Angela M. Zoss, Ph.D.

Visualization for Data Science with R

To my family.

I'm so grateful for your support.

Contents

List of Figures

List of Tables

About This Book

Note: This book is a work in progress, with a full draft expected in April of 2022.

Proposal

This book combines instruction on writing R code with building basic graphic design skills in a way that is unusual in data science literature. The book will guide readers through a series of projects, each designed to cover both how visualizations work in R and how visualizations can be designed to have the greatest impact. Far more than a "do this, then this" checklist, this book will focus on building understanding, confidence, and the ability to transfer skills to other tools and design contexts. It will avoid technical jargon that our target audience is unlikely to have encountered before. To accommodate learners who don't have time to work through an entire book, each chapter will operate independently, covering a specific set of tasks that all make sense together as part of a visualization project. For those who would like extra practice, there will be several types of hands-on exercises, from those that are entirely prescribed to those that allow readers to apply new techniques to problems in their own areas.

The book will have solutions (in the form of completed code and sample output) for all exercises. While not a textbook, the book will also include a brief teacher's guide for courses that might want to use one or more chapters to structure lessons in a course. The book will also have a website, including links to Open Access content, solutions, and related resources like video tutorials.

The target audience of this book would be professionals who are having to learn data science techniques on the job, likely at an under-resourced organization or company. These newly minted data professionals may feel comfortable in Excel but have only just started to learn R for processing data. They have never used a programming language to build a visualization before, and even creating charts in Excel has often been a frustrating and mystifying process. They appreciate that R is freely available and are able to get started on a

data science project, but the idea of creating publication-quality visualizations using only code is daunting.

Increasingly, programs of study with a focus on preparing students for professional careers in under-resourced fields, like public policy and even management, include courses on data analysis and communication using freely available software. This book, while not a textbook, could easily be used for a semester-long course, titled something like "Practical data visualization for the modern workforce." A chapter could be covered each week, and larger projects could help learners synthesize chapters into a complete set of analyses and communication materials.

Why read this book

This book will be:

- Written for non-academics, beginning programmers
- Each chapter stands alone
- Covers pressing modern issues, like accessibility and ethics
- Focuses on freely available software
- Combines hands-on exercises with basic graphic design principles

Structure of the book

- Chapter 1: Overview of common visualizations and how to read them
- Chapter 2: Building basic visualizations with ggplot2
- Chapter 3: Working with textual data in ggplot2
- Chapter 4: Customizing the design of ggplot2 visualizations
- Chapter 5: Avoiding unethical design practices
- Chapter 6: Building ggplot2 visualizations into print publications
- Chapter 7: Basic accessibility for static visualizations
- Chapter 8: Exploring interactivity in visualizations with plotly and crosstalk
- Chapter 9: Using RMarkdown to build websites for projects
- Chapter 10: Using RMarkdown to build dashboards for projects
- Chapter 11: Basic usability for interactive visualizations
- Chapter 12: Teacher's guide

Software information and conventions

I used the **knitr** package (Xie, 2015) and the **bookdown** package (Xie, 2021) to compile my book. My R session information is shown below:

```
xfun::session_info()
```

```
## R version 4.1.2 (2021-11-01)
## Platform: aarch64-apple-darwin20 (64-bit)
## Running under: macOS Monterey 12.2.1
##
## Locale: en_US.UTF-8 / en_US.UTF-8 / en_US.UTF-8 / C / en_US.UTF-
8 / en_US.UTF-8
##
## Package version:
##   base64enc_0.1.3 bookdown_0.24   bslib_0.3.1
##   cli_3.2.0       compiler_4.1.2  digest_0.6.29
##   evaluate_0.15   fastmap_1.1.0   fs_1.5.2
##   glue_1.6.2      graphics_4.1.2  grDevices_4.1.2
##   highr_0.9       htmltools_0.5.2 jquerylib_0.1.4
##   jsonlite_1.8.0  knitr_1.37      magrittr_2.0.2
##   methods_4.1.2   R6_2.5.1        rappdirs_0.3.3
##   rlang_1.0.2     rmarkdown_2.12  rstudioapi_0.13
##   sass_0.4.0      stats_4.1.2     stringi_1.7.6
##   stringr_1.4.0   tinytex_0.37    tools_4.1.2
##   utils_4.1.2     xfun_0.30       yaml_2.3.5
```

Package names are in bold text (e.g., **rmarkdown**), and inline code and filenames are formatted in a typewriter font (e.g., `knitr::knit('foo.Rmd')`). Function names are followed by parentheses (e.g., `bookdown::render_book()`).

Angela Zoss

About the Author

Angela is the Assessment & Data Visualization Analyst[1] in the Assessment & User Experience Department[2] in the Duke University Libraries[3]. She has many years of experience in teaching and training, predominantly focusing on teaching data visualization to university students, faculty, and staff. She is also active in several open source development projects, including FOLIO[4] and Wax[5].

[1] https://library.duke.edu/about/directory/staff/angela.zoss
[2] https://library.duke.edu/about/depts/assessment-user-experience
[3] https://library.duke.edu/
[4] https://github.com/folio-org/
[5] https://github.com/minicomp/wax

FIGURE 1: Angela M. Zoss, Ph.D.

Introduction

Outline

- Intro to visualization in data science
  - communicating clearly is worth time & effort
  - automating graphical output saves time and isn't too hard
- What is this book trying to accomplish
  - Written for non-academics, beginning programmers;
    - The target audience of this book would be professionals who are having to learn data science techniques on the job, likely at an under-resourced organization or company. These newly minted data professionals may feel comfortable in Excel but have only just started to learn R for processing data. They have never used a programming language to build a visualization before, and even creating charts in Excel has often been a frustrating and mystifying process. They appreciate that R is freely available and are able to get started on a data science project, but the idea of creating publication-quality visualizations using only code is daunting.
  - Focuses on freely available software
  - Combines hands-on exercises with basic graphic design principles
    - This book combines instruction on writing R code with building basic graphic design skills in a way that is unusual in data science literature. The book will guide readers through a series of projects, each designed to cover both how visualizations work in R and how visualizations can be designed to have the greatest impact. Far more than a "do this, then this" checklist, this book will focus on building understanding, confidence, and the ability to transfer skills to other tools and design contexts. It will avoid technical jargon that our target audience is unlikely to have encountered before. To accommodate learners who don't have time to work through an entire book, each chapter will operate independently, covering a specific set of tasks that all make sense together as part of a visualization project. For those who would like extra practice, there will be several types of hands-on exer-

cises, from those that are entirely prescribed to those that allow readers to apply new techniques to problems in their own areas.

- What this book covers (table of contents)
  - Covers pressing modern issues, like accessibility and ethics
- How to use this book
  - Each chapter stands alone
  - The book will have solutions (in the form of completed code and sample output) for all exercises.
  - The book will also have a website, including links to Open Access content, solutions, and related resources like video tutorials.
  - Increasingly, programs of study with a focus on preparing students for professional careers in under-resourced fields, like public policy and even management, include courses on data analysis and communication using freely available software. This book, while not a textbook, could easily be used for a semester-long course, titled something like "Practical data visualization for the modern workforce." A chapter could be covered each week, and larger projects could help learners synthesize chapters into a complete set of analyses and communication materials.
  - While not a textbook, the book will also include a brief teacher's guide for courses that might want to use one or more chapters to structure lessons in a course.

1

Overview of common visualizations and how to read them

This book will cover how to create a variety of visualizations using R. One of the first things you should do to improve your skills creating visualizations is to become familiar with the kinds of visualizations that are possible and the different features of each.

Effective visualization design relies on a solid understanding of how data properties, visualization types, and audience characteristics interact to help people make sense of a visualization. In this chapter, we'll look at a series of common visualization types, and we'll break down how each is meant to be read. Understanding these basic visualization types will create a solid foundation for communicating your data science work to a broad audience.

1.1 Visualization components

As we discuss different visualizations, we will also be talking about different components within the visualizations. In figure 1.1 below, the major components of the visualization are labeled: the main title, the subtitle, the x axis title, the y axis title, the panel, the horizontal and vertical gridlines, and the axis labels and tick marks for both axes. Almost all of the visualizations we cover in this book will use these basic components.

In this set of basic visualization components, we see two components labeled as an axis. These axes are called the x and y axes, and they always appear in these positions: the x axis always goes left to right, and the y axis always goes up and down.

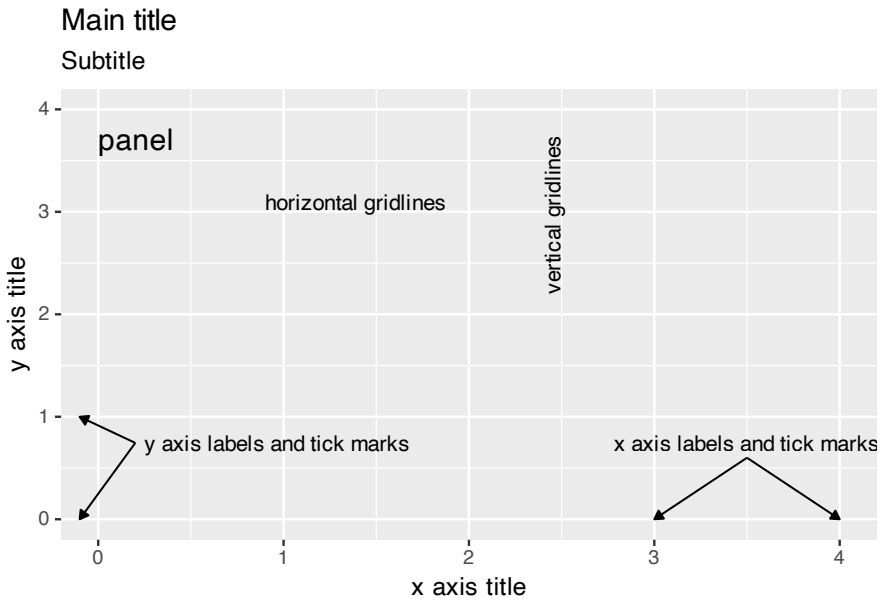


FIGURE 1.1: Visualization components, labeled.

1.2 Bar Chart

The bar chart is possibly the most common type of visualization. In this type of visualization, the basic shape being used to represent data values is a rectangle. In a traditional bar chart, each rectangle (or bar) has exactly the same width, and the height of the bar is representative of some data value. To create a simple bar chart, the data set should have one column that contains textual (or categorical) data and one column that contains numerical data. A common way to create these two columns is to start with one categorical data column and count the number of records for each category to create the numerical column.

In the sample bar chart above, the categorical variable is displayed on the x axis and the numerical variable is displayed on the y axis. This results in a classic style of bar chart where each bar has the same width and the heights are proportionate to a data value. Each bar has a starting value of zero on the y axis. Each bar travels upward from zero and stops at the correct data value.

When reading this visualization, we are comparing the lengths of bars in order to understand patterns within the numerical data values from our data set.

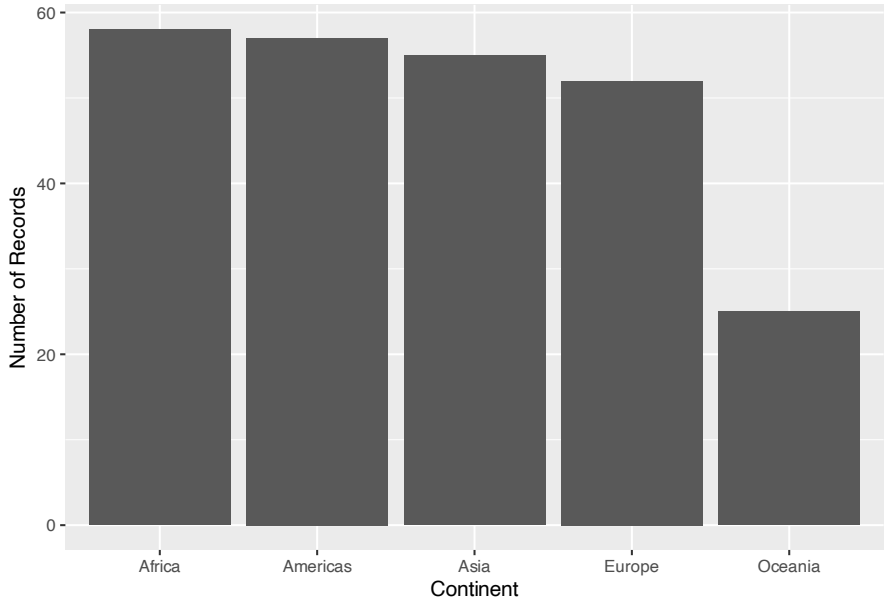


FIGURE 1.2: A sample bar chart.

The power of the bar chart lies in how precisely we can detect differences in the end points of the bars. This is something that people naturally do quite well when all of the bars start at the same lowest point, or baseline.

Bar charts are especially effective if the bars that have small differences in lengths appear close to each other. In the above chart, this is accomplished by arranging the bars so they appear with the highest data values on the left and the lowest data values on the right.

Here is another example of a bar chart where the data values include both positive and negative values. For data values that are negative, the bar travels downward from zero and stops at the correct data value. The x axis title and labels appear at the bottom of the panel, below the lowest data values.

For stylistic reasons, bar charts may also appear with the bars oriented horizontally instead of vertically. In that case, each bar will have the same height, and the widths of the bars will vary based on the data values. The text (or categorical) axis will then be the y axis, and the numerical axis will be the x axis.

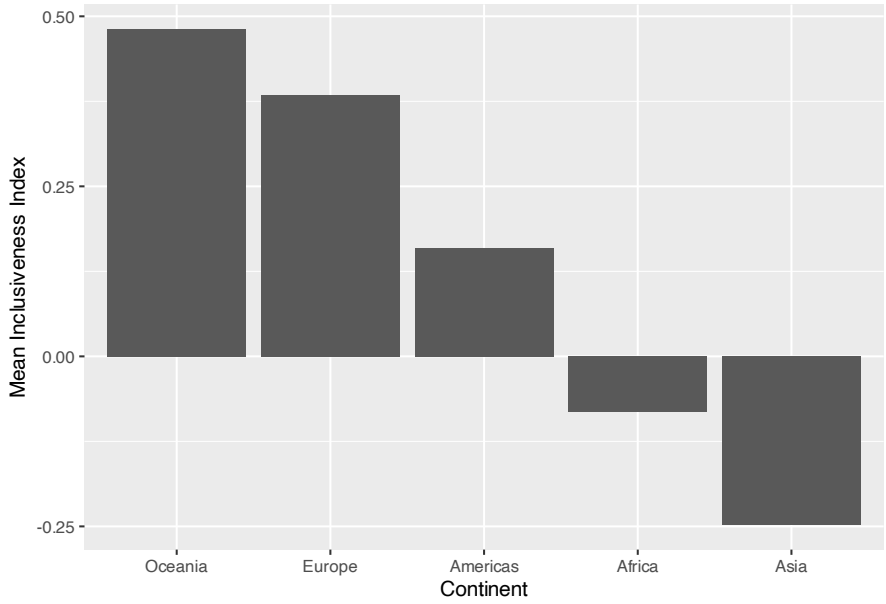


FIGURE 1.3: A sample bar chart with both positive and negative values.

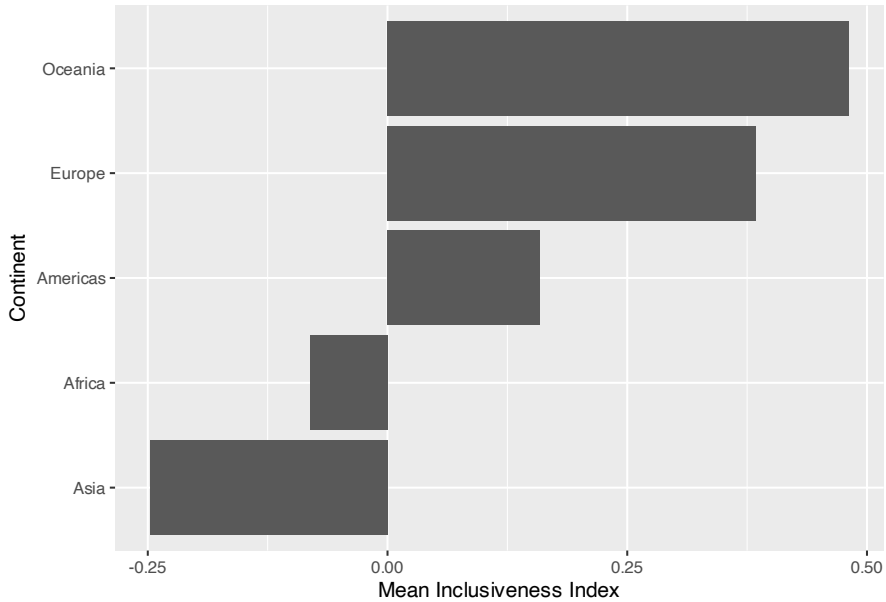


FIGURE 1.4: A sample bar chart, with the bars oriented horizonally.

1.2.1 Variations

Sometimes, we may have data that would work well for a bar chart, but we may want to explore a variation that would change the basic shape of the chart. There is one main variation of a bar chart: the lollipop chart.

1.2.1.1 Lollipop plot

The lollipop plot is a simple variation on the bar chart. In a lollipop plot, the bars are replace by a long line with a circle at the end, creating something that looks like a lollipop. Apart from the different shapes used, the lollipop plot works just like a bar chart. The circles draw attention to the data value, but the lines extending to the axis reinforce the length comparisons.

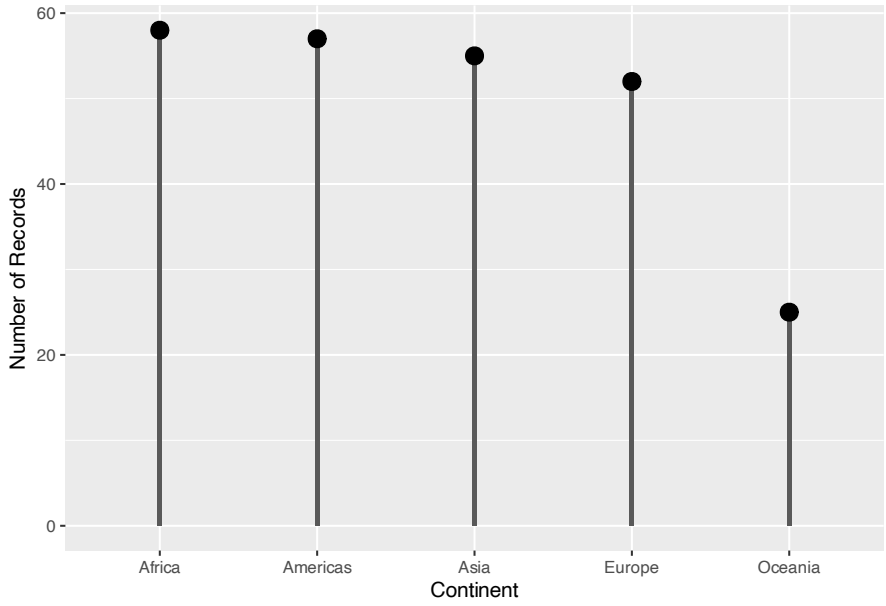


FIGURE 1.5: A sample lollipop plot.

1.2.2 Add a variable: Bar charts with color

A simple bar chart includes one categorical variable and one numerical variable. Sometimes, however, it is useful to explore the patterns in relation to a second categorical variable. Adding another categorical variable to a bar chart usually means using color to represent the extra variable.

With a stacked bar chart, the additional variable is used to segment the bars into separate, colored regions. In this example, the bar for each continent is subdivided into groups of countries based on their values for Inclusive Index.

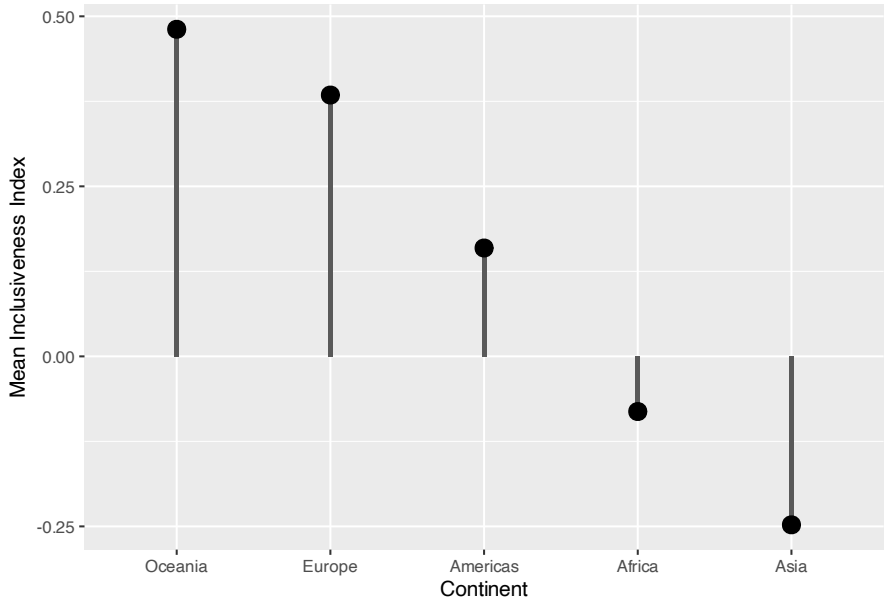


FIGURE 1.6: A sample lollipop plot with both positive and negative values.

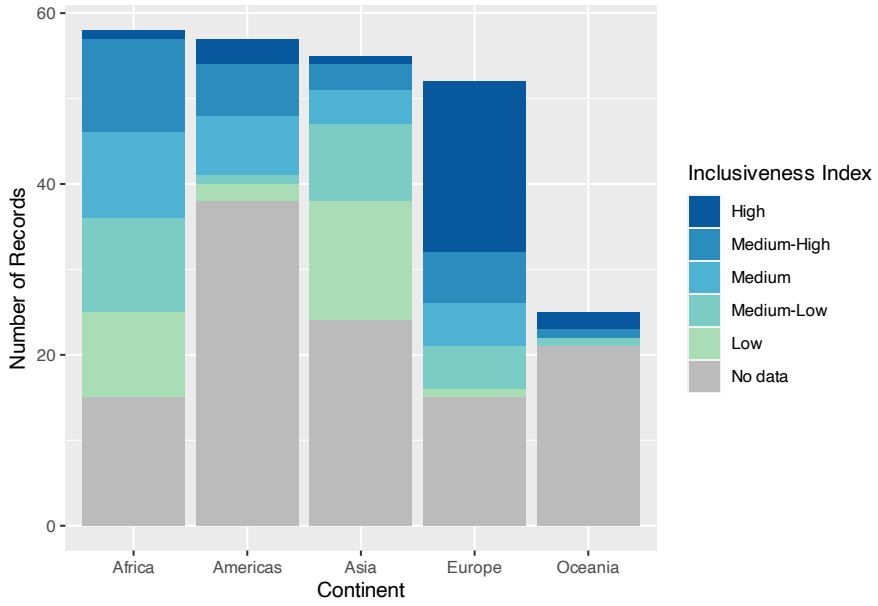


FIGURE 1.7: A sample stacked bar chart.

With this stacked bar chart, you can still see the total number of records for each Continent, but what happens when you try to compare the different segments inside the bar chart? Starting at the bottom, it seems to work out okay. All of the bars for the "No data" category start at the same baseline (the x axis), and we can read these segments like a normal bar chart. But what happens with the "Low" segments right above them? And the "Medium-Low" segments above those? Every time we have a group of segments that aren't lined up with each other, we have to try to guess how tall the bar is in comparison to the other bars in the group. The farther apart the segments are, the harder it is to make that comparison.

Another variation of the stacked bar chart is the "filled" stacked bar chart. Instead of using the raw counts to determine the lengths of the bars, in the filled stacked bar chart, the full bars are all stretched to have the same height, and each segment becomes the percentage of the records in each bar. (Notice how the y axis changes from "Number" to "Percentage.") This is useful if the percentages matter more than the raw counts, but it doesn't fix any of the concerns with comparing different segments without a common baseline.

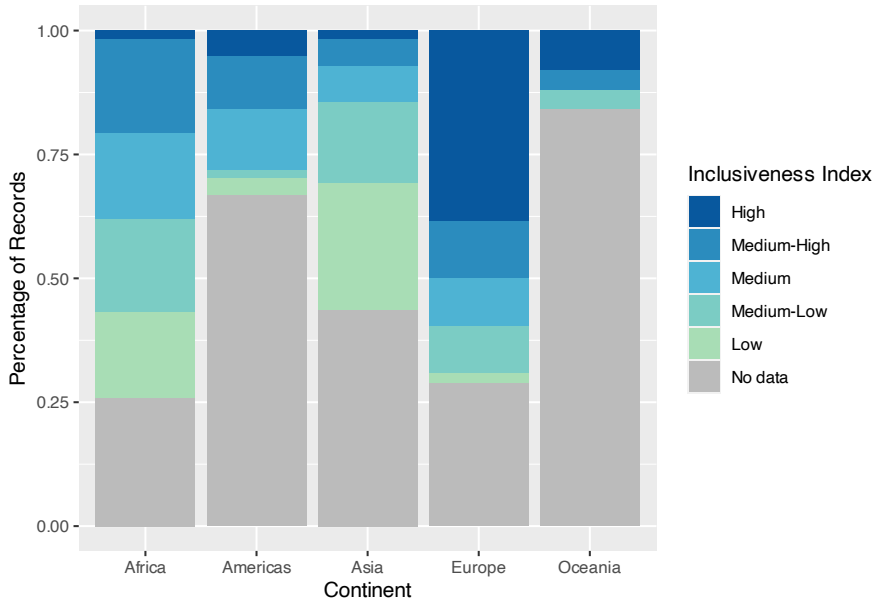


FIGURE 1.8: A sample filled stacked bar chart.

An alternative to the stacked bar chart is called the grouped bar chart. In the grouped bar chart, every segment starts from the x axis. Each continent forms a group of bars, and each option of the Inclusiveness Index is a separate bar.

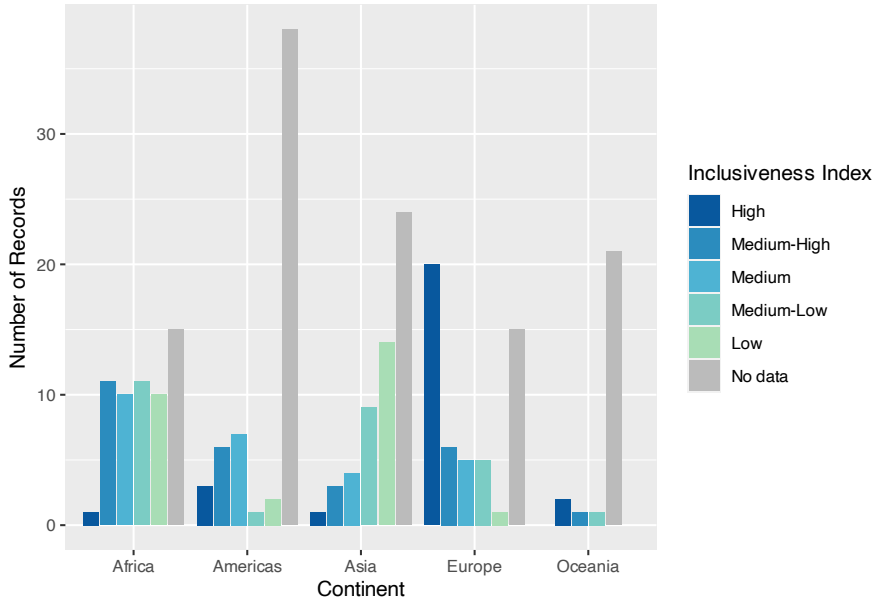


FIGURE 1.9: A sample grouped bar chart.

1.2.2.1 Dumbbell plot

Stacked and grouped bar charts show some of the limitations of bar charts for making complex comparisons. The rectangles in the bar chart take up a large amount of space. Think back to the lollipop plot, where it's the circles that directly represent the data value. Converting bars to something like circles opens up the ability to make more direct comparisons.

For example, let's say we want to compare two continents more directly: Asia and Europe.

This chart groups all of the segments by continent, which makes it easy to compare different Index categories within a single continent. What if we want to bring more attention to the difference between continents for each category? We could always switch which category is the primary division on the x axis and which is represented by color.

This improves our ability to compare the continents directly because the bars are directly next to each other. The amount of space the bars take up is still pretty large, though. If we combine this chart with something like a lollipop plot, we get one last variation: a dumbbell plot.

With a dumbbell plot, we use a circle to represent the data values, just like the lollipop. Instead of having a line that extends all the way to the axis, though, we use a line to connect the two dots in each category of Inclusiveness Index.

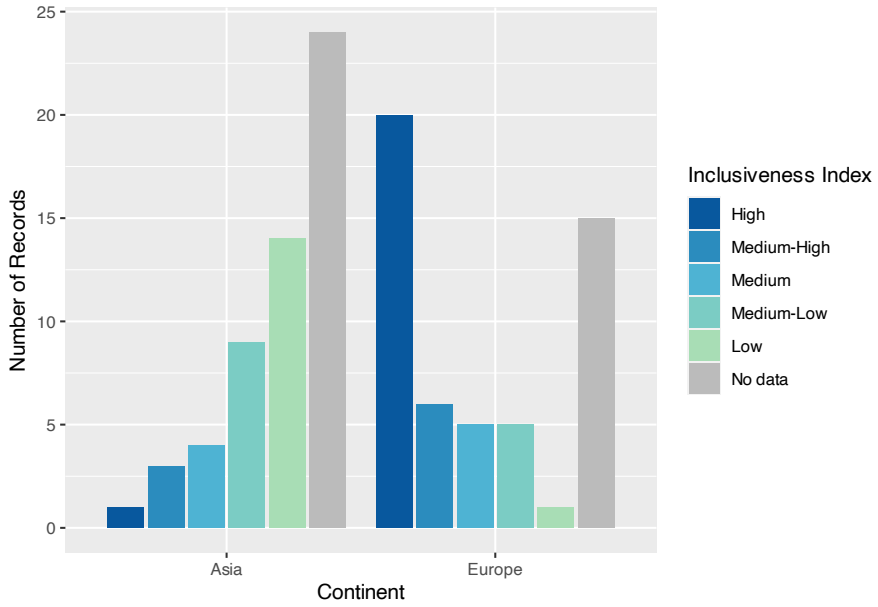


FIGURE 1.10: A grouped bar chart focusing on two continents.

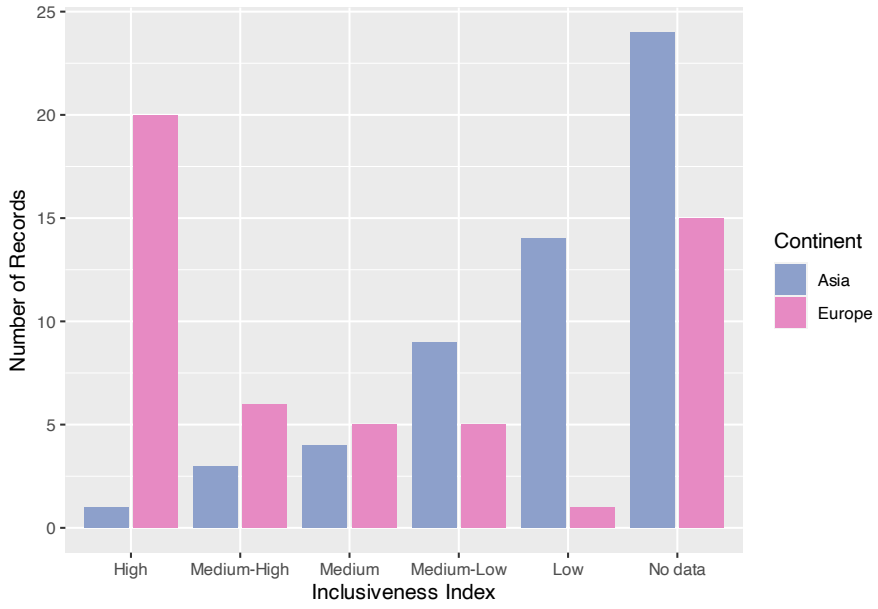


FIGURE 1.11: The same grouped bar chart with a different arrangement of the categorical variables.

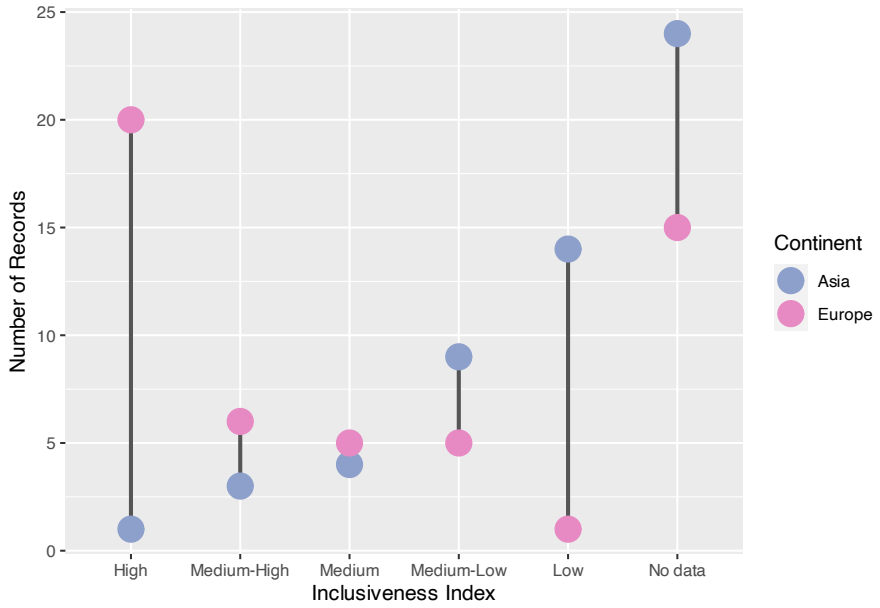


FIGURE 1.12: A dumbbell plot comparing two continents.

With a dumbbell plot, there are three main trends we can explore in the chart. We can focus on the green circles to see the pattern in the Asia data values. We can focus on the blue circles to see the pattern in the Europe data values. Finally, we can focus on the lengths of the lines connecting the circles to compare the continents at each level of the Inclusiveness Index. This chart time is an efficient way to compare these data values, but remember that it can be difficult to compare the lengths of shapes when they don't have the same baseline.

1.3 Scatter Plot

The scatter plot is another common visualization type. This type of visualization displays one circle for each record in the data set. The position of the circle is based on the values of two different numerical variables, one of which is associated with the x axis and the other of which is associated with the y axis.

This visualization is a way to show a relationship between the two numerical variables displayed on the axes. A relationship between the variables means that a change in one variable would predict a specific kind of change in the

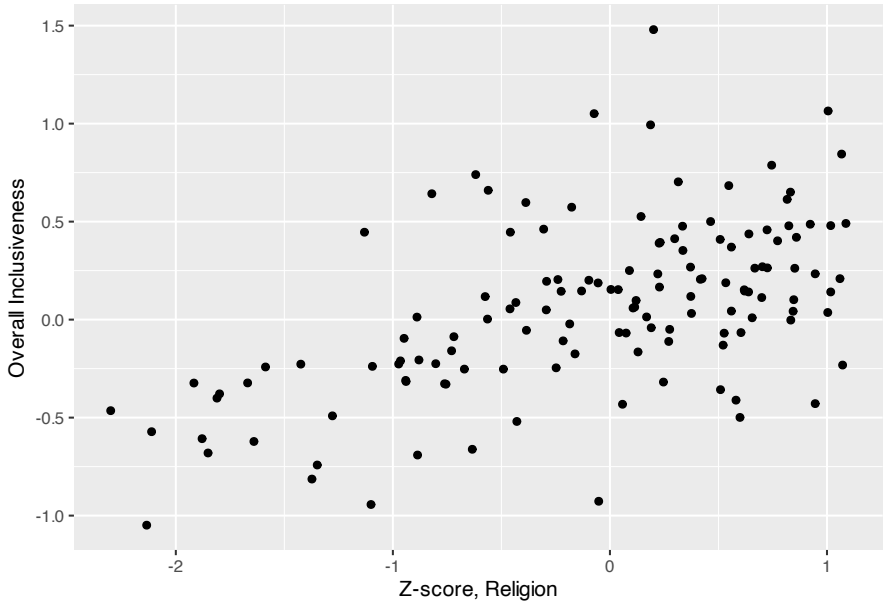
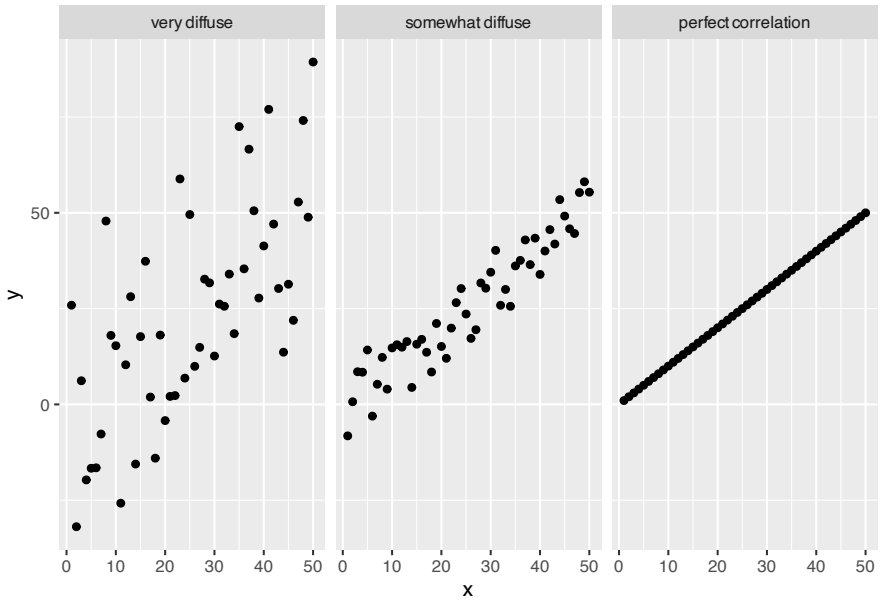


FIGURE 1.13: A sample scatter plot.

other variable. For example, one kind of relationship is a positive correlation, which means that an increase in one variable is associated with an increase in the other variable. Points with higher values on the x axis tend to have higher values on the y axis. Similarly, points with lower values on one axis tend to have lower values on the other axis.

When two numerical variables have a positive correlation, it shows up on the scatter plot as a diagonal pattern of circles, from the bottom left corner of the chart to the upper right corner of the chart. The closer it looks to a straight line, rather than a diffuse pattern, the stronger the relationship is.



There are a few types of patterns that might show up when looking at a scatter plot. Instead of a positive correlation, the variables could have a negative correlation: high values of one variable are associated with low values of the other variable. This relationship shows up on the scatter plot as a diagonal patter from the top left corner to the bottom right corner.

Both positive and negative correlations are linear relationships - they look like lines on the chart. There are also nonlinear relationships that look like different kinds of curves. An exponential relationship looks like a curve that starts mostly horizontal and then curves up dramatically, ending up almost vertical. A logarithmic relationship is a curve that starts mostly vertical and bends over dramatically, ending up almost horizontal. Other curvilinear patterns of dots might be better represented by other mathematical functions.

What does it mean to see a shape in a scatter plot? A detectable shape in a scatter plot is a suggestion that there might be a statistically powerful relationship between these variables. The chart, however, is not a substitute for a statistical analysis. Using statistical analyses to explore the relationship between two variables is called modeling.

Sometimes a scatter plot will be combined with a statistical model to explore the connection between the data points and an ideal relationship. For example, you may see a scatter plot where there is a correlation between the variables combined with a linear model (represented by a straight line drawn on top of the points).

When you see a line on a scatter plot like this, it is showing the linear model

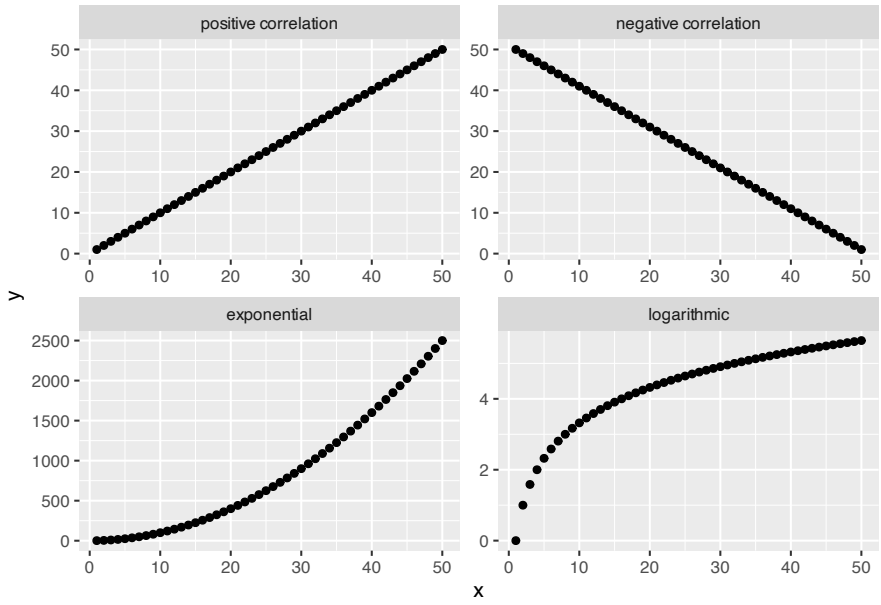


FIGURE 1.14: Different relationships between numerical variables.

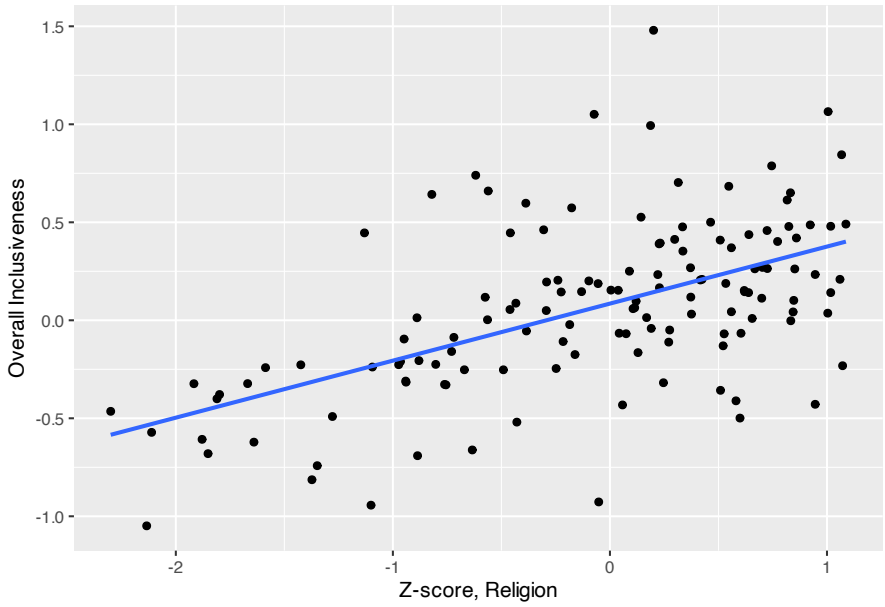


FIGURE 1.15: A sample scatter plot with a linear model overlaid over the points.

that best represents the relationship between the x and y variables. In this case, the relationship between the variables is not very strong, so the points look more like a cloud than the straight line of the linear model.

1.3.1 Variations

The following charts are variations on a scatter plot. They build on a basic scatter plot by summarizing the data and making it easier to see patterns.

1.3.1.1 Contour or density plot

Sometimes a dataset is too large for a scatter plot to be effective. With a large number of data points, there can be too much overlap between the circles to see the dominant patterns. In this instance, it can be helpful to calculate the density of data points across the chart and visualize the density instead of (or in addition to) the points. This is called a contour or density plot.

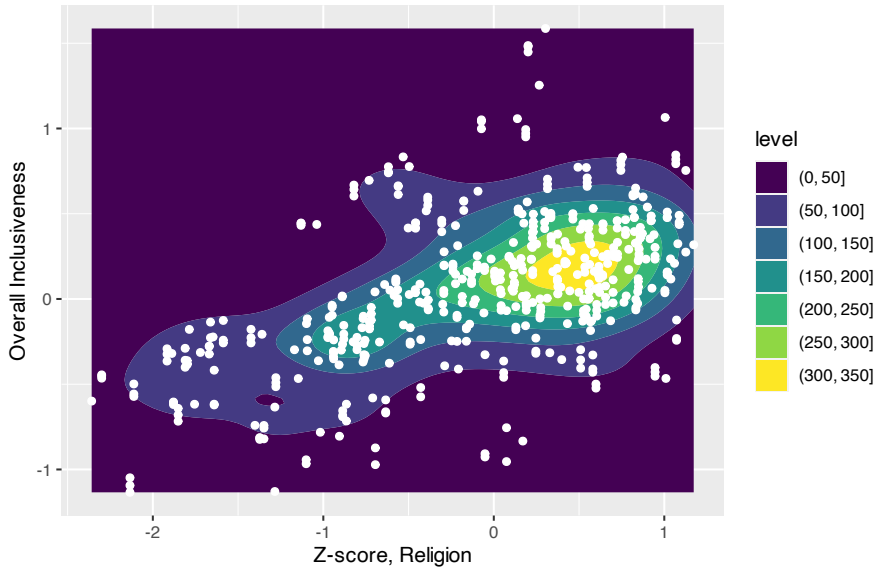


FIGURE 1.16: A sample contour plot with scatter plot points on top.

1.3.1.2 Binned scatter plot

In a contour plot, the density calculation detects regions of high density in a scatter plot. Another way of summarizing the distribution of points across the plot is to divide the plot into an even grid and then to count the points inside each region. This is often called "binning." Common types of binning

are rectangular (splitting the plot up using a rectangular grid) and hexagonal (splitting the plot up using a grid of interlocking hexagons).

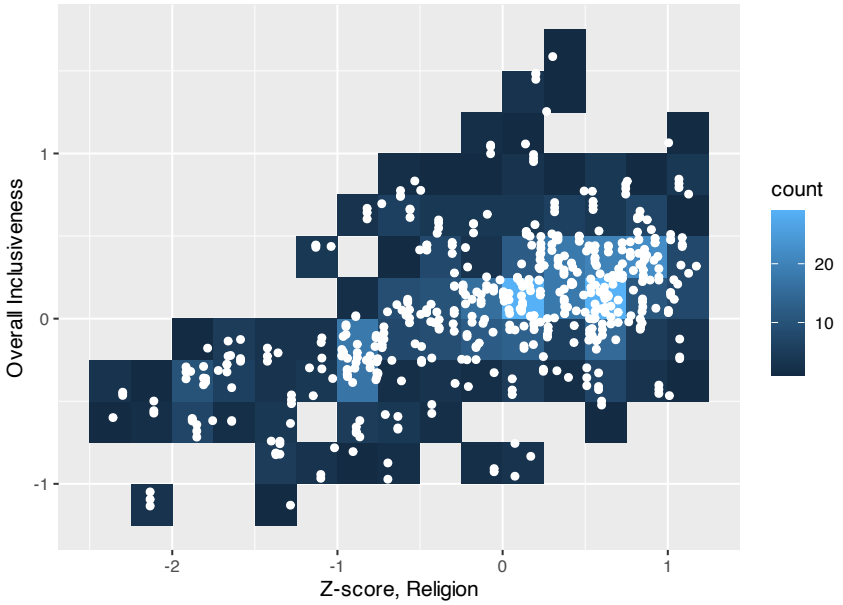


FIGURE 1.17: A sample scatter plot binned with a rectangular grid.

1.3.2 Add a variable: Scatter plot with color

So far, our scatter plots have still only been used to visualize the relationship between two numerical variables. In some datasets, it can be helpful to consider how an additional variable interacts with the scatter plot pattern. One way to incorporate an additional variable is to change the color of the points in the scatter plot according to the third variable. For example, you can associate the color of the points with a categorical variable to show whether different subsets of the points cluster in different parts of the graph.

In the above chart, the color represents the continent; that is, each continent shows as a separate color. We're looking for a relationship between the pattern of the colors and the spatial pattern of the points. In this chart, the points associated with Europe do overall seem to cluster in the upper-right corner of the graph, meaning that on the whole the European countries tend to rate highly on both the Z-score value for religious inclusiveness and the overall inclusiveness score. African countries also tend to have high scores for religious inclusiveness, but they don't rate as highly for overall inclusiveness.

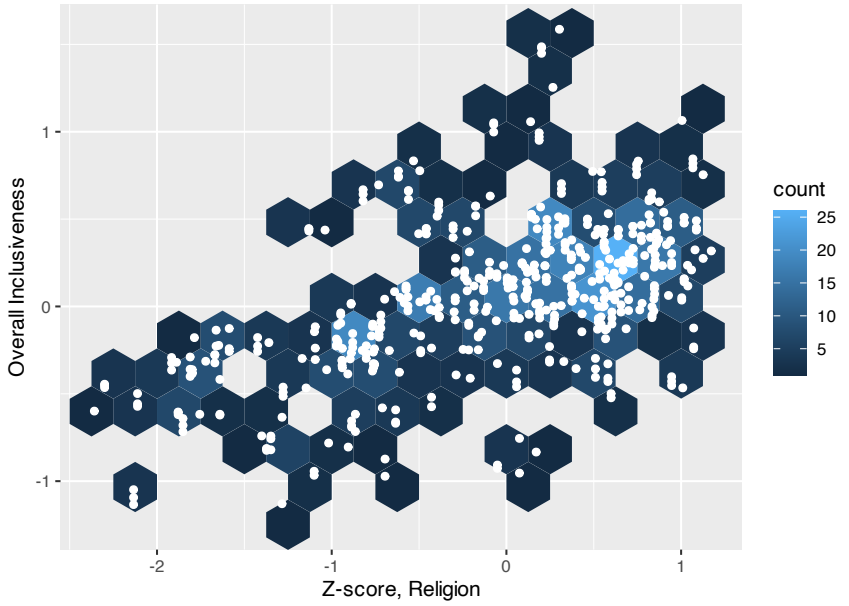


FIGURE 1.18: A sample scatter plot binned with a hexagonal grid.

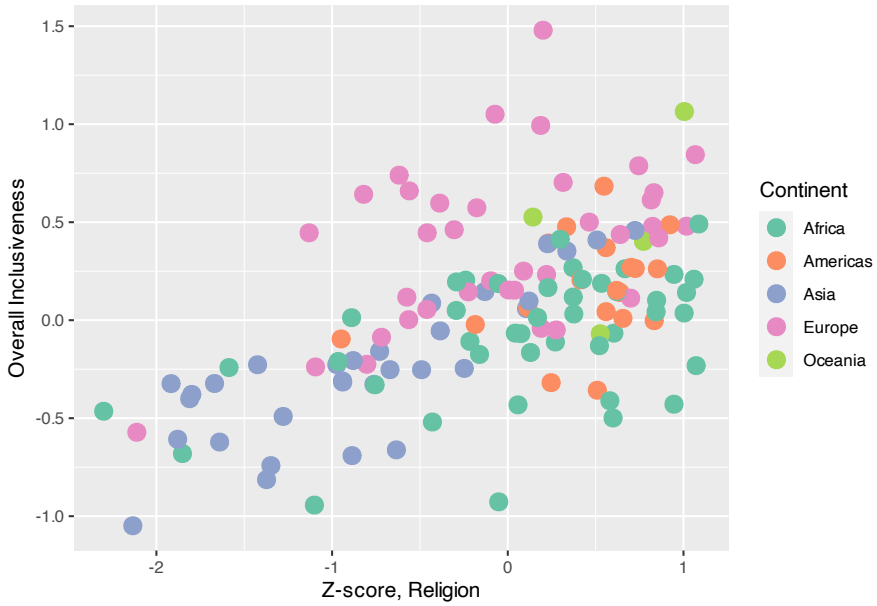


FIGURE 1.19: A sample scatter plot with color categories.

You can also use color to visualize a third numerical variable instead of a categorical variable.

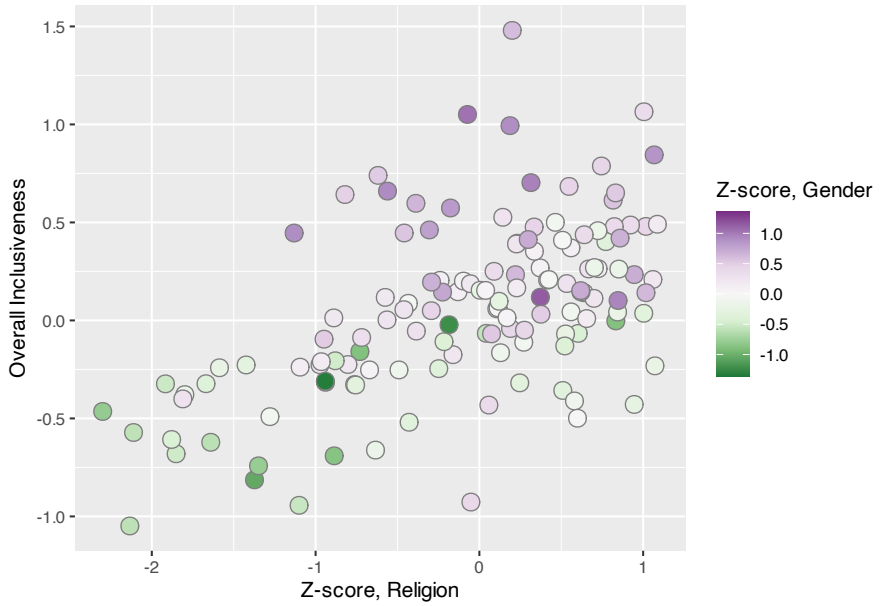


FIGURE 1.20: A sample scatter plot with a color gradient.

In this chart, the values in the "Z-score, Gender" variable are associated with a color gradient. Positive values of this new variable show in the chart as increasingly darker shades of purple. Negative values show as increasingly darker shades of green.

With any chart, adding more variables runs the risk of creating visual confusion that makes it harder (not easier) to see interesting patterns. For example, in the chart above, we see that purples mostly occur on the top half of the chart and greens on the lower half of the chart. Beyond that, though, it's hard to identify a strong relationship between the strength of the color values and either of the axes. If colors are not concentrating in a particular region of the plot, adding a third variable may not be necessary for this chart.

1.3.3 Add a variable: Bubble chart

A bubble chart is another way of adding a third variable to a scatter plot. Unlike adding color to the chart, however, a bubble plot works best when you are adding a third numerical variable. That's because in a bubble plot, the additional variable is represented by changing the size of the bubble. Representing a categorical variable by changing the size of the circle isn't as natural

as using different colors. It's hard to focus on all of the bubbles of a particular size to try to identify clusters.

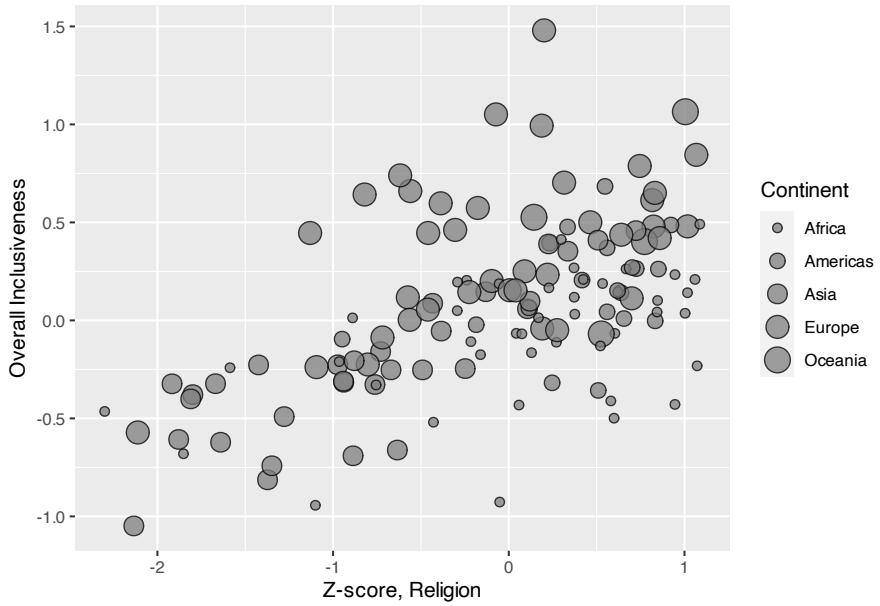


FIGURE 1.21: A sample bubble chart with categories.

If you use a numerical variable to size the bubbles, the goal of the chart becomes similar to when color is used to add a third numerical variable: explore whether the sizes of the bubbles changes in a meaningful way in relation to the axes.

There is one property of bubble charts that is different from scatter plots with added color. When you have a variable that has both positive and negative values, it may be a slightly less natural fit for the size of the bubbles. The size of an object like a circle is naturally a positive value. A circle doesn't itself have negative size. If the variable uses negative and positive in an abstract sense, though, the size of a bubble can still help differentiate low and high values.

With bubble charts, color is still available to display another variable if there is anything else that might interact with the three numerical variables.

Be cautious, again, with adding too many variables to a single chart. If adding a variable doesn't reveal anything new about the data, it is probably getting in the way of a pattern that does exist.

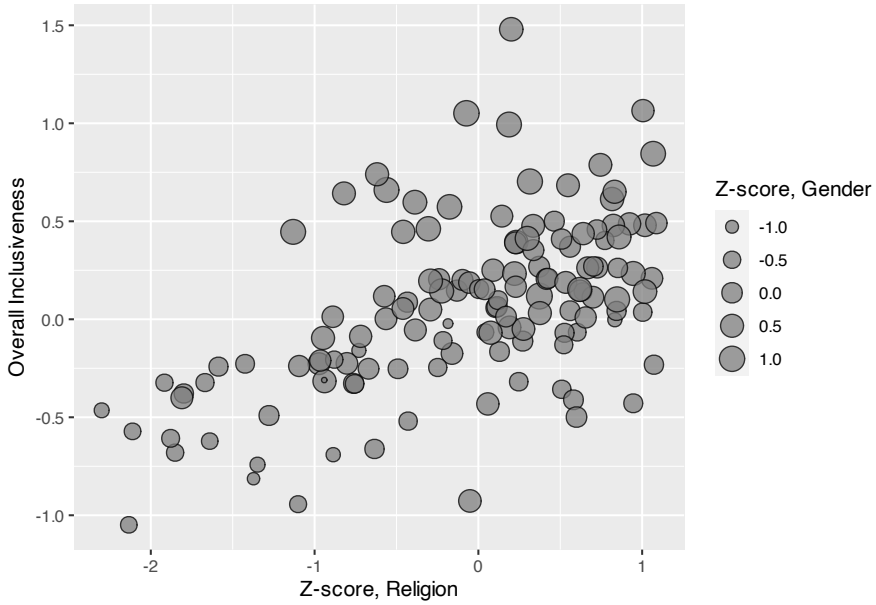


FIGURE 1.22: A sample bubble chart with a size gradient.

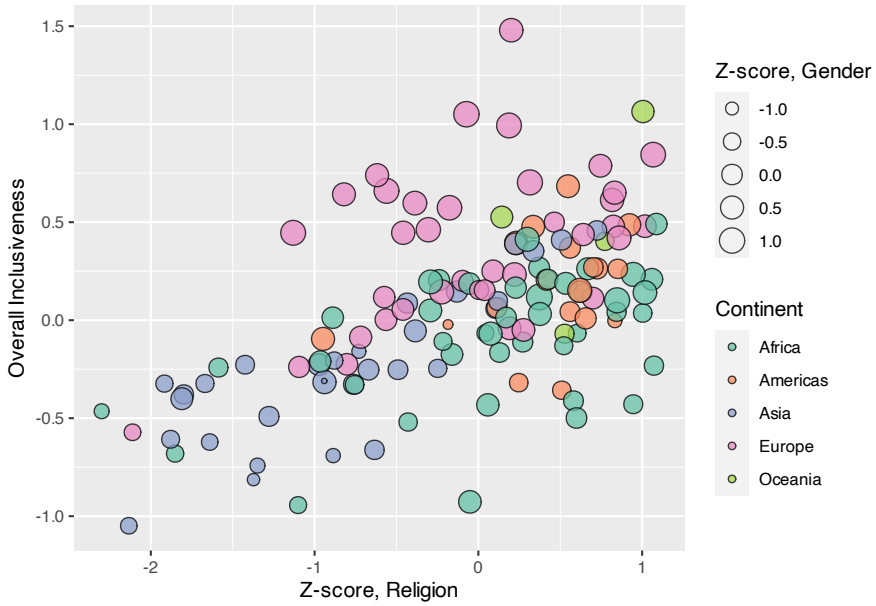
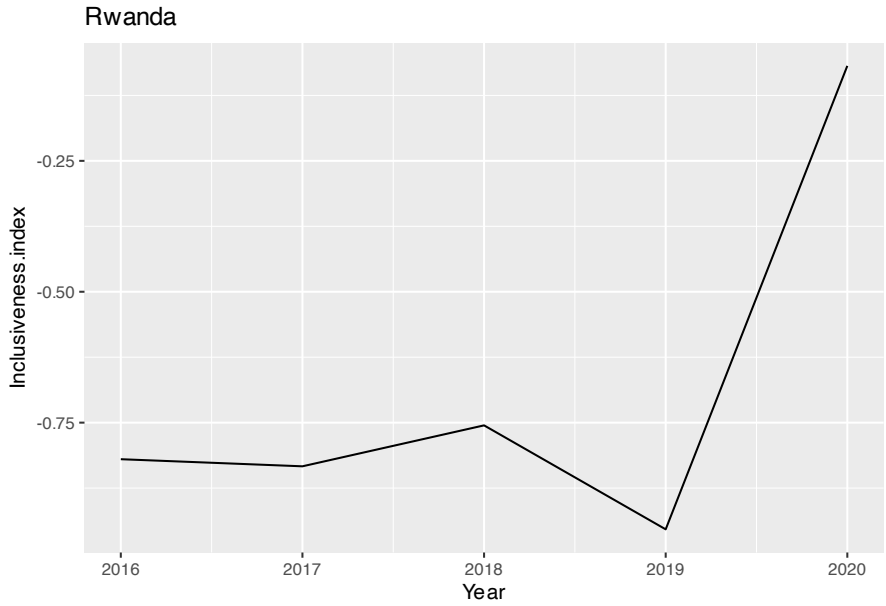


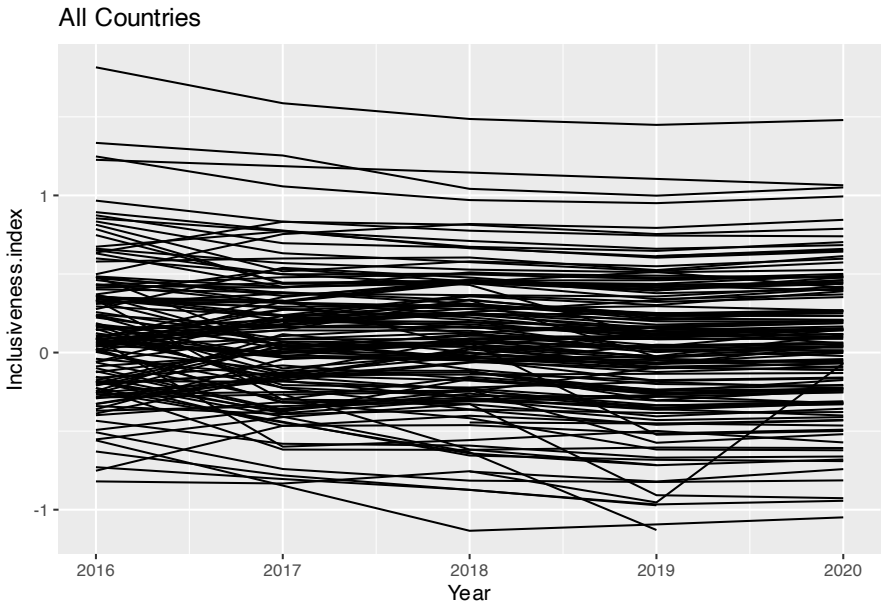
FIGURE 1.23: A sample bubble chart with color categories.

1.4 Line Chart

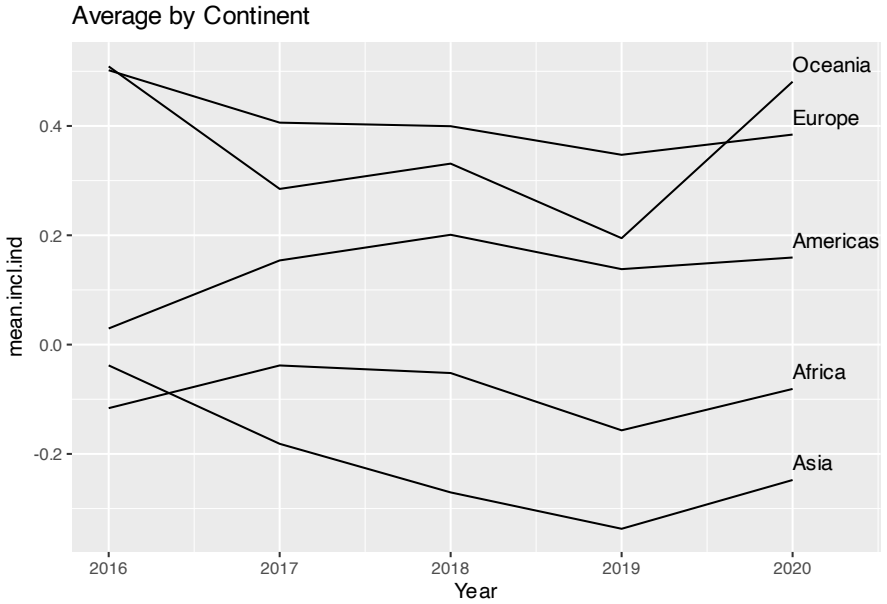
A line chart is typically used to explore the change in a numerical variable over time. The variable is measured over a series of time points (for example, years, days, seconds). The time variable is associated with the x axis of the chart, and the numerical variable is associated with the y axis. Instead of placing a circle for each record in the data set, like you would see in a scatter plot, the chart uses a line that travels through all of the correct data points. Connecting related points with a line gives the data a sense of continuity and motion. Line charts are often used to look for dramatic increases or decreases in the numerical value at particular points in time.



Line charts can show change over time for one entity or for many entities. In the example above, there is a single line representing a country. You can also use a line chart to show multiple lines at once. For example, if each line is a country, you can display many countries on the same chart to allow for comparisons between countries as well as over time.



As with the other visualizations we've looked at, it is easy to create a line chart with so much data that it is hard to see patterns. The goal of the line chart is to show trends over time, but if all of the lines look flat, or if there are so many lines that you cannot detect the variations that are there, it may be useful to include only a small number of the lines at once, or to combine groups of the lines together.

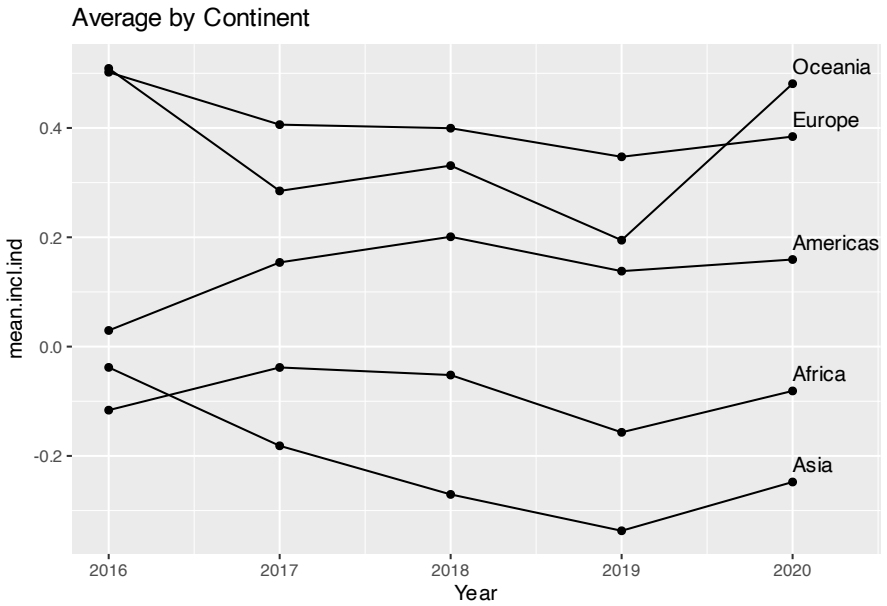


In this version of the chart, the countries from each continent have been combined into a single line using the mean of the countries' values for each year. Another way to reduce the number of lines in the line chart would have been to limit the chart to the countries from a single continent.

Also notice that when you limit the number of lines you use in a line chart, it becomes much easier to identify each line with a label. Not every visualization includes labels for every data point. For example, in a scatter plot, you may not label the points if the important information is the overall trend. You may have the same situation in a line chart if all of the lines are moving in a similar direction, or they all experience a dramatic change at the same point. If it is important to identify specific lines in your chart, however, it is best to use only a few lines so that it is possible to identify each line directly and trace it individually across the entire chart.

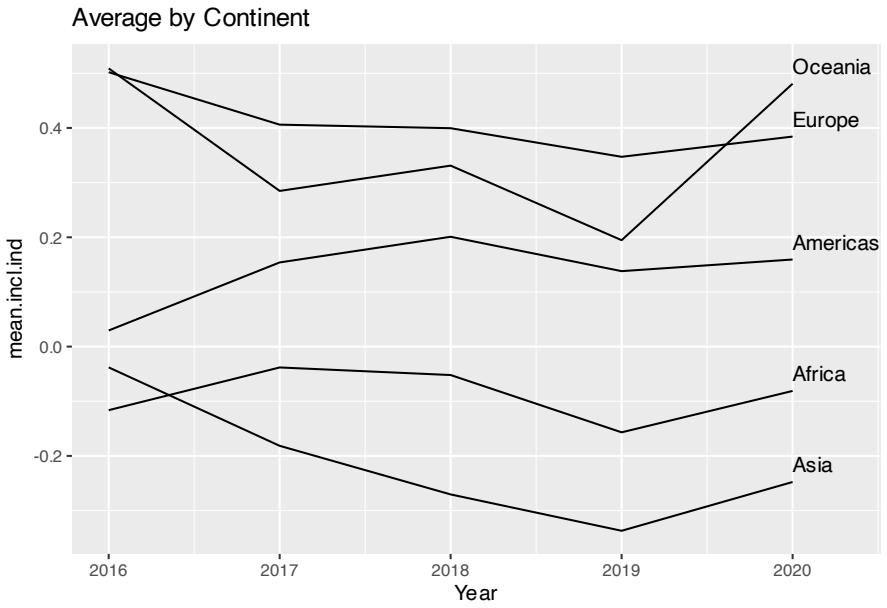
1.4.0.1 Using circles to highlight the position of the data points

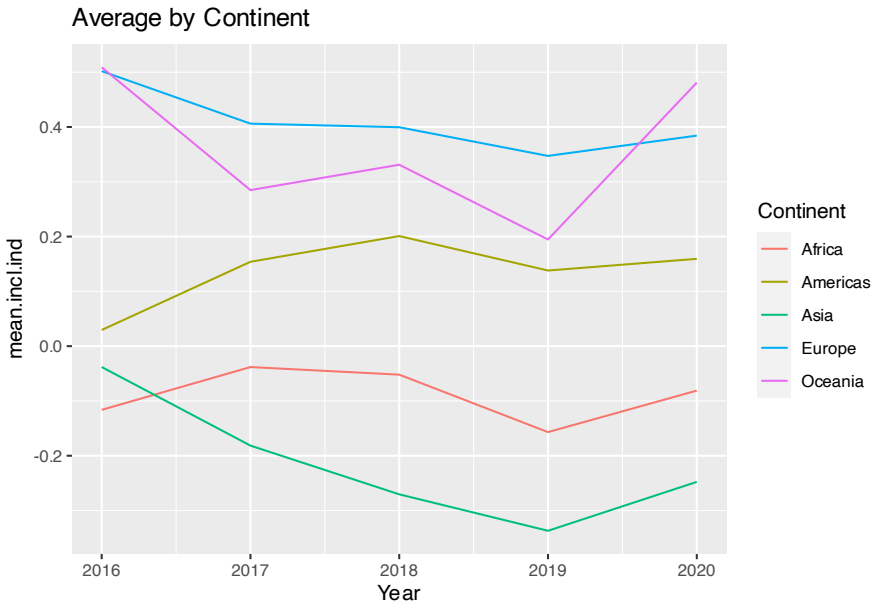
Some line charts include circles along the lines. These circles indicate that the data set included a value at that point in the line. This can be especially useful if lines travel through data at irregular time intervals. For example, if the data is collected every year except that there was a gap of three years in the middle, the circles would clearly indicate which years are included in the dataset and which are not. Normally the lines on a line chart will connect straight from one point in time to the next point in time, and unless the line turns up or down, it can be hard to know exactly where the real data values are.

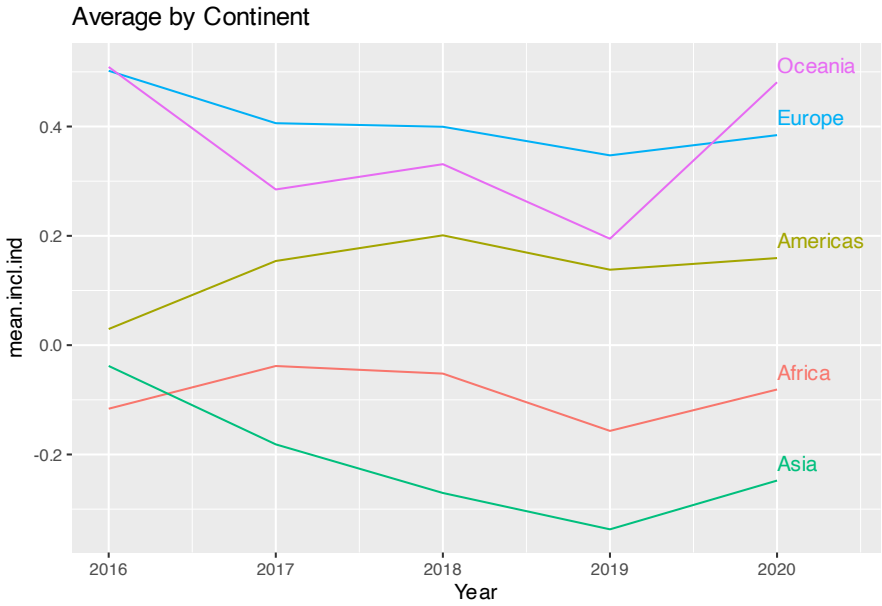


1.4.0.2 Using colors to distinguish individual lines

Sometimes line charts use color to differentiate the lines from each other. Each line gets its own color, and the color legend clarifies the name of the line. Even for charts where it is important to individually identify the lines, it can quickly become overwhelming to see a separate color for each line. Even a small number of lines can require a set of colors that include colors that are difficult to tell apart from each other. In additional, it can be confusing and time consuming to have to consult a legend to identify an individual line. While color may be a nice complement to a simple line chart, it is easier to read a line chart when the lines are labeled directly instead of relying on a legend.


Average by Continent
Oceania
Europe
Americas
Africa
Asia
0.4
0.2
0.0
-0.2
mean.incl.ind
2016
2017
2018
2019
2020
Year



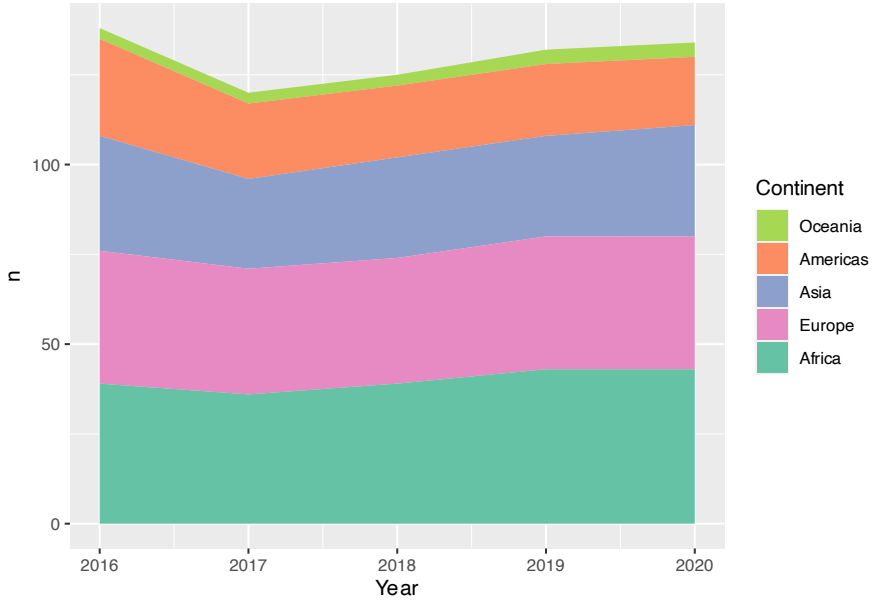


1.4.1 Variations

There is one primary variation of the line chart: the area chart.

1.4.1.1 Area chart

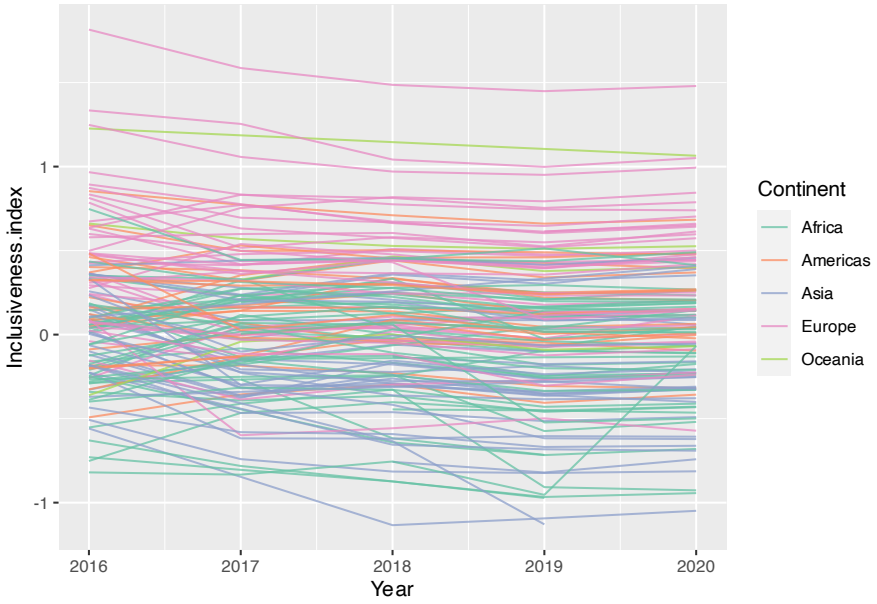
An area chart looks like a line chart where everything under the line, down to the x axis, has been colored in. If you took a typical line chart and color everything in under each line, though, you would have overlapping colors for much of the chart. It is more common to see a stacked area chart, which follows the same principles as a stacked bar chart. One line is chosen as the bottom line, and the values for each additional line are stacked on top of that line. The top most line shows the overall total of the data in the chart. For this reason, area charts should be used to show data points that can be added together, like counts.



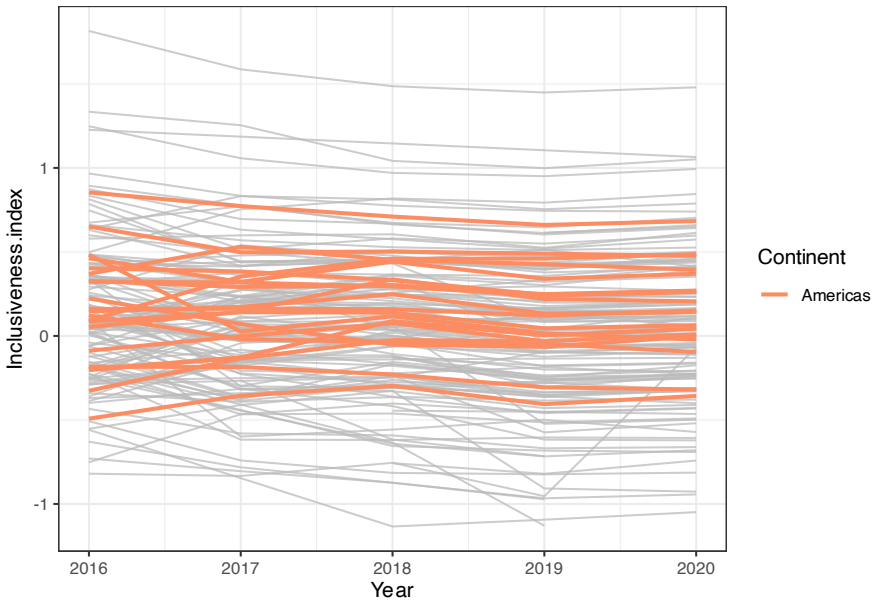
Stacked area charts raise concerns that are similar to the ones raised by stacked bar charts. The values for the lowest line and the overall total are easy to read, but the inner areas do not have a consistent baseline. It can be very difficult to see if an area has its own decrease at a particular year or if it is just reacting to a decrease in one of the lines below it.

1.4.2 Add a variable: Line chart with color

A common way to add a third variable to a line chart is to use color to represent a categorical variable, grouping the lines into different categories. This third variable could be unrelated to the variables already in the line chart, or it could be generated from the patterns in the chart – for example, creating a categorical variable that indicates whether the values are increasing or decreasing over time.

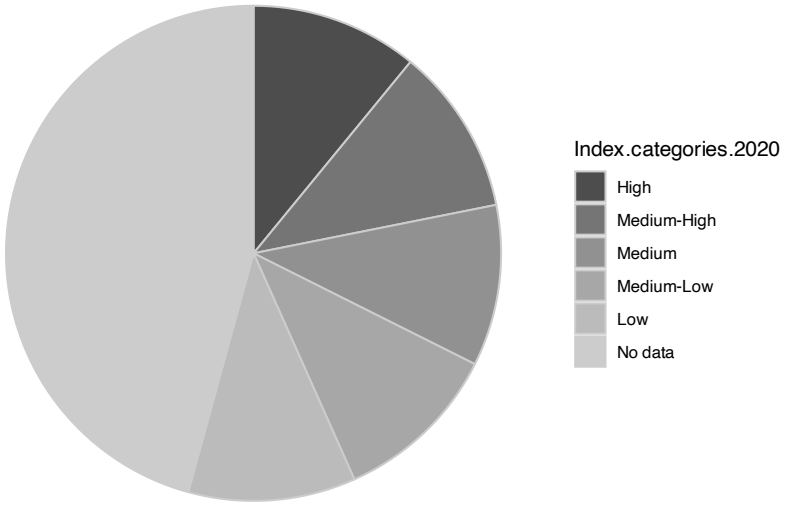


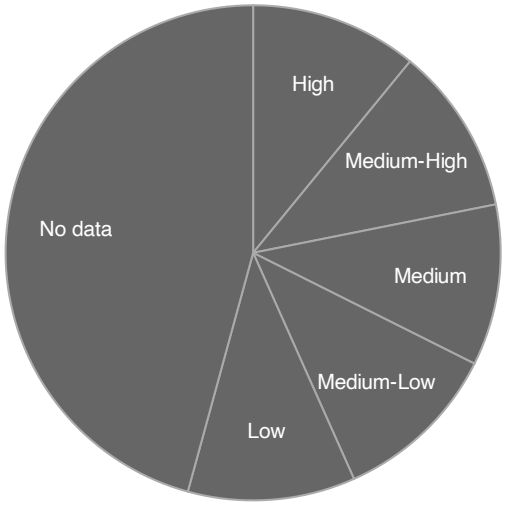
In the above chart, the individual country lines are colored by their continent. While this does reveal some general patterns for the continents, it also adds complexity to an already busy chart. Instead of using a different color for all possible continents, it may be easier to read a chart that focuses on a single continent and uses color to highlight just those countries.

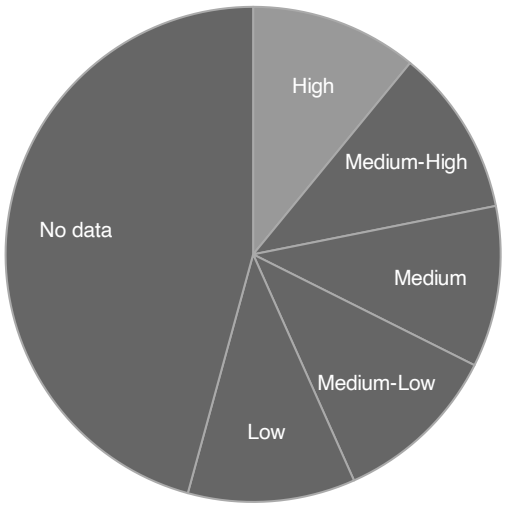


1.5 Pie Chart

- same data as a bar chart - categorical variable and numerical variable
- unlike a bar chart, the form of the pie chart means that only certain kinds of data make sense - categories that combine together to form a whole, numbers that are positive and make sense as a percentage of a whole
- hard to compare the size of wedges that are about the same size, especially when they're rotated in space
- typically the wedges "start" at the top, and then usually the order goes around clockwise or counter-clockwise
- order matters (like bar chart)
- typical way of telling pie wedges apart is by color, but you can leave them all the same color and label them directly, like a line chart
- what are you looking for in a pie chart? You are looking for how the data are distributed between the wedges - are they mostly equal? Is one much larger than you would expect? Much smaller? How much of the total goes to each category?
- these comparisons become more complicated when you have a lot of wedges. If it's important to understand small changes between the wedges, that can be difficult in a pie chart.





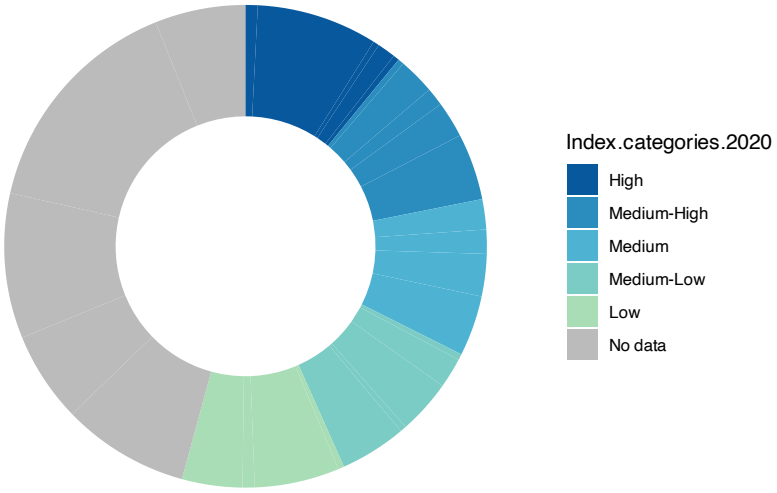


1.5.1 Variations

- primary variation is donut chart

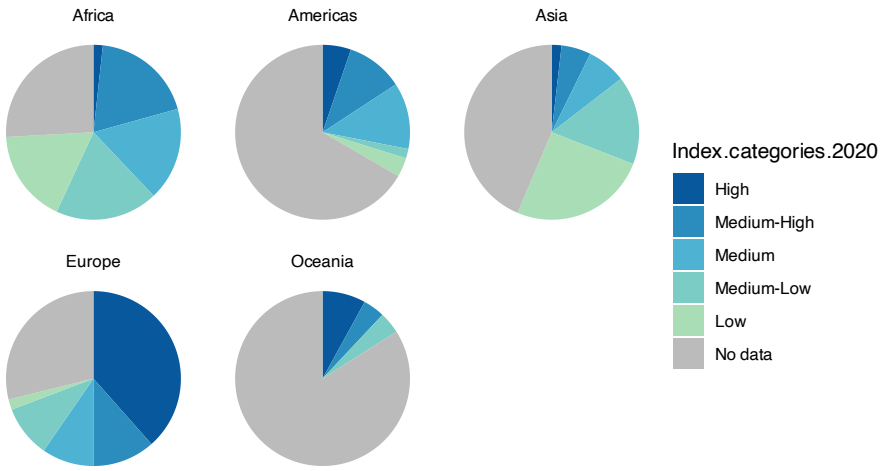
1.5.1.1 Donut chart

- just like pie chart, but the center is missing



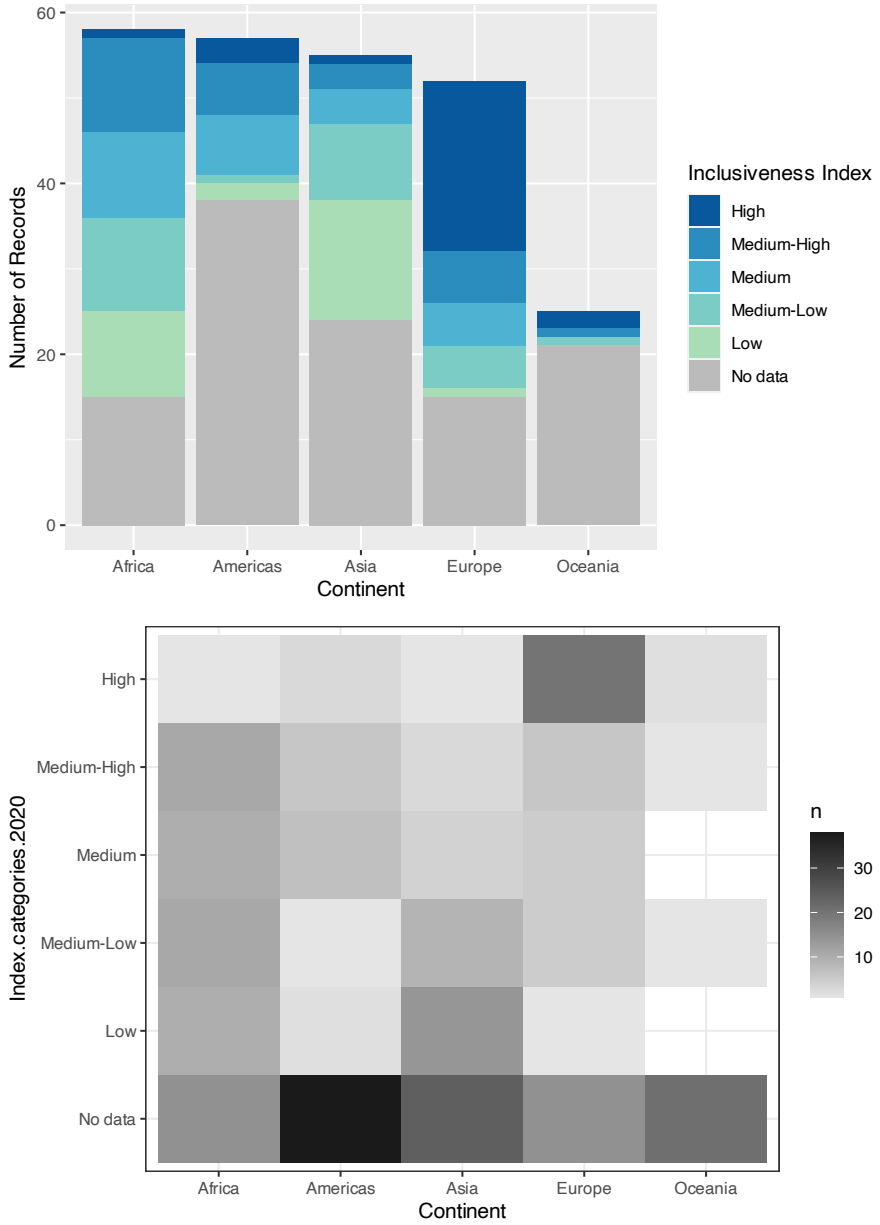
1.5.2 Add a variable: Small multiples

- It is very difficult - and likely inadvisable - to try to visualize an additional variable inside a basic pie chart.
- Instead, if you have a third categorical variable that relates to the data in the pie chart, you could try creating a separate pie chart for each category in that variable. we call that technique using "small multiples." Small multiples are actually useful for a variety of visualizations, especially if the visualization is already complex and you don't want to add to the complexity by including an additional variable.

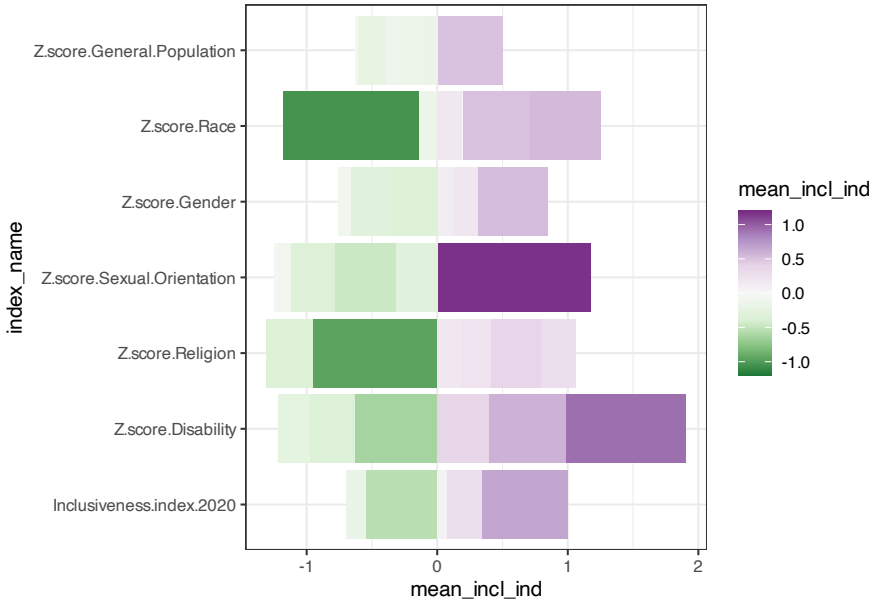


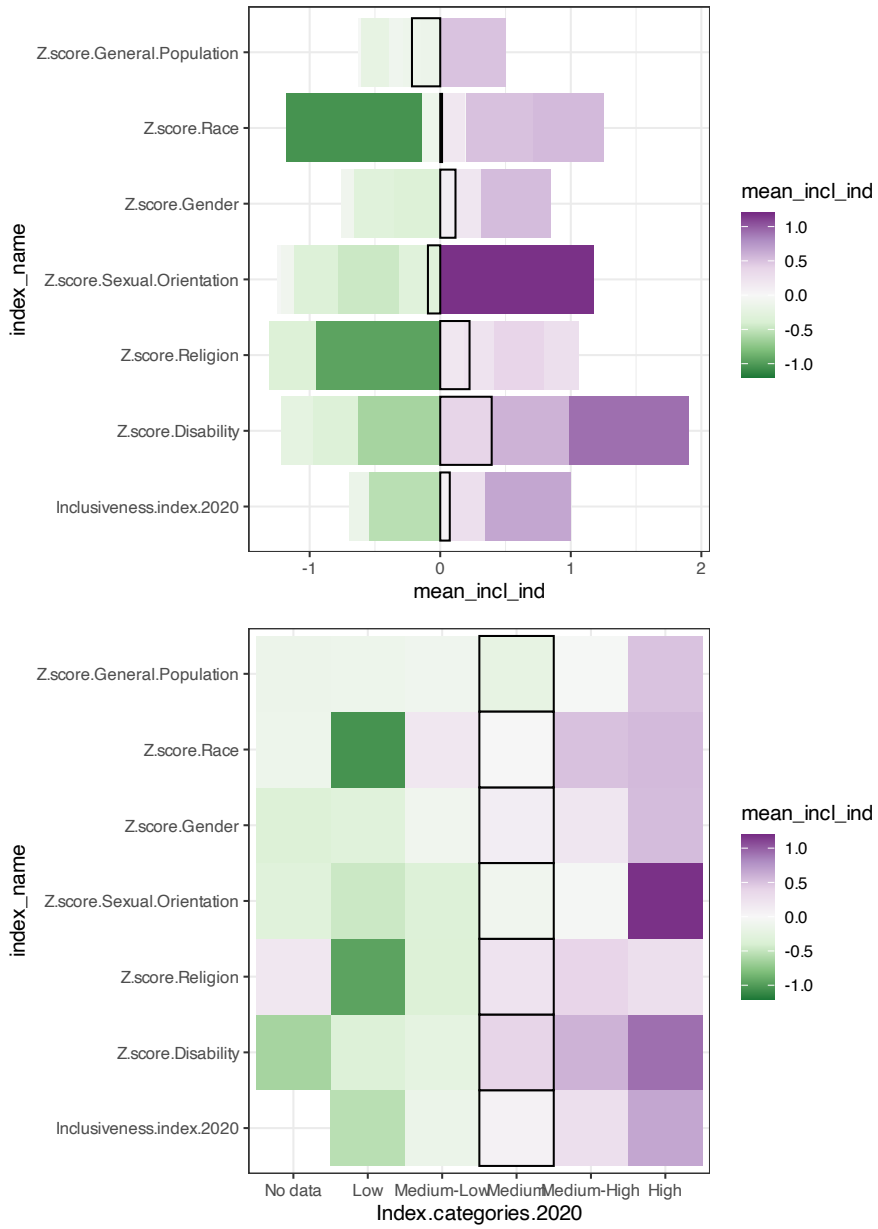
1.6 Heat Map

- two categorical variables and a numerical variable
- sort of like a stacked bar chart, but instead of showing the numerical variable as the size of the bar segment, you show it as a change in color.
- for example, longer segments -> darker colors in the grid
- bonus - no moving baseline problem
- problem - still hard to precisely compare values because just trying to compare color shades



* additional bonus, though - possibly better for negative values, since it can be hard to represent negative values in a stacked bar where totals matter, or where it doesn't make sense to be adding the data together at all * order only stays consistent if you are using all positive numbers; if there are negative values, those get pulled out and appear under the axis. * introduce diverging color

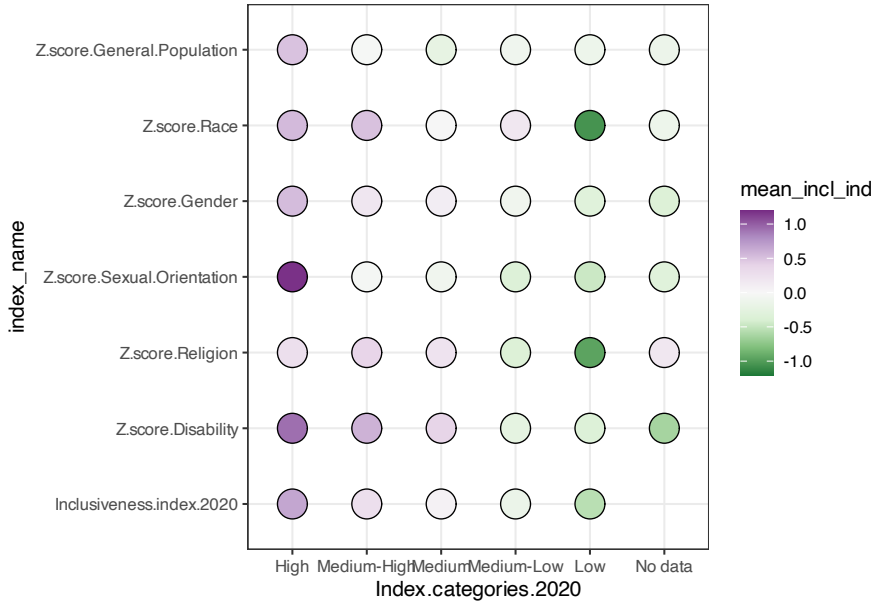

Z.score.General.Population
Z.score.Race
Z.score.Gender
Z.score.Sexual.Orientation
Z.score.Religion
Z.score.Disability
Inclusiveness.index.2020
index_name
-1
0
1
2
mean_incl_ind
mean_incl_ind
1.0
0.5
0.0
-0.5
-1.0



1.6.1 Variations

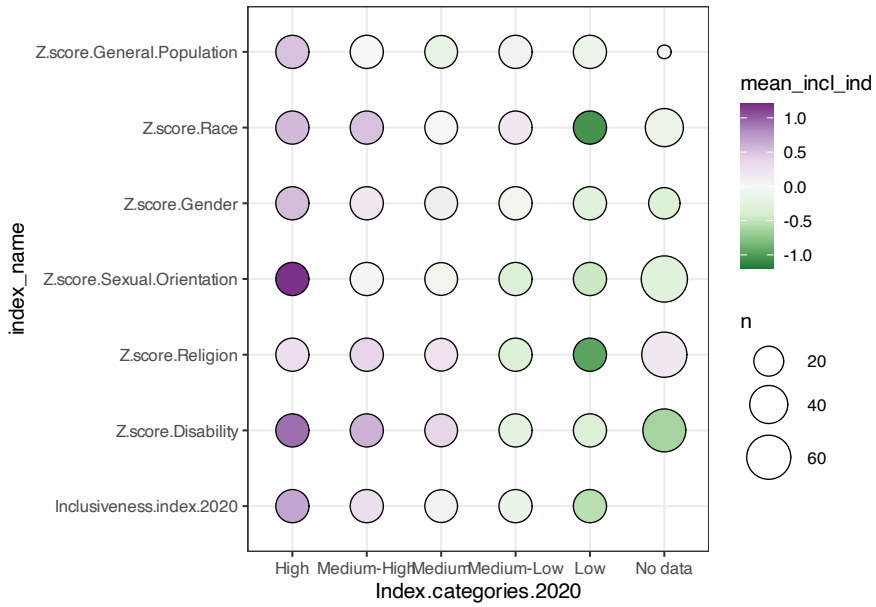
- main variation is to keep the matrix format but maybe change the shape

1.6.1.1 Circle matrix



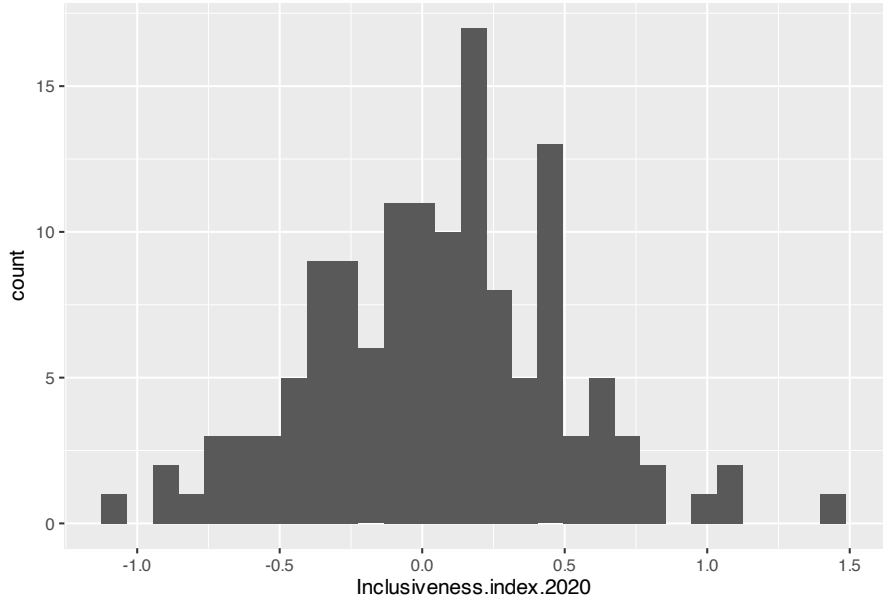
1.6.2 Add a variable: Bubble matrix

- by using a circle, you can also modulate the size



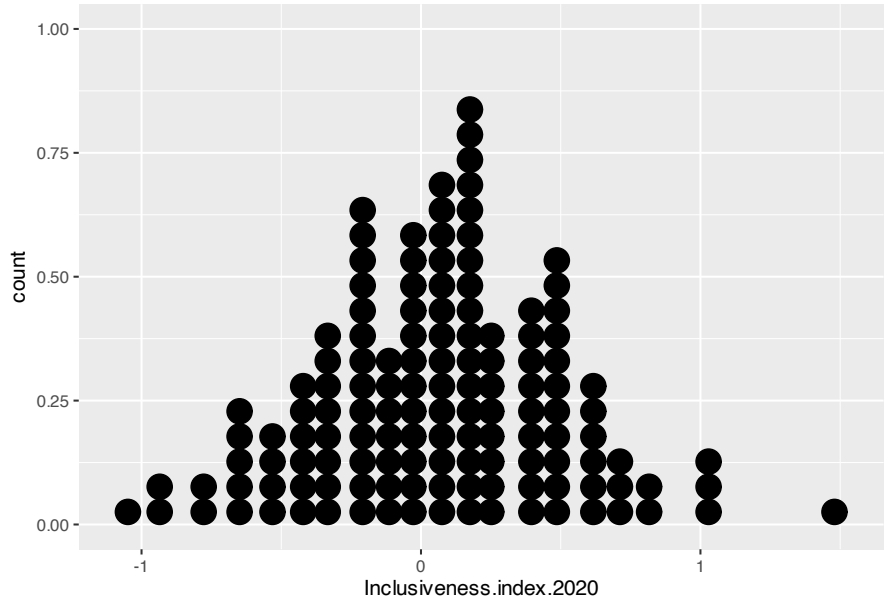
1.7 Histogram

- Largely one-variable, one-axis plots
- goal is to look at the distribution of a numerical variable
- tend to look for a few common patterns - bell curve (with or without skew), pareto plot, bimodal distribution, etc.
- shape will depend on binning, whether the numerical variable is an integer or floating point, etc.

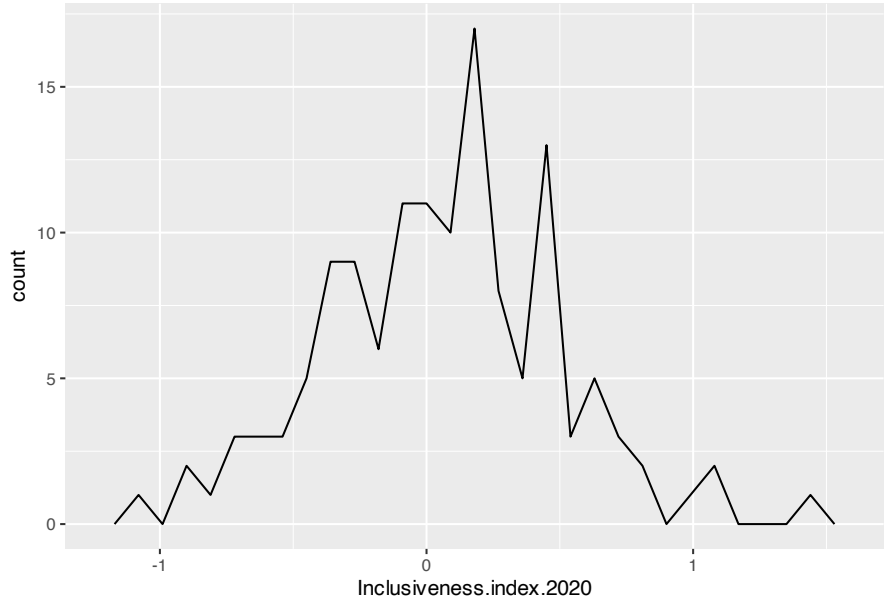


1.7.1 Variations

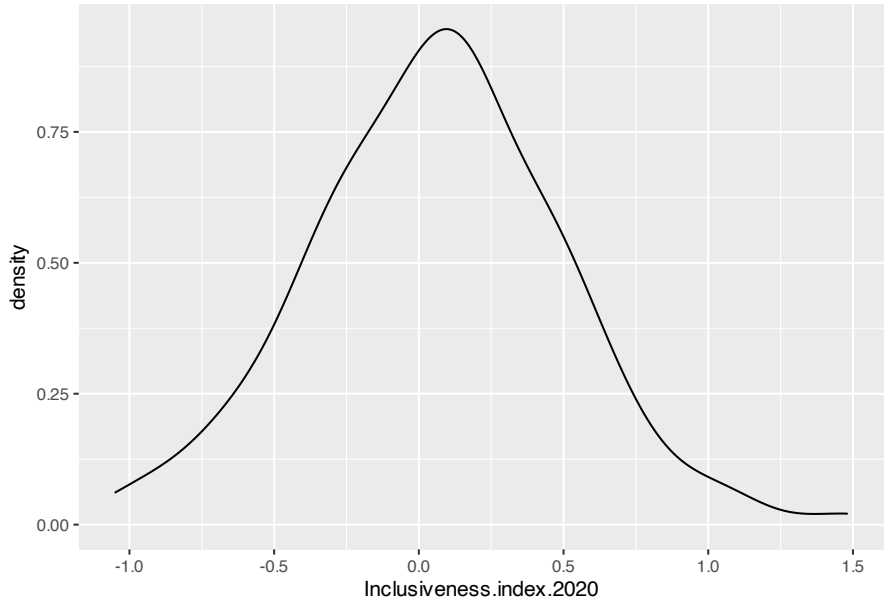
1.7.1.1 Dot plot



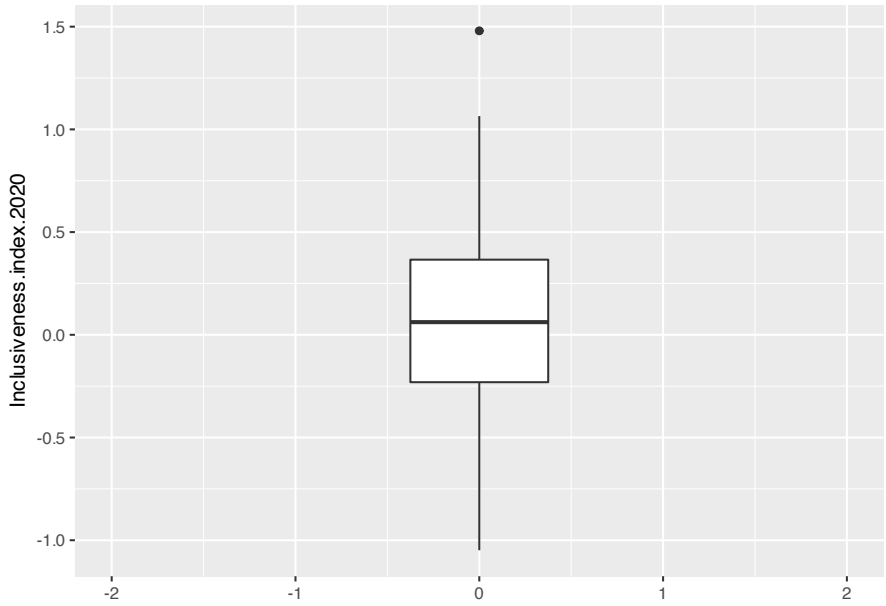
1.7.1.2 Frequency Polygon



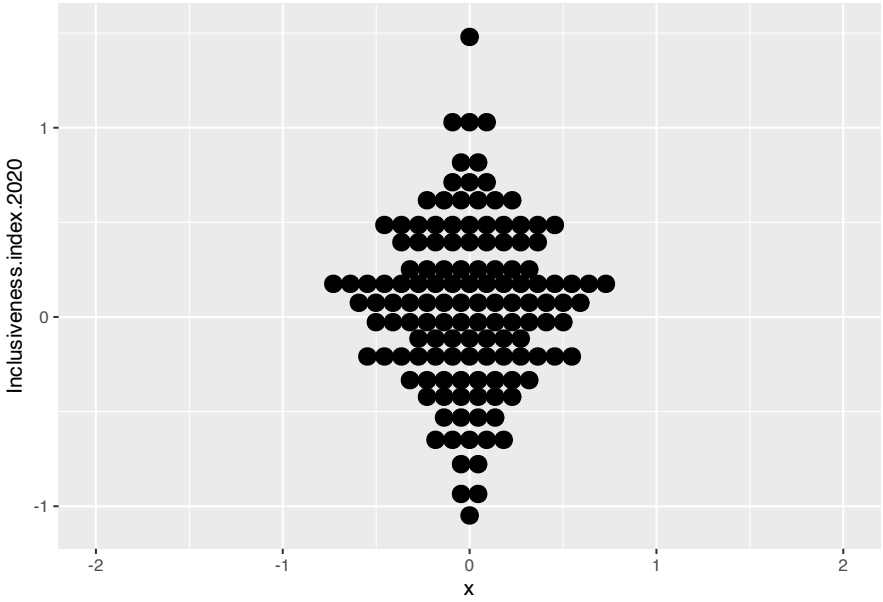
1.7.1.3 Density



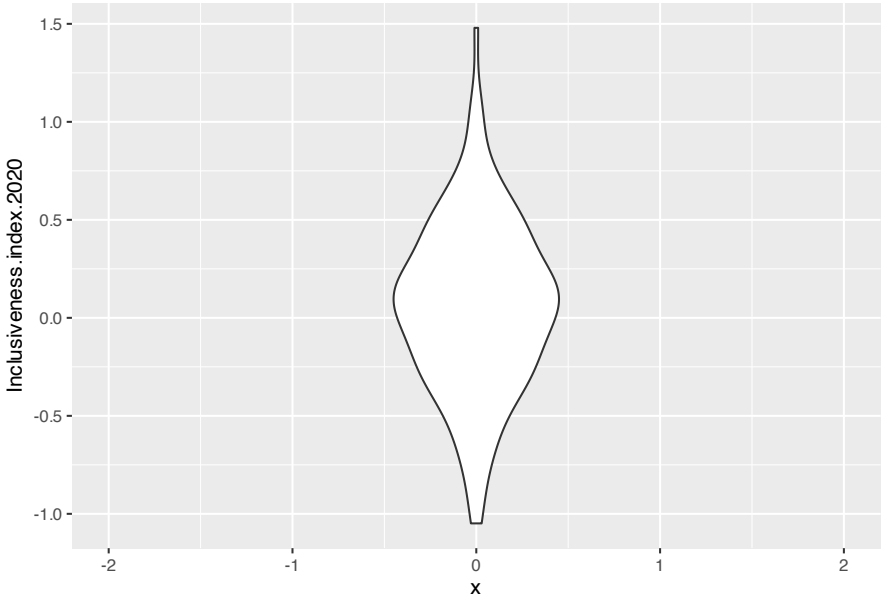
1.7.1.4 Box Plot



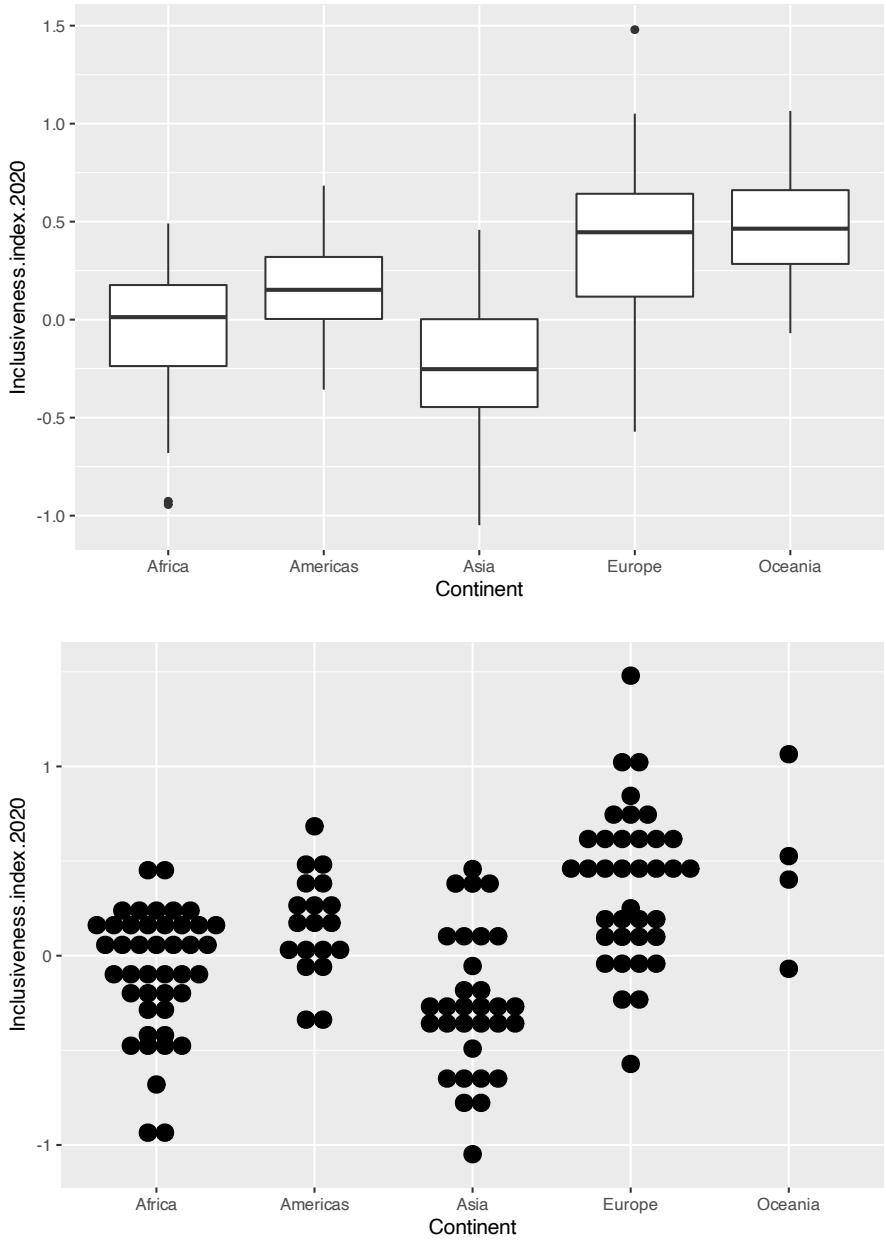
1.7.1.5 Centered dot plot / Bee swarm

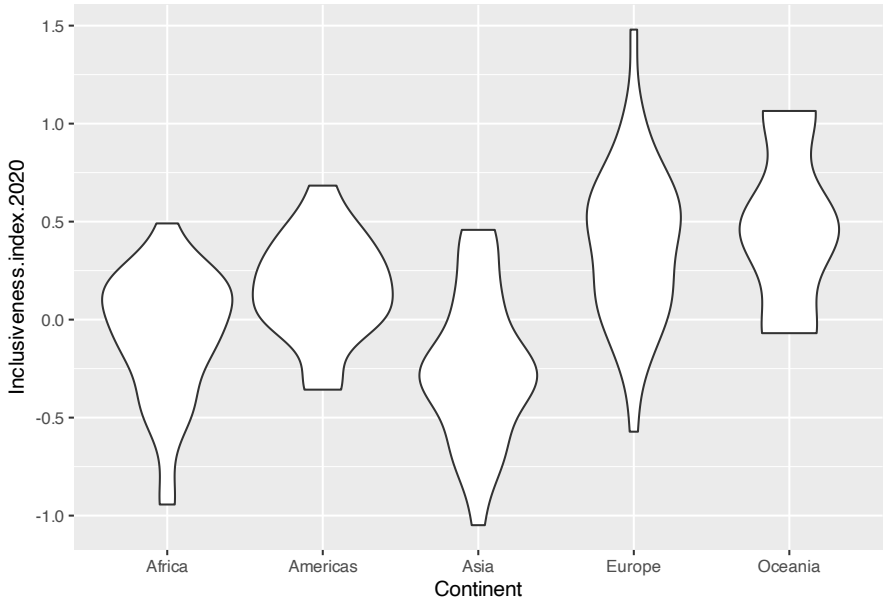


1.7.1.6 Violin plot

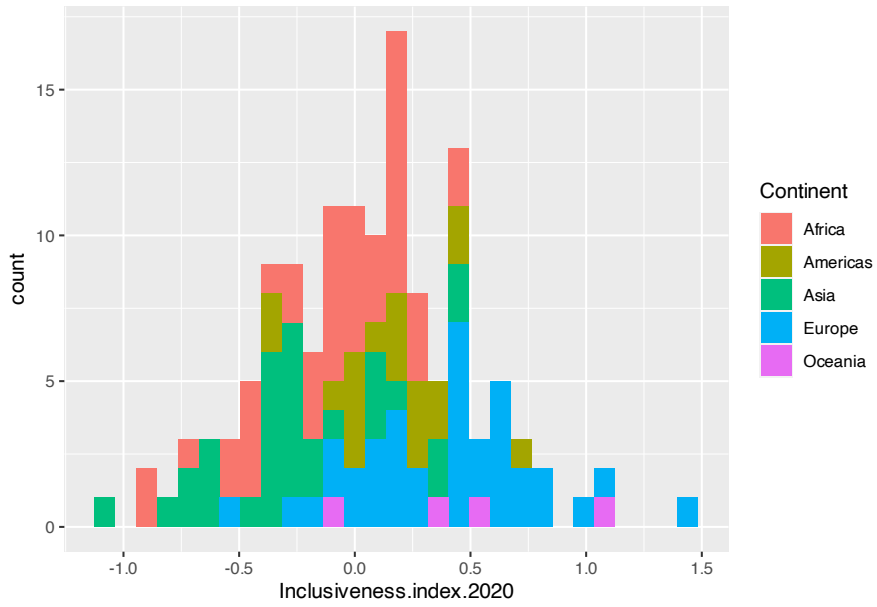


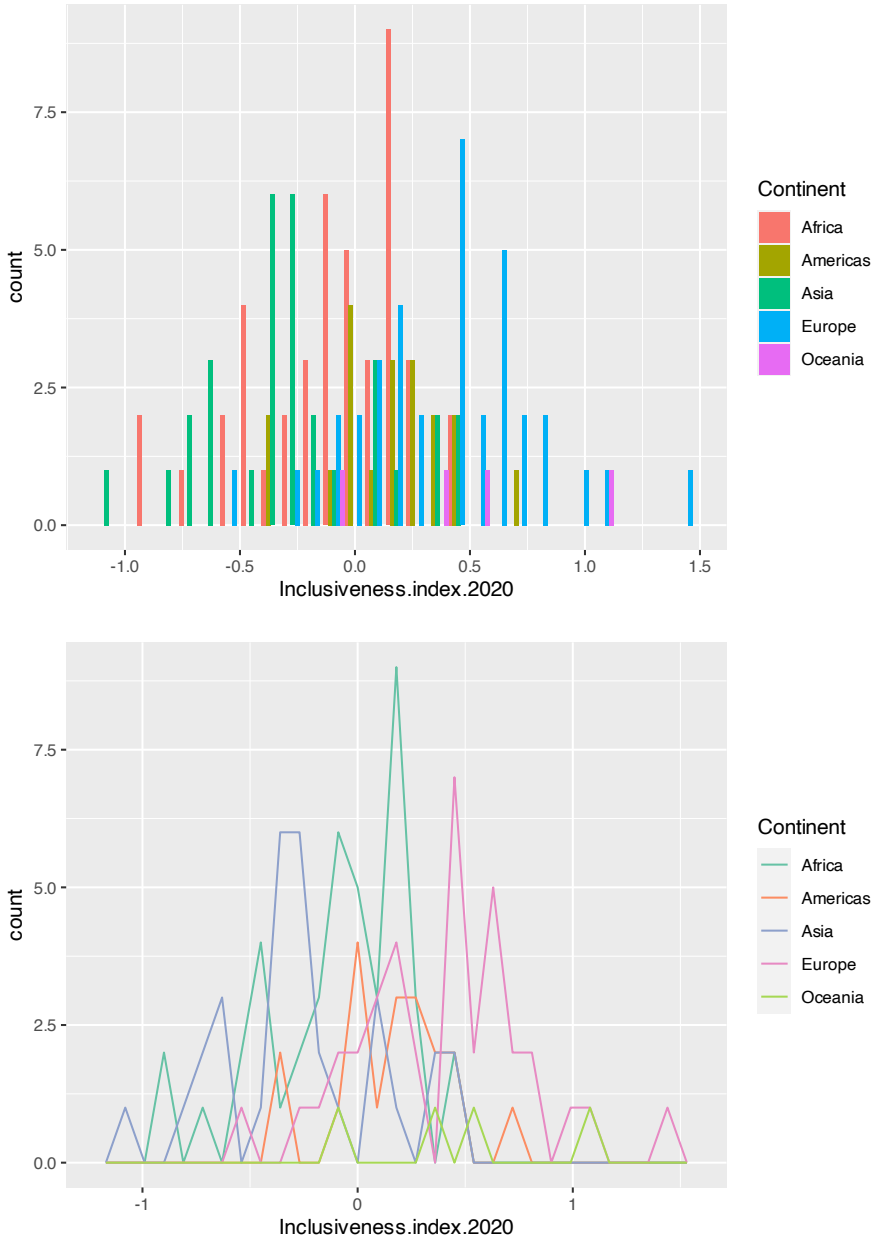
1.7.2 Add a variable: Separate groups on x axis

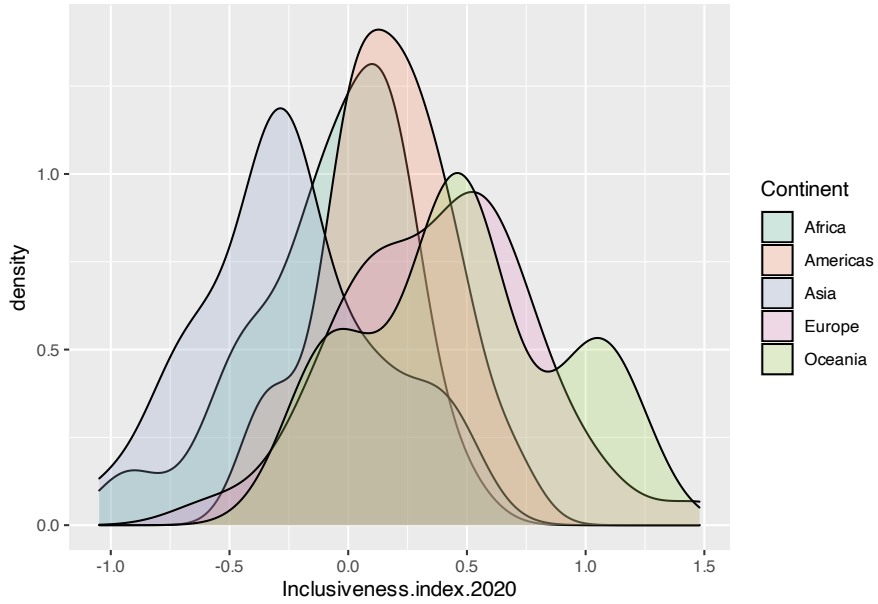




1.7.3 Add a variable: Separate groups with color

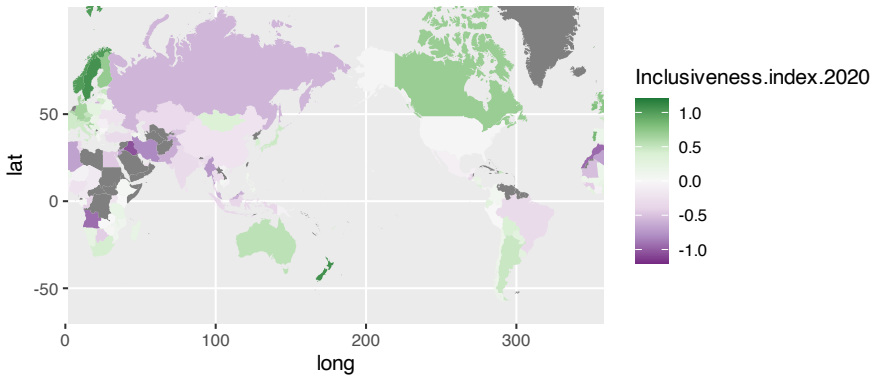



count
7.5
5.0
2.5
0.0
-1.0
-0.5
0.0
0.5
1.0
1.5
Inclusiveness.index.2020
Continent
Africa
Americas
Asia
Europe
Oceania
count
7.5
5.0
2.5
0.0
-1
0
1
Inclusiveness.index.2020
Continent
Africa
Americas
Asia
Europe
Oceania

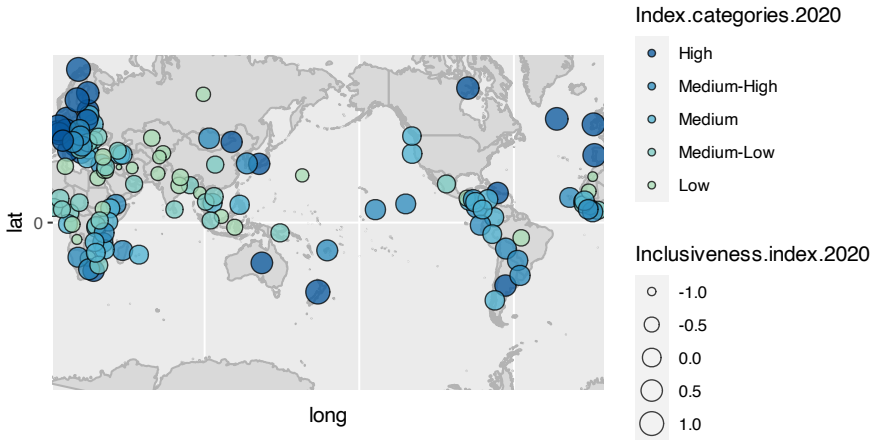

density
1.0
0.5
0.0
-1.0
-0.5
0.0
0.5
1.0
1.5
Inclusiveness.index.2020
Continent
Africa
Americas
Asia
Europe
Oceania

1.8 Maps

1.8.1 Choropleth



1.8.2 Proportional Symbol Map



2

Building basic visualizations with ggplot2

In Chapter 1, we covered a variety of visualization types and how to read them. All of those visualizations were created using an R package called **ggplot2**. This package is a very common way of creating visualizations in R, partly because it is so versatile. Once you understand the basics of how to create **ggplot2** visualizations, you can use the same techniques to produce an enormous range of visualizations.

In this chapter, we'll focus on the basic components of the R code you need to write to create **ggplot2** visualizations. We'll use the word *syntax* to refer to the components you need to create and the way you need to organize those components. Instead of trying to create a bunch of different visualizations, we start with one simple visualization and keep adding to it to learn more about **ggplot2** features. In future chapters we will cover other kinds of visualizations and more advanced **ggplot2** features.

2.1 Basic ggplot2 syntax

To learn the syntax of a **ggplot2** visualization, it helps to understand how **ggplot2** thinks of a visualization (Figure 2.1).

In **ggplot2**, a visualization is called a "plot." Each plot uses one or more visualization types like the ones we learned in Chapter 1 (e.g., bar chart, scatter plot). The visualization type is called a "geom," which is short for geometric object. You can think of the geom as the shape of the visual elements in the plot.

Each possible geom has default visual properties. This makes sense if we step back and think about a geom as a type of visualization. Every bar chart, for example, has a few things in common with every other bar chart: the numbers are represented by rectangles that vary in length, and there is a separate bar for each value of a categorical variable. To say that another way, a bar chart by default has one or more bars lined up along one axis, and the other axis allows you to compare the lengths of those bars. The default visual properties

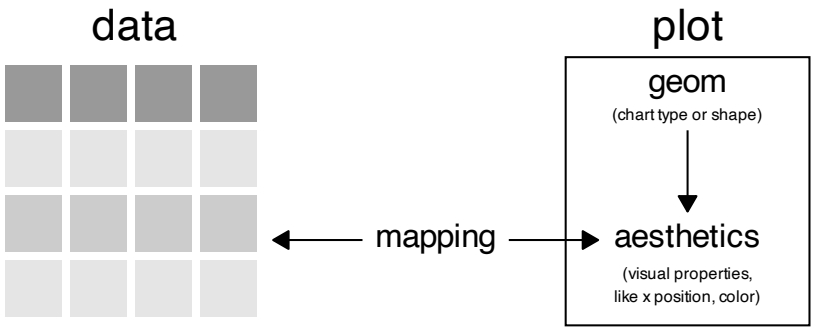


FIGURE 2.1: The ggplot2 model of a visualization.

for a bar chart, then, are that it has two axes: one that serves to organize the bars, and one that serves to measure the lengths.

In **ggplot2**, the visual properties associated with each geom are called "aesthetics." The aesthetics vary a bit from geom to geom, but there are some aesthetics that are required for almost all geoms. The most common aesthetics are called "x" and "y", and they stand for the x and y axes. Most geoms use an x axis to organize one variable and a y axis to organize another. There are plenty of other optional aesthetics, however, that describe how a geom should look. The bars in a bar chart don't just have a position on the x axis and a length on the y axis. They also have a fill color, a fill transparency value, a border color, a border size, and a border line type.

I think of a geom's aesthetics as questions on a form that I need to fill out for the plot to work correctly. If something is option, like the fill color, I don't really have to answer that question on the form. There is a default value **ggplot2** will use if I don't have a specific value in mind. For the x and y axes, though, **ggplot2** does not have a default. Those are required, and **ggplot2** will give me an error if I try to make a plot without providing that information.

How do I tell **ggplot2** what values to use for the aesthetics, like the x and y axes? I need to create a "mapping" between the dataset I want to visualize and the geom's aesthetics. Again, I think if we step back, this is pretty logical. We need to get our data into our visualization. Different parts of the data

need to be associated with different parts of our visualization. That process of creating associations between our dataset and our visualization is called mapping. In **ggplot2**, we will map variables from our dataset to aesthetics in our geom one by one. For each aesthetic we want to specify, we will pick a variable and map it to that aesthetic. In a bar chart with vertical bars, we would map a categorical variable to the x aesthetic and a numerical variable to the y aesthetic. For horizontal bars, the categorical variable would be mapped to the y aesthetic and the numerical variable to the x aesthetic.

Looking again at our diagram in figure 2.1, we can summarize it like this. A plot contains one or more geoms that specify the visualization type. A geom determines the required and optional aesthetics for that part of the plot. To bring data into the visualization, create a mapping between variables in a dataset and the (required or options) aesthetics of the geom.

Now that we understand how **ggplot2** thinks about the structure of a plot, we can turn to the code that we need to write.

2.1.1 Simple Plot Template

In R, there is often many different ways of writing code to accomplish your goals. The same is true in **ggplot2**. We'll start with learning a template for a simple **ggplot2** plot, and then we will learn some additional ways of writing the code that may be easier or work better for different kinds of visualizations.

```
# start with the main plot function and set the data frame
ggplot(data = data_frame) +

# next add a geometry layer and create the mapping between data frame and aesthetics
  geom_...(mapping = aes(aesthetics1 = data_field1, aesthetics2 = data_field))
```

2.2 Example: Vineyards in the Finger Lakes Region of New York State

2.2.1 Step 1: Add Main Plot Function

```
ggplot()
```

2.2.2 Step 2: Set the Data

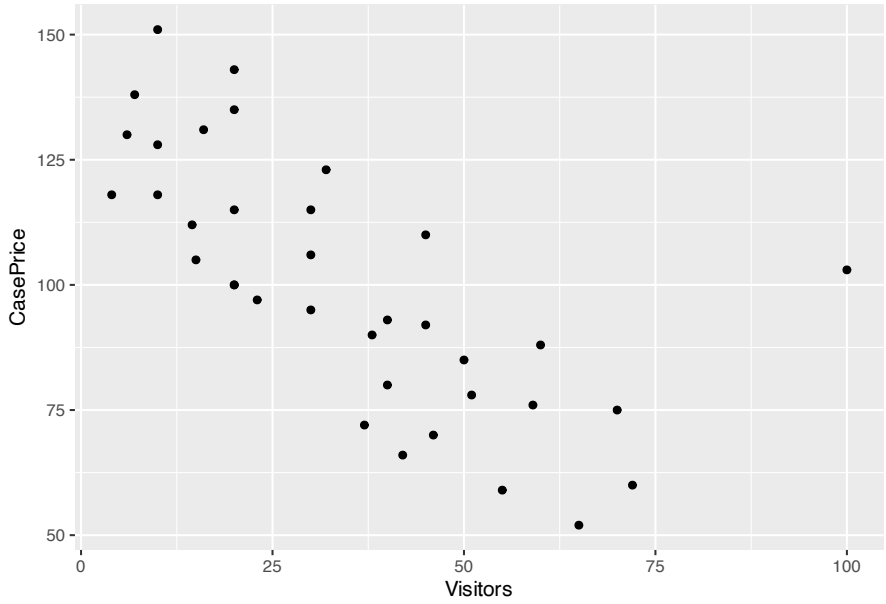
```
ggplot(data = vineyards)
```

2.2.3 Step 3: Choose a shape layer

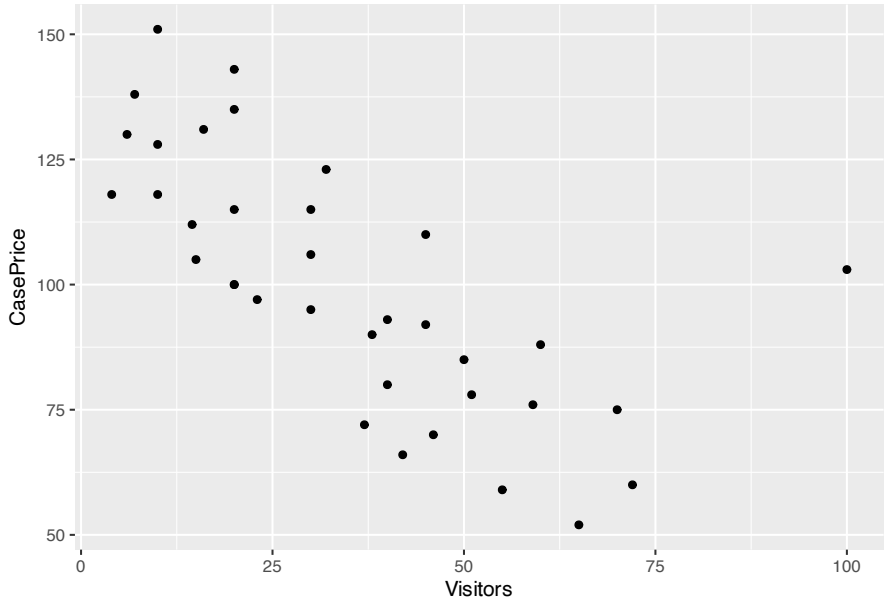
```
# This will throw an error.
ggplot(data = vineyards) +
  geom_point()

# Error: geom_point requires the following missing aesthetics: x and y Run `rlang::last_error()` t
```

2.2.4 Step 4: Map Variables to Aesthetics

```
# Start with the required aesthetics: x and y.
# We'll be mapping x to the Visitors column and y to the CasePrice column.

mapped_variables <- aes(x=Visitors, y=CasePrice)

ggplot(data = vineyards) +
  geom_point(mapping = mapped_variables)
```

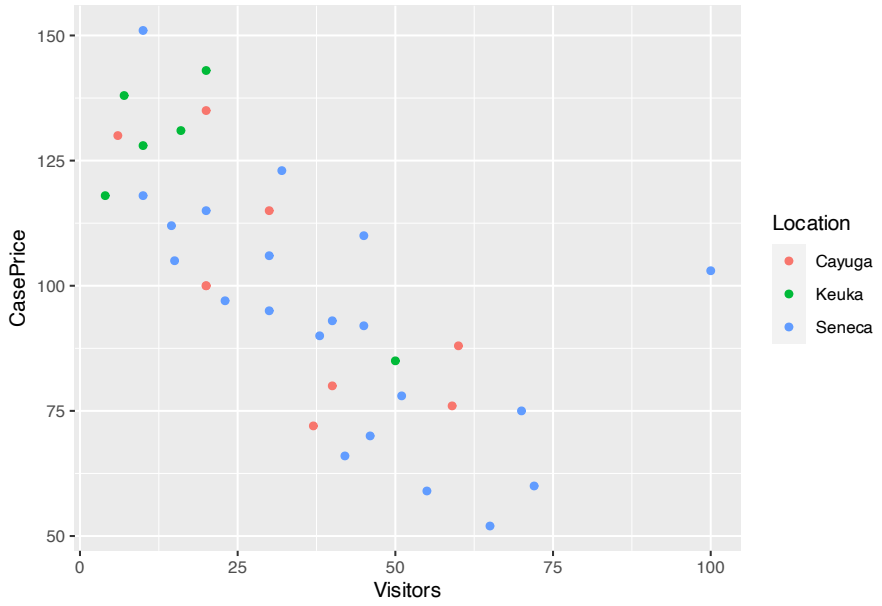



```
# or, all in one step:

ggplot(data = vineyards) +
  geom_point(mapping = aes(x=Visitors, y=CasePrice))
```



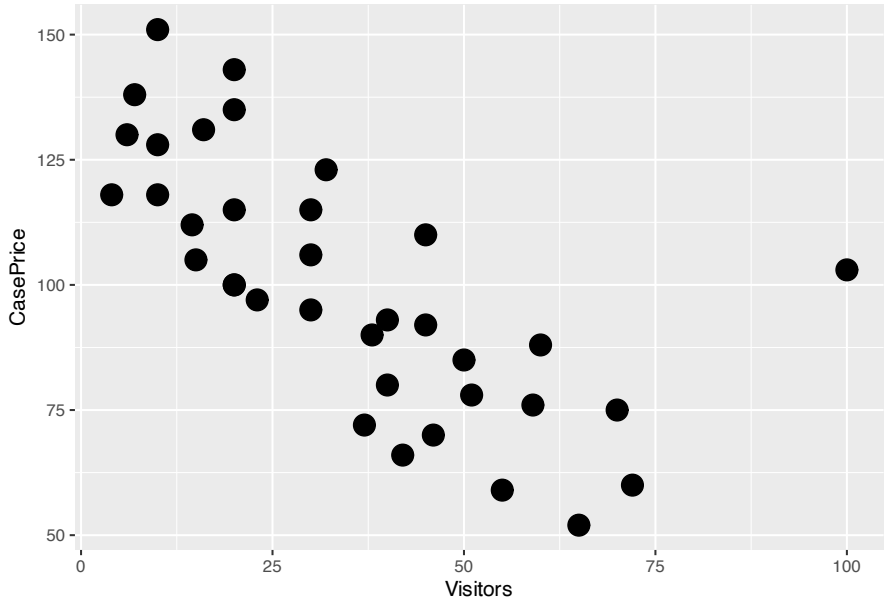
Now, we'll add a third variable. We'll also map the Location variable to the `color` aesthetic.

```
# Start with the required aesthetics: x and y.
# We'll be mapping x to the Visitors column and y to the CasePrice column.

ggplot(data = vineyards) +
  geom_point(mapping = aes(x=Visitors, y=CasePrice, color=Location))
```

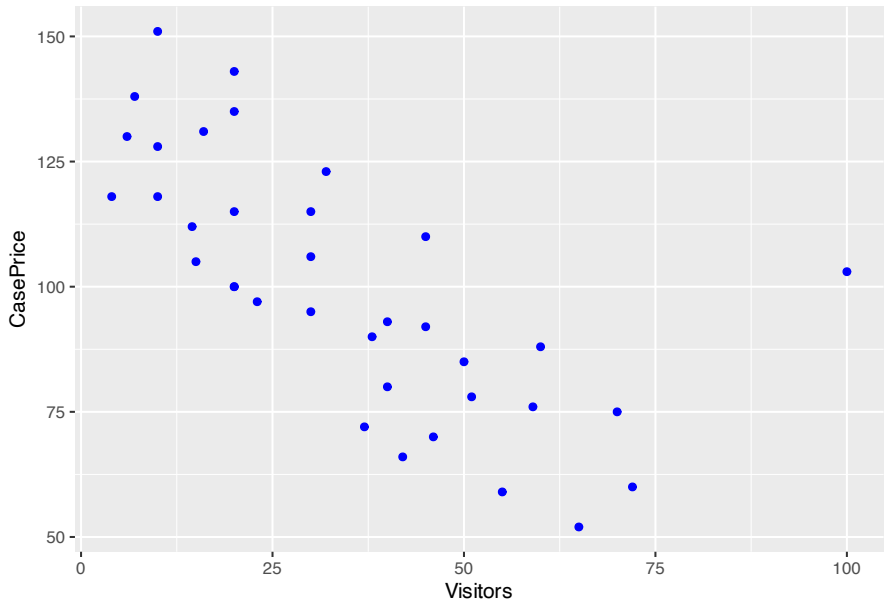


2.2.5 Step 5: Adjust the Defaults

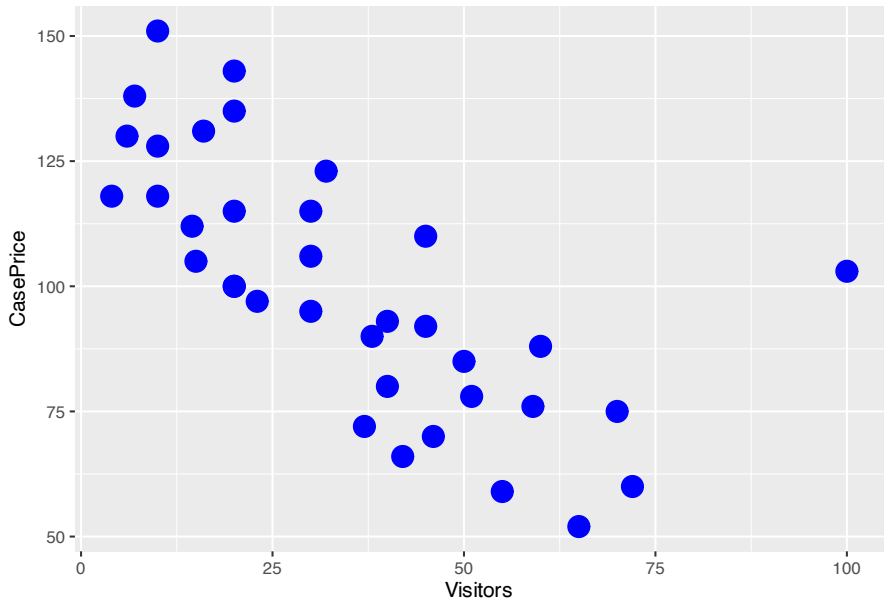
```
# Changing the default size of all of the points
ggplot(data = vineyards, mapping = aes(x=Visitors, y=CasePrice)) +
  geom_point(size=5)
```



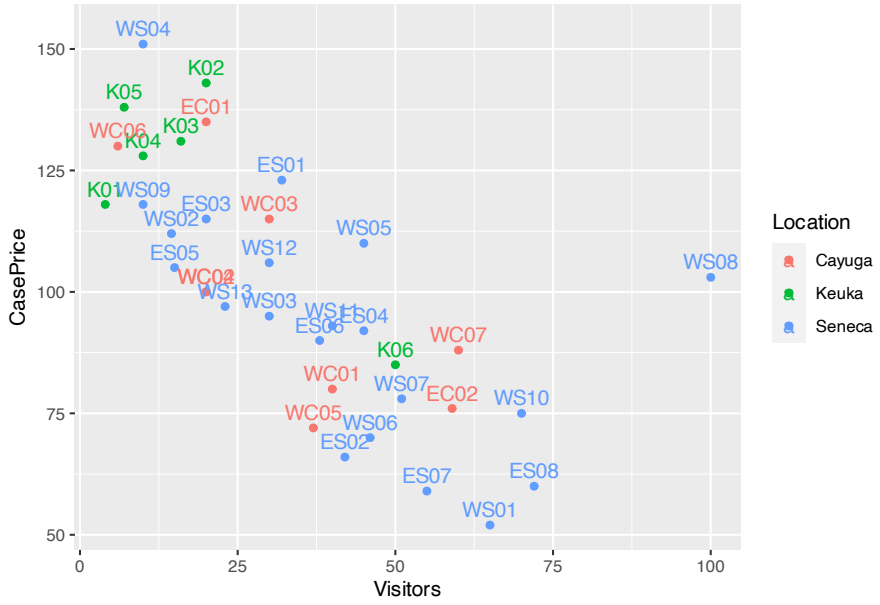

```
# Changing the default color of all of the points
ggplot(data = vineyards, mapping = aes(x=Visitors, y=CasePrice)) +
  geom_point(color="blue")
```

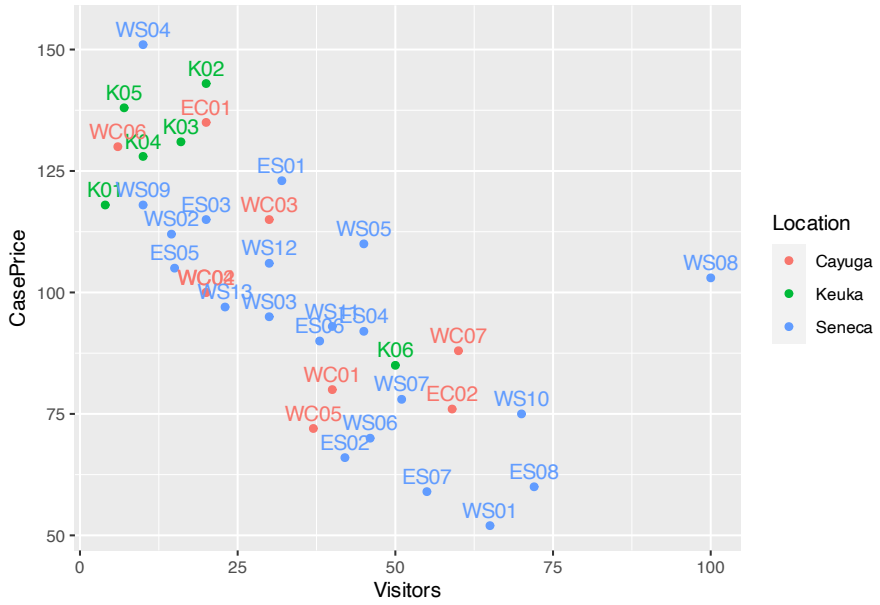


```
# Changing both size and color
ggplot(data = vineyards, mapping = aes(x=Visitors, y=CasePrice)) +
  geom_point(size=5, color="blue")
```



2.3 Inheritance





2.4 How to Debug Code

3

Working with textual data in ggplot2

Note on cleaning data: use duke_enrollment (either by status or school) to talk about factors. Have Semester, which is really a time-based variable. Need to combine with Year to get the real sequence of enrollment.

3.1 What is a text variable?

3.2 Visualizing text variables

3.2.1 Common mappings for text variables

3.2.2 Ordering

3.2.3 Long Text Labels

3.2.4 Picking Colors

3.2.5 Small Multiples

4

Customizing the design of ggplot2 visualizations

4.1 Common Data Modifications

4.2 Labels

4.3 Scales

4.4 Themes

5

Avoiding unethical design practices

5.1 Due Diligence for Analysis

- If your visualization is making a claim, you need to have investigated other explanations of the patterns, have a reasonable certainty that the pattern is meaningful, trust/verify that the data were corrected appropriately and are appropriate for the analysis you're conducting

5.2 Promote Dignity and Agency

- Nothing about us without us
- Choosing labels and colors
- Provide sufficient information for users to make their own meaning

5.3 Reduce Distortion

5.3.1 Color

5.3.2 Shapes

- If shape changes in both dimensions (width and height), make sure data is represented correctly by area, not just one dimension

5.3.3 Truncated Axes

5.4 Compare Like Things

5.4.1 Dual Axis

6

Building ggplot2 visualizations into print publications

6.1 Exporting ggplot2 Visualizations

6.2 Using RMarkdown to create PDF reports

6.3 Using R Packages to Build Word and PowerPoint Files

7

Basic accessibility for static visualizations

7.1 Accessible Text Accompanying the Visualization

- Accessible Font
- Embedded tags/styles
- Large Text
- High Contrast

7.2 Neurodivergence

- Limited Visual Confusion

7.3 Low Vision

- Large text
  - "output-examples" file[1]
- High color contrast
  - Both marks/text on background and labels on marks
  - Check with savonliquide package[2]

[1] https://github.com/amzoss/RVis-2Day/blob/master/Day%201/templates/output-examples.md

[2] https://github.com/feddelegrand7/savonliquide

7.4 Color Vision Deficiency

7.4.1 Dual encoding (never just color)

- Line color – also vary line type
- Point color – also vary point shape
- https://www.youtube.com/watch?v=mbi_JVC1arM

7.4.2 Color palettes

- colorspace package[3]

7.5 Screen Reader Users

7.5.1 Alternative Text

In R, R Markdown:

- fig.alt[4] in code chunk (new, just for HTML output)
- fig.cap[5] in code chunk as backup
- embedded images: write alt text between square brackets
- New: ggplot2 v3.3.4 adds alt option in labs()[6], with plans to propagate to Rmd, Shiny

Writing good alt text for visualizations[7]

7.5.2 Longer Descriptions

Longer descriptions: savonliquide package[8]

7.5.3 Converting graphics to sound, touch, text, table

- sonify package

[3] http://colorspace.r-forge.r-project.org/index.html
[4] https://blog.rstudio.com/2021/04/20/knitr-fig-alt/
[5] https://bookdown.org/yihui/rmarkdown/r-code.html
[6] https://ggplot2.tidyverse.org/reference/labs.html
[7] https://nightingaledvs.com/writing-alt-text-for-data-visualization/
[8] https://github.com/feddelegrand7/savonliquide

- tactileR package
- BrailleR package[9]
  - Note: set plot title, subtitle, caption using labs()

Accessible Data Science for the Blind Using R[10]

7.6 Accessibility Resources

- savonliquide package[11]
- Making betteR figures: Accessibility and Universal Design[12]
- Highlights from the DVS accessibility fireside chat[13]

[9] https://r-resources.massey.ac.nz/BrailleRInAction/GGPlot.html
[10] https://jooyoungseo.com/post/ds4blind/
[11] https://github.com/feddelegrand7/savonliquide
[12] https://bookdown.org/ybrandvain/Applied-Biostats/betteRfigs.html#accessibility-and-universal-design
[13] https://nightingaledvs.com/highlights-from-the-dvs-accessibility-fireside-chat/

8

Exploring interactivity in visualizations with plotly and crosstalk

8.1 Interactivity for Exploration

- Components: Picking the right visual elements
- Input: Giving users the right controls
  - Can support generalized workflows, custom subsetting, changing parameters, personalizing output
- Layout Arranging everything in the right place

8.2 Interactive Components

- functionality built in for users to change appearance of chart or what is included/in focus

8.2.1 About Websites and HTML

8.2.2 Simple Markdown Websites

8.3 HTML Widgets

8.3.1 Plotly

8.3.2 DataTables

8.4 CrossTalk

- Coordinated Views
- Input Controls

9

Using RMarkdown to build websites for projects

move before interactive visualizations? combine?

9.1 Static Websites

9.2 Hosting

9.3 Associating a Website with an R Project

10

Using RMarkdown to build dashboards for projects

11

Basic usability for interactive visualizations

move earlier?

How to measure usability?

MEELS

- Memorability
- Efficiency
- Errors
- Learnability
- Satisfaction

https://www.nngroup.com/articles/usability-101-introduction-to-usability/

Three independent measures * Accuracy (Percent Correct) * Time to Complete Each Scenario * Satisfaction

https://www.usability.gov/get-involved/blog/2006/03/complete-picture-with-usability-testing.html

11.1 What do users want to do with a dashboard

Typical User Performance Tasks

- Retrieve absolute value: "What was the price of barrel of oil in February 2015?"
- Retrieve ratio value: "About what is the ratio of the cost of a sandwich to the total cost of room service in Seattle?"
- Find extremum: "In which country is the average internet speed the fastest?"
- Make comparisons: "Which metro system has more ridership?"
- Find correlations/trends: "What is the relationship between the height and weight of these individuals?"

http://dx.doi.org/10.1109/TVCG.2016.2598920

Dashboard activities

- Navigation between pages
- Customizing page content with searches, filters
- Locating relevant content
- Reading content and charts

Tasks for dashboards * Check that things are going ok * Check that a process had completed * Troubleshoot a problem * Support a tactical decision * Decide where to allocate resources * Try to better understand a complex system

http://www.jeromecukier.net/2015/02/07/dashboards-as-products/

11.1.1 Conducting an Initial User Study

11.2 How to Test a Dashboard

Different things to test

- Tone and style
- Comprehension
- Usefulness
- Readability
- Organization
- Navigation
- Interaction
- Engagement and influence

https://www.uxmatters.com/mt/archives/2009/12/testing-content-concepts.php

11.2.1 Preliminary Questions

- Can act as a warm up before scenarios/tasks
- Good for first impressions
- Avoid leading, closed, or vague questions

https://www.nngroup.com/articles/leading-questions/

Sample Preliminary Questions

Ask: “Have you heard of [site]?” _____Yes _____No

If yes, ask: “Tell me what you know about them.”

Bring the test participant to [site]. Ask: “Just from looking at this site, what

kinds of information do you think you could get from this site? Please be specific."

Ask: "Who do you think this site is designed for? Why?" (Probe: public, health professionals, etc.)

https://www.usability.gov/how-to-and-tools/resources/templates/what-testnote-takers-guide.html

11.2.2 5-second test for first impressions

Evaluating memorability, style, satisfaction

Show the page for only 5 seconds, then ask (e.g.): * "What is the purpose of the page?" * "What are the main elements you can recall?" * "Who do you think the intended audience is?" * "Did the design/brand appear trustworthy?" * "What was your impression of the design?"

https://fivesecondtest.com/

11.2.3 Think-aloud Scenarios

Evaluating efficiency, errors, learnability

Present 3-5 Think-Aloud Scenarios

Specific tasks you designed the visualization for, based on initial user testing * Make the task realistic * Make the task actionable * Avoid giving clues and describing the steps

https://www.nngroup.com/articles/thinking-aloud-the-1-usability-tool/
https://www.nngroup.com/articles/task-scenarios-usability-testing/

"First click" testing as a variation

Describe scenario, then ask what participant would click on first https://www.usability.gov/how-to-and-tools/methods/first-click-testing.html

11.2.4 Closing Questions

Evaluating memorability, style, satisfaction

- What is your overall impression to [site]?
- What is your impression of the search capability?
- Do you feel this site is current? Why?
- What did you like best about the site?
- What did you like least about the site?

https://www.usability.gov/how-to-and-tools/resources/templates/what-testnote-takers-guide.html

One question to rule them all

Single Ease Question (SEQ)

Overall, this task was?

Very Difficult…Very Easy (7-point scale)

https://measuringu.com/single-question/

12

Teacher's guide

sample text

We talk about the *FOO* method in this chapter.

12.1 Live Coding

12.2 Sample Data

A

Datasets

Duke Enrollment

Duke enrollment[1]

Sample of Duke Enrollment By School dataset, Table A.1.

Bar Chart

Figure A.1.

Coral Resilience Data

Protecting coral reefs[2]

Figure A.2.

```
## Warning: Removed 1 rows containing missing values
## (geom_point).
```

[1] https://doi.org/10.7924/r4db82p1j
[2] https://doi.org/10.7924/G8348HFP

TABLE A.1: A sample from the Duke Enrollment By School dataset.

Year	Semester	Origin	Region	Sex	School	Count
1970	Fall	Alabama	United States	Female	Trinity	11
1970	Fall	Alabama	United States	Female	Graduate	7
1970	Fall	Alabama	United States	Female	Divinity	1
1970	Fall	Alabama	United States	Female	Law	1
1970	Fall	Alaska	United States	Female	Trinity	1
1970	Fall	Alaska	United States	Female	Graduate	1

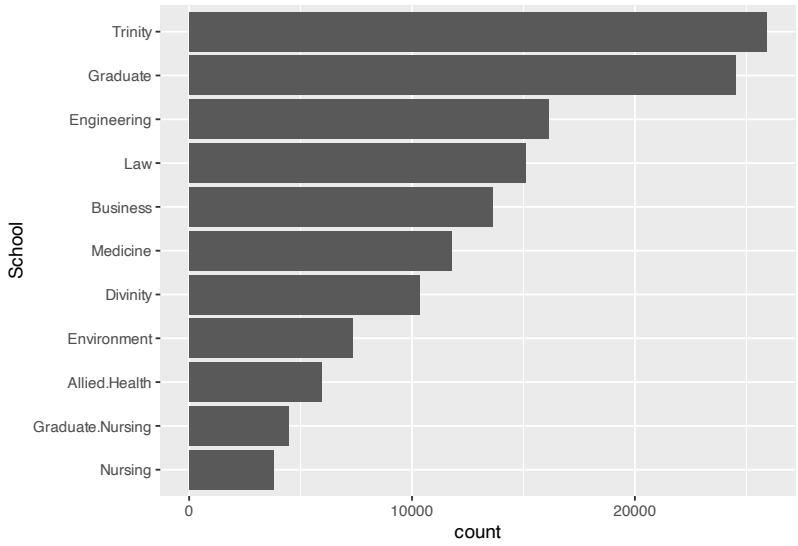


FIGURE A.1: Total Duke Enrollment by School

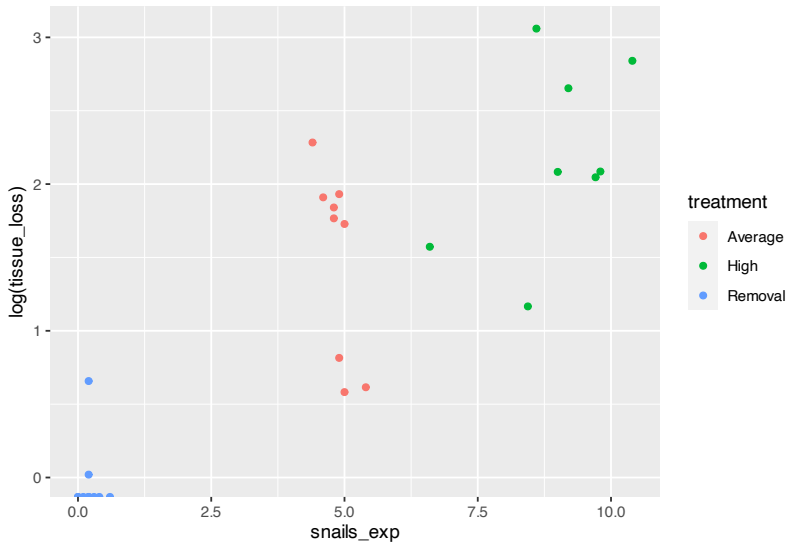
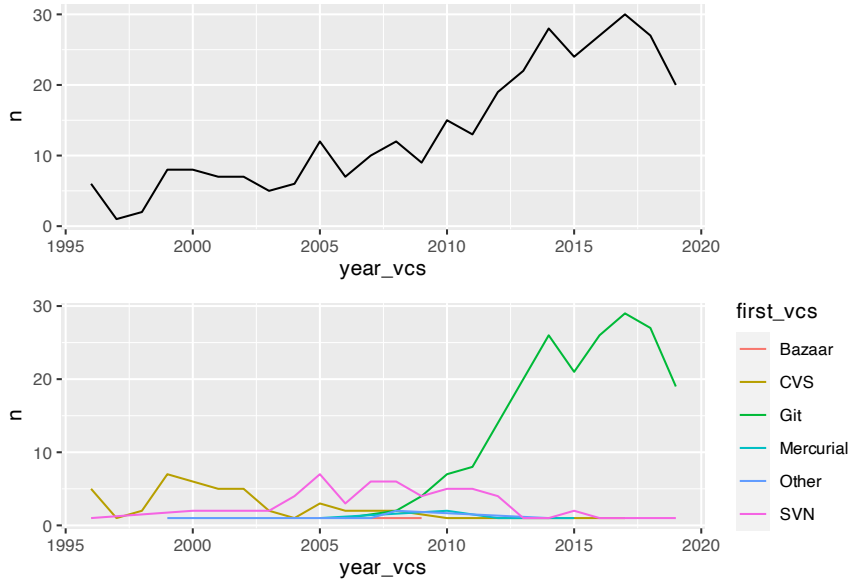


FIGURE A.2: Log of tissue loss by snail density

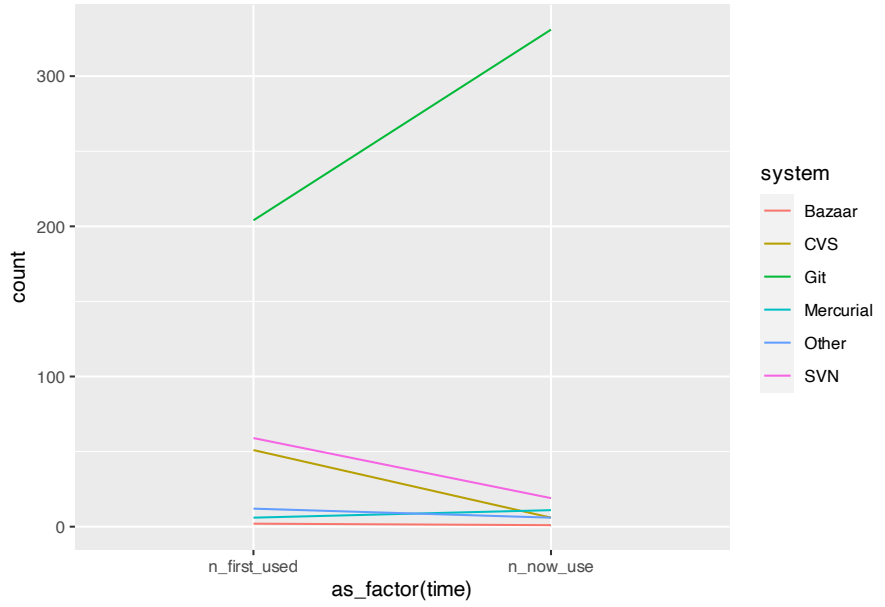
Git Experience

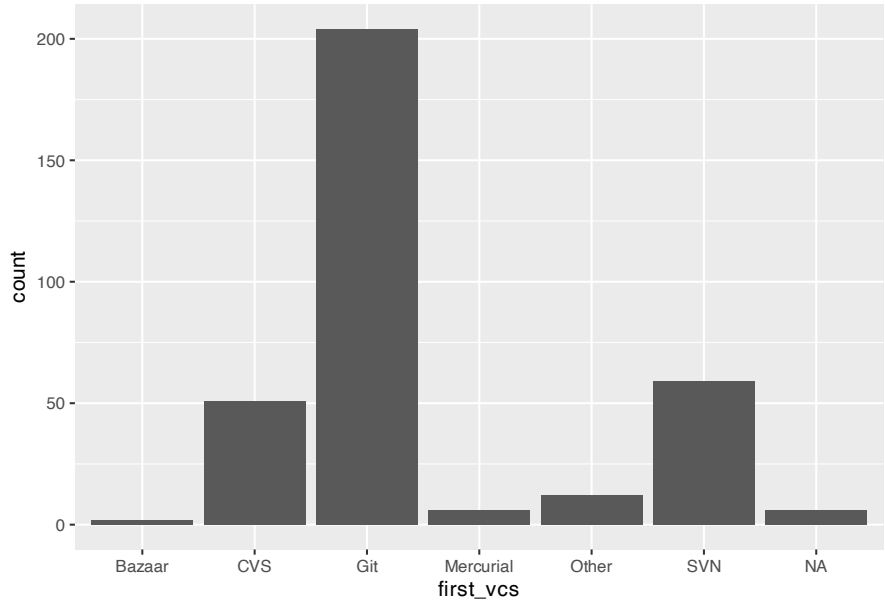
A Behavioral Approach to Understanding the Git Experience[3]

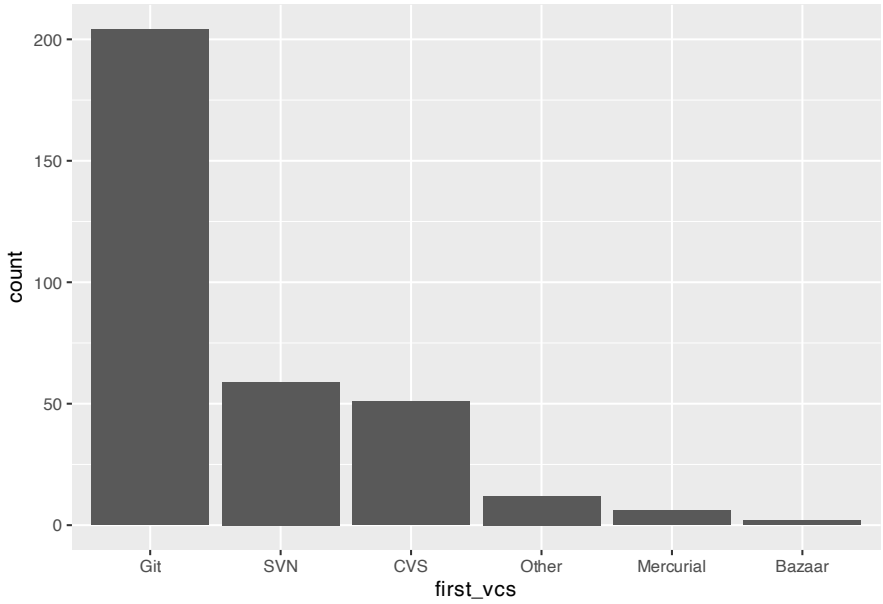


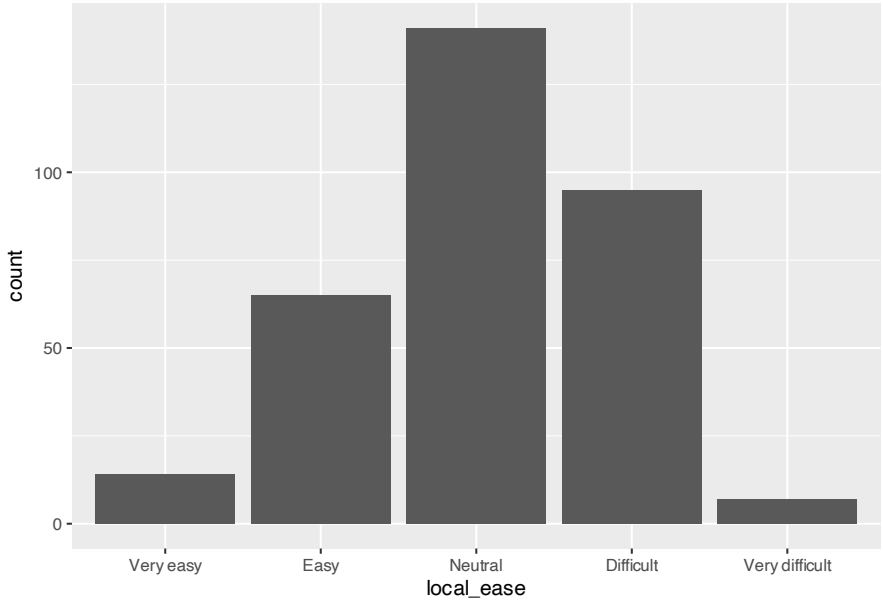

```
## Joining, by = "system"
```

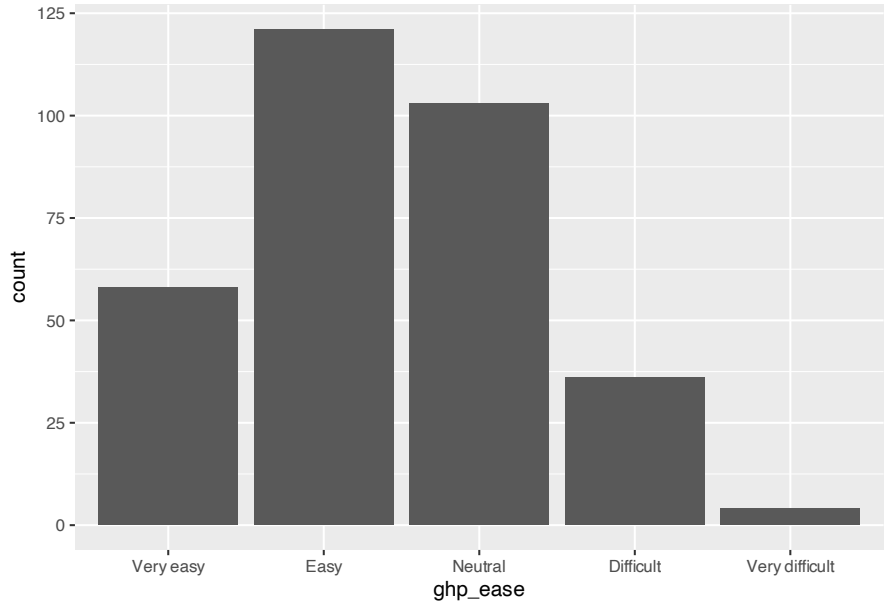
[3] https://osf.io/57tb8/

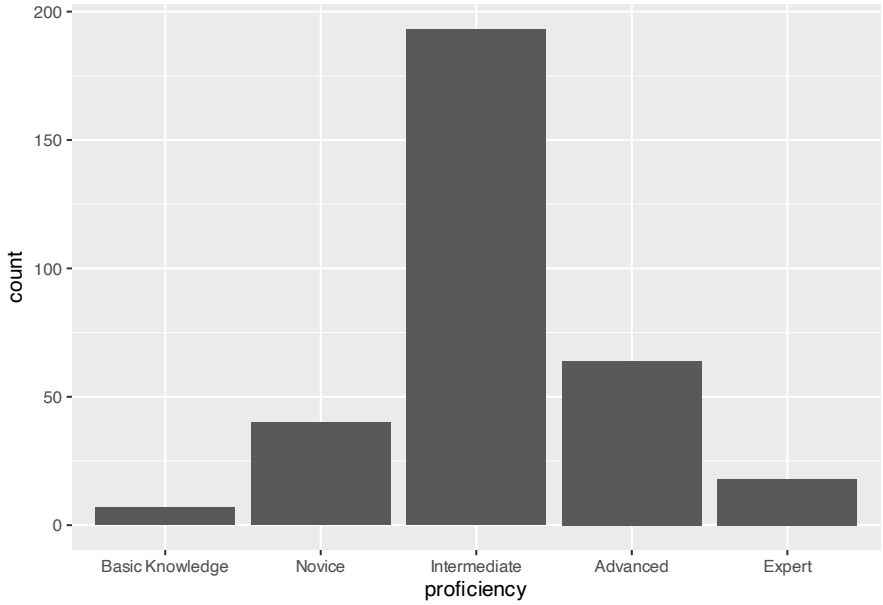

300
200
100
0
count
n_first_used
n_now_use
as_factor(time)
system
Bazaar
CVS
Git
Mercurial
Other
SVN

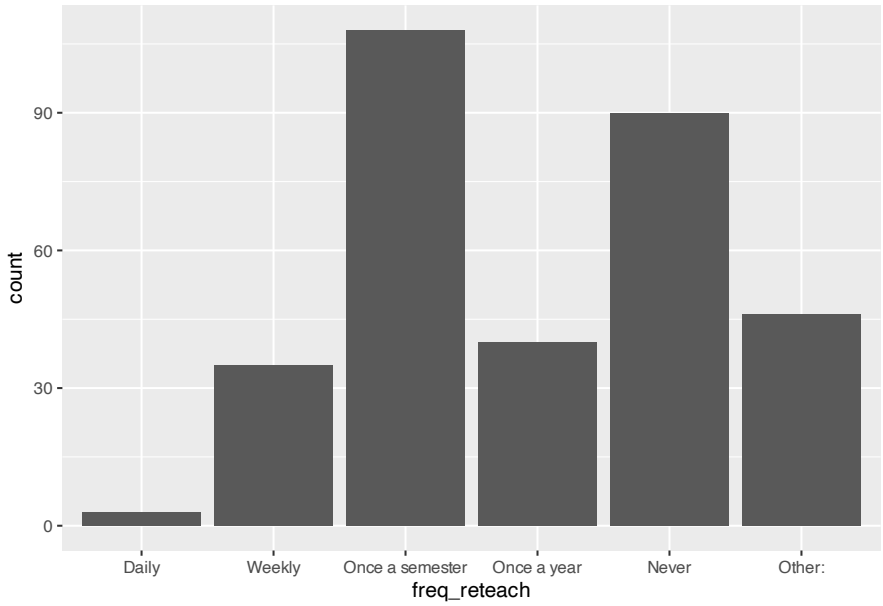

200
150
100
50
0
count
Bazaar
CVS
Git
Mercurial
Other
SVN
NA
first_vcs

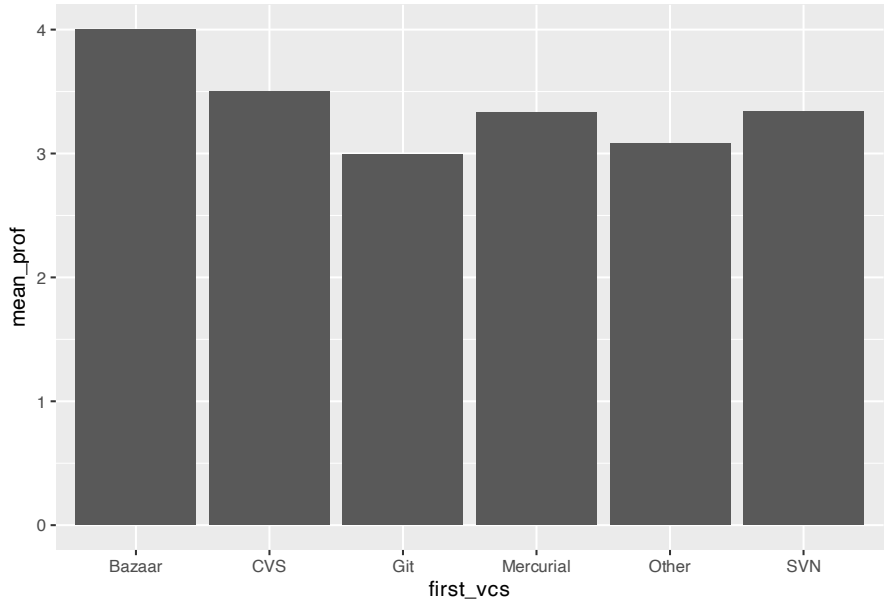

200
150
100
50
0
count
Git
SVN
CVS
Other
Mercurial
Bazaar
first_vcs

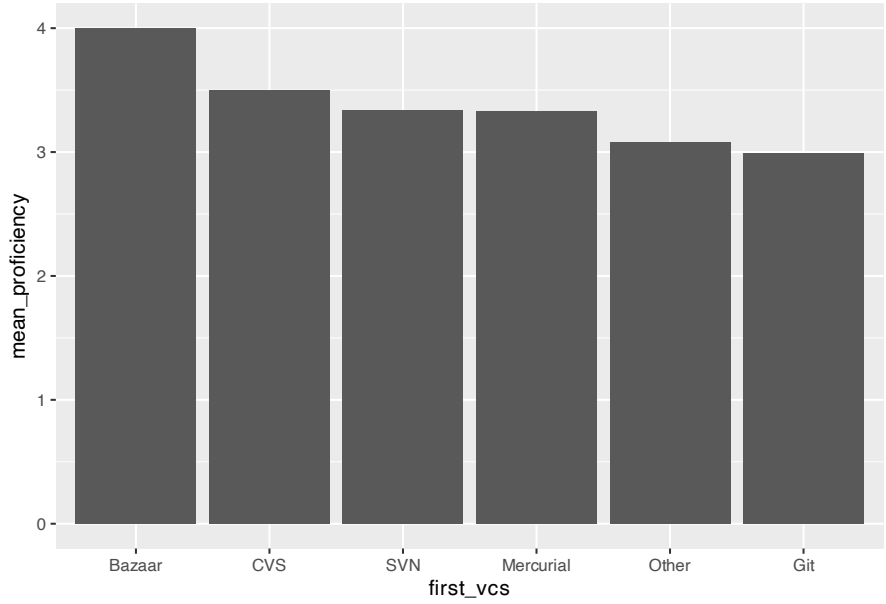

100
50
0
count
Very easy
Easy
Neutral
Difficult
Very difficult
local_ease


125
100
75
50
25
0
count
Very easy
Easy
Neutral
Difficult
Very difficult
ghp_ease

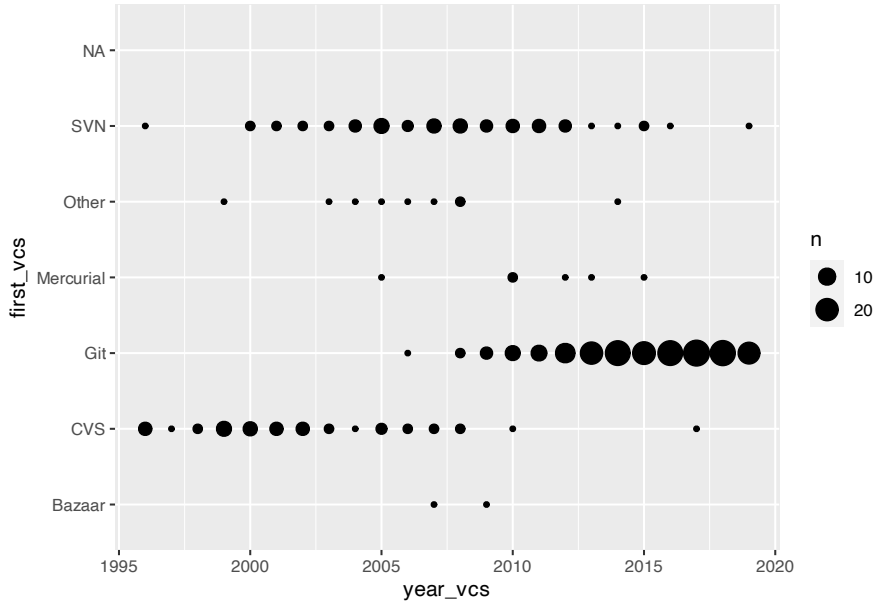

200
150
100
50
0
count
Basic Knowledge
Novice
Intermediate
Advanced
Expert
proficiency


90
60
30
0
count
Daily
Weekly
Once a semester
Once a year
Never
Other:
freq_reteach

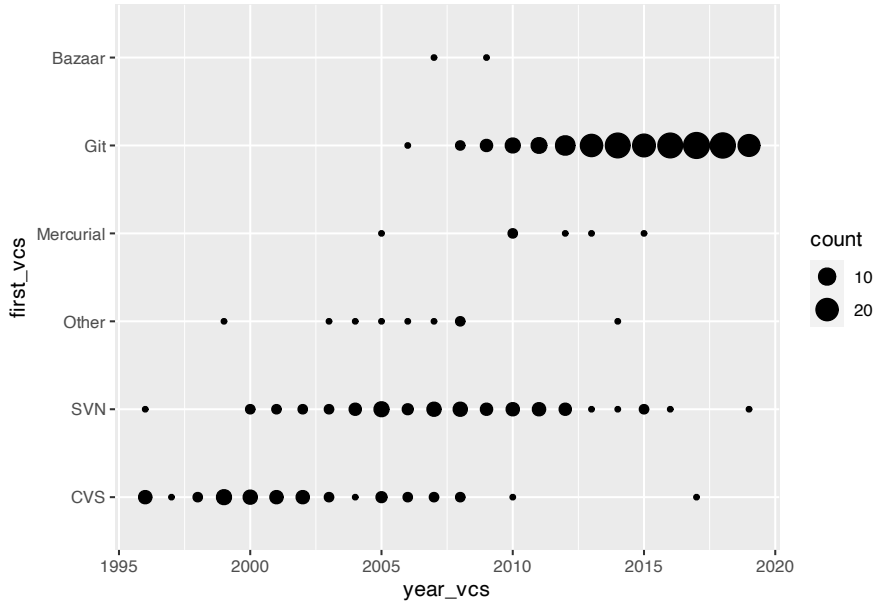




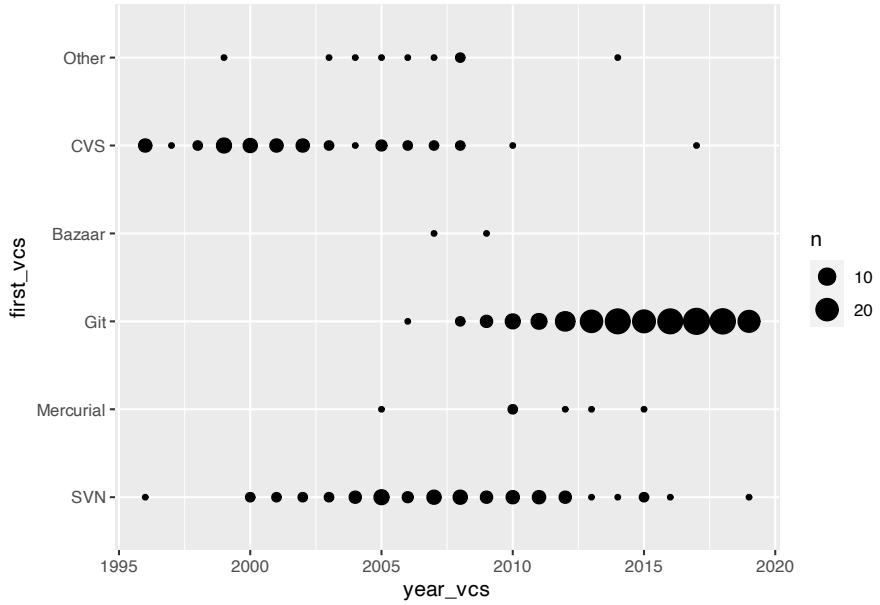

```
## Warning: Removed 15 rows containing non-finite values
## (stat_sum).
```



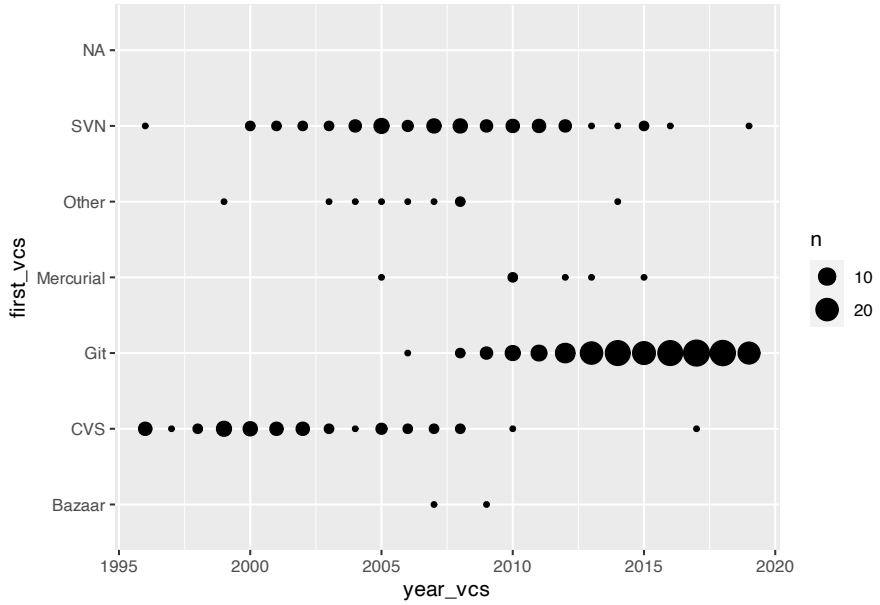

```
## `summarise()` has grouped output by 'year_vcs'. You
## can override using the `.groups` argument.
```




```
## Warning: Removed 9 rows containing non-finite values
## (stat_sum).
```

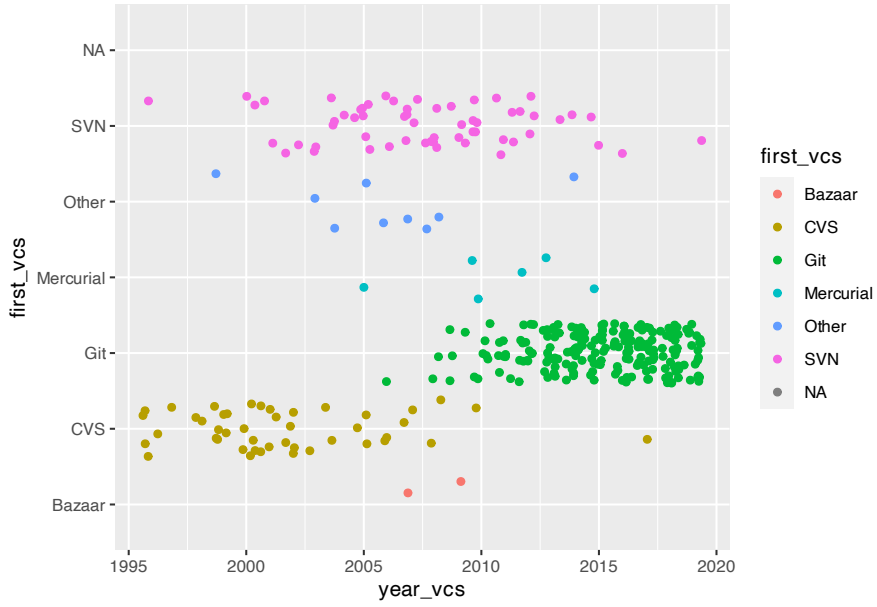



```
## Warning: Removed 3 rows containing missing values
## (geom_point).
```

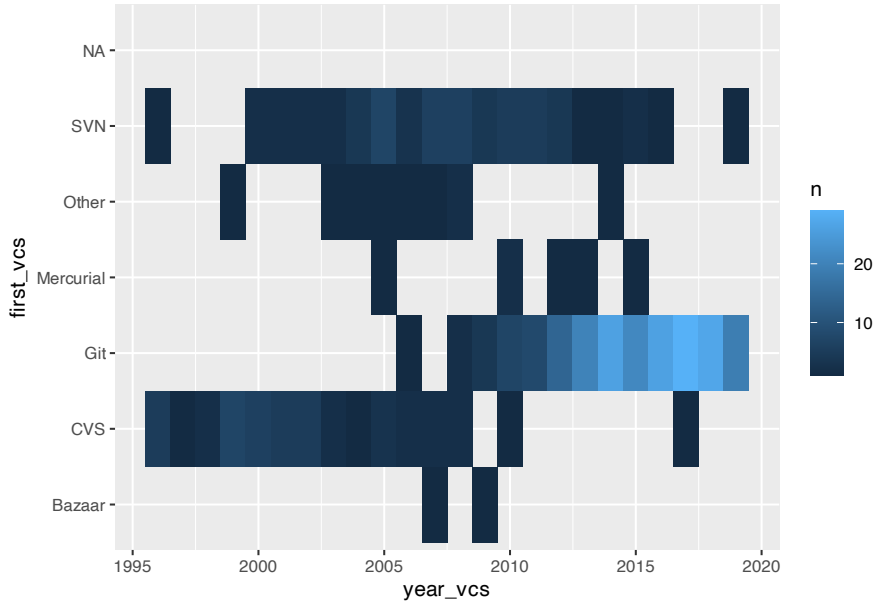


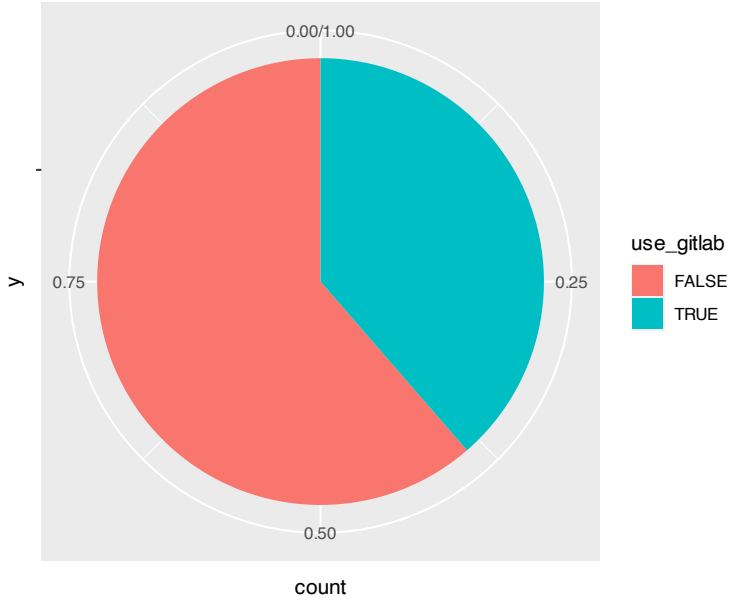

```
## Warning: Removed 15 rows containing missing values
## (geom_point).
```


```
## Warning: Removed 15 rows containing missing values
## (geom_point).
```




```
## Warning: Removed 3 rows containing missing values
## (geom_tile).
```


NA
SVN
Other
Mercurial
Git
CVS
Bazaar
first_vcs
1995
2000
2005
2010
2015
2020
year_vcs
n
20
10


0.00/1.00
0.25
0.50
0.75
y
count
use_gitlab
FALSE
TRUE

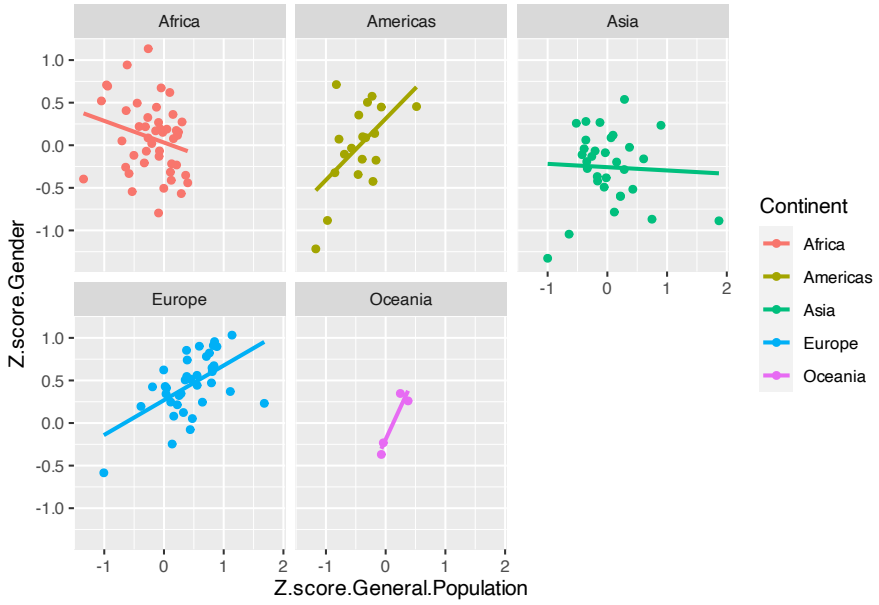
Inclusiveness Index

Inclusiveness Index[4]

```
## `geom_smooth()` using formula 'y ~ x'
```

```
## Warning: Removed 111 rows containing non-finite values
## (stat_smooth).
```

```
## Warning: Removed 111 rows containing missing values
## (geom_point).
```

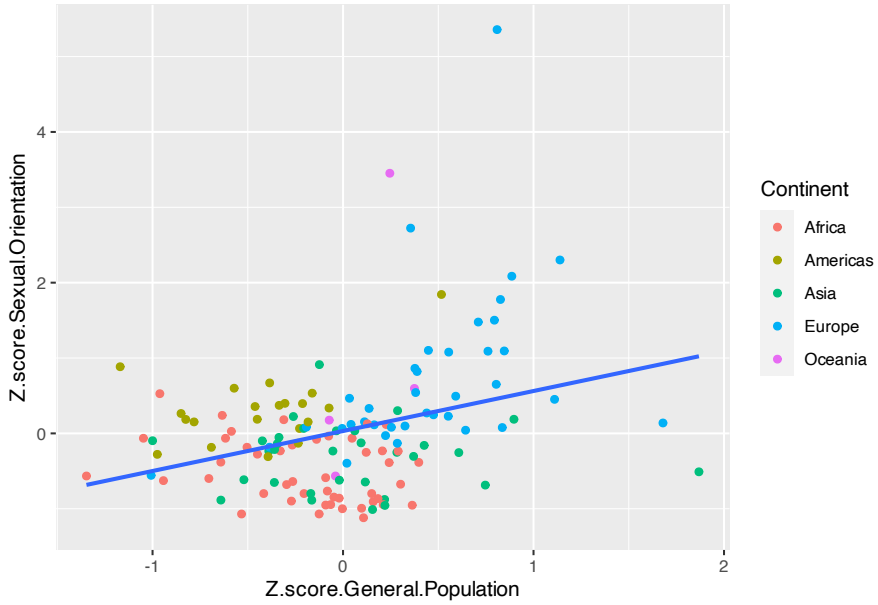



```
## `geom_smooth()` using formula 'y ~ x'
```

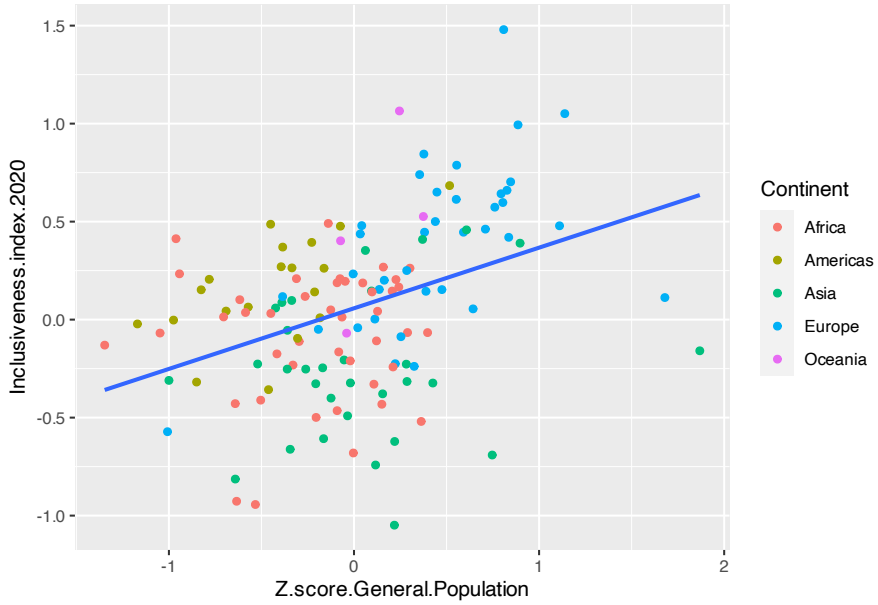
```
## Warning: Removed 109 rows containing non-finite values
## (stat_smooth).
```

```
## Warning: Removed 109 rows containing missing values
## (geom_point).
```

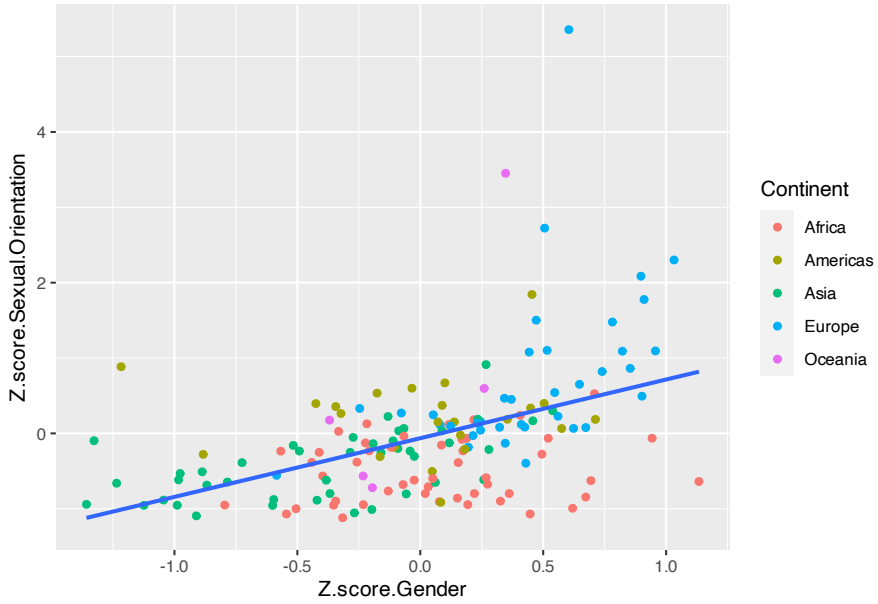
[4] https://belonging.berkeley.edu/inclusivenessindex/data-and-resources




```
## `geom_smooth()` using formula 'y ~ x'

## Warning: Removed 113 rows containing non-finite values
## (stat_smooth).

## Warning: Removed 113 rows containing missing values
## (geom_point).
```

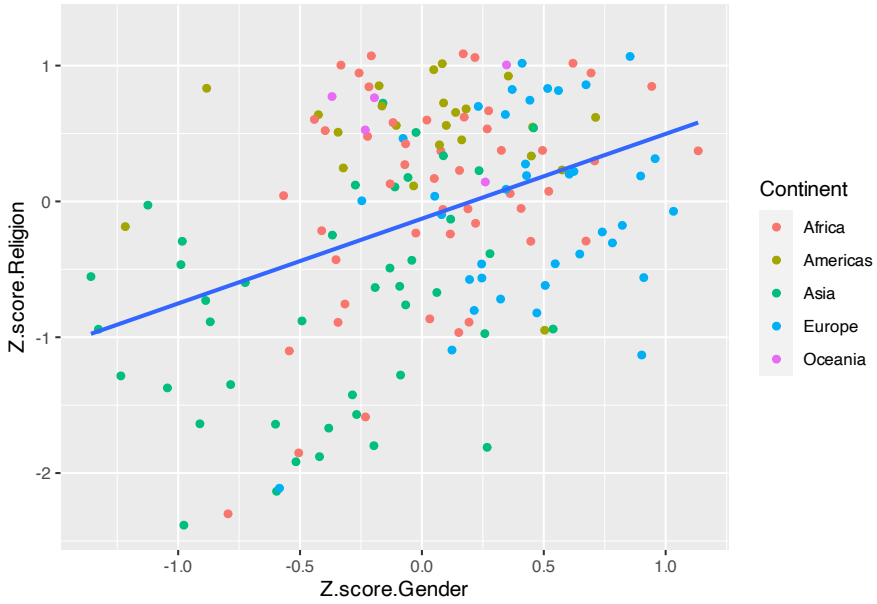



```
## `geom_smooth()` using formula 'y ~ x'

## Warning: Removed 90 rows containing non-finite values
## (stat_smooth).

## Warning: Removed 90 rows containing missing values
## (geom_point).
```

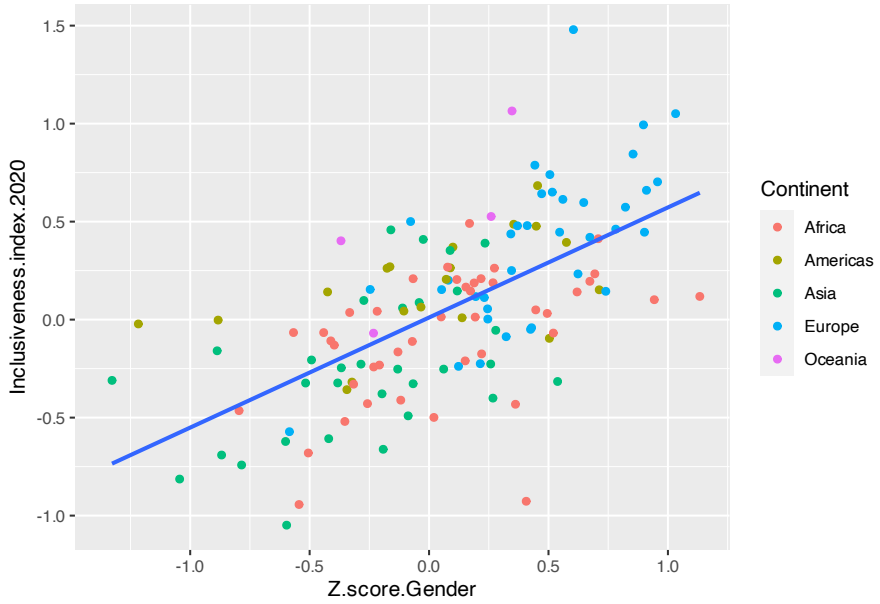



```
## `geom_smooth()` using formula 'y ~ x'
```

```
## Warning: Removed 90 rows containing non-finite values
## (stat_smooth).
## Removed 90 rows containing missing values
## (geom_point).
```

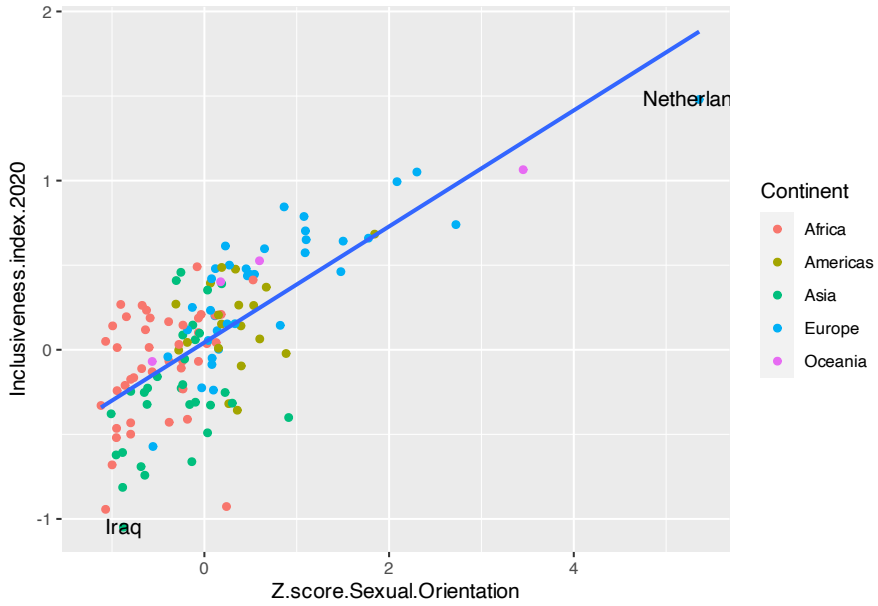



```
## `geom_smooth()` using formula 'y ~ x'

## Warning: Removed 113 rows containing non-finite values
## (stat_smooth).

## Warning: Removed 113 rows containing missing values
## (geom_point).
```



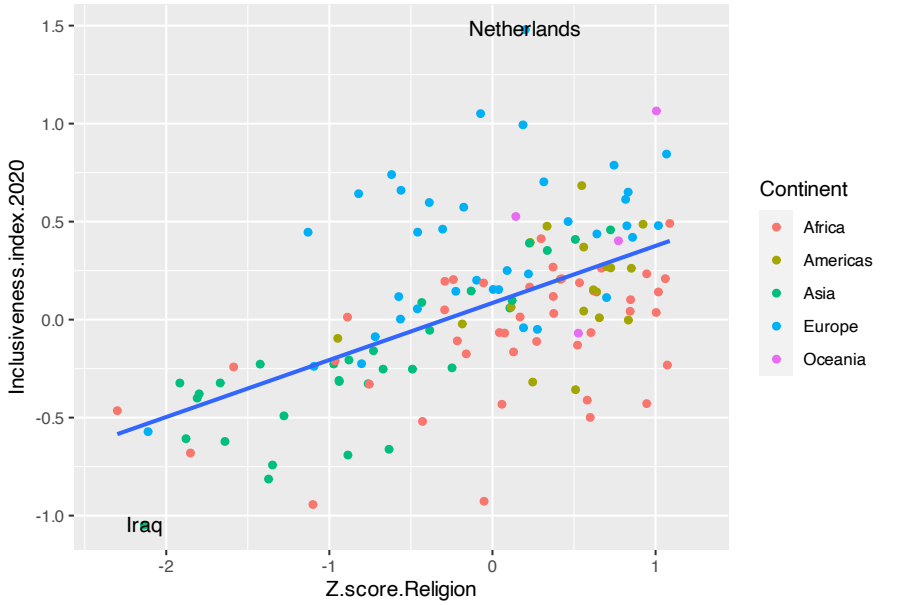

```
## `geom_smooth()` using formula 'y ~ x'
```

```
## Warning: Removed 113 rows containing non-finite values
## (stat_smooth).
## Removed 113 rows containing missing values
## (geom_point).
```

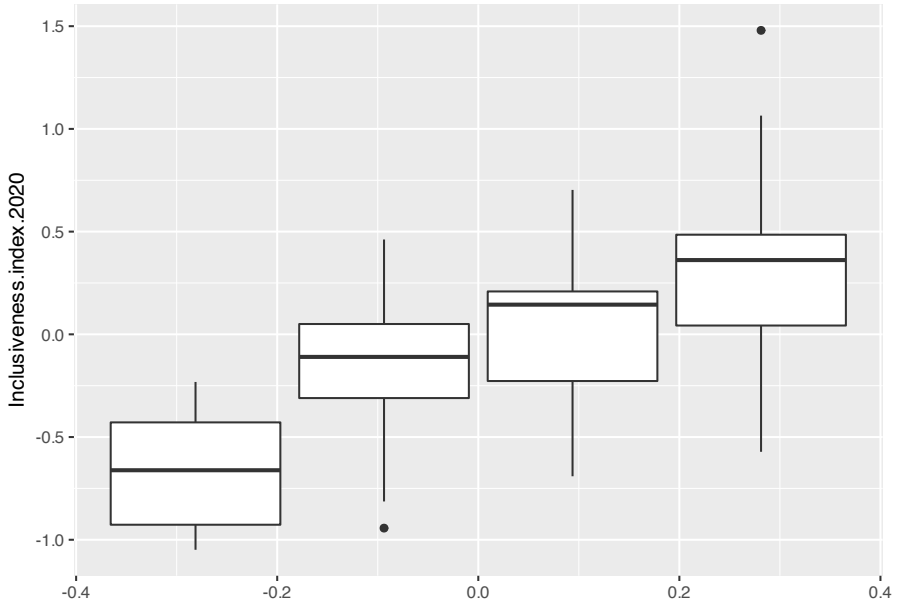


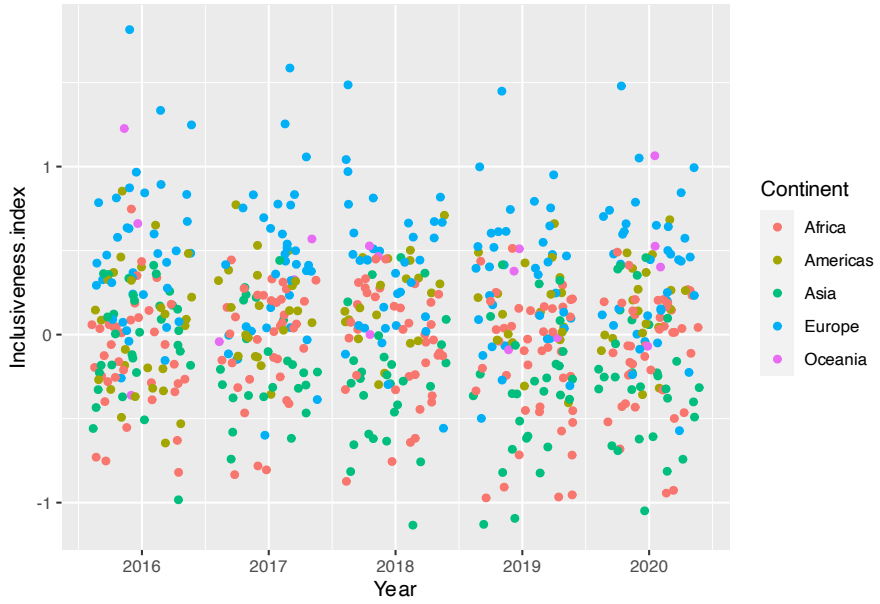

```
## `geom_smooth()` using formula 'y ~ x'
```

```
## Warning: Removed 113 rows containing non-finite values
## (stat_smooth).
## Removed 113 rows containing missing values
## (geom_point).
```




```
## Warning: Removed 113 rows containing non-finite values
## (stat_boxplot).
```





Candidate Demographics

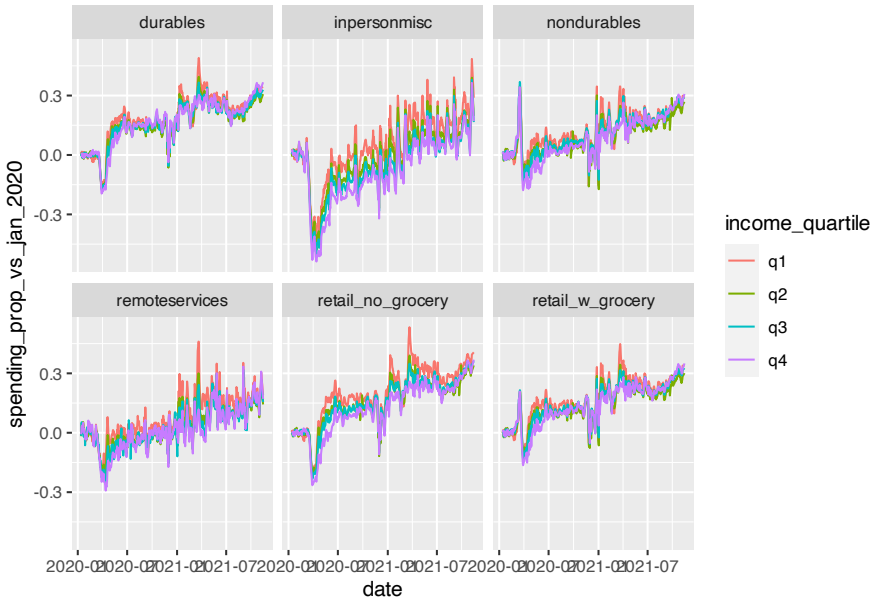
Candidate Demographics[5]

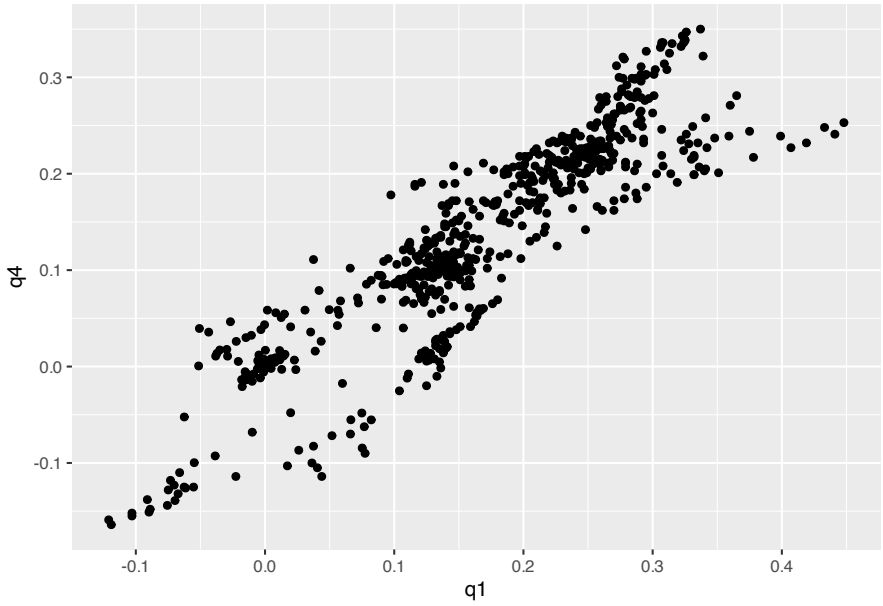
Includes State, Candidate Name, Candidate Party, Office Name, White/Non-White, Race, Gender, Race/Gender Category, Office Level; 4 years (2012, 2014, 2016, 2018), over 40k records

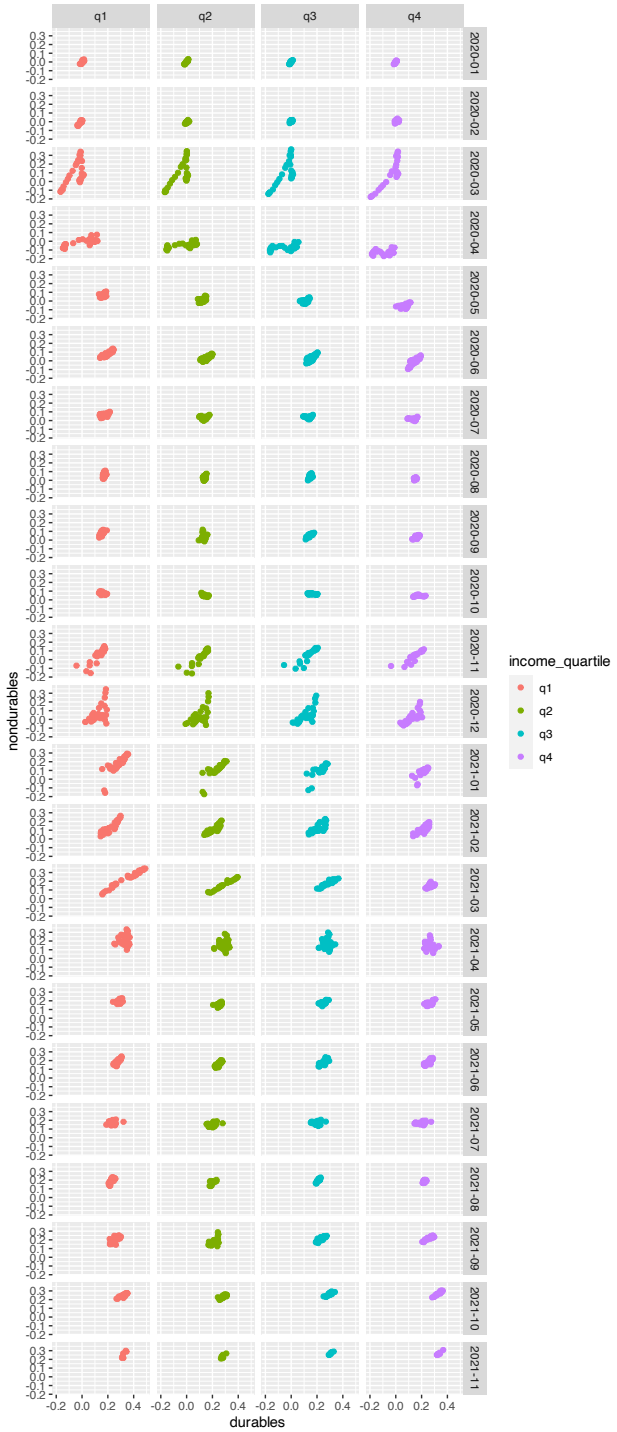
Affinity Spending

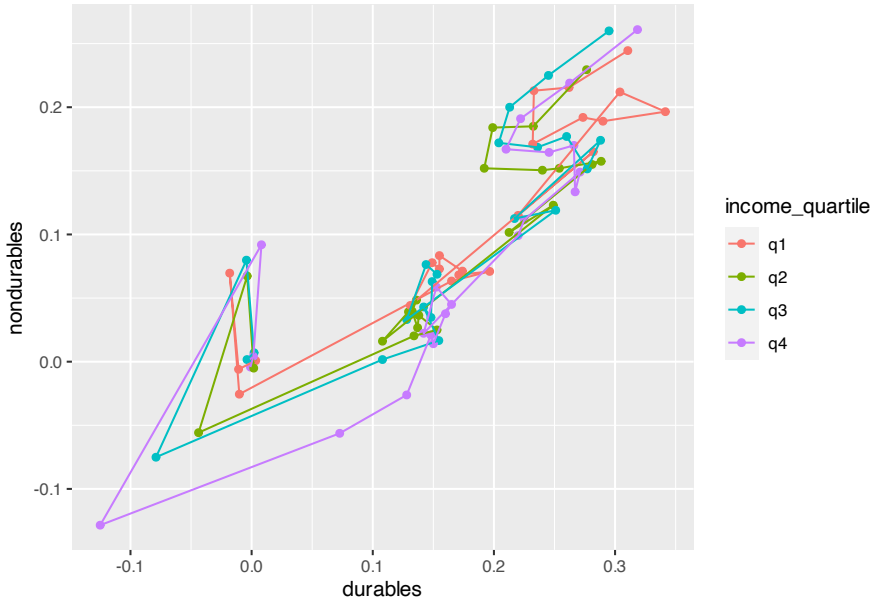
Affinity Spending[6]

[5] https://wholeads.us/research/rising-tide-ballot-demographics/
[6] https://github.com/OpportunityInsights/EconomicTracker


durables
inpersonmisc
nondurables
remoteservices
retail_no_grocery
retail_w_grocery
spending_prop_vs_jan_2020
0.3
0.0
-0.3
date
income_quartile
q1
q2
q3
q4


q4
q1
0.3
0.2
0.1
0.0
-0.1
0.4

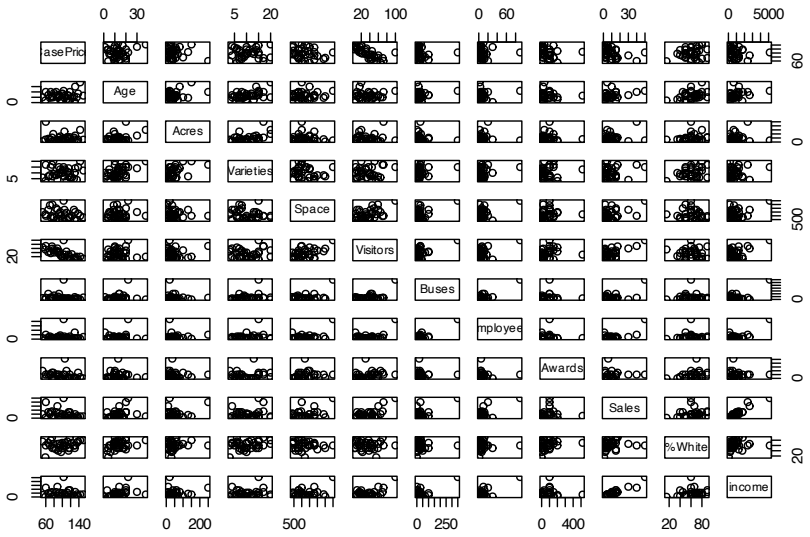

q1
q2
q3
q4
2020-01
2020-02
2020-03
2020-04
2020-05
2020-06
2020-07
2020-08
2020-09
2020-10
2020-11
2020-12
2021-01
2021-02
2021-03
2021-04
2021-05
2021-06
2021-07
2021-08
2021-09
2021-10
2021-11
nondurables
durables
income_quartile
q1
q2
q3
q4

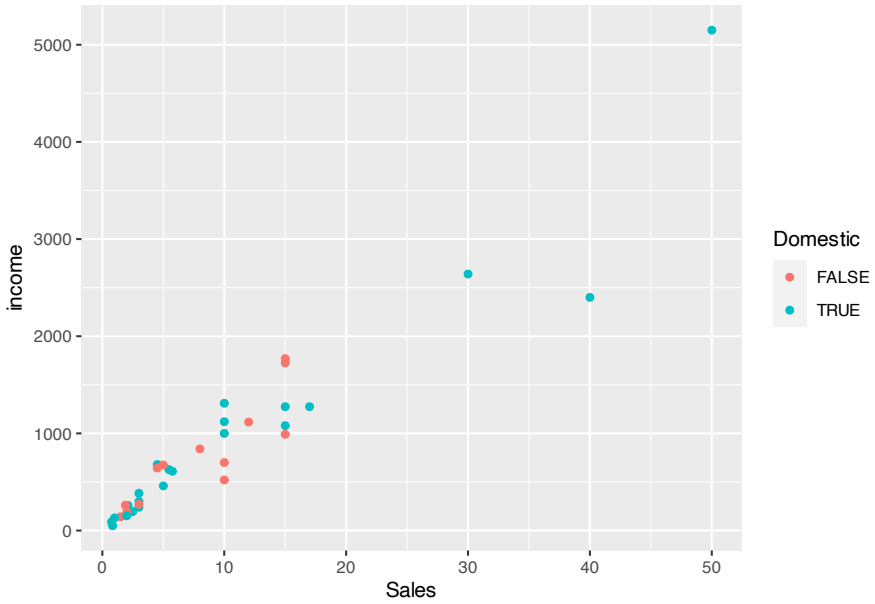


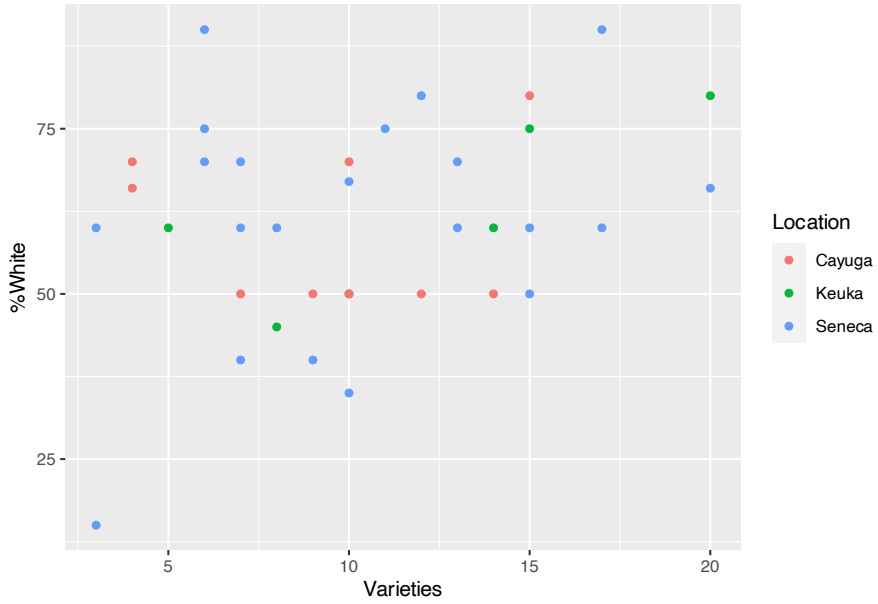
Vineyards

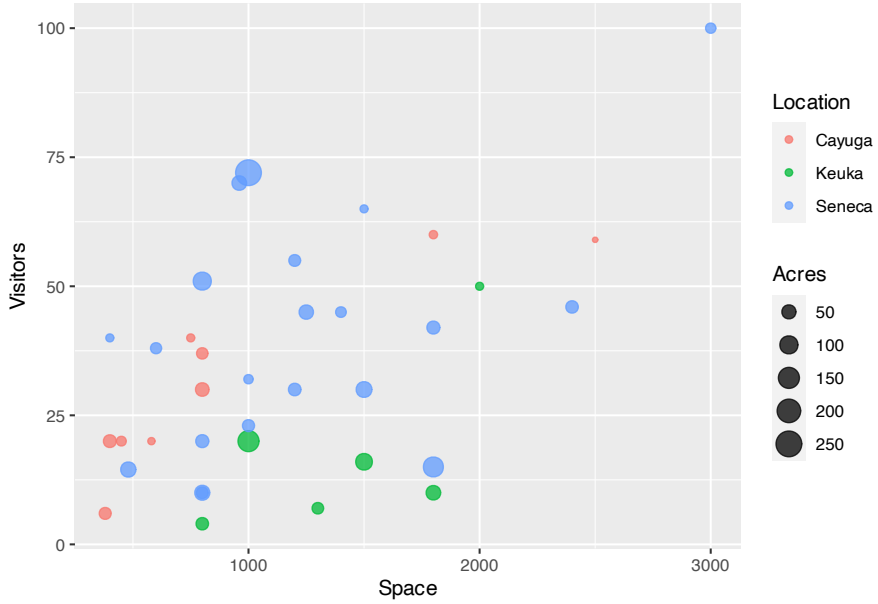
Vineyards[7]

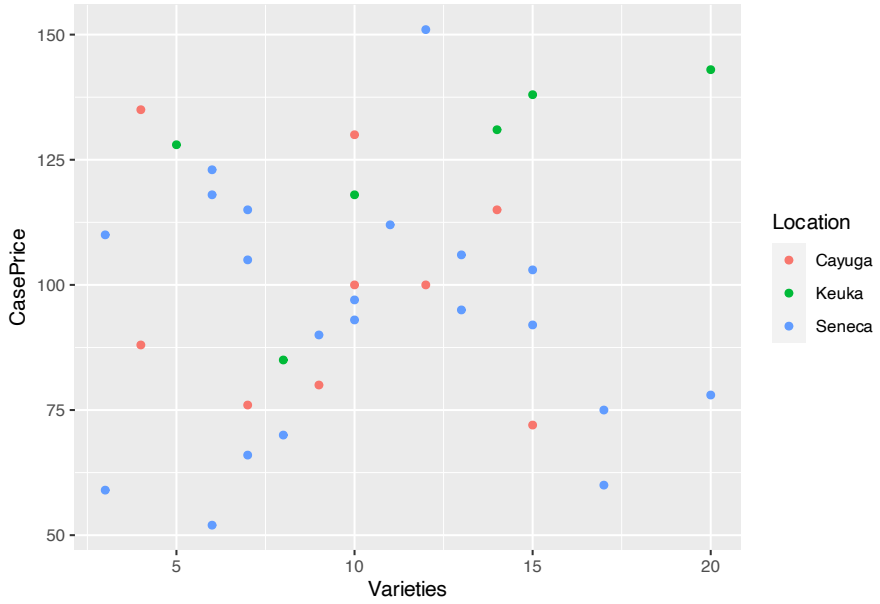
[7] https://dasl.datadescription.com/datafile/vineyards/

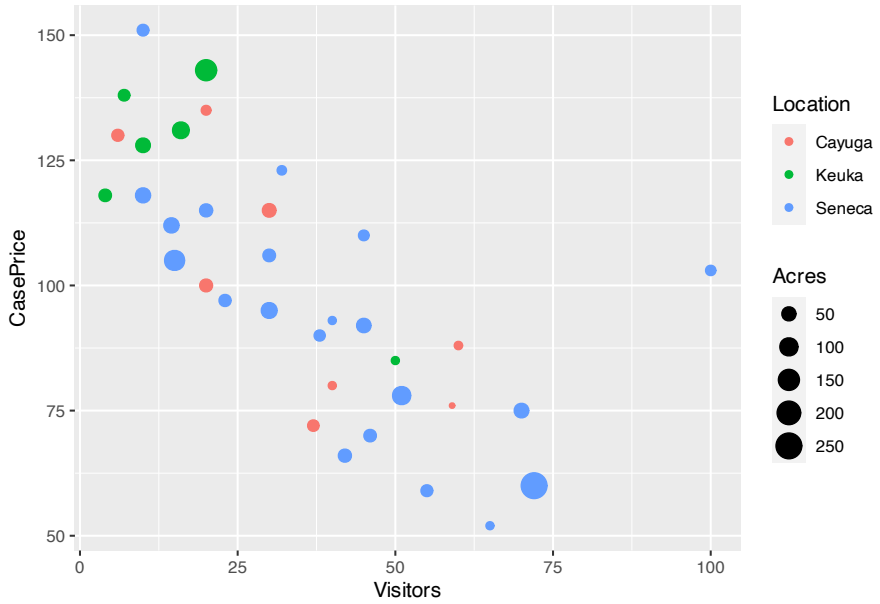

asePric
Age
Acres
Varieties
Space
Visitors
Buses
mployee
Awards
Sales
%White
income

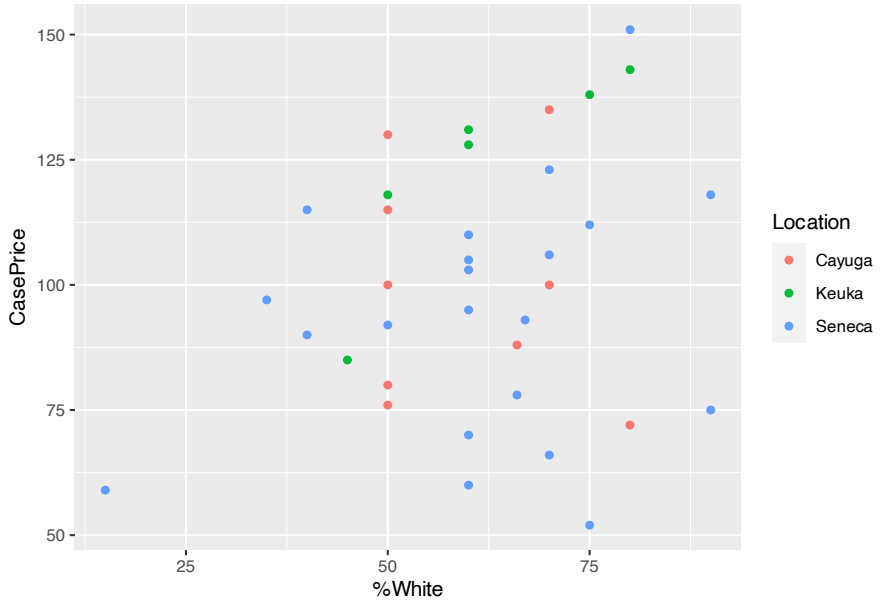

income
Sales
Domestic
FALSE
TRUE

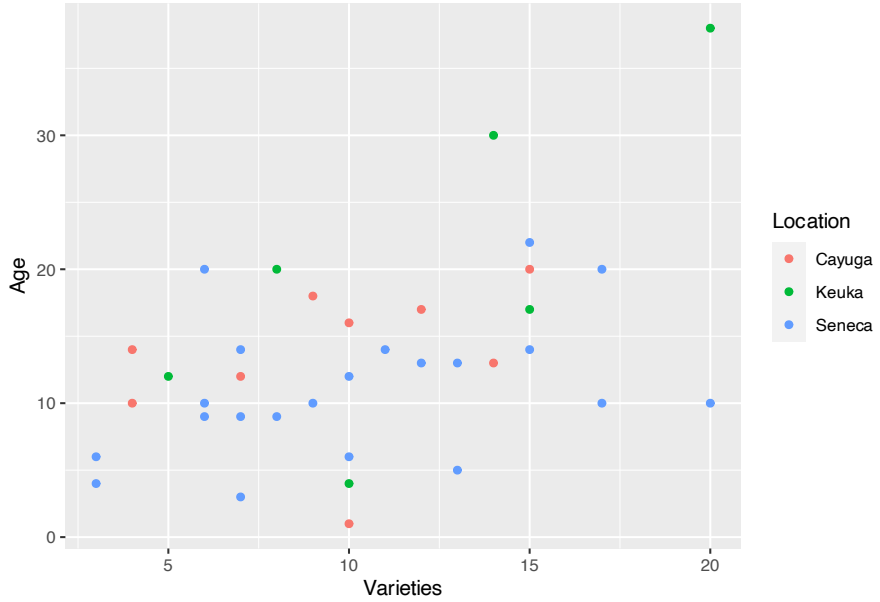

%White
75
50
25
5
10
15
20
Varieties
Location
Cayuga
Keuka
Seneca

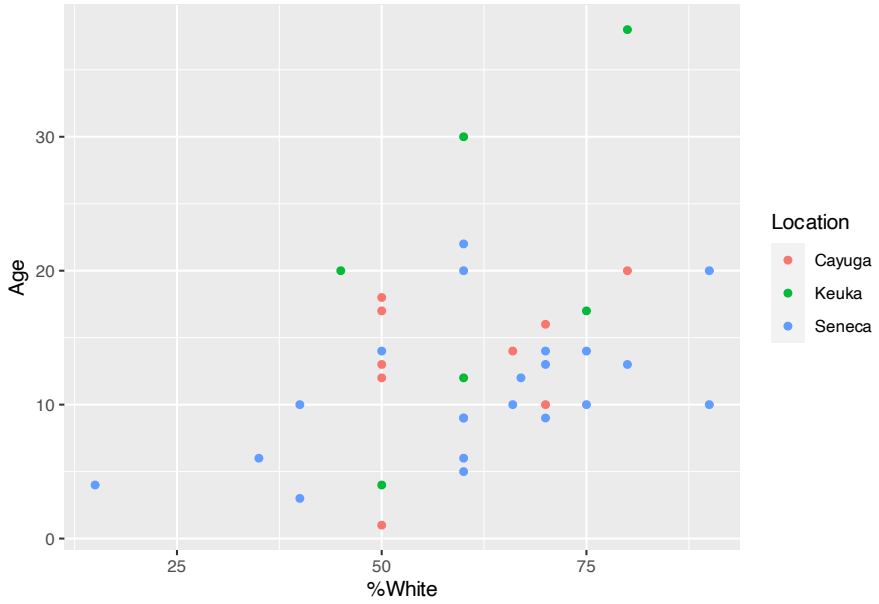

Visitors
100
75
50
25
0
1000
2000
3000
Space
Location
Cayuga
Keuka
Seneca
Acres
50
100
150
200
250

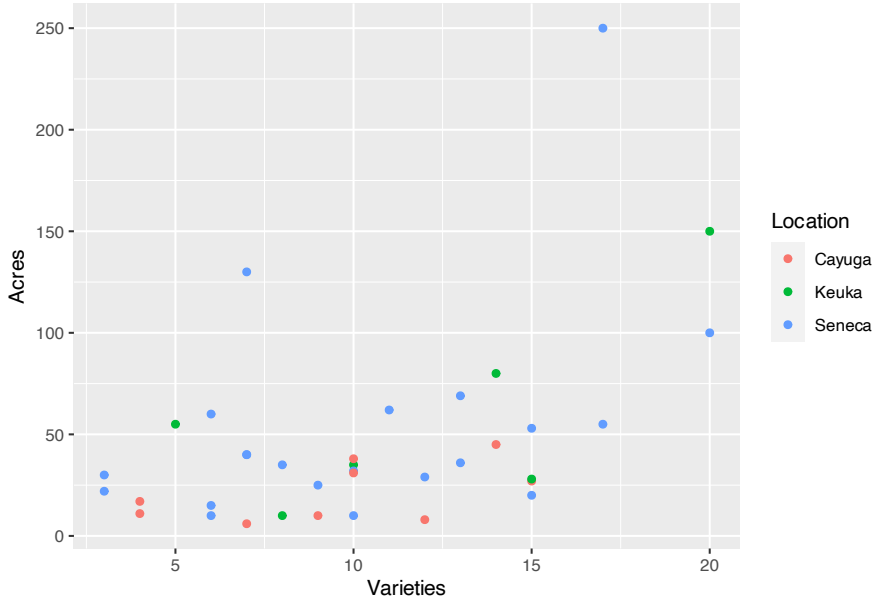

CasePrice
150
125
100
75
50
5
10
15
20
Varieties
Location
Cayuga
Keuka
Seneca

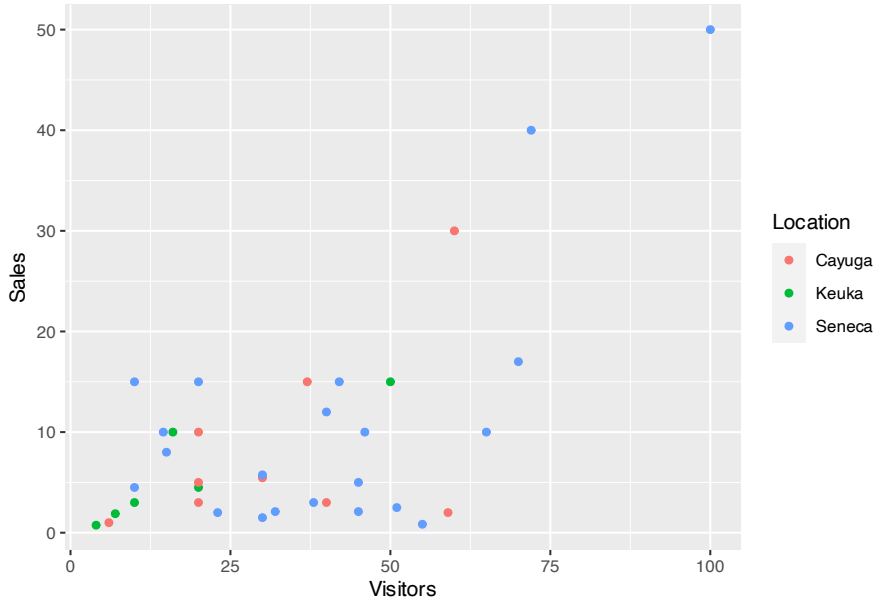

CasePrice
150
125
100
75
50
0
25
50
75
100
Visitors
Location
Cayuga
Keuka
Seneca
Acres
50
100
150
200
250

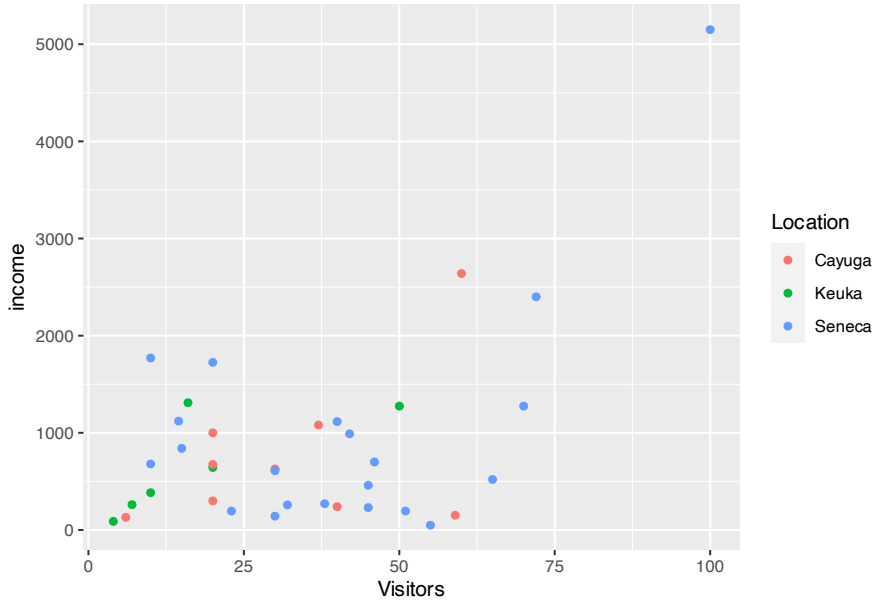

150
125
100
75
50
CasePrice
25
50
75
%White
Location
Cayuga
Keuka
Seneca

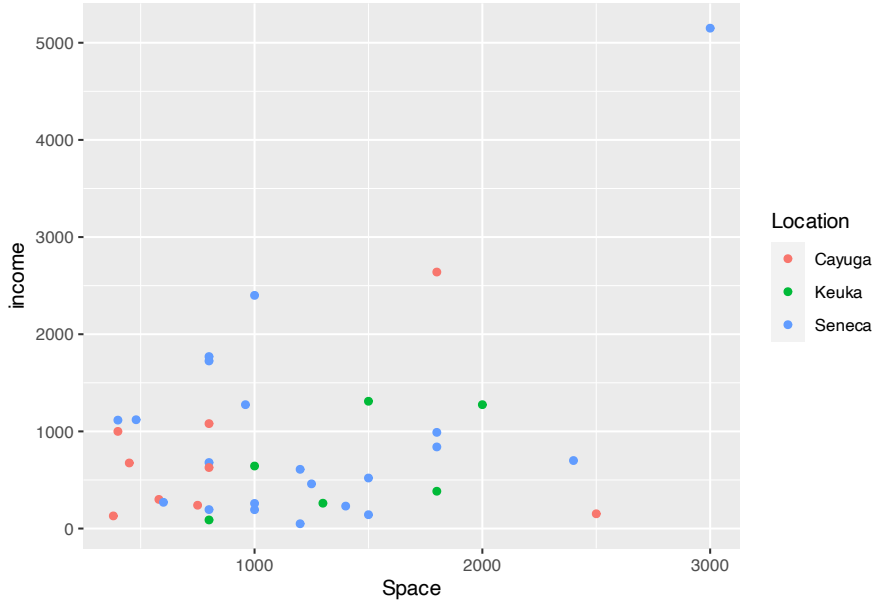

30
20
10
0
Age
5
10
15
20
Varieties
Location
Cayuga
Keuka
Seneca


Age
30
20
10
0
25
50
75
%White
Location
Cayuga
Keuka
Seneca


Acres
250
200
150
100
50
0
5
10
15
20
Varieties
Location
Cayuga
Keuka
Seneca


Sales
Visitors
0
10
20
30
40
50
0
25
50
75
100
Location
Cayuga
Keuka
Seneca


income
5000
4000
3000
2000
1000
0
0
25
50
75
100
Visitors
Location
Cayuga
Keuka
Seneca


income
5000
4000
3000
2000
1000
0
1000
2000
3000
Space
Location
Cayuga
Keuka
Seneca

Bibliography

Xie, Y. (2015). *Dynamic Documents with R and knitr*. Chapman and Hall/CRC, Boca Raton, Florida, 2nd edition. ISBN 978-1498716963.

Xie, Y. (2021). *bookdown: Authoring Books and Technical Documents with R Markdown*. R package version 0.24.

Index