ENTMLGY 6702 Entomological Techniques and Data Analysis

R Activity 3: Data Visualization

Due: 9/19/2023

1 Introduction

Data visualization is a key step in data analysis. Base `R` has some quick and easy ways to make graphs and, with lots of work and patience, one could produce a publication quality figure using just base R. In my estimation, most R users now rely on the `tidyverse`, specifically `ggplot2`, to produce high quality graphics without too much tinkering (there will always be *some* tinkering, no matter which software you use). There are several fantastic `ggplot2` tutorials online. Here are some I seem to frequent:

- https://r4ds.had.co.nz/data-visualisation.html
- https://ggplot2.tidyverse.org/
- https://www.datanovia.com/en/blog/ggplot-legend-title-position-and-labels/
- http://www.cookbook-r.com/Graphs/Legends_(ggplot2)/

Information on using "themes" in `ggplot` (more details below):

- https://ggplot2.tidyverse.org/reference/ggtheme.html

Changing the shape of plotted points (more details below):

- http://www.sthda.com/english/wiki/r-plot-pch-symbols-the-different-point-shapes-available-in-r.

Using different colors in R (more details below):

- https://www.nceas.ucsb.edu/sites/default/files/2020-04/colorPaletteCheatsheet.pdf

Given the wonderful online resources for R graphics, we will only cover the basics in class. In practice, you will rarely produce the exact same figure twice and each graph usually has some idiosyncrasy (e.g., changing a trend line from solid to dashed, adding text for your summary statistics, italicizing one word in an axis title, etc). Understanding the logic of `ggplot2` and Googling can help you produce *a lot* of different graphs.

2 Graphing windows

One goal of this course is to enable you to produce publication quality figures. Part of that process is rendering the figure in the exact size it will appear in print. To create a plotting window outside of R studio of say, 3 inches wide by 2 inches tall, you would run:

```
windows(width=3,height=2)
```

This is advantageous because journals typically provide the preferred figure dimensions in their author guidelines and with the above code you can create a plot in those exact dimensions. Doing so circumvents scaling issues that often arise when submitting a huge figure that the journal downsizes for print. In the plotting window we just activated, you can save the figure as a PDF, which is typically more than adequate for publication.

Oh... if you are a Mac user, I *think* the following will open up a plotting window.

```
quartz(title="",width=3,height=2)
```

3 Base R graphics

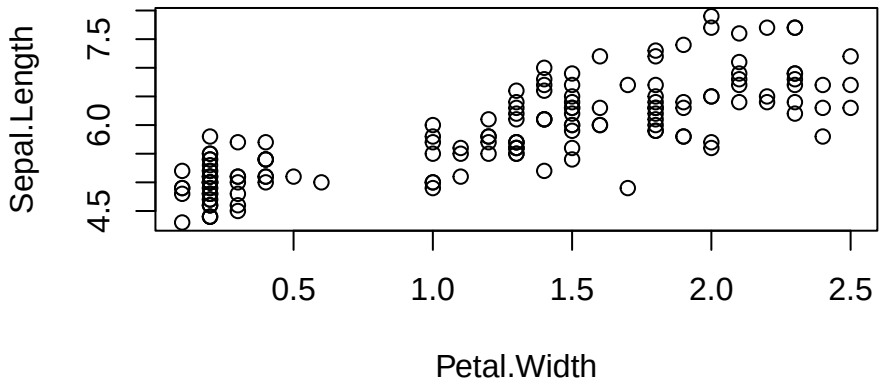
We are going to continue working with the Iris data for this example. We could load it in again for practice. . . or you could install the package `datasets` and then load that package into R. The `datasets` package has several practice data sets. . . including `iris`.

```
library(datasets)
head(iris)
```

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
## 1          5.1         3.5          1.4         0.2  setosa
## 2          4.9         3.0          1.4         0.2  setosa
## 3          4.7         3.2          1.3         0.2  setosa
## 4          4.6         3.1          1.5         0.2  setosa
## 5          5.0         3.6          1.4         0.2  setosa
## 6          5.4         3.9          1.7         0.4  setosa
```

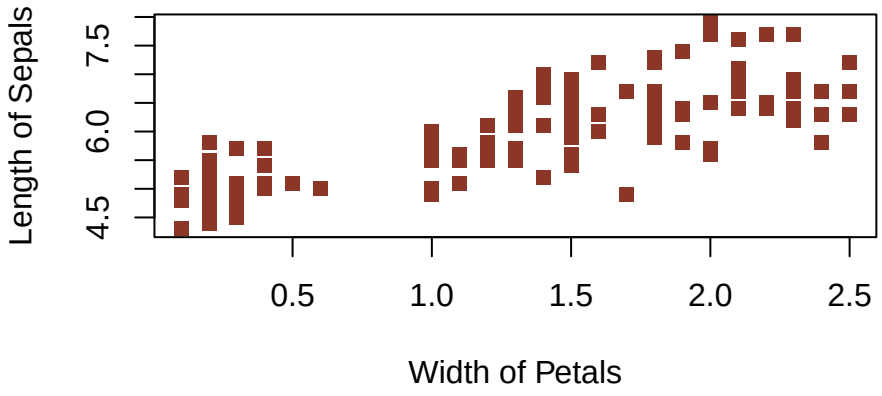
You have already made a few plots using base R. Here is an example of a scatterplot of `Sepal.Length` on the y-axis and `Petal.Width` on the x-axis (which I will often write as "`Sepal.Length` as a function of `Petal.Width`").

```
plot(Sepal.Length~Petal.Width, data=iris)
```



We can change the type of points using `pch`, the color of points using `col`, and axis labels (`xlab`,`ylab`) all in the `plot()` function.

```
plot(Sepal.Length~Petal.Width, data=iris, pch=15, col="tomato4", xlab="Width of Petals",
     ylab="Length of Sepals")
```



4 ggplot2

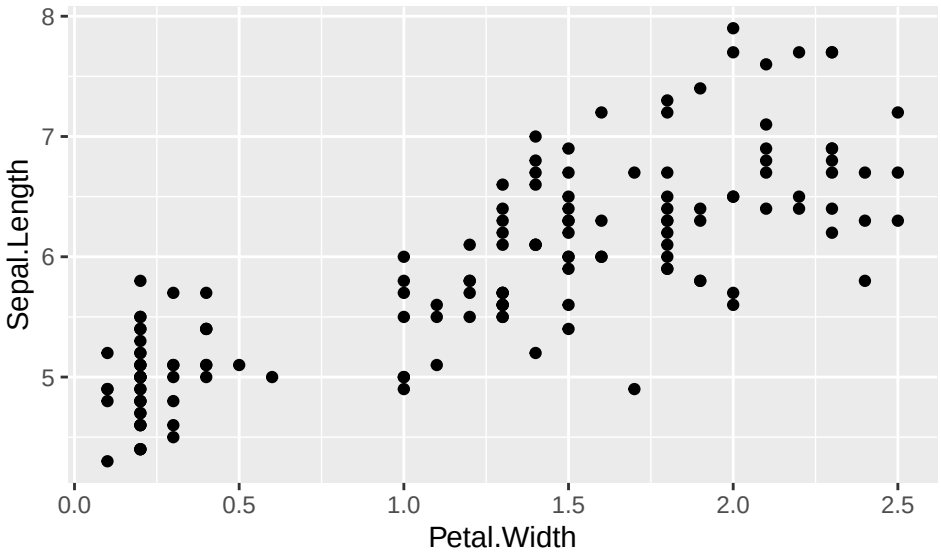
Imagine we wanted to do something more complex, like render points belonging to each `Species` a unique color. That would require substantial effort using base graphics. It would also require a lot more code to remove the box around the figure, adjust the length of tick marks, change the location of axis labels, etc. to make a "pretty" graph. `ggplot2` has been developed to make pretty graphs, so you have to worry less about aesthetics and can focus on the message you are trying to communicate in the figure. Loading in `tidyverse` loads in `ggplot2`. You could also load in the `ggplot2` package on its own.

```
library("tidyverse")
library("ggplot2")
```

4.1 ggplot2: scatterplots

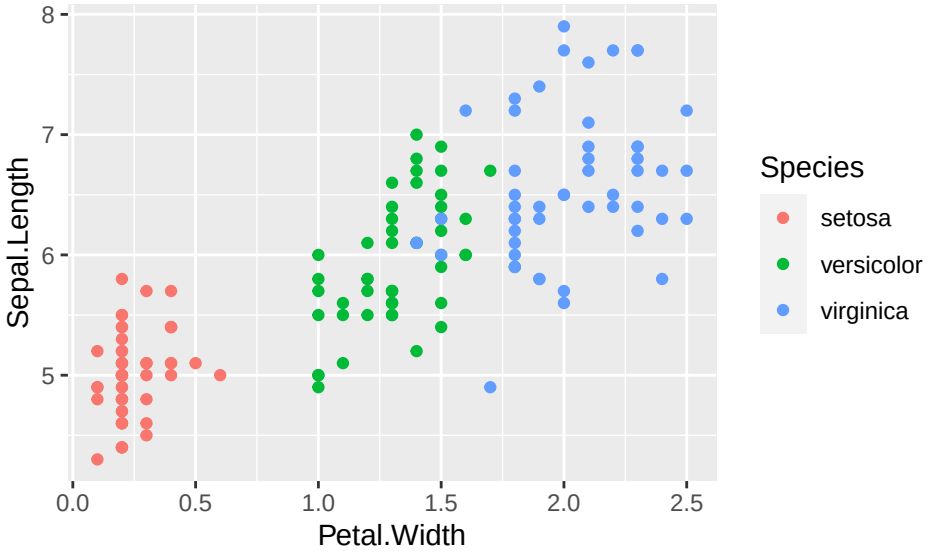
If we want to make the same basic scatterplot as above using `ggplot2`, we would use the `ggplot()` command. Notice instead of pipe operators, we are adding components to our graph using `+` signs.

```
ggplot(data=iris, mapping=aes(x=Petal.Width, y=Sepal.Length))+
  geom_point()
```



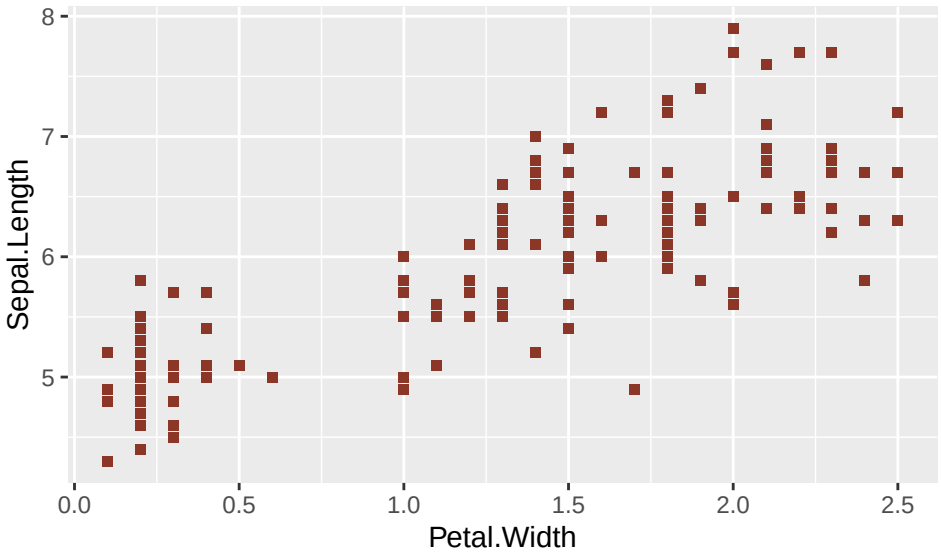
That isn't exactly pretty. But we can change certain aspects very quickly. We will just add `col=Species` to the `aes()` argument (aka the aesthetics argument).

```
ggplot(data=iris, mapping=aes(x=Petal.Width, y=Sepal.Length, col=Species))+
  geom_point()
```



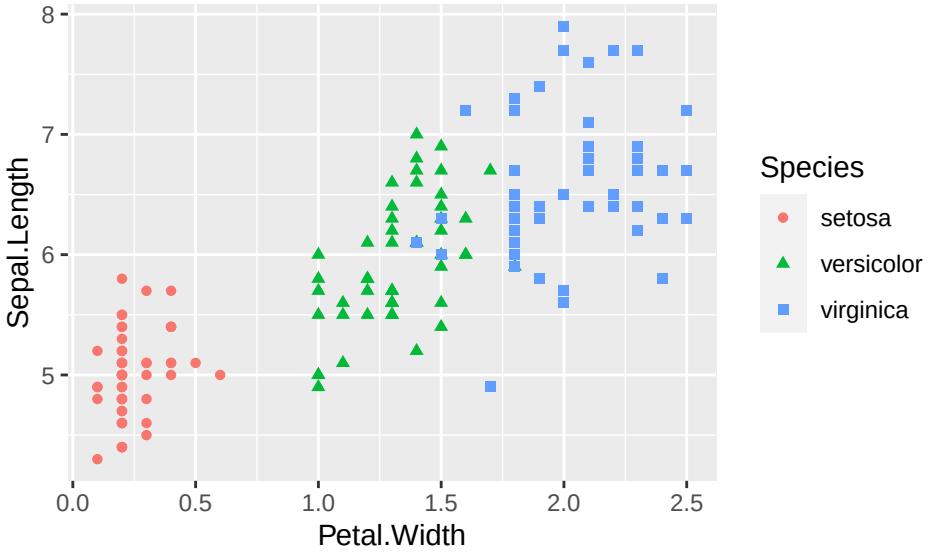
You will notice a legend was produced automatically. In base R, we would have needed to add the legend manually and specified each component, such as telling R which `Species` is represented by which color. We can also change the shape and color of all points. Note that the `shape` and `col` arguments have moved from the `aes()` argument (which customizes based on categories - like `Species` - in your data) to the `geom_point()` argument (which customizes "globally").

```
ggplot(iris, mapping=aes(x=Petal.Width, y=Sepal.Length))+
 geom_point(shape=15, col="tomato4")
```



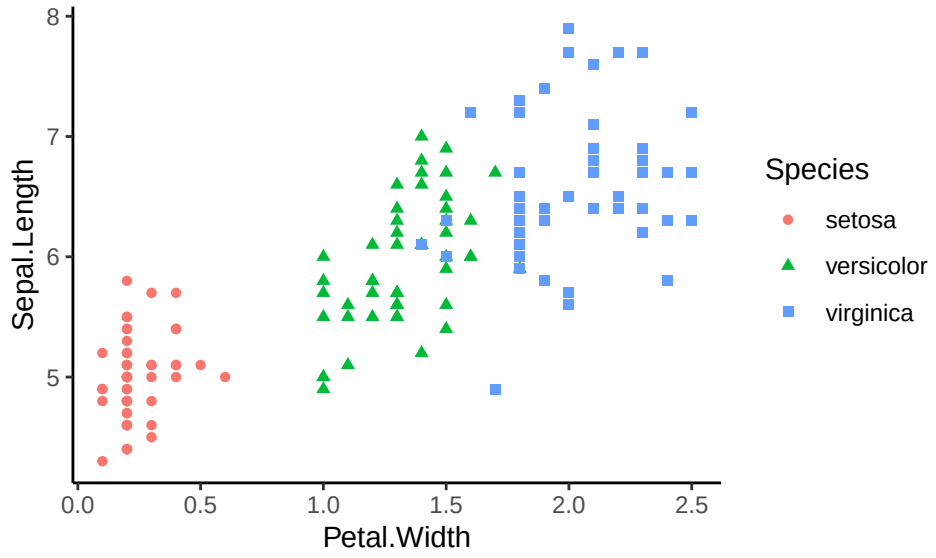
We can also change the shape of points by `Species`. Notice the `shape` argument has been placed in the `aes()` function this time around and not in the `geom_point()` argument as we did in the previous example. I think of `mapping` as R mapping aspects of my data to the graph: `shape` is now set equal to `Species` - a column in my data frame - rather than a number or symbol.

```
ggplot(iris, mapping=aes(x=Petal.Width, y=Sepal.Length, col=Species, shape=Species))+
   geom_point()
```



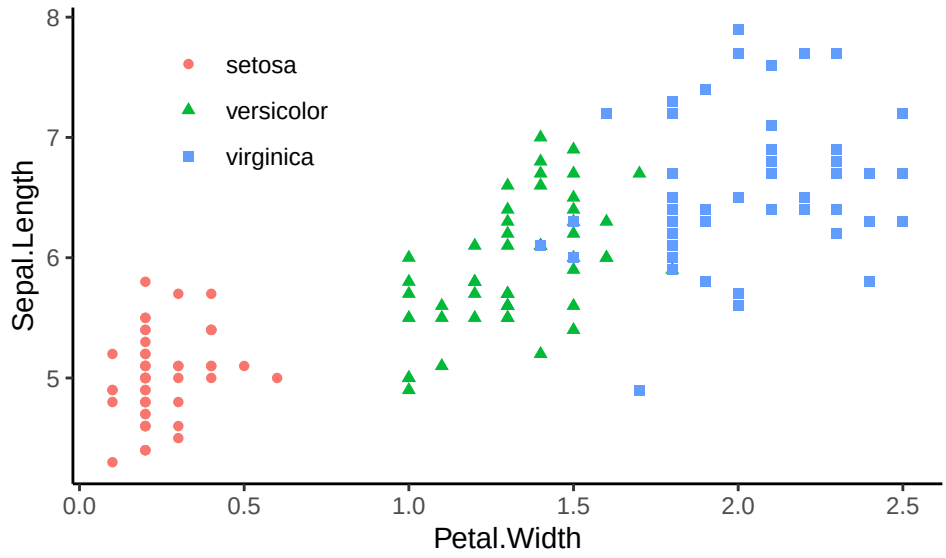
There are lots of ways to change aspects of the overall appearance of your `ggplot()` graphs - many of them are achieved by the use of `themes()`. You can add a built-in theme, such as `theme_classic()`:

```
ggplot(iris, mapping=aes(x=Petal.Width, y=Sepal.Length, col=Species, shape=Species))+
   geom_point()+
    theme_classic()
```



And you can change the theme with additional `theme()` arguments added to the `ggplot()` graph. Here we will move the legend onto the graph and remove the legend title. Notice the `legend.position` coordinates are between 0 and 1 and do not reflect the actual x-y coordinates on the graph.

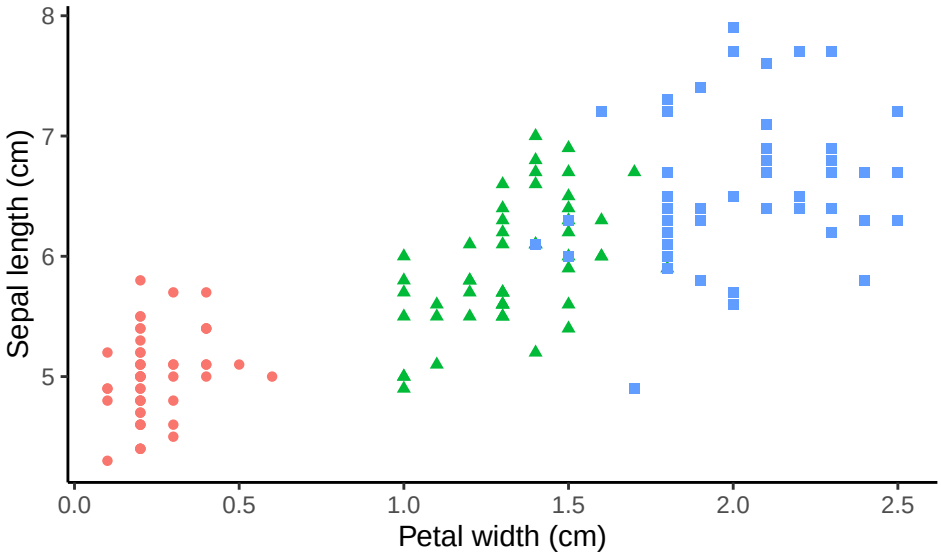
```
ggplot(iris, mapping=aes(x=Petal.Width, y=Sepal.Length, col=Species, shape=Species))+
    geom_point()+
    theme_classic()+
    theme(legend.position = c(0.2,0.8), legend.title = element_blank())
```



Here we will remove the legend and change the axis labels to be a bit more informative.

```
ggplot(iris, mapping=aes(x=Petal.Width, y=Sepal.Length, col=Species, shape=Species))+
    geom_point()+
    theme_classic()+
    xlab("Petal width (cm)")+
    ylab("Sepal length (cm)")+
```

```
    theme(legend.position = "none")
```

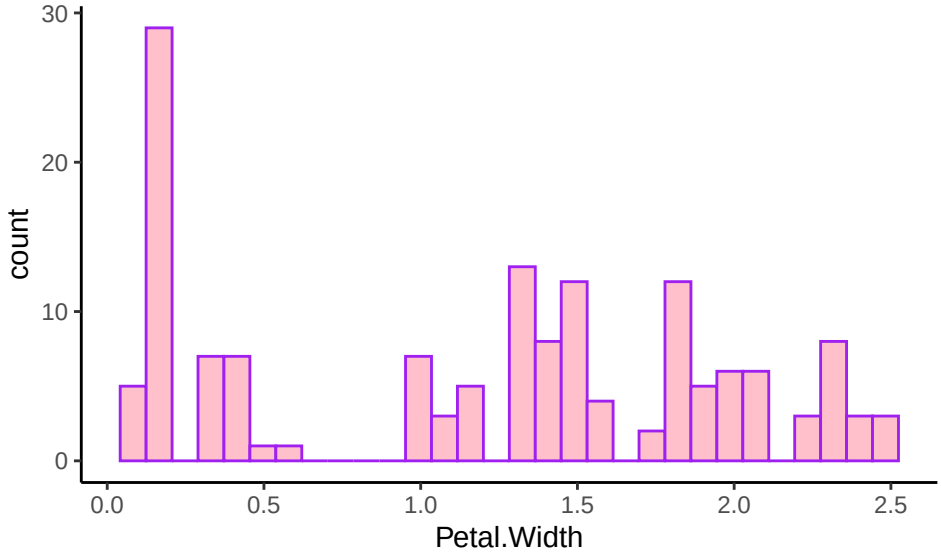


I would keep the legend, but otherwise the above graph (to me) looks ready to go. I typically use base graphics during the data exploration stage. When I am ready to develop a figure intended for peer review, I switch to `ggplot2`, create a plotting window with the correct dimensions for my target journal, and get to tinkering.

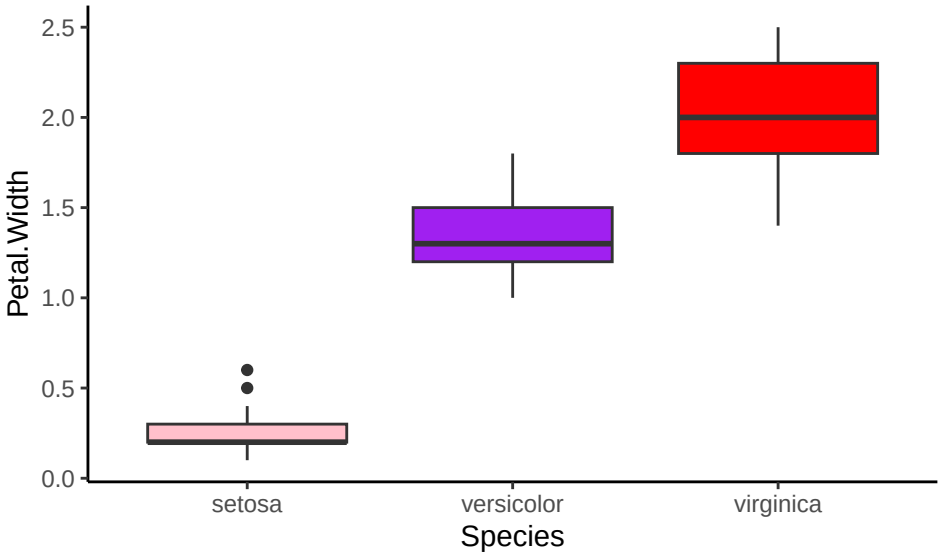
4.2 ggplot2: other plots

`ggplot()` can produce a variety of graphics (including maps). Given there are books written on the subject, an exhaustive list of examples is beyond the scope of this tutorial (and course); I provide an example of a histogram and boxplot. Reading the primary literature is a useful way to see how other scientists are presenting data similar to yours. Once you have the plot you want to make in mind, head to Google to find the code you need!

```
ggplot(iris, mapping=aes(x=Petal.Width))+
    geom_histogram(color="purple", fill="pink")+
    theme_classic()+
    theme(legend.position = "none")
```




```
ggplot(iris, mapping=aes(x=Species, y=Petal.Width, fill=Species))+
    geom_boxplot()+
    theme_classic()+
    scale_fill_manual(values = c("pink", "purple", "red"))+
    theme(legend.position = "none")
```



5 R Activity

All the data files you need for this activity are provided on GitHub. For some of these questions, you might need to head to Google or one of the tutorials linked above. I cannot stress this enough: knowing how to Google the question you are trying to answer in a meaningful way will help you solve most coding problems.

1. Download the `cherrytree` data set from Canvas and load it into R using `read.table()`. Note this file is a `.csv` file (comma separated values).
2. Make a scatterplot of `Height` as a function of `Girth` using `ggplot2`. Use the theme `theme_bw()` and change the black circles to orange, hollow triangles.
3. Make a boxplot of `Height` and as a function of `Variety` using `ggplot2`. Change the x-axis label to "`Prunus variety`" and the y-axis label to "`Height (feet)`". Use the theme `theme_classic()` and fill the variety A box as `dark blue` and the variety B box as `yellow`.
4. Make the same plot as the previous question, but make it a violin plot instead of a box plot and remove the legend.