Day Nine: Visualization Conventions

SDS 192: Introduction to Data Science

Lindsay Poirier Statistical & Data Sciences, Smith College

Spring 2022

Introduction

The goals of this lab are to provide you with practice in producing legible and effective data visualuzations in `R`. Specifically, you will be practicing.

1. Visualization Principles in `R`
2. How to read the `ggplot` cheatsheets
3. How to adjust the style and attributes of a plot
4. How to apply color to plots
5. How to deal with overplotting

Pre-requisites:

Since we will be working with color palettes today, we are going to install two color palettes in `R`. Enter the following in your `R` Console.

```
install.packages("RColorBrewer")
```

> Note: Remember, we only need to install packages once to have their functions available for us in our `R` environments.

Run the line of code below to load the functions in the tidyverse library into our workspace, along with the viridis and RColorBrewer packages.

```
library(tidyverse)
```

```
## Warning: package 'readr' was built under R version 4.1.2
```

```
library(RColorBrewer)
```

Today we will be working with an incredible data resource from the University of Wisconsin Population Health Institute. It aggregates data from a number of government sources to produce county health indicators for every county in the US. Something to keep in mind when reviewing this data:

> While this is a powerful data source for visualizing health disparities, particularly in smaller counties, there can be large degrees of uncertainty in the reporting of certain health measures. We won't be working with this dataset's confidence intervals today, so we do want to consider that the visualizations that we see are not a perfect reflection of county health.

Run the code below to load in the datasets we will be working with today.

```
library(tidyverse)
read_File = readLines("https://www.countyhealthrankings.org/sites/default/files/media/document/analytic
county_health_2021 <- read.csv(textConnection(read_File[-2]), header = TRUE, sep=",")

county_health_2021 <-
  county_health_2021 %>%
  rename_all(~str_replace_all(.,"\\.","_")) %>%
  rename_all(~str_replace_all(.,"___","_")) %>%
  filter(State_FIPS_Code != 00 & County_FIPS_Code != 0) %>%
  select(State_FIPS_Code:Release_Year, ends_with("value"), -starts_with("X"))

ma_county_health_2021 <- county_health_2021 %>%
  filter(State_Abbreviation == "MA")

ne_county_health_2021 <- county_health_2021 %>%
  filter(State_Abbreviation %in% c("MA", "RI", "CT", "VT", "NH", "ME"))
```

Visualization Principles

While there are a variety of ways we can display information, some ways can help make data more clear and legible than others. Data visualization and communication researchers have shown how our eyes are often not very good at picking up on the nuanced differences between certain kinds of visual cues than others. For instance, there is pretty widespread consensus that pie charts are a terrible way to visualize data because our eyes are not very good at distinguishing the difference between angles.

Further, some methods of visualization present data with more integrity to their original form than others. Visualization expert Edward Tufte argues that we can pursue "graphical integrity" by aiming to keep our **data-to-ink** ratio on a plot as close to 1 as possible. This means that the amount of ink we add to the plot should be comparable to the amount of data we display on a plot, minimizing decoration. The goal is to avoid other distractions in order to focus our attention on just the data.

It's important to note that clarity and legibility is not always the goal of data visualization. Feminist data colleagues have shown how sometimes we design visualizations to elicit emotion or provoke contemplation, and that decoration can sometimes animate that. Further, there are critical tradeoffs to balancing data-to-ink on a plot. Researchers have shown how these conventions can send the signal that data emerged from nowhere in particular, rather than from people with certain standpoints, ideas, and biases. They have shown that minimizing decoration doesn't necessarily remove human judgment from the data (because human judgment is always in our data) but instead serves the rhetorical purpose of convincing us to trust the data. As we think about how to design our data visualizations, we might contemplate our goals for display and how the visualization conventions we choose to implement advance them.

Today, we are going to work through a lab that demonstrates how we can apply some formatting conventions to data visualizations to make them more legible and to present them with more integrity. Before we begin, though I'd like to draw your attention to what I think is the greatest resource out there for ggplot - the ggplot cheatsheet.

I reference this cheatsheet probably more than any other cheatsheet in `R`. There are so many ways you can customize a plot with ggplot, and this summarizes many of those customizations!

Note how this cheatsheet is organized. There are headings for things like:

- Basics
- Geom functions

- Scales
- Coordinate systems

The first tip to find what you're looking for is to consider what heading the graphic element will likely fall under. Also note that, for many functions on this cheatsheet:

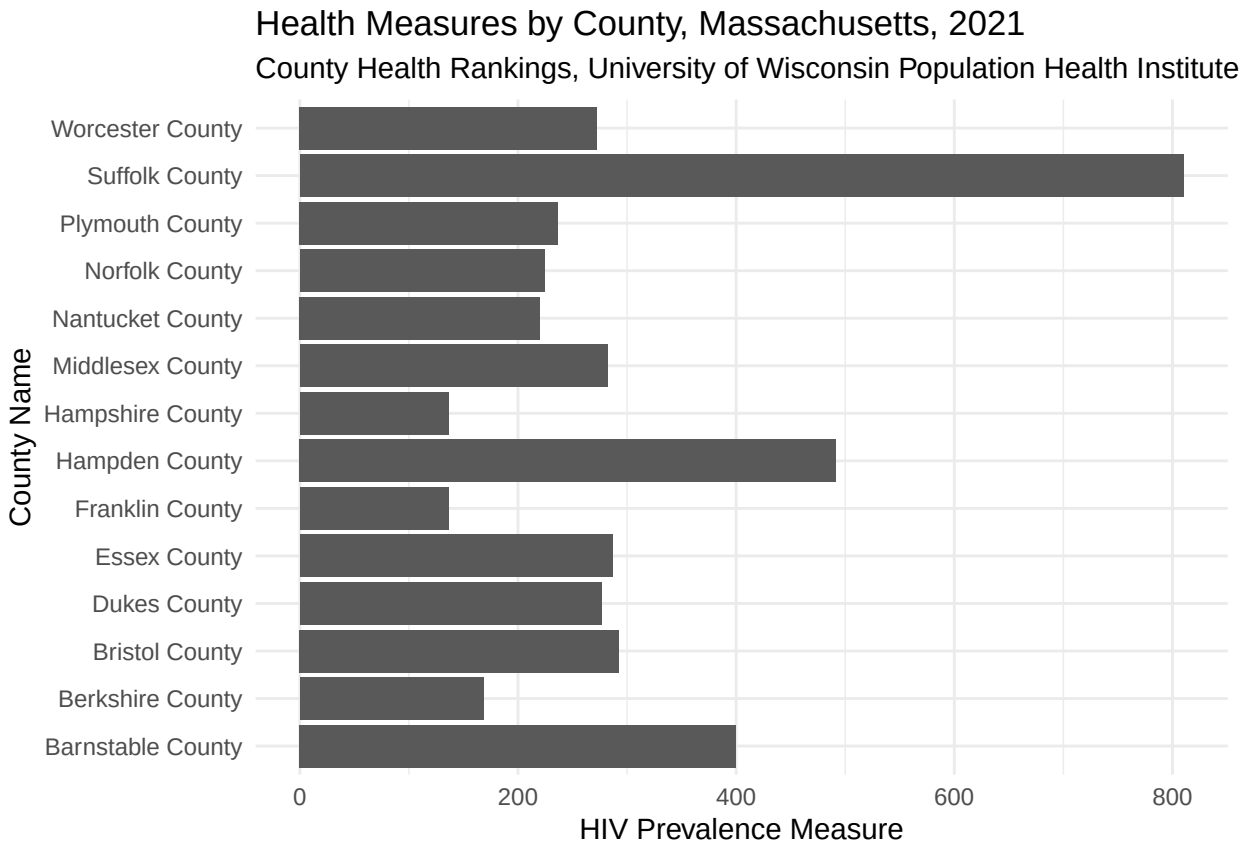
1. an image is provided for how the plot will transform,
2. the function and its arguments are referenced in bold, and
3. a text description of what will happen when you apply the function is provided

Finally note that most of these functions listed on this cheatsheet are appended to the `ggplot` function call with a `+` sign.

Exercise 1: Reference the `ggplot()` cheatsheet to add a subtitle to this plot and to set the theme to *minimal* to balance the data-to-ink ratio.

Your subtitle should be the data's source: County Health Rankings, University of Wisconsin Population Health Institute

```
ggplot(ma_county_health_2021,
       aes(x = Name, y = HIV_prevalence_raw_value)) +
  geom_col() +
  coord_flip() +
  labs(title = "Health Measures by County, Massachusetts, 2021",
       x = "County Name",
       y = "HIV Prevalence Measure",
       subtitle = "County Health Rankings, University of Wisconsin Population Health Institute") +
  theme_minimal()
```



What kinds of disparities in HIV prevalence are indicated here? What social and historical conditions might explain these disparities?

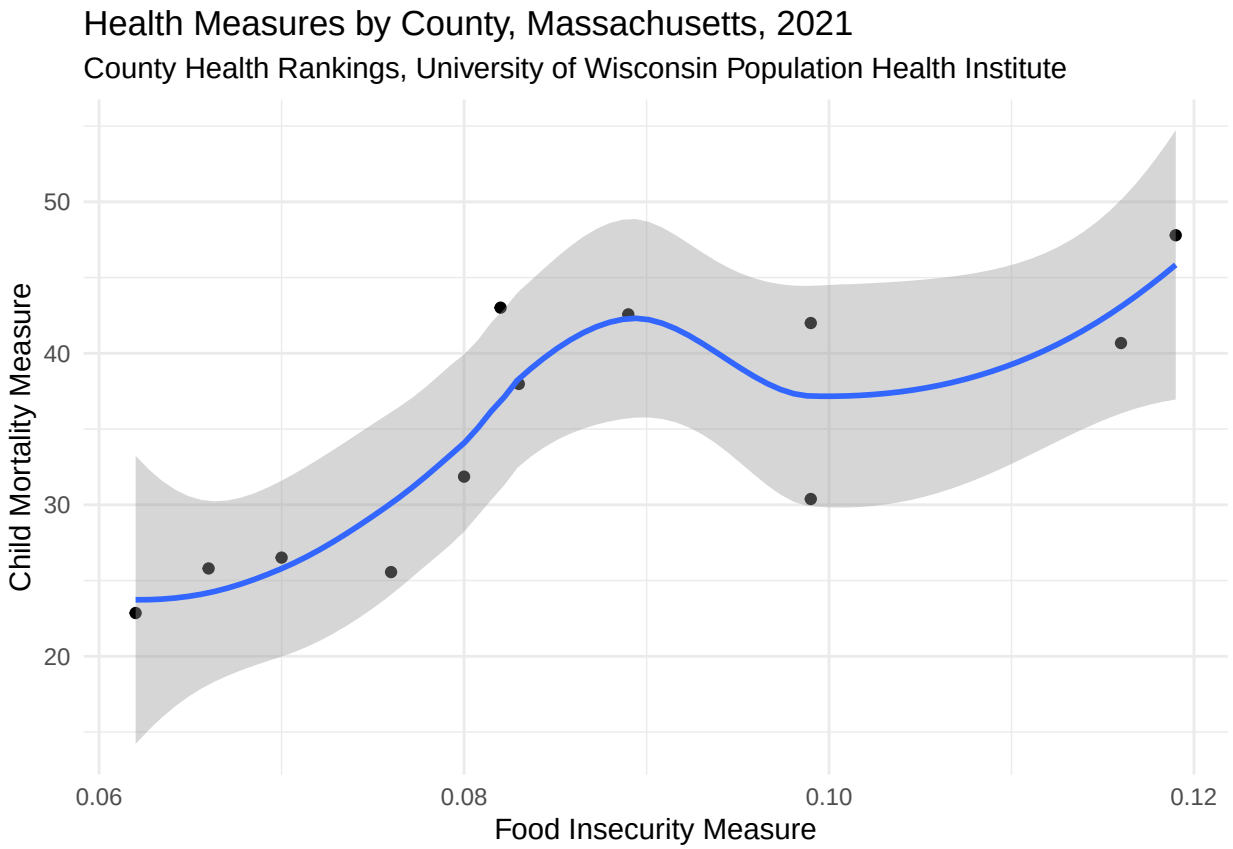
Exercise 2: Reference the `ggplot()` cheatsheet to add a subtitle to this plot (same as above), to set the theme to *minimal*, and add a trend line.

Hint: This line adds **smoothed** conditional means, or a regression line, to the plot.

```
ggplot(ma_county_health_2021,
       aes(x = Food_insecurity_raw_value, y = Child_mortality_raw_value)) +
  geom_point() +
  labs(title = "Health Measures by County, Massachusetts, 2021",
       x = "Food Insecurity Measure",
       y = "Child Mortality Measure",
       subtitle = "County Health Rankings, University of Wisconsin Population Health Institute") +
  theme_minimal() +
  geom_smooth()
```

```
## Warning: Removed 2 rows containing non-finite values (stat_smooth).
```

```
## Warning: Removed 2 rows containing missing values (geom_point).
```



Reflection: What kind of relationship between food security and child mortality do we see here?

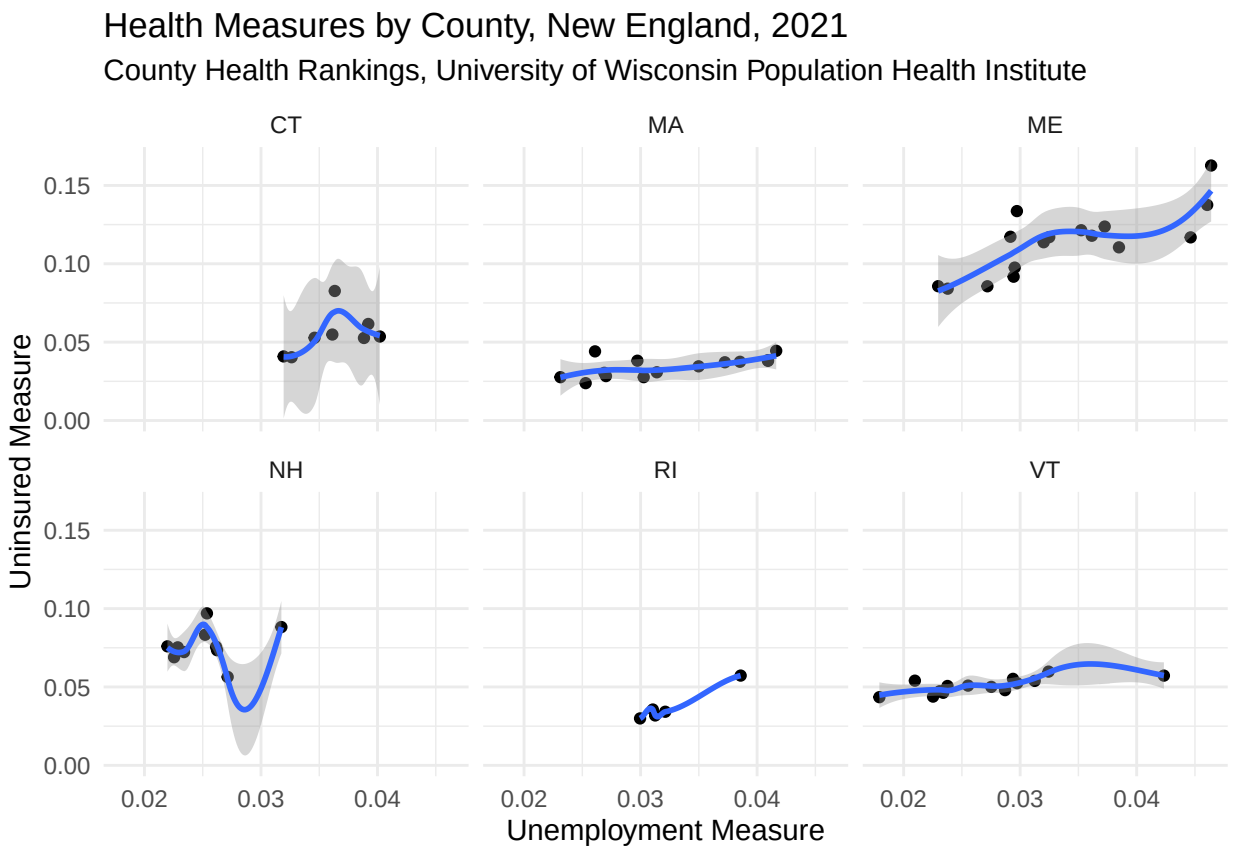
Small Multiples

Sometimes, the narrative a plot conveys can be misleading until we starting layering in additional variables. For instance, in the following plot we look at the correlation between unemployment rates and uninsured individuals in New England counties. ... but different states in New England implement different policies when it comes to both employment and insurance, so it may make more sense to divide this plot out by state. We can do this by creating small multiples using the `facet_wrap()` function. Reference the cheatsheet to determine how to facet the following plot by state.

Exercise 3: Divide this plot by state using `facet_wrap()`.

```
ggplot(ne_county_health_2021,
       aes(x = Unemployment_raw_value, y = Uninsured_raw_value)) +
  geom_point() +
  geom_smooth() +
  labs(title = "Health Measures by County, New England, 2021",
       x = "Unemployment Measure",
       y = "Uninsured Measure",
       subtitle = "County Health Rankings, University of Wisconsin Population Health Institute") +
  theme_minimal() +
  facet_wrap(vars(State_Abbreviation))
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



Reflection: In which states do counties have the highest uninsured measures? In which states do counties have the lowest?

Adding Size to Plots

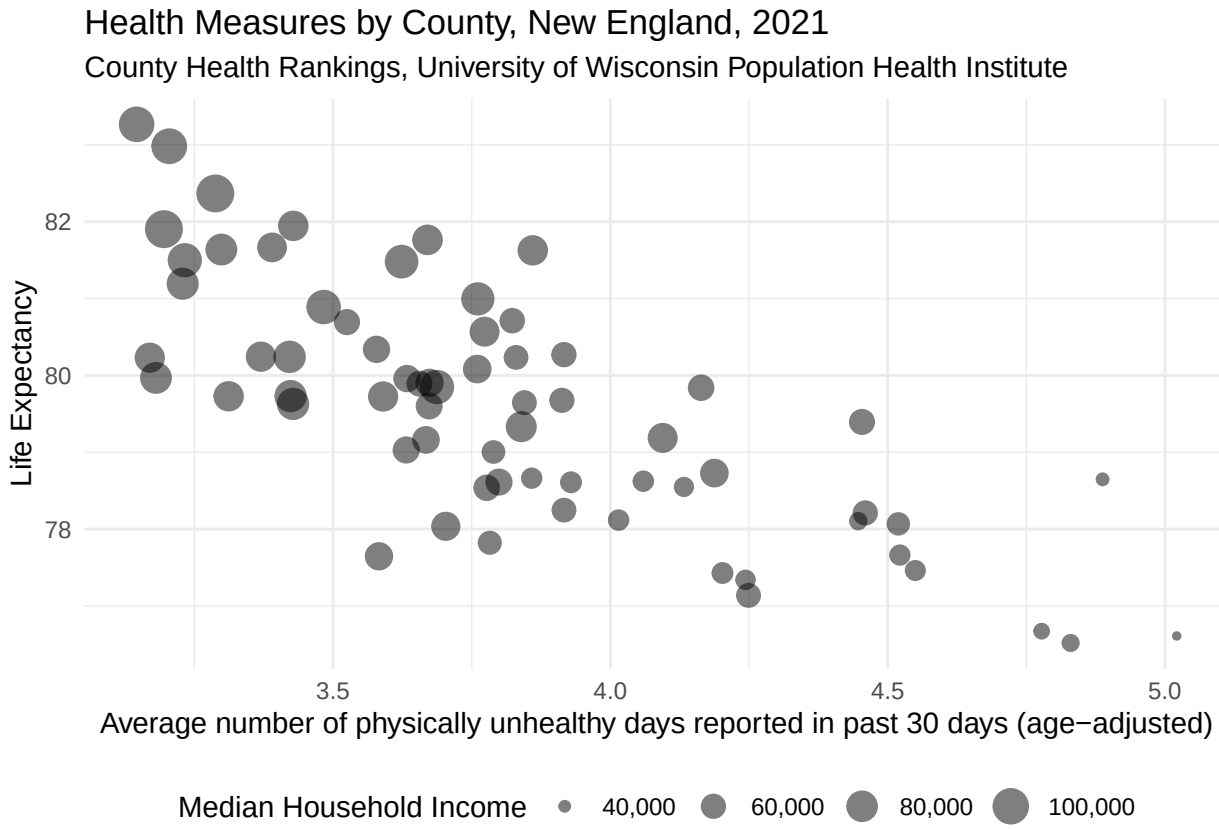
Another way we can layer more variables onto a plot is through the *size* aesthetic. In addition to supplying variables to the x and y in the `aes()` function, we can supply a third variable to the *size* argument. When adding this third variable, a legend will automatically be added to the plot indicating what the range in size means. Just like we set the labels for our x and y axis, we need to set the labels for the legend. We can also adjust the legend's position on our plot.

Exercise 4: Adding a Size Aesthetic to Plot

Reference the `ggplot()` cheatsheet to adjust the following plot in these ways:

- Set the *size* of the point to the median income value
- Change the label for the legend (hint: you will supply the name of the aesthetic argument in the `labs()` function)
- Set the legend's position to the bottom of the plot

```
ggplot(ne_county_health_2021,
       aes(x = Poor_physical_health_days_raw_value,
           y = Life_expectancy_raw_value,
           size = Median_household_income_raw_value)) +
  geom_point(alpha = 0.5) +
  labs(title = "Health Measures by County, New England, 2021",
       x = "Average number of physically unhealthy days reported in past 30 days (age-adjusted)",
       y = "Life Expectancy",
       size = "Median Household Income",
       subtitle = "County Health Rankings, University of Wisconsin Population Health Institute") +
  theme_minimal() +
  theme(legend.position = "bottom") +
  scale_size_continuous(labels = scales::comma)
```



Bonus!: See if you can figure out how to convert the income values in the legend from scientific to comma notation. (Hint: You might reference how we converted decimals to percentages along our axes in the last lab.)

Reflection: What does this plot tell us that we couldn't see if we only considered the variables on the x and y axis?

Adding Color to Plots

As you read this week, there are three types of color palettes that can be added to a plot:

- Sequential: A uni-directional ordering of shades.
- Qualitative: A discrete set of colors.
- Divergent: A bi-directional ordering of shades.

We're going to use the `RBrewer` package to add all three to these plots. To start, enter the following in your Console to check out this package's color palettes:

```
display.brewer.all(colorblindFriendly = TRUE)
```

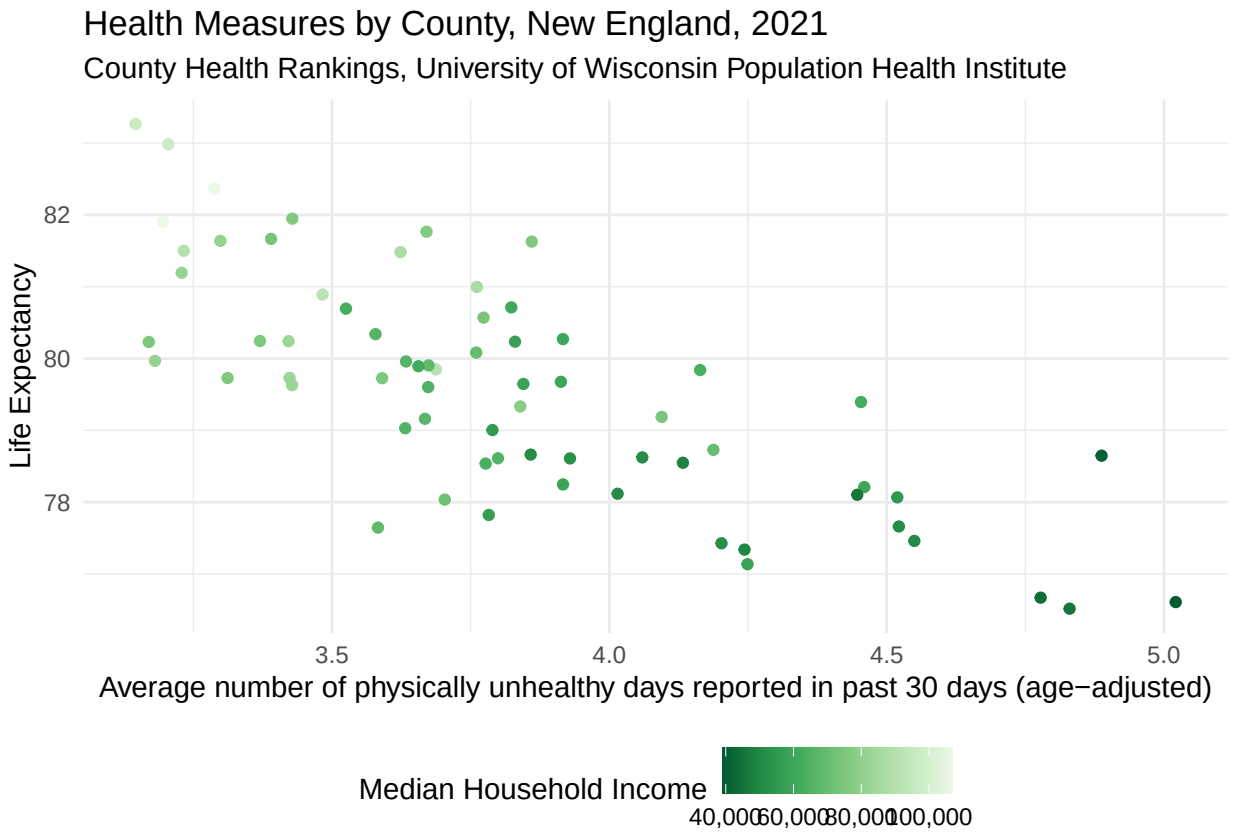
The first set is sequential, the second is qualitative, and the third is divergent.

Exercise 5: Add a Sequential Color Aesthetic to Plot

Copy and paste the plot you created in Exercise 4 into the code snippet below. Change the *size* argument in your `aes()` function to *col* (short for color). Be sure to also change this argument in your `labs()` function to update your legend label. Finally, append the following argument to the end of your plot:

```
scale_color_distiller(palette = "Greens", labels = scales::comma)
```

```
ggplot(ne_county_health_2021,
       aes(x = Poor_physical_health_days_raw_value,
           y = Life_expectancy_raw_value,
           col = Median_household_income_raw_value)) +
  geom_point() +
  labs(title = "Health Measures by County, New England, 2021",
       x = "Average number of physically unhealthy days reported in past 30 days (age-adjusted)",
       y = "Life Expectancy",
       col = "Median Household Income",
       subtitle = "County Health Rankings, University of Wisconsin Population Health Institute") +
  theme_minimal() +
  theme(legend.position = "bottom") +
  scale_color_distiller(palette = "Greens", labels = scales::comma)
```



Why do we use a sequential pattern here instead of a divergent pattern? Because Median Household Income is a continuous variable that starts at 0 and increases in one direction. There's not a middle or neutral value that we are comparing incomes around. We should only use divergent color palettes in cases when we are comparing distances in two directions from some middle point. For instance, maybe I would use a divergent palette to visualize temperatures above or below freezing.

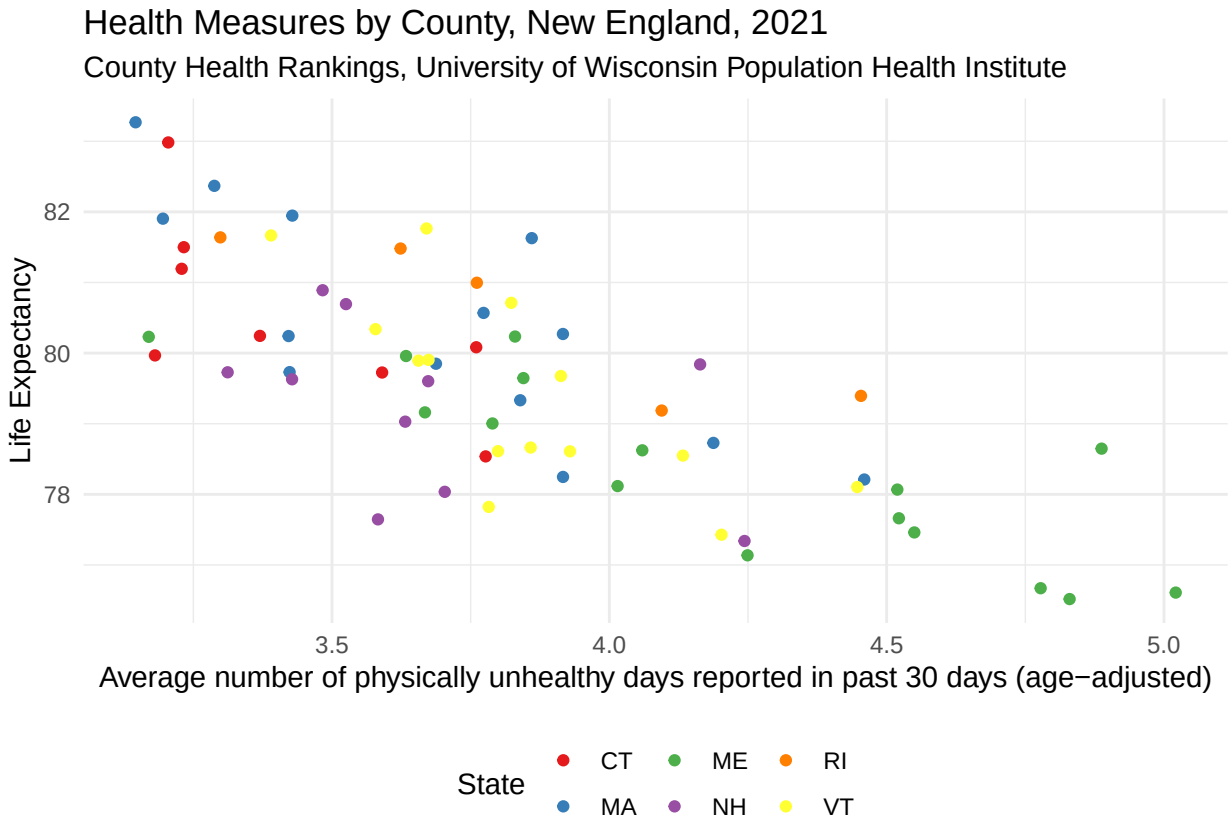
Exercise 6: Add a Qualitative Color Aesthetic to Plot

Copy and paste the plot we created above into the code snippet below. Instead of coloring the points by Median Household Income, color them by state. Be sure to update your legend title. Finally, convert the sequential palette into a categorical palette. To determine how to do this, direct your attention to the *Color and Fill Scales* section of the `ggplot()` cheatsheet. How should the word "distiller" be edited in this function call in order create a palette with a discrete set of colors? Once you determine this, update the function call and the palette to "Set2."

Hint: You don't need to change the word "color" to "fill" when adjusting the scale function name. We apply the color argument to points and lines, and the fill argument to bars and shapes. This is a point plot, so we will stick with color!

```
ggplot(ne_county_health_2021,
       aes(x = Poor_physical_health_days_raw_value,
           y = Life_expectancy_raw_value,
           col = State_Abbreviation)) +
  geom_point() +
```

```
  labs(title = "Health Measures by County, New England, 2021",
       x = "Average number of physically unhealthy days reported in past 30 days (age-adjusted)",
       y = "Life Expectancy",
       col = "State",
       subtitle = "County Health Rankings, University of Wisconsin Population Health Institute") +
  theme_minimal() +
  theme(legend.position = "bottom") +
  scale_color_brewer(palette = "Set1")
```



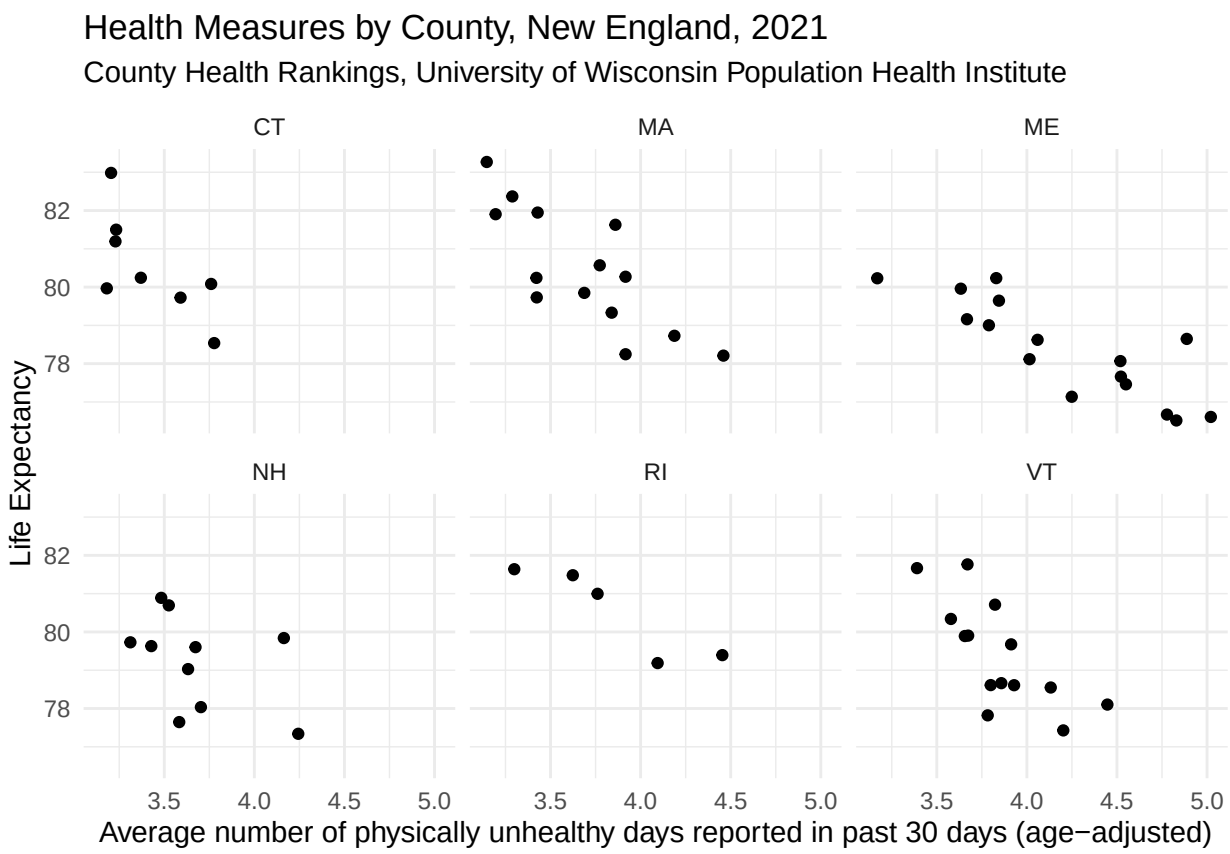
Reflection: To what extent can you identify disparities in life expectancies and unhealthy days across states? Which states tend to be higher and which lower?

Exercise 7: Facet the Plot by State

Copy and paste the plot you created above into the code snippet below. Remove the *col* aesthetic and all labels/legends and then facet the plot by state.

```
ggplot(ne_county_health_2021,
       aes(x = Poor_physical_health_days_raw_value,
           y = Life_expectancy_raw_value)) +
  geom_point() +
  labs(title = "Health Measures by County, New England, 2021",
       x = "Average number of physically unhealthy days reported in past 30 days (age-adjusted)",
       y = "Life Expectancy",
       subtitle = "County Health Rankings, University of Wisconsin Population Health Institute") +
```

```
  theme_minimal() +
  facet_wrap(vars(State_Abbreviation))
```



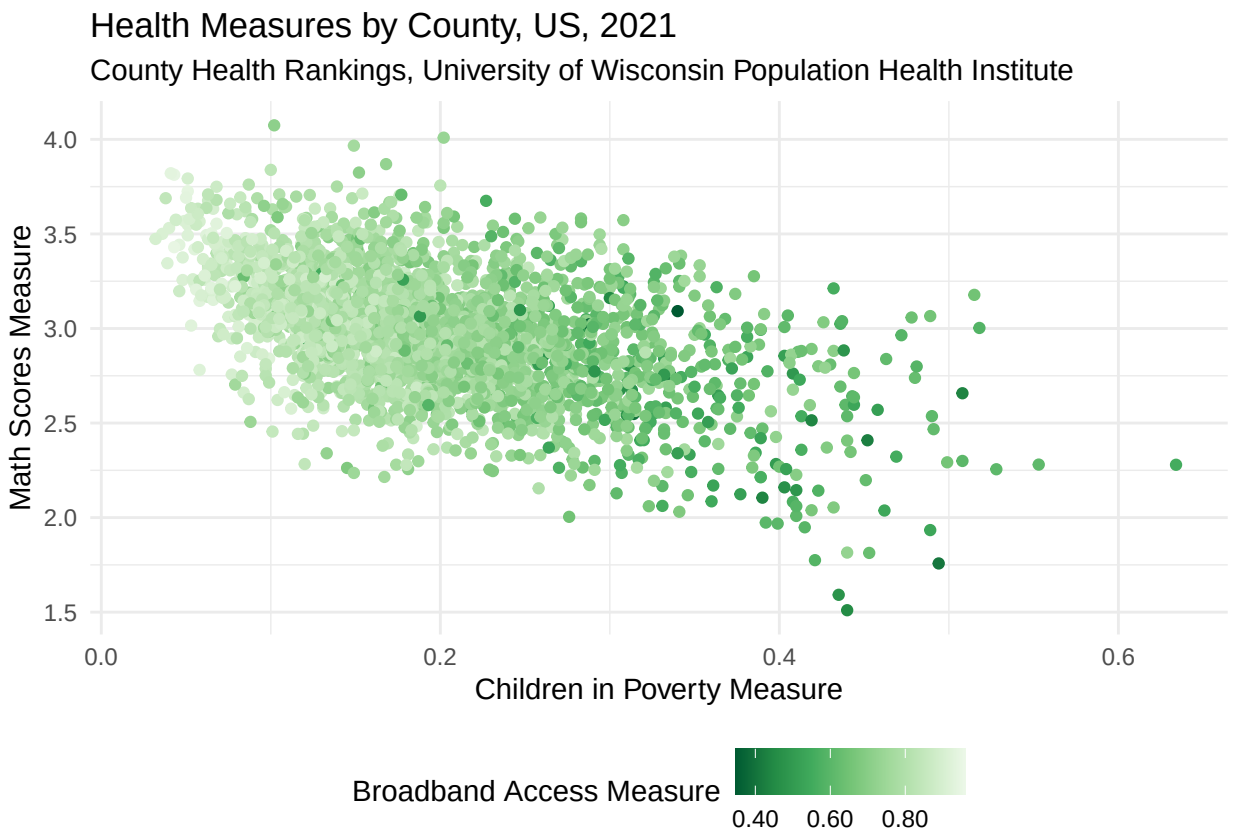
Reflection: Which of the plots you created in the last two exercises better enables you to compare health disparities across states?

Overplotting

The final topic for today's lab is overplotting - when we have so much overlapping data represented on a plot that it becomes difficult to draw any conclusions from it. Run the code below to see an example of this

```
ggplot(county_health_2021,
       aes(x = Children_in_poverty_raw_value,
           y = Math_scores_raw_value,
           col = Broadband_access_raw_value)) +
  geom_point() +
  labs(title = "Health Measures by County, US, 2021",
       x = "Children in Poverty Measure",
       y = "Math Scores Measure",
       subtitle = "County Health Rankings, University of Wisconsin Population Health Institute",
       col = "Broadband Access Measure") +
  theme_minimal() +
  theme(legend.position = "bottom") +
  scale_color_distiller(palette = "Greens", labels = scales::comma)
```

```
## Warning: Removed 448 rows containing missing values (geom_point).
```



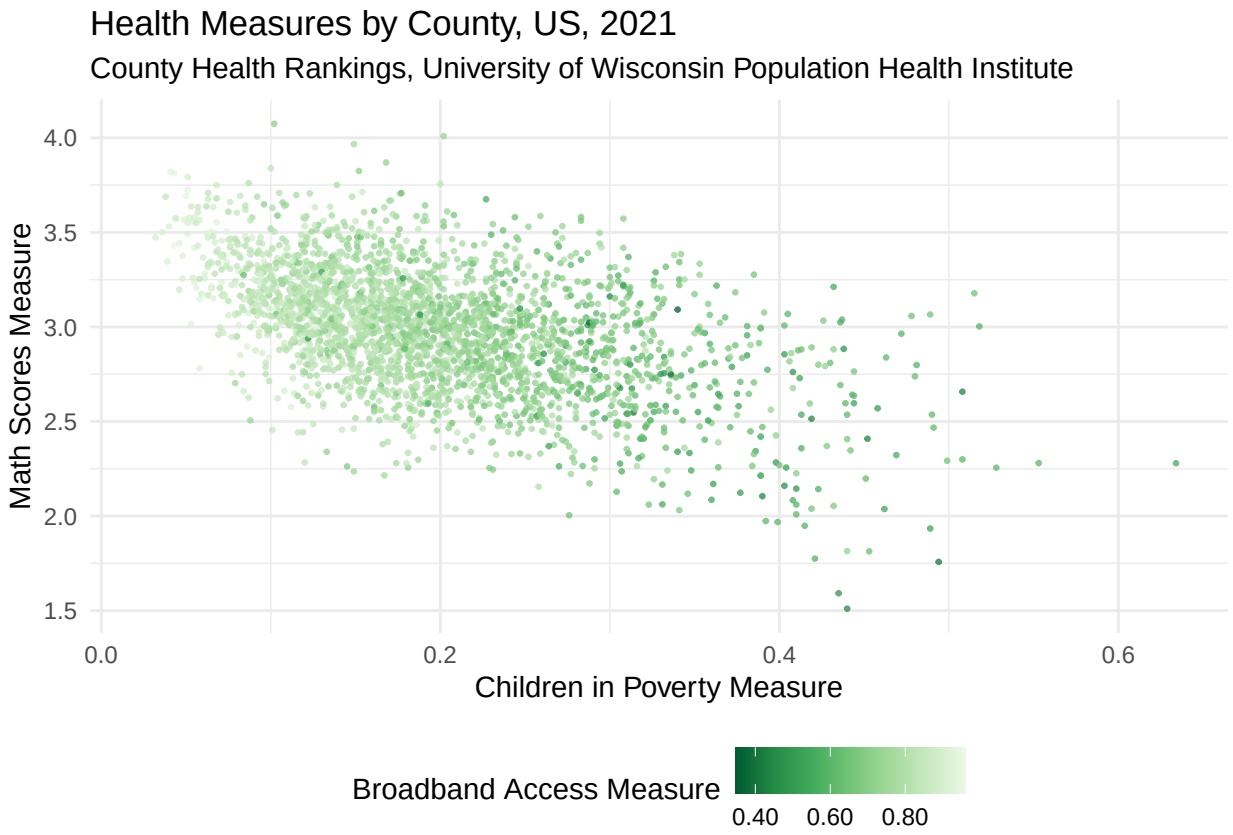
Fortunately, there are some adjustments we can make to deal with overplotting. First, we can adjust the transparency of the shapes on our plot by setting the *alpha* argument in our geom function to a number between 1 and 0. We can also reduce the size of the shapes on our plot by setting the *size* argument in our geom function to a number between 1 and 0.

Exercise 8: Adjust the Alpha and Size of Points on Plot

Copy and paste the plot I created above into the code snippet below. In the `geom_point()` function, set the *alpha* to 0.75 and the *size* to 0.5. Notice the adjustments to the plot as you update these attributes.

```
ggplot(county_health_2021,
       aes(x = Children_in_poverty_raw_value,
           y = Math_scores_raw_value,
           col = Broadband_access_raw_value)) +
  geom_point(alpha = 0.75, size = 0.5) +
  labs(title = "Health Measures by County, US, 2021",
       x = "Children in Poverty Measure",
       y = "Math Scores Measure",
       subtitle = "County Health Rankings, University of Wisconsin Population Health Institute",
       col = "Broadband Access Measure") +
  theme_minimal() +
  theme(legend.position = "bottom") +
  scale_color_distiller(palette = "Greens", labels = scales::comma)
```

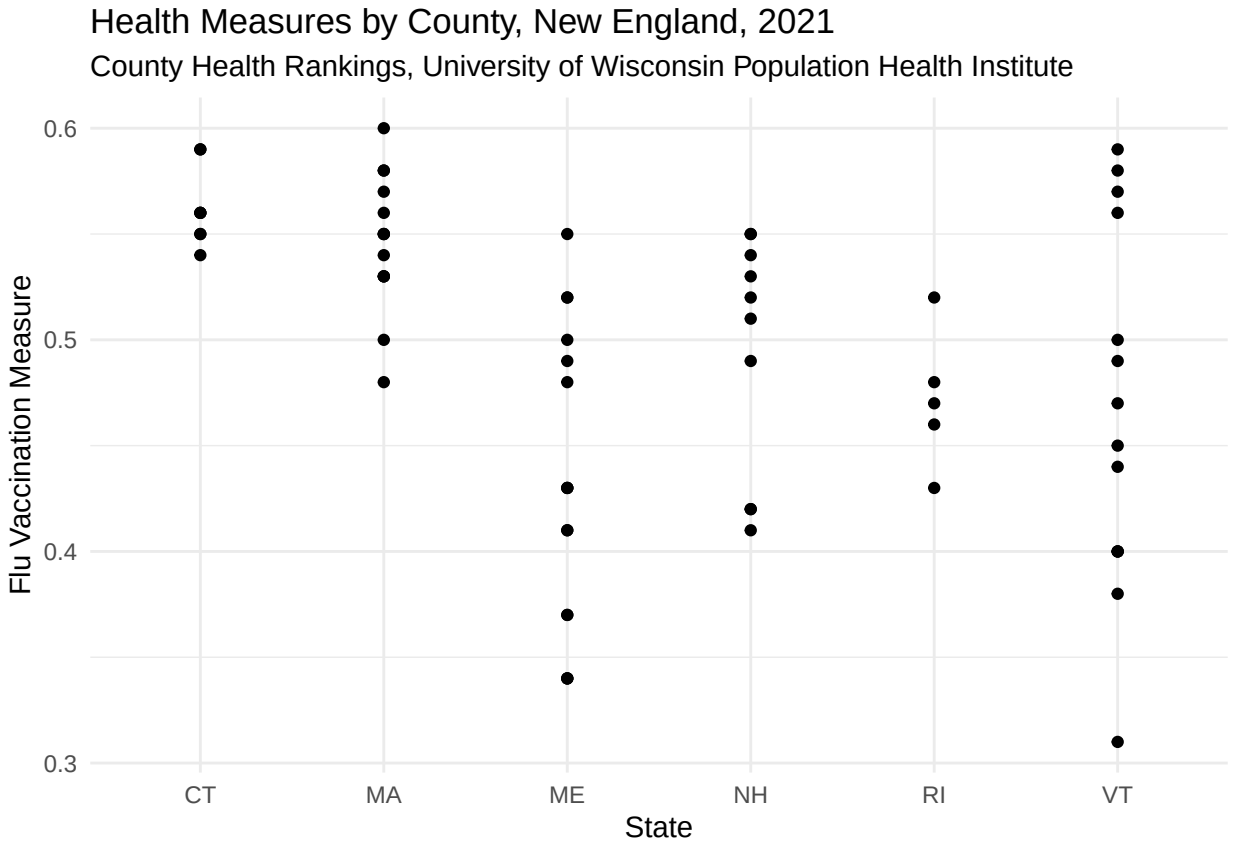
```
## Warning: Removed 448 rows containing missing values (geom_point).
```



Reflection: What conclusions would you draw from this plot?

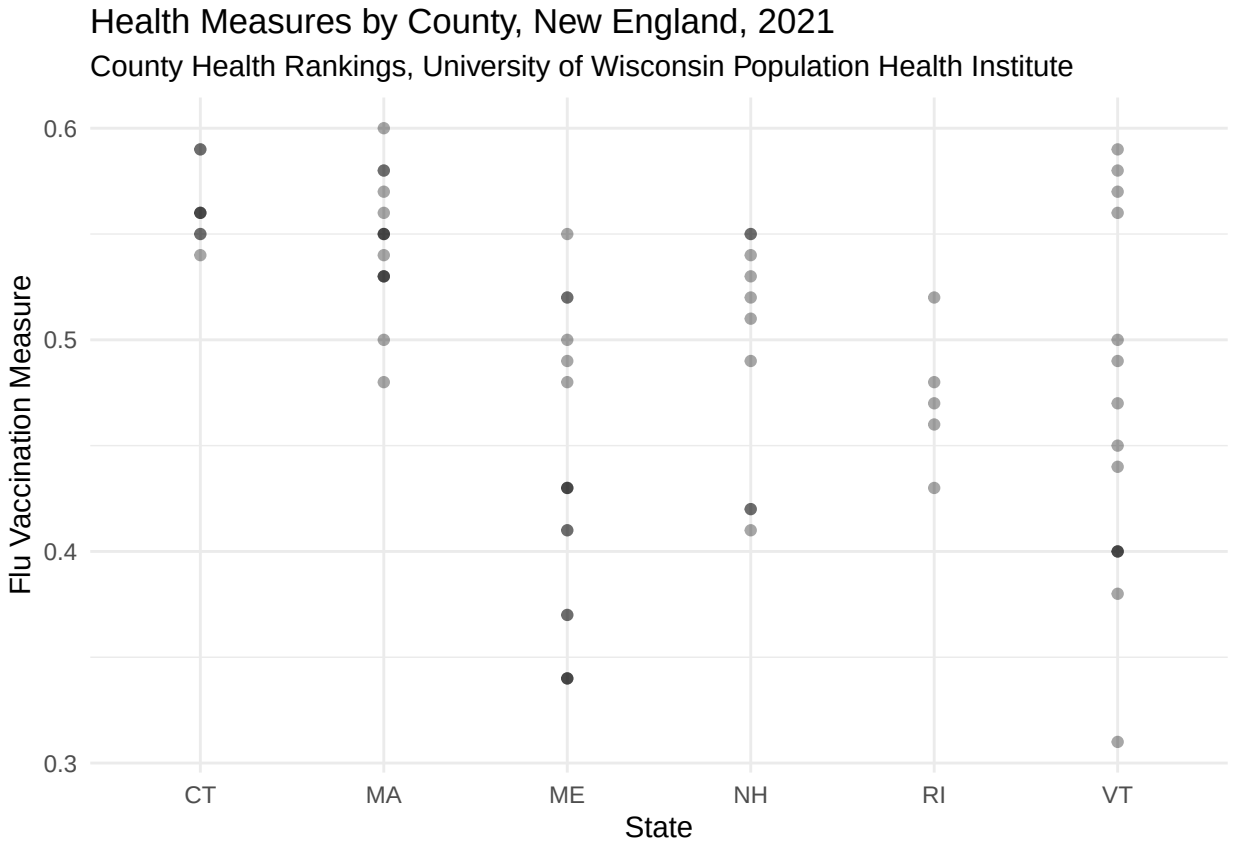
Sometimes, overplotting can also be an issue when we are plotting points discretely. Run the code below and review the plot.

```
ggplot(ne_county_health_2021,
       aes(x = State_Abbreviation,
           y = Flu_vaccinations_raw_value)) +
  geom_point() +
  labs(title = "Health Measures by County, New England, 2021",
       x = "State",
       y = "Flu Vaccination Measure",
       subtitle = "County Health Rankings, University of Wisconsin Population Health Institute") +
  theme_minimal()
```



While it may look fine at first glance, it turns out that some counties have the exact same flu vaccination measure and are overlapping each other on that plot. We could add an alpha argument to see where there are overlaps.

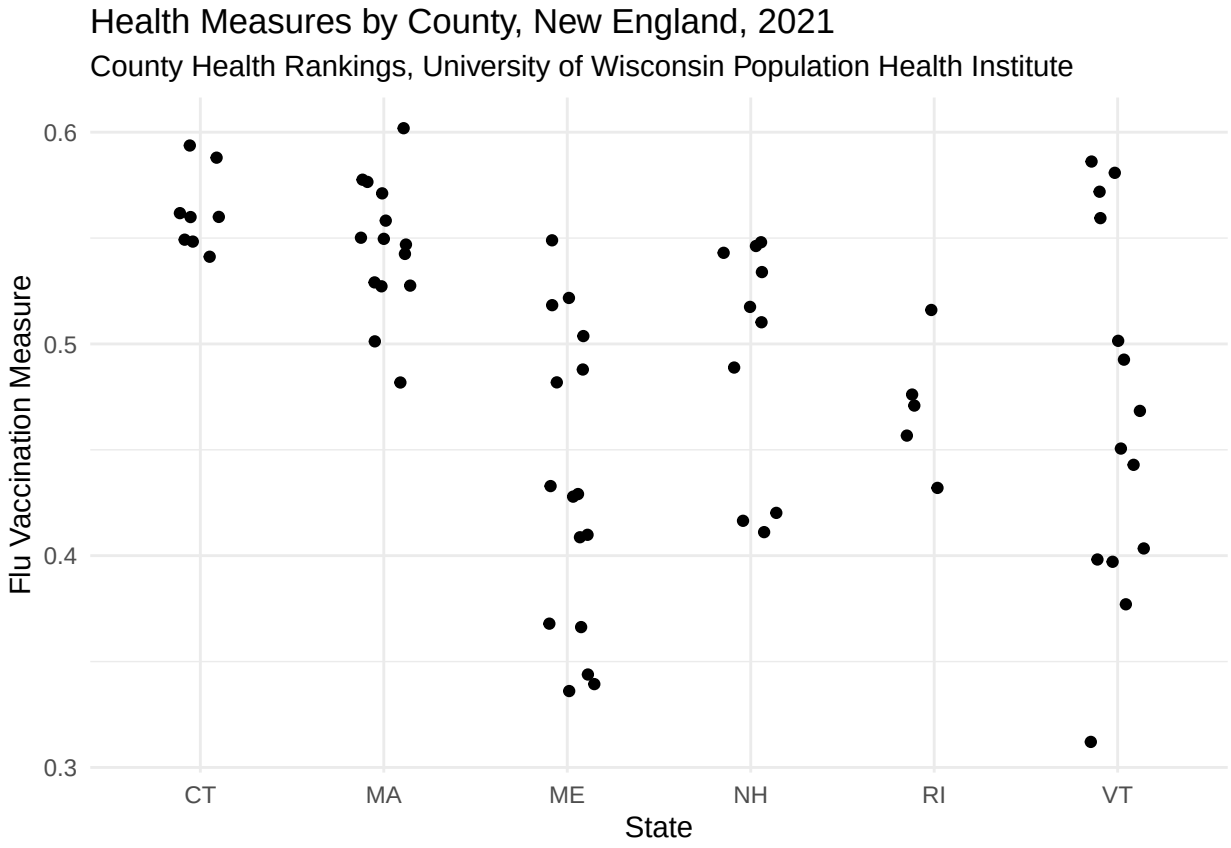
```
ggplot(ne_county_health_2021,
       aes(x = State_Abbreviation,
           y = Flu_vaccinations_raw_value)) +
  geom_point(alpha = 0.35) +
  labs(title = "Health Measures by County, New England, 2021",
       x = "State",
       y = "Flu Vaccination Measure",
       subtitle = "County Health Rankings, University of Wisconsin Population Health Institute") +
  theme_minimal()
```



However, in this case, to be sure that every county is represented on the plot, a better option, would be to add *jitter* to the plot. Jitter offsets the points from their original position slightly so that we can make out points that overlap. ## Exercise 9: Add jitter to the Plot

Copy the plot that I created above into the code snippet below. Using the cheatsheet, find the geom function for creating a jittered plot. Set the *width* of the jitter to 0.15

```
ggplot(ne_county_health_2021,
       aes(x = State_Abbreviation,
           y = Flu_vaccinations_raw_value)) +
  geom_jitter(width = 0.15) +
  labs(title = "Health Measures by County, New England, 2021",
       x = "State",
       y = "Flu Vaccination Measure",
       subtitle = "County Health Rankings, University of Wisconsin Population Health Institute") +
  theme_minimal()
```



Reflection: Which New England states have counties with the highest flu vaccination rates? In which states are there greater disparities in vaccination rates across counties?