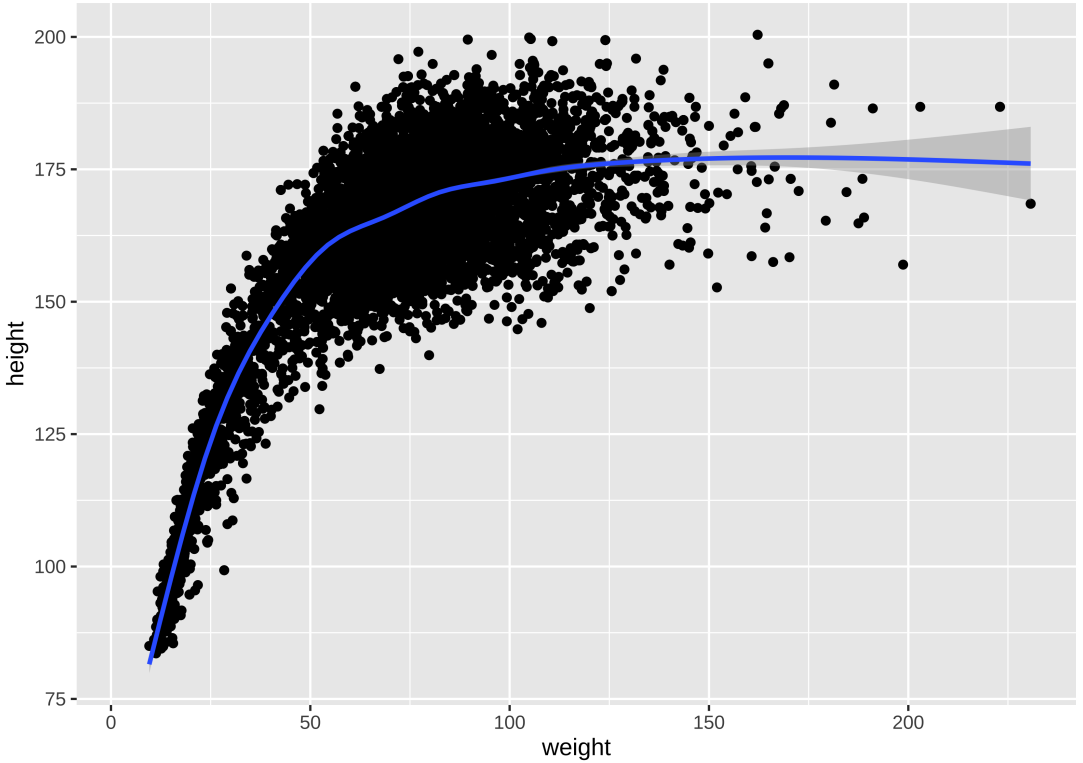
Your Turn (solution 1)

```
SmallNhanes %>%
  ggplot(mapping = aes(x = weight, y = height)) +
  geom_point() +
  geom_smooth()
```



Your Turn (solution 2)

```
SmallNhanes %>%
  ggplot() +
  geom_point(mapping = aes(x = weight, y = height)) +
  geom_smooth(mapping = aes(x = weight, y = height))
```



Variables, Aestheitcs, and Geoms

Variables, Aestheitcs, and Geoms (1)

Each graph needs a variable or value, an aesthetic, and geom (the accompanying graphic, geometry)

```
geom_point(mapping = aes(x = weight, y = height)) + # layer 1
geom_smooth(mapping = aes(x = weight, y = height)) # layer 2
```

variable	aesthetic	geom
weight	position = x	dots = point
height	position = y	dots = point
weight	position = x	line = smooth
height	position = y	line = smooth

These have the same aesthetics! What if we added a layer with a variable mapped to a different aesthetic?

Variables, Aestheitcs, and Geoms (2)

But we can add *more* variables, map them to *different* aesthetics, and *adding* another `geom` layer

Add another layer, coloring the points by `gender`

```
SmallNhanes %>%
  ggplot() +
  geom_point(mapping = aes(x = weight, y = height)) +
  geom_point(mapping = aes(color = gender))
```

variable	aesthetic	geom
`weight`	position = `x`	dots = `point`
`height`	position = `y`	dots = `point`
`gender`	color = `color`	dots = `point`

Variables, Aestheitcs, and Geoms (3)

ERROR!

```
SmallNhanes %>%
  ggplot() +
  geom_point(mapping = aes(x = weight, y = height)) +
  geom_point(mapping = aes(color = gender))
```

```
# <error/rlang_error>
# geom_point requires the following missing aesthetics: x and y
```

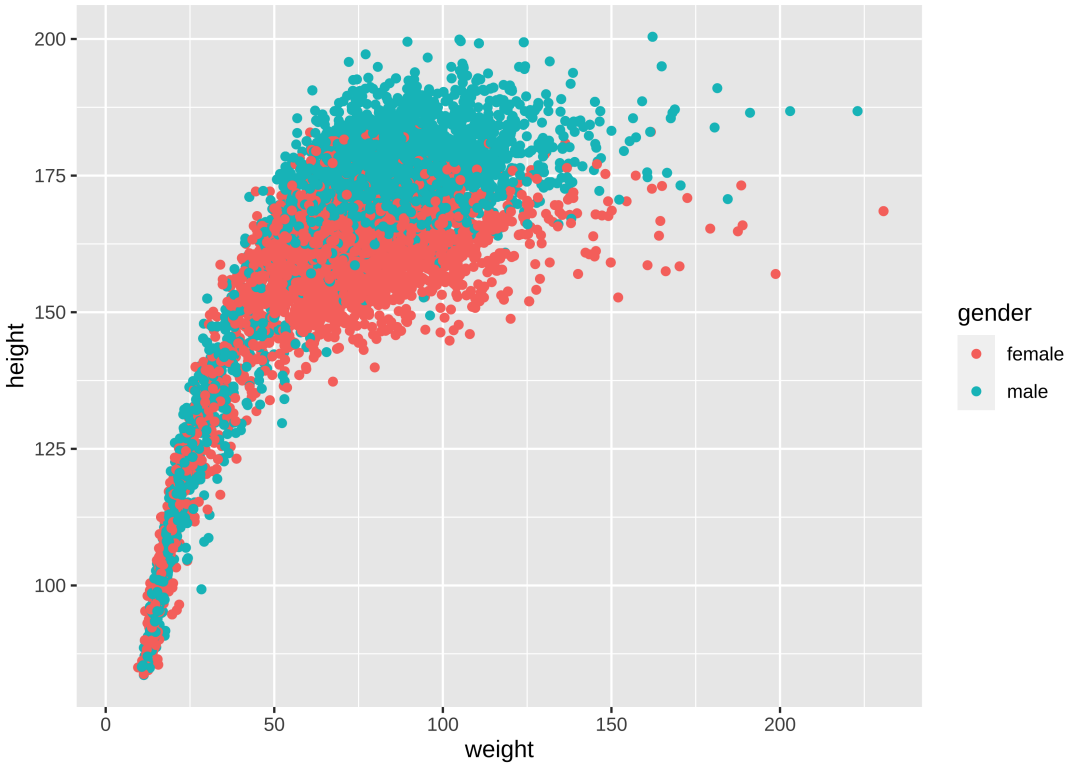
SOLUTION

All geoms have required aesthetics--map variables globally

```
SmallNhanes %>%
  ggplot(aes(x = weight, y = height)) +
    geom_point(aes(color = gender))
```

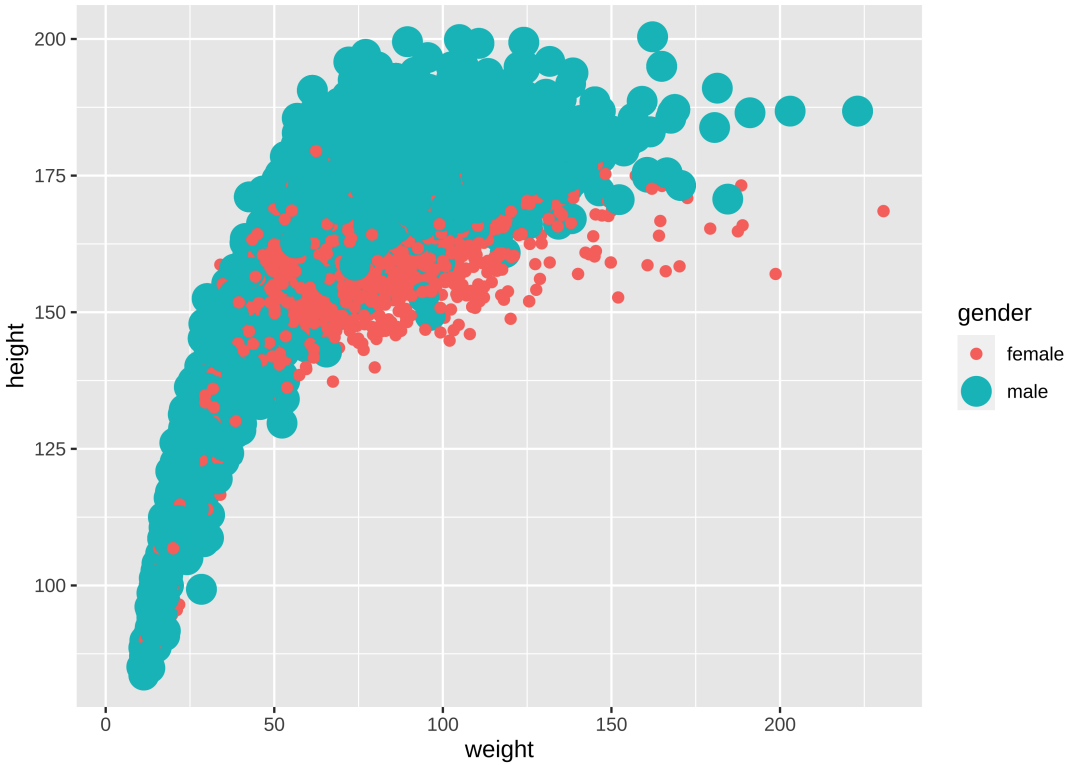
Aesthetics: color

```
SmallNhanes %>%
  ggplot(aes(x = weight, y = height)) +
    geom_point(aes(color = gender))
```



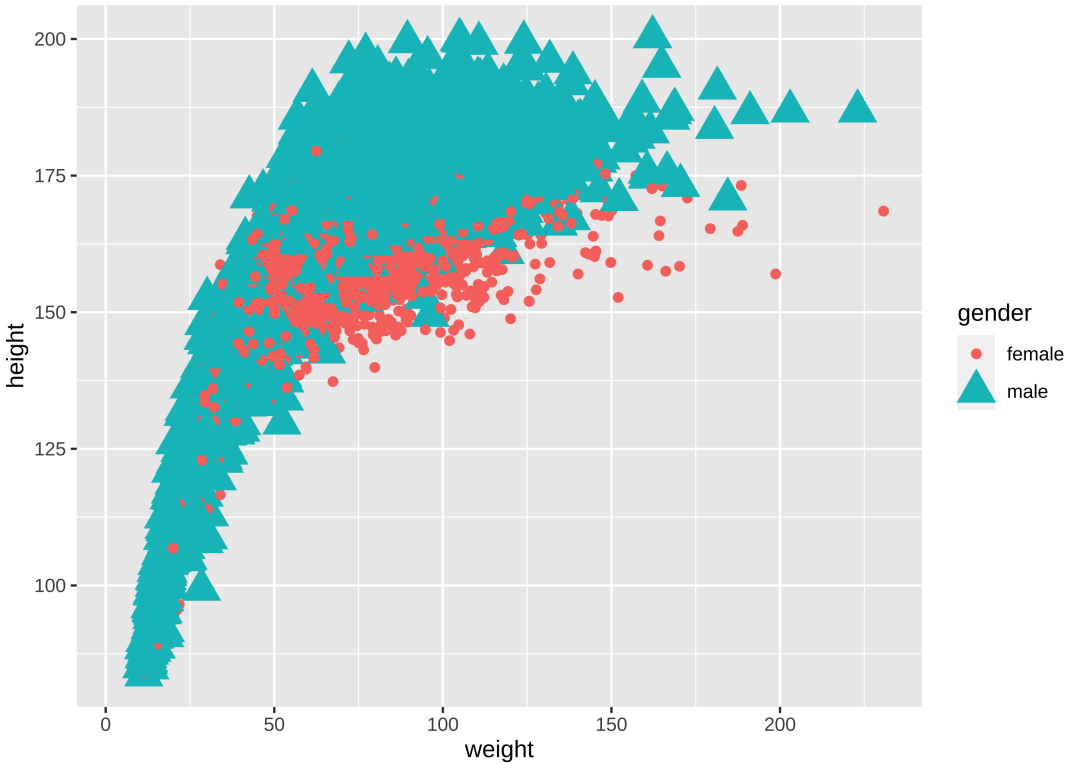
Aesthetics: size

```
SmallNhanes %>%
  ggplot(aes(x = weight, y = height)) +
    geom_point(aes(color = gender, size = gender))
```



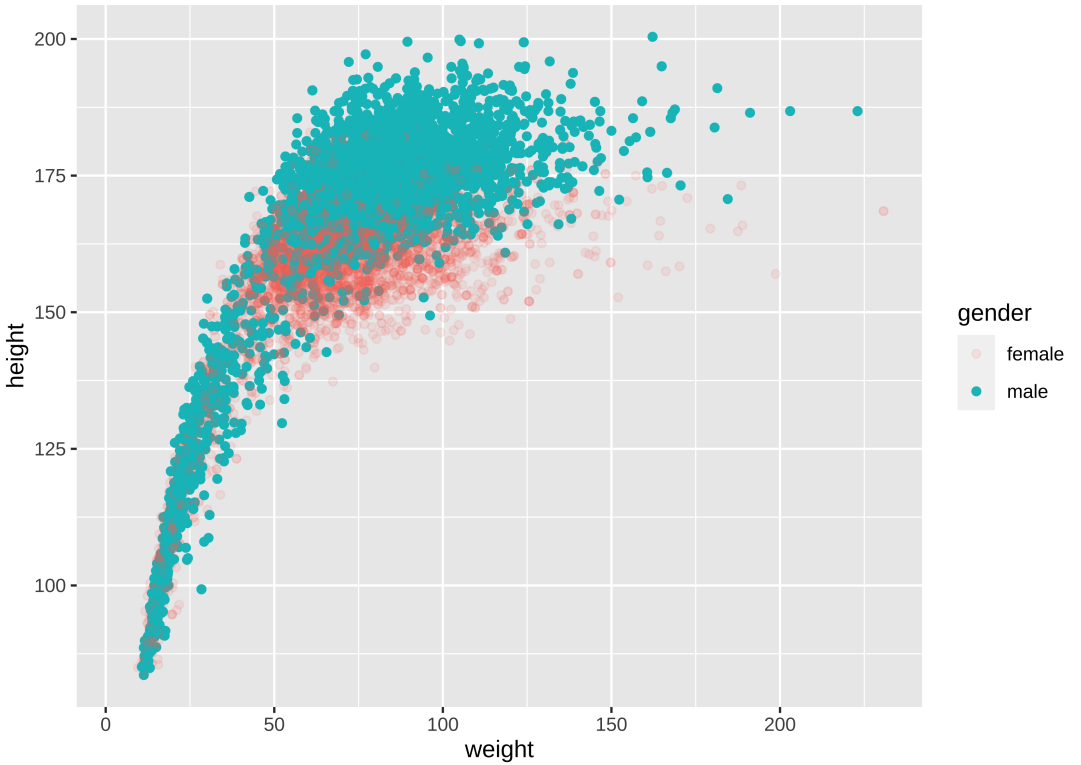
Aesthetics: shape

```
SmallNhanes %>%
  ggplot(aes(x = weight, y = height)) +
    geom_point(aes(color = gender, size = gender, shape = gender))
```



Aesthetics: alpha (opacity)

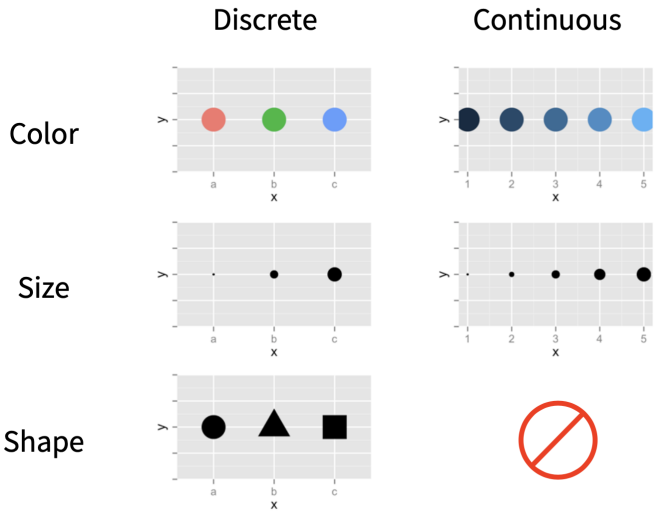
```
SmallNhanes %>%
  ggplot(aes(x = weight, y = height)) +
    geom_point(aes(color = gender, alpha = gender))
```



Aesthetic mappings

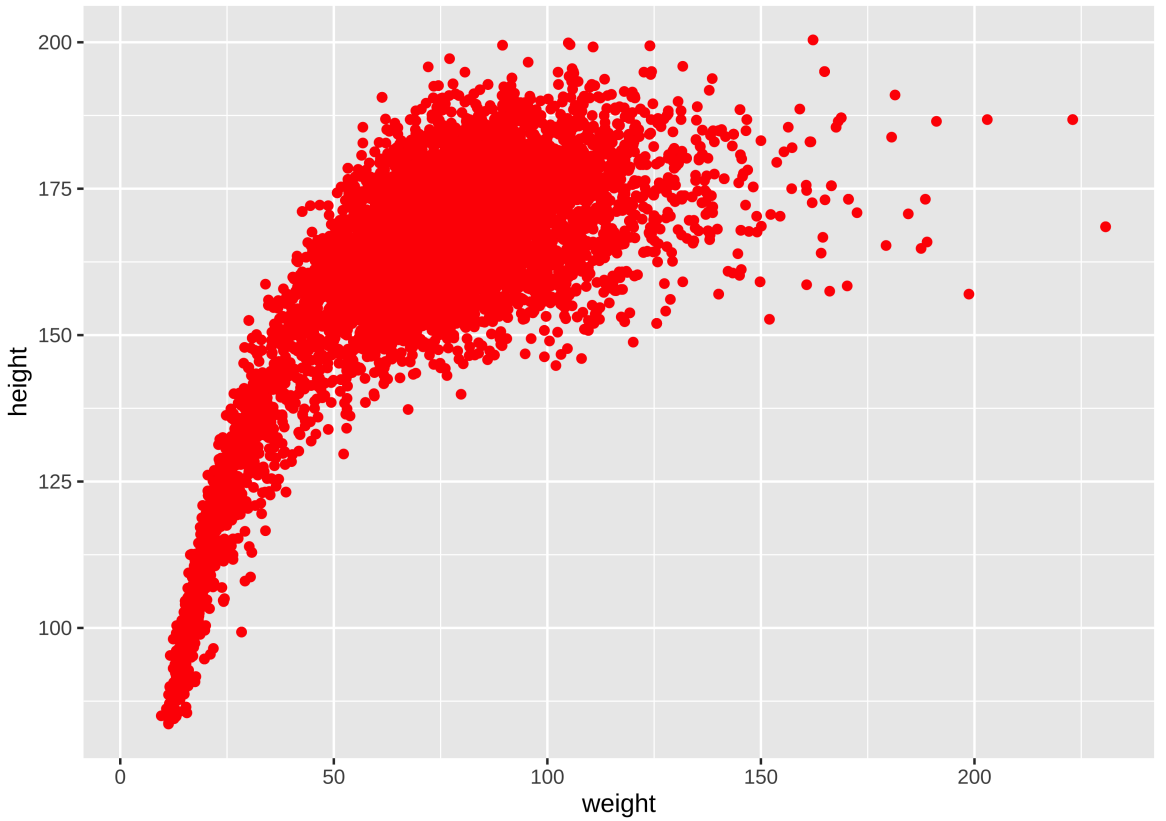
Legend is automatically included

Continuous variables best with `size`



Setting values vs. mapping variables

How can we create this plot?

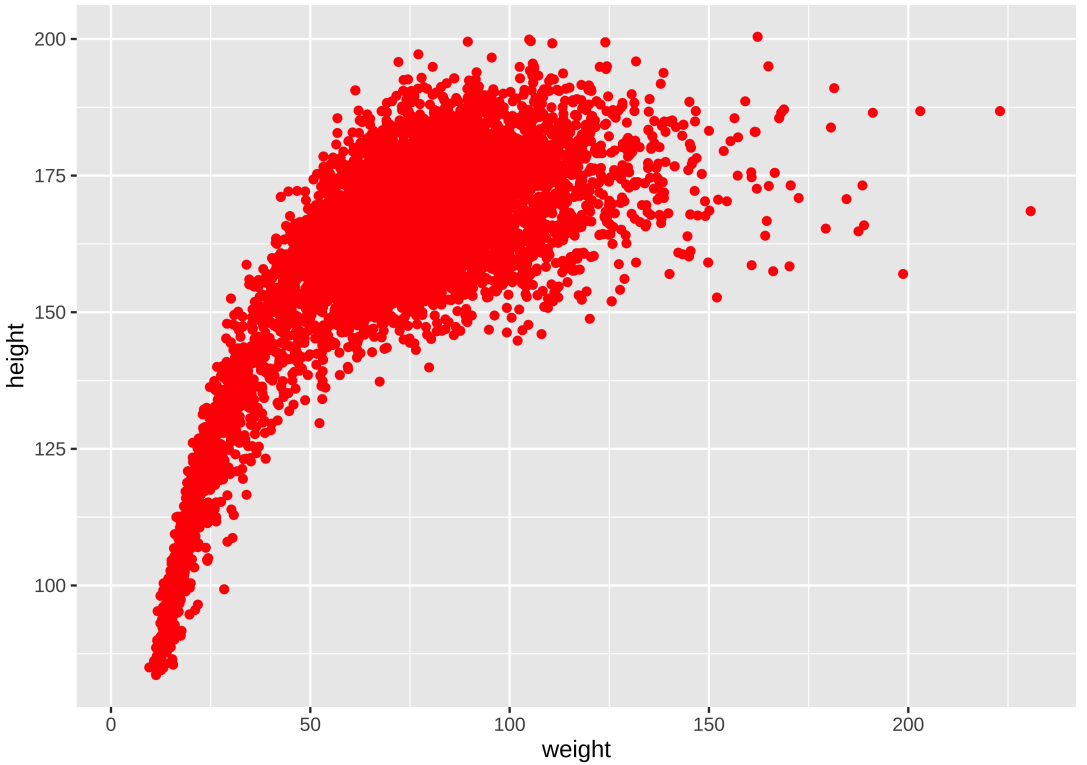


Inside aes( )

```
SmallNhanes %>%
  ggplot(aes(x = weight, y = height)) +
    geom_point(aes(color = "red")) # inside aes
```

Outside aes( )

```
SmallNhanes %>%
  ggplot(aes(x = weight, y = height)) +
  geom_point(color = "red") # outside aes
```



What happened?

This expected a variable, not a value (`"red"`).

```
SmallNhanes %>%
  ggplot(aes(x = weight, y = height)) +
  geom_point(aes(color = "red")) # "value" in aes
```

Geoms (geometric objects)

Geoms

These are visual elements used to represent the data of the graph

Examples include:

- `geom_boxplot`
- `geom_col`
- `geom_line`
- `geom_smooth`

See the cheatsheet for more examples:

https://bit.ly/ggplot2-cheat

Your Turn

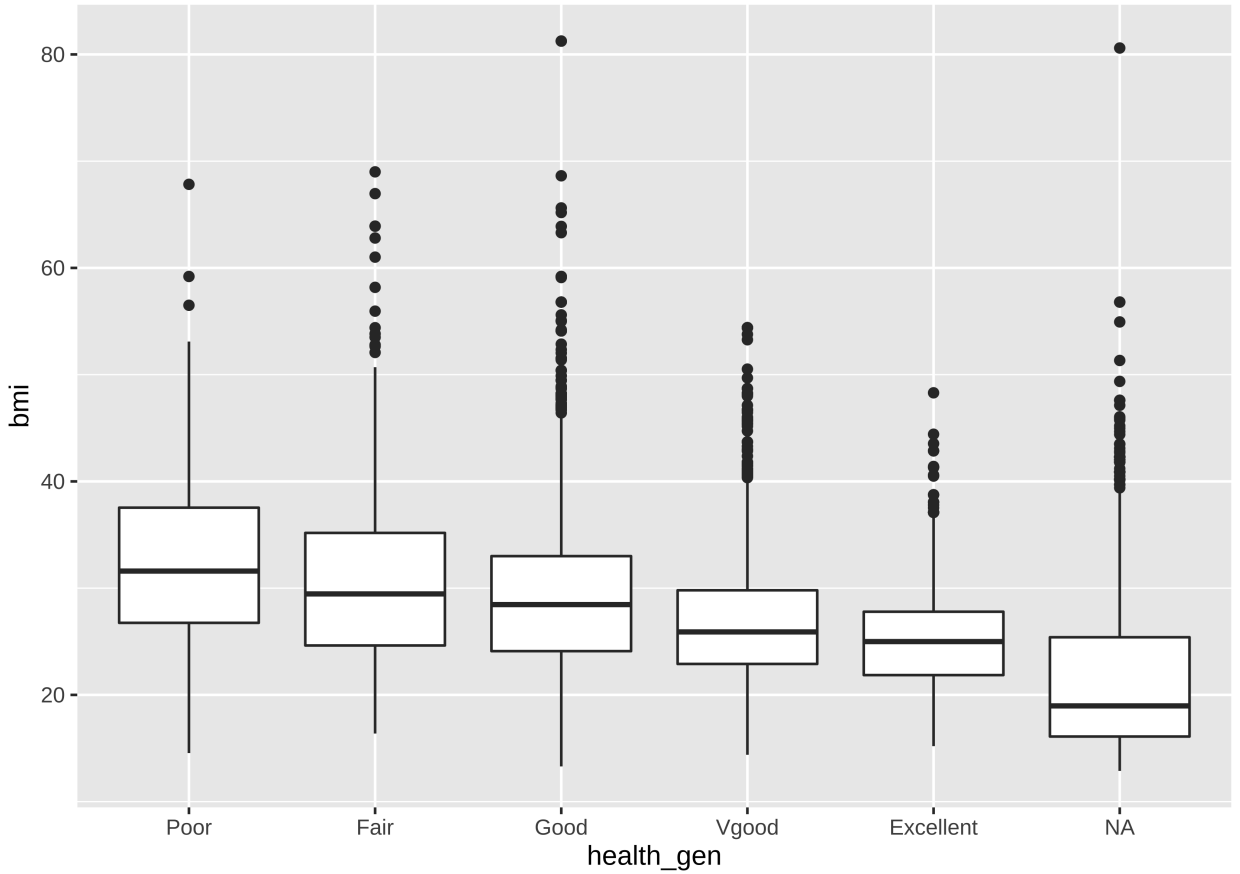
How does BMI vary across levels of self-reported general health?

Complete the code below:

Map the variables locally inside the `geom_boxplot()` function

```
SmallNhanes %>%
  ggplot() %>%
  geom_boxplot(mapping = aes(x = __________, y = ___))
```

```
SmallNhanes %>%
  ggplot() +
  geom_boxplot(mapping = aes(x = health_gen, y = bmi))
```



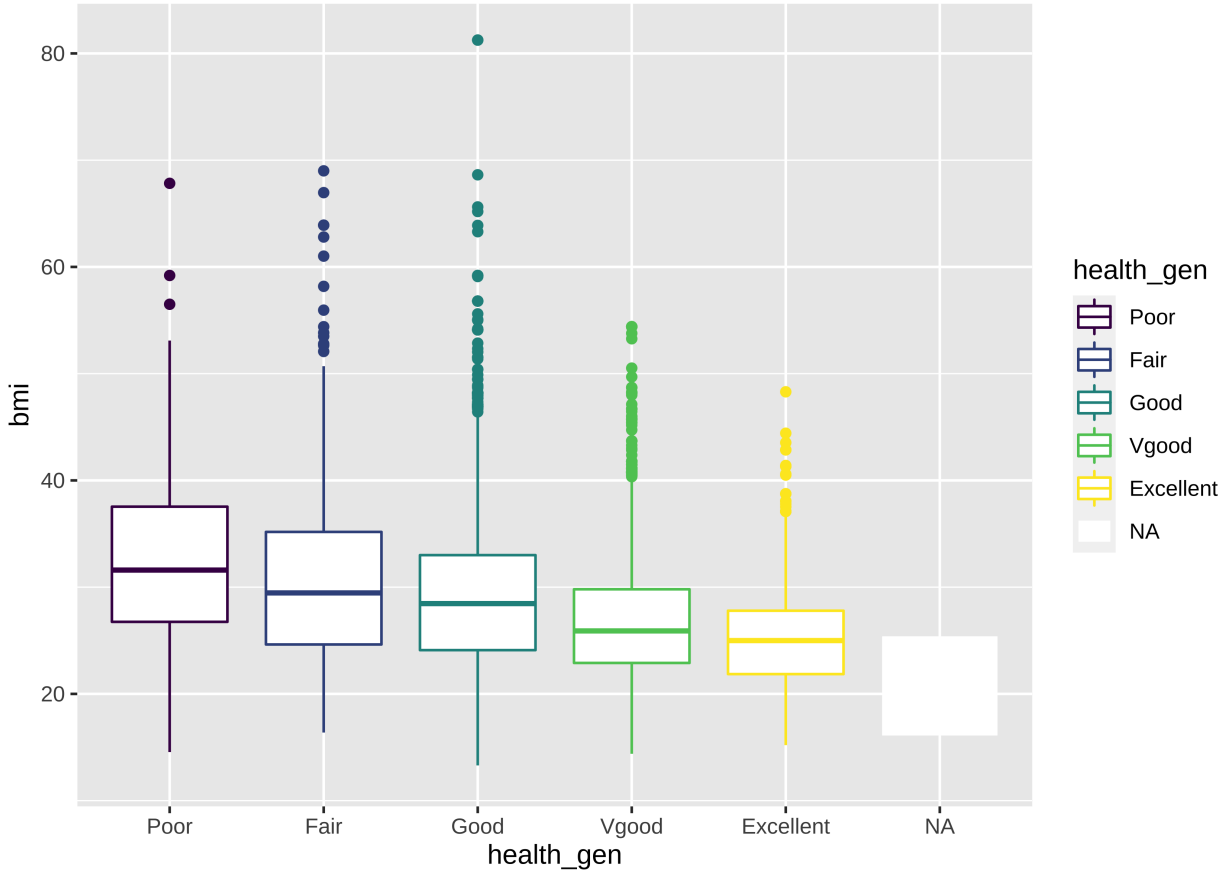
Box-plots are great for seeing how a continuous variable varies across a categorical variable

Your Turn

Fill in the code below to change the colors in the boxplot for each level of **`health_gen`**

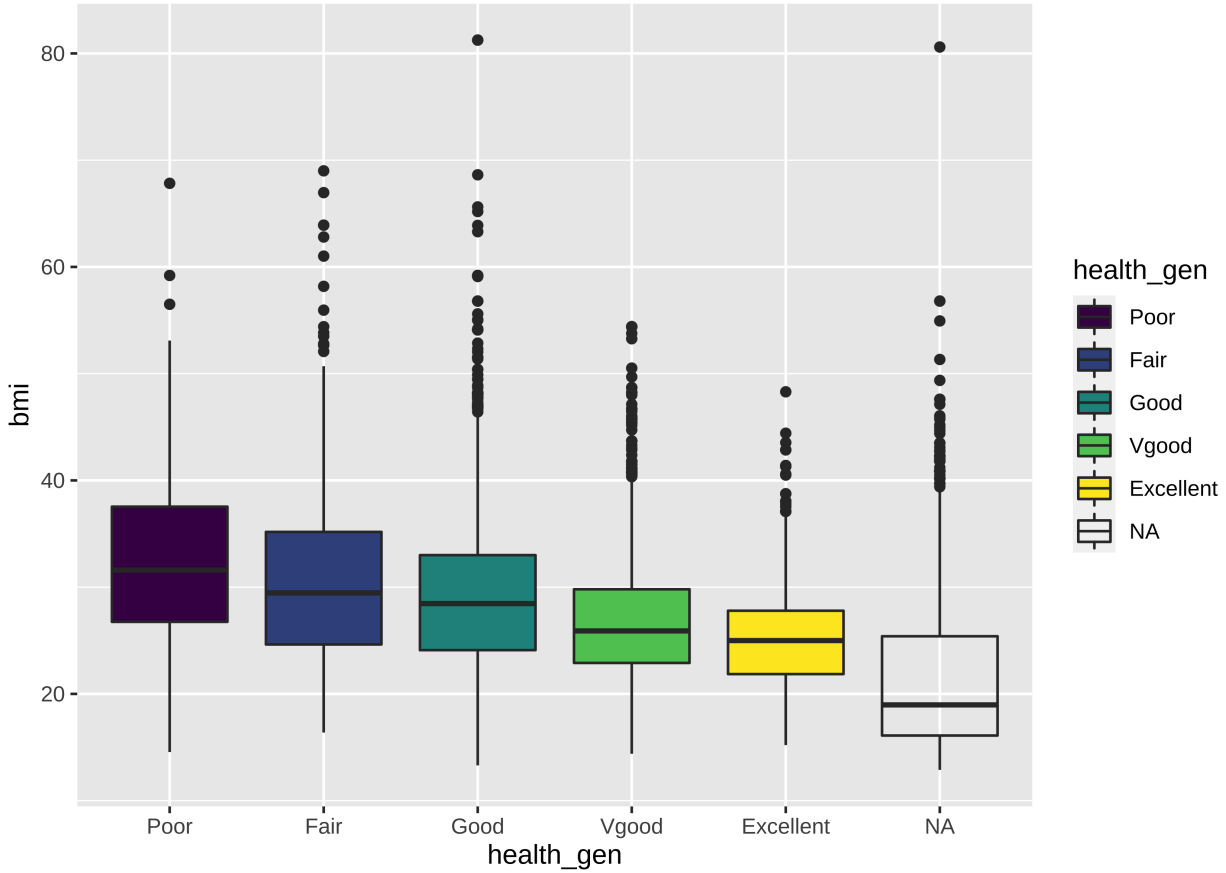
```
SmallNhanes %>%
  ggplot() +
  geom_boxplot(
    aes(x = health_gen, y = bmi, _____ = health_gen))
```

```
SmallNhanes %>%
  ggplot() +
  geom_boxplot(
    aes(x = health_gen, y = bmi, color = health_gen))
```



Color is not the setting we want here...

```
SmallNhanes %>%
  ggplot() +
  geom_boxplot(
    aes(x = health_gen, y = bmi, fill = health_gen))
```



Fill is better

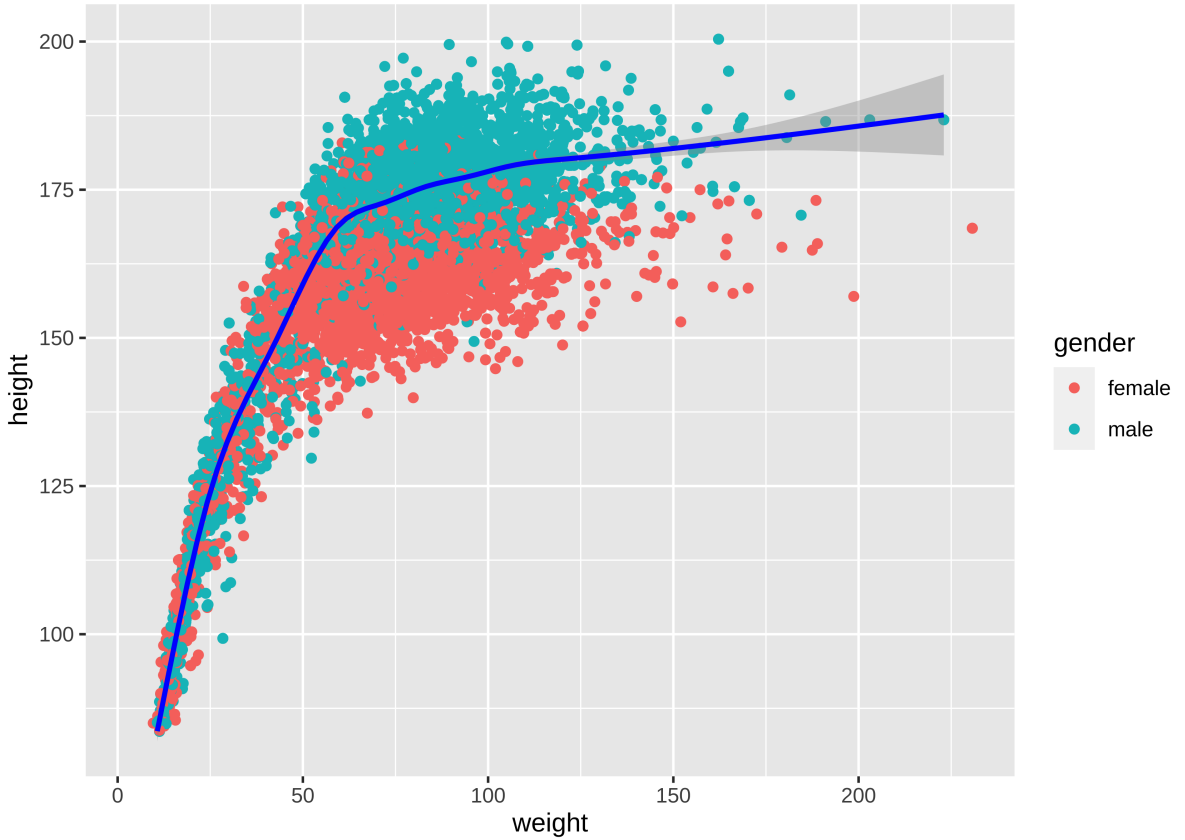
Adding layers

The 'infinitely extensible' part of **`ggplot2`** is where we start to really see it's power

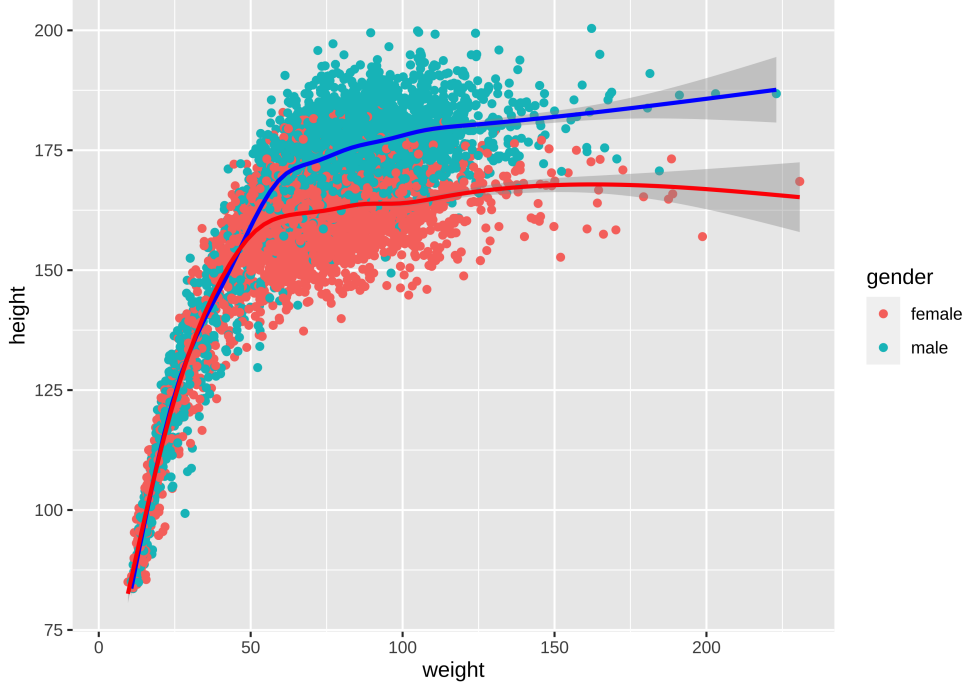
Consider the relationship between **`height`** and **`weight`** again

```r
SmallNhanes %>%
  ggplot(aes(x = weight, y = height)) + # global
  geom_point(aes(color = gender))
```

```r
SmallNhanes %>%
  ggplot(aes(x = weight, y = height)) +
  geom_point(aes(color = gender)) +
  geom_smooth(data = # data 2
                filter(SmallNhanes, gender == "male"), # layer 2
              aes(x = weight, y = height),
              color = "blue")
```



```r
SmallNhanes %>%
  ggplot(aes(x = weight, y = height)) +
  geom_point(aes(color = gender)) +
  geom_smooth(data =
                filter(SmallNhanes, gender == "male"),
              aes(x = weight, y = height),
              color = "blue") +
    geom_smooth(data = # data 3
                  filter(SmallNhanes, gender == "female"), # layer 3
              aes(x = weight, y = height),
              color = "red")
```



Facets

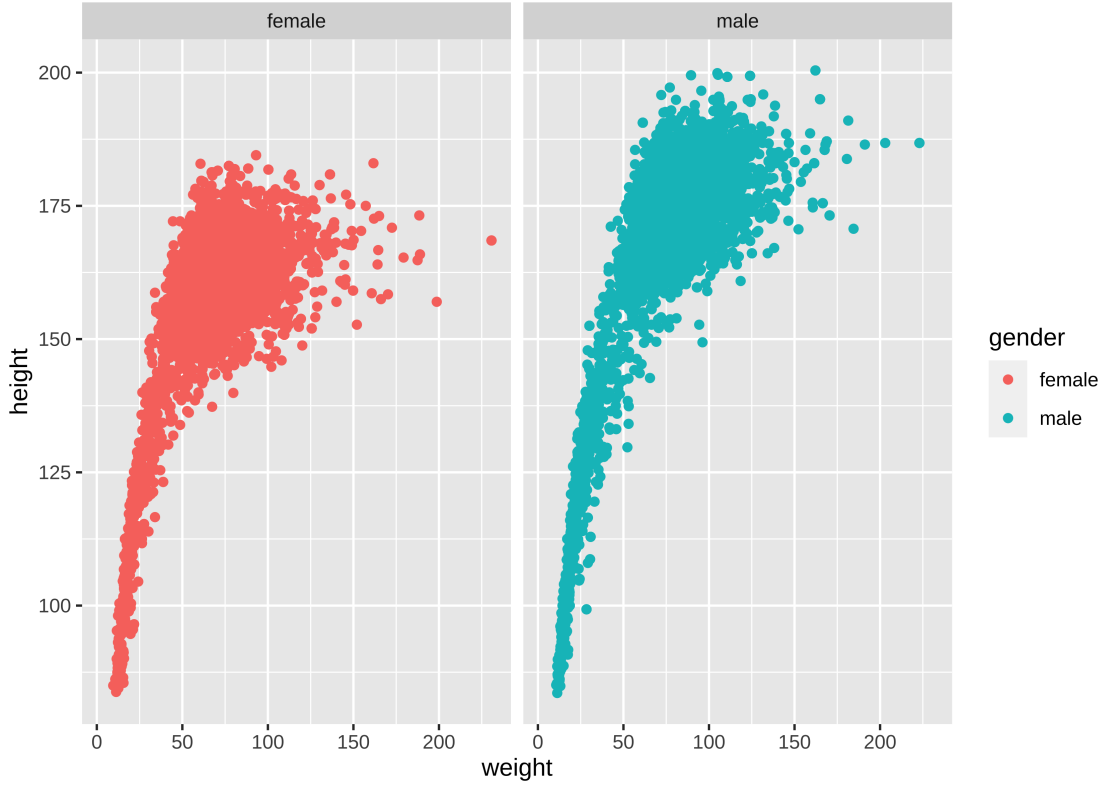
Facetting

Facet layers display subplots for levels of categorical variables

Facet layer	Display
`facet_wrap(. ~ gender)`	Plot for each level of `gender`
`facet_wrap(race1 ~ gender)`	Plot for each level of `gender` and `race`
`facet_wrap(. ~ gender, ncol = 1)`	Specify the number of columns
`facet_wrap(. ~ gender, nrow = 1)`	Specify the number of rows

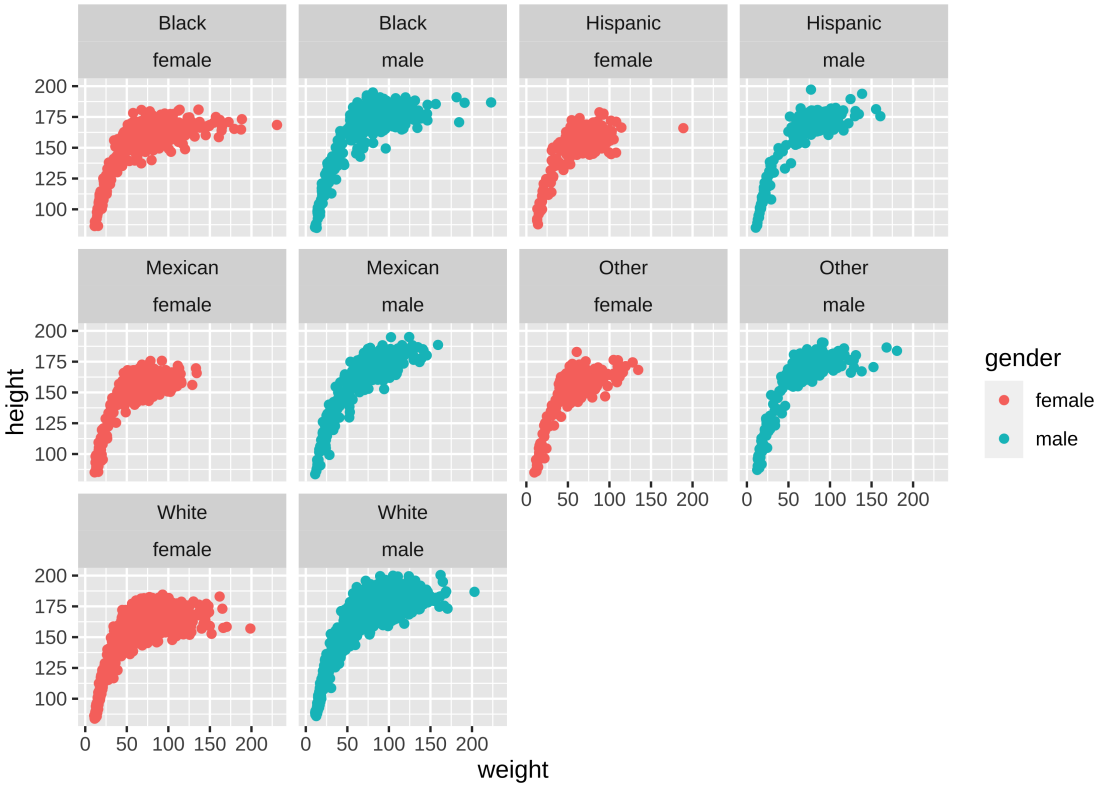
Facet Single Variable

```
SmallNhanes %>%
  ggplot(aes(x = weight, y = height)) +
  geom_point(aes(color = gender)) +
  facet_wrap(. ~ gender)
```



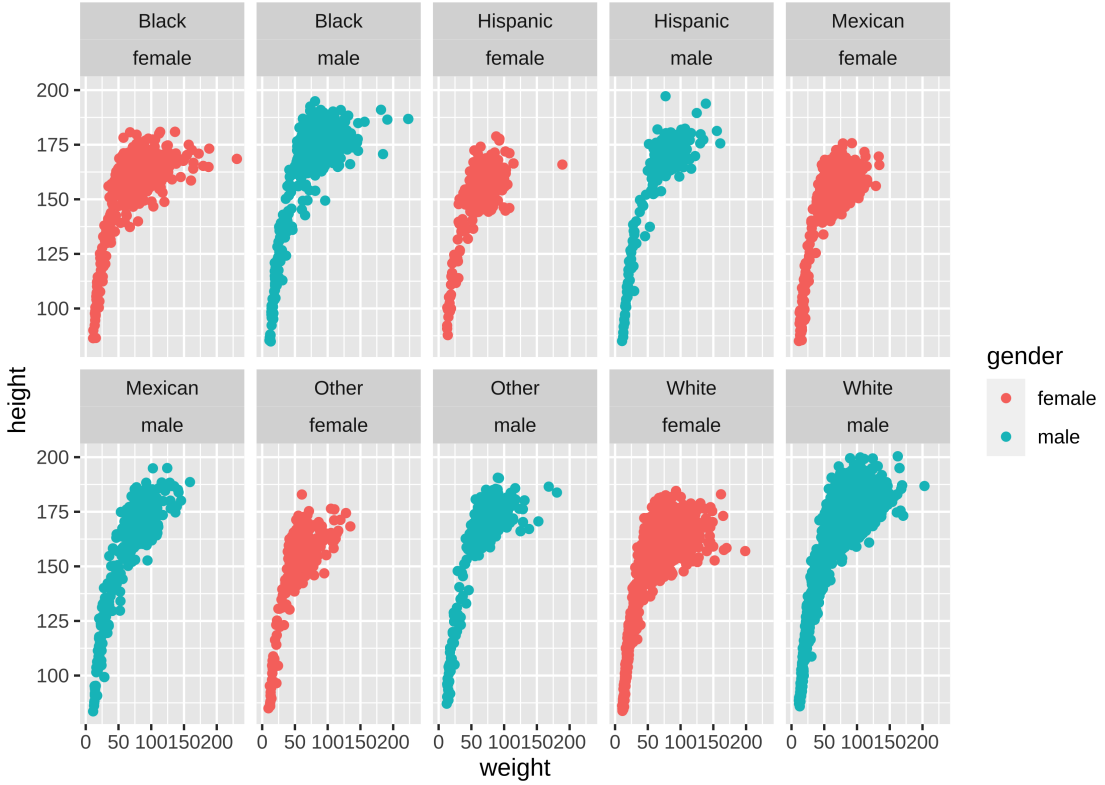
Facet Two Variables

```
SmallNhanes %>%
  ggplot(aes(x = weight, y = height)) +
  geom_point(aes(color = gender)) +
  facet_wrap(race1 ~ gender)
```



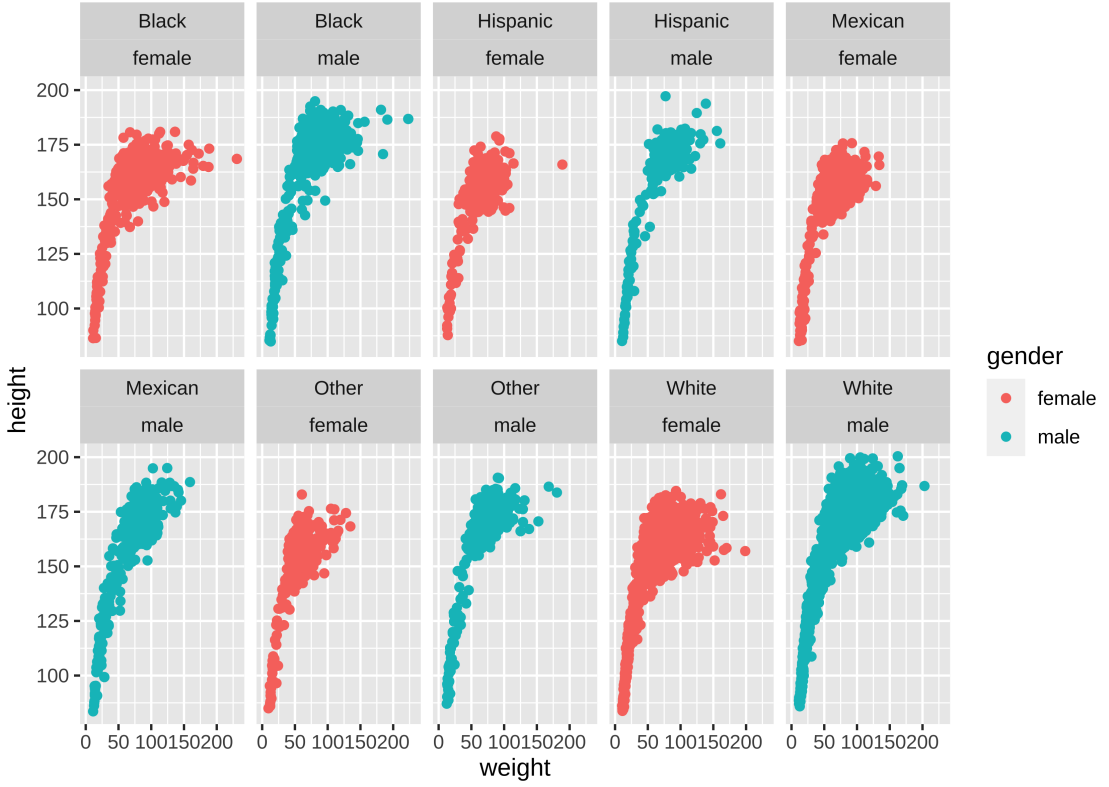
Facet: Set Columns

```
SmallNhanes %>%
  ggplot(aes(x = weight, y = height)) +
  geom_point(aes(color = gender)) +
  facet_wrap(race1 ~ gender, ncol = 5)
```



Facet: Set Rows

```
SmallNhanes %>%
  ggplot(aes(x = weight, y = height)) +
  geom_point(aes(color = gender)) +
  facet_wrap(race1 ~ gender, nrow = 2)
```



Recap

1) Introduction the grammar of graphics syntax

2) Identifying graph aesthetics (position, color, shape, opacity, etc.)

3) Recognizing and using **`geoms`** (**`geom_point`**, **`geom_smooth`**, etc.)

4) Facetting graphs (**`facet_wrap`** with 1 or two variables)

More resources

The `ggplot2` book

`ggplot2` on the tidyverse website

Flowing Data