Data visualization using ggplot

Prepared by Dr. Jose Isagani Janairo

2023

ggplot2

In the past activity, you learned how to create plots using the built-in R base graphics. While the R base graphics is simple and gets the job done, it lacks in customization and visual aesthetics. The ggplot2 package addresses these gaps, wherein this package enables the creation of beautiful visualizations with just a few lines of code. In this activity, we will learn how to create plots using the ggplot2 package. Description of each type of plot and when to use them will be discussed in the lecture. In this activity, our focus is to learn how to use ggplot2 to create visually appealing visualizations.

Install the package and load it by calling the library function. For our first plot, we will create a histogram using the rock data.

```
set.seed(1234)

library(ggplot2)
data(rock)
```

Explore your dataset.

```
summary(rock)
```

```
##       area            peri            shape              perm        
##  Min.   : 1016   Min.   : 308.6   Min.   :0.09033   Min.   :   6.30  
##  1st Qu.: 5305   1st Qu.:1414.9   1st Qu.:0.16226   1st Qu.:  76.45  
##  Median : 7487   Median :2536.2   Median :0.19886   Median : 130.50  
##  Mean   : 7188   Mean   :2682.2   Mean   :0.21811   Mean   : 415.45  
##  3rd Qu.: 8870   3rd Qu.:3989.5   3rd Qu.:0.26267   3rd Qu.: 777.50  
##  Max.   :12212   Max.   :4864.2   Max.   :0.46413   Max.   :1300.00
```

```
head(rock)
```

```
##   area    peri     shape perm
## 1 4990 2791.90 0.0903296  6.3
## 2 7002 3892.60 0.1486220  6.3
## 3 7558 3930.66 0.1833120  6.3
## 4 7352 3869.32 0.1170630  6.3
## 5 7943 3948.54 0.1224170 17.1
## 6 7979 4010.15 0.1670450 17.1
```

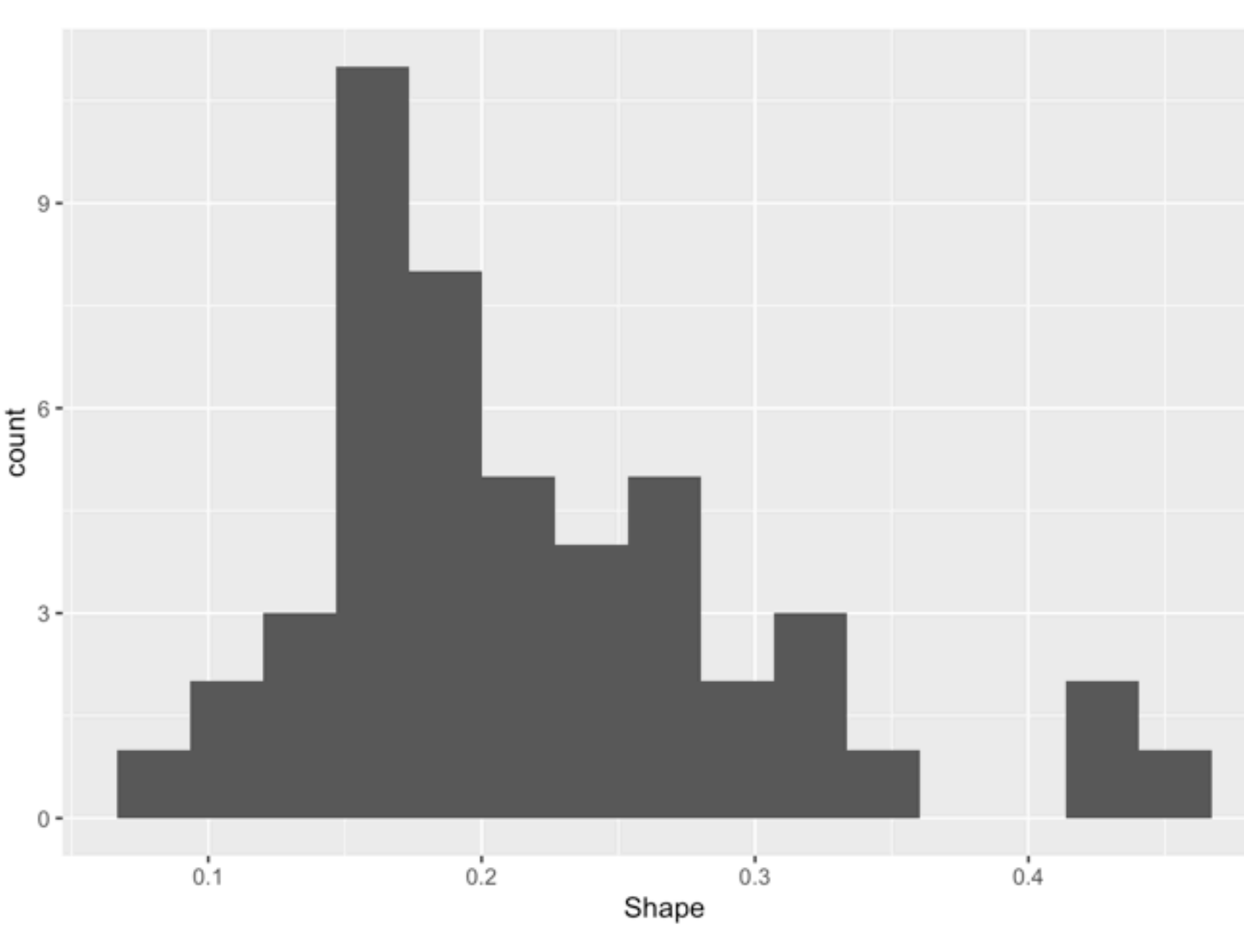
```
str(rock)
```

```
## 'data.frame':    48 obs. of  4 variables:
##  $ area : int  4990 7002 7558 7352 7943 7979 9333 8209 8393 6425 ...
##  $ peri : num  2792 3893 3931 3869 3949 ...
##  $ shape: num  0.0903 0.1486 0.1833 0.1171 0.1224 ...
##  $ perm : num  6.3 6.3 6.3 6.3 17.1 17.1 17.1 17.1 119 119 ...
```

Now that we have an idea on the contents and nature of our data, we can proceed and visualize it. First, we are going to create a histogram.

Histogram

```
ggplot(data = rock, (aes(x = shape))) + geom_histogram(bins = 15) +labs(x = "Shape")
```



Carefully study the line of code used to produce the histogram.

The **ggplot** function is used to create the plot. Here are the essential elements of the ggplot2 syntax:

Data: The dataset you want to visualize. It's the foundation of your plot and contains the variables you'll use for aesthetics and mapping. **Aesthetics (aes):** Aesthetic mappings define how variables in the dataset are visually represented in the plot. Aesthetics include things like position (x and y axes), color, shape, size, and transparency. **Geometries (geom):** Geometries are the visual elements used to represent the data points. Different geoms correspond to different types of plots, such as points for scatter plots, lines for line charts, bars for bar charts, and more.

There are other miscellaneous elements that enrich the visualizations, such as themes. We'll learn more of that later on. For now, the **data**, **aesthetics**, and **geometries** are the critical elements we need to create a plot.

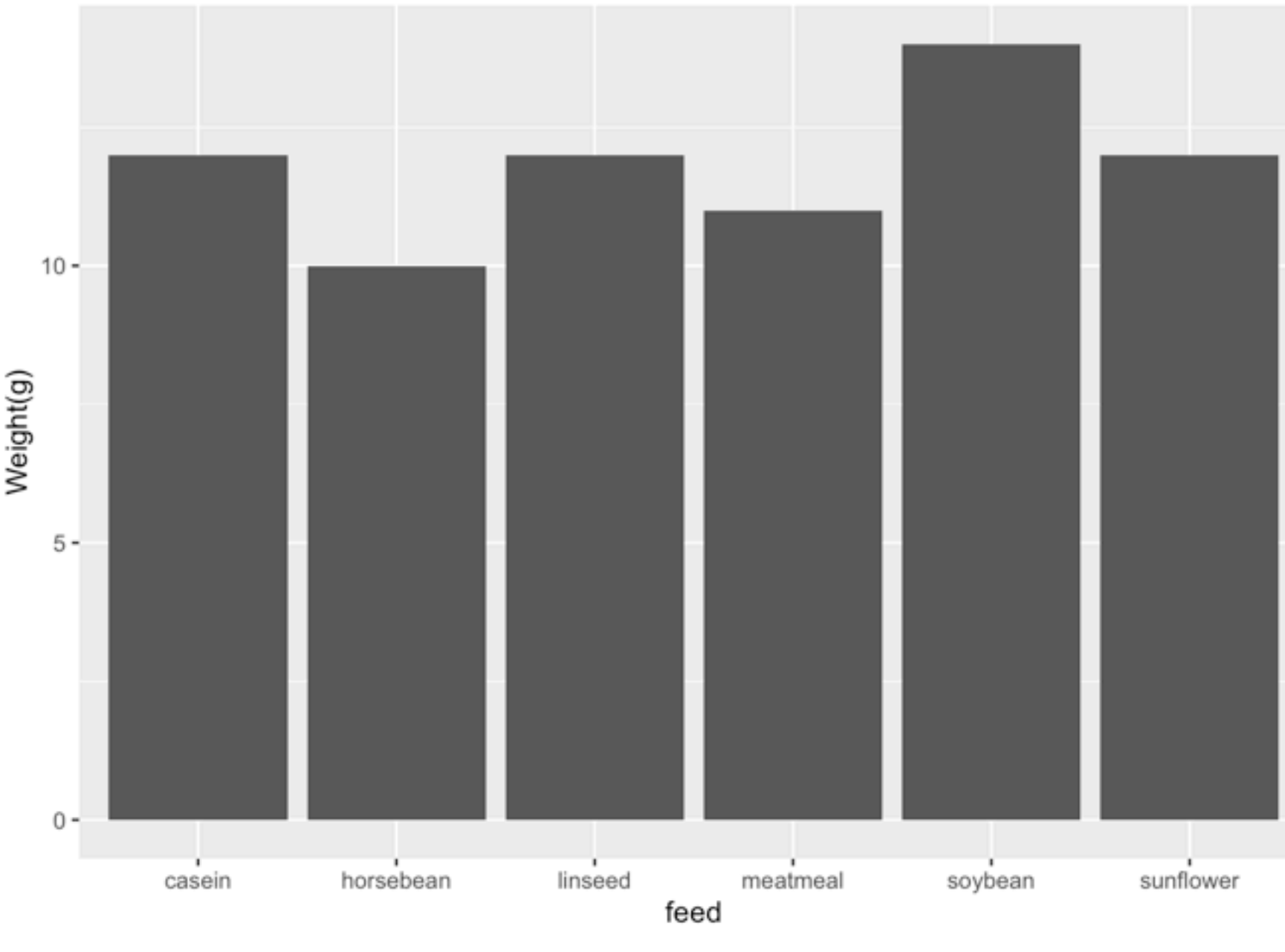
Did you notice how well-structured and systematic the syntax is? This is because ggplot2 is built on the principle of **Grammar of Graphics**, which is a systematic framework for creating and understanding data visualizations.

Going back to our histogram, can you identify which part of the syntax specified that we will be creating a histogram? How about specifying that we will create a histogram for the variable shape? **Hands-on Output 1 (HOO1): Using the same dataset, create a histogram for the variable area, and set the bins to 10.**

Bar plot

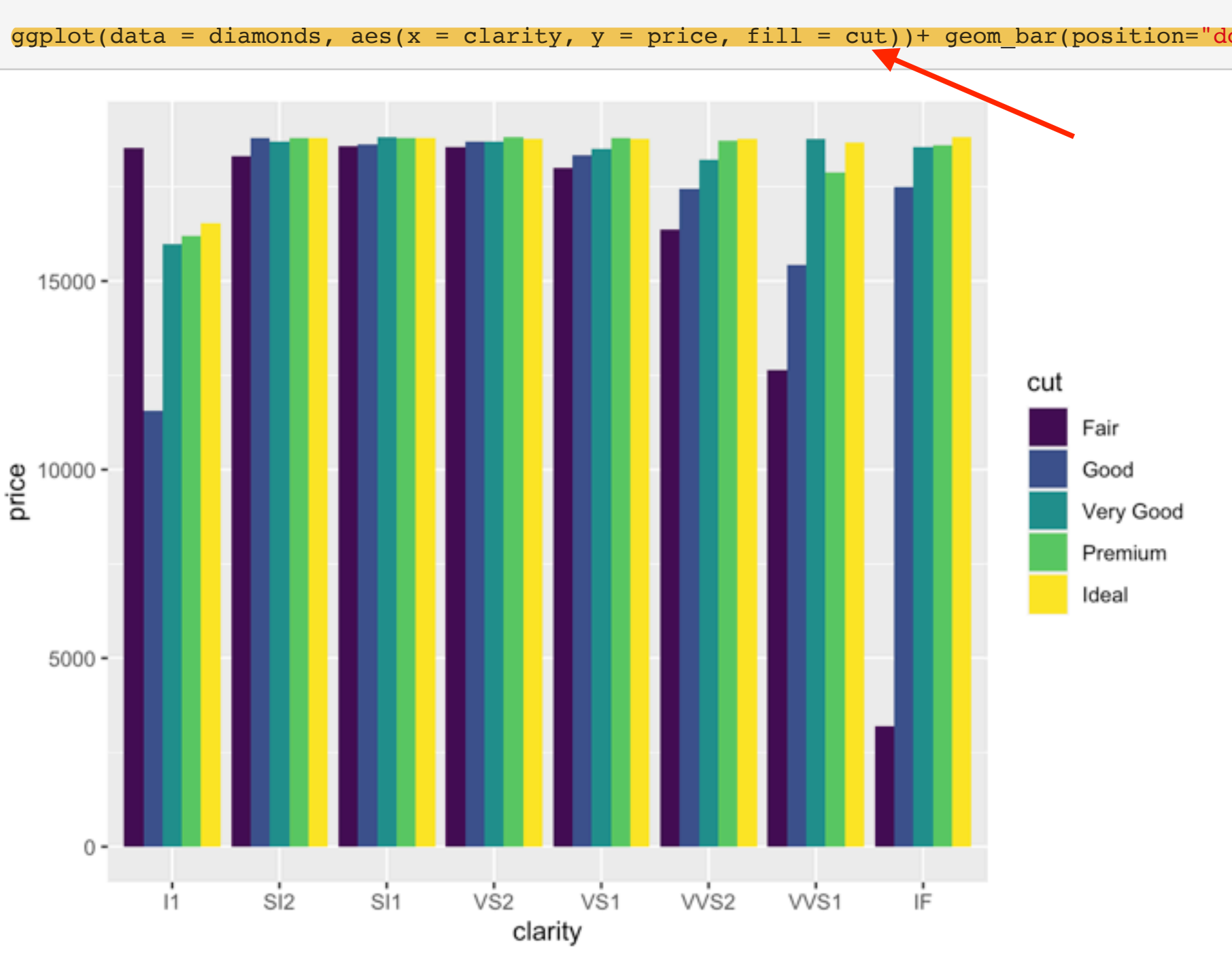
In creating bar plots, we use the same syntax but change the **geom** argument. For the bar plot, we will be using the **chickwts** dataset. Explore the dataset before you execute the plotting function.

```
data("chickwts")

ggplot(data = chickwts, aes(x = feed)) + geom_bar() + labs(y = "Weight(g)")
```



We can also create a grouped bar plot based on the factor variable. For this, we will use the **diamonds** dataset. Again, explore your dataset first before plotting.

```
data(diamonds)

ggplot(data = diamonds, aes(x = clarity, y = price, fill = cut))+ geom_bar(position="dodge", stat="identity")
```

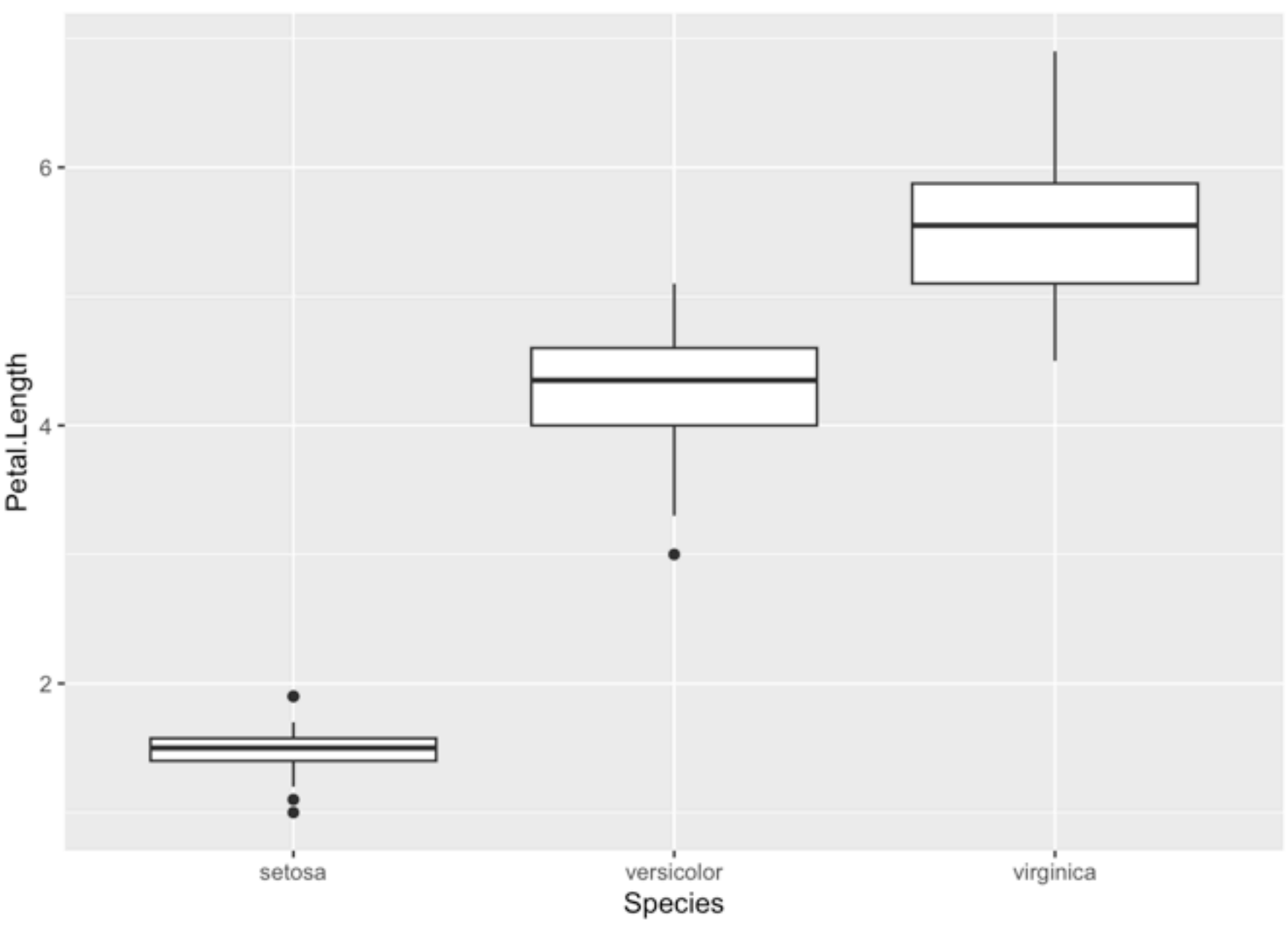


HOO2: Which variable was used as the basis of the grouping?

Boxplots

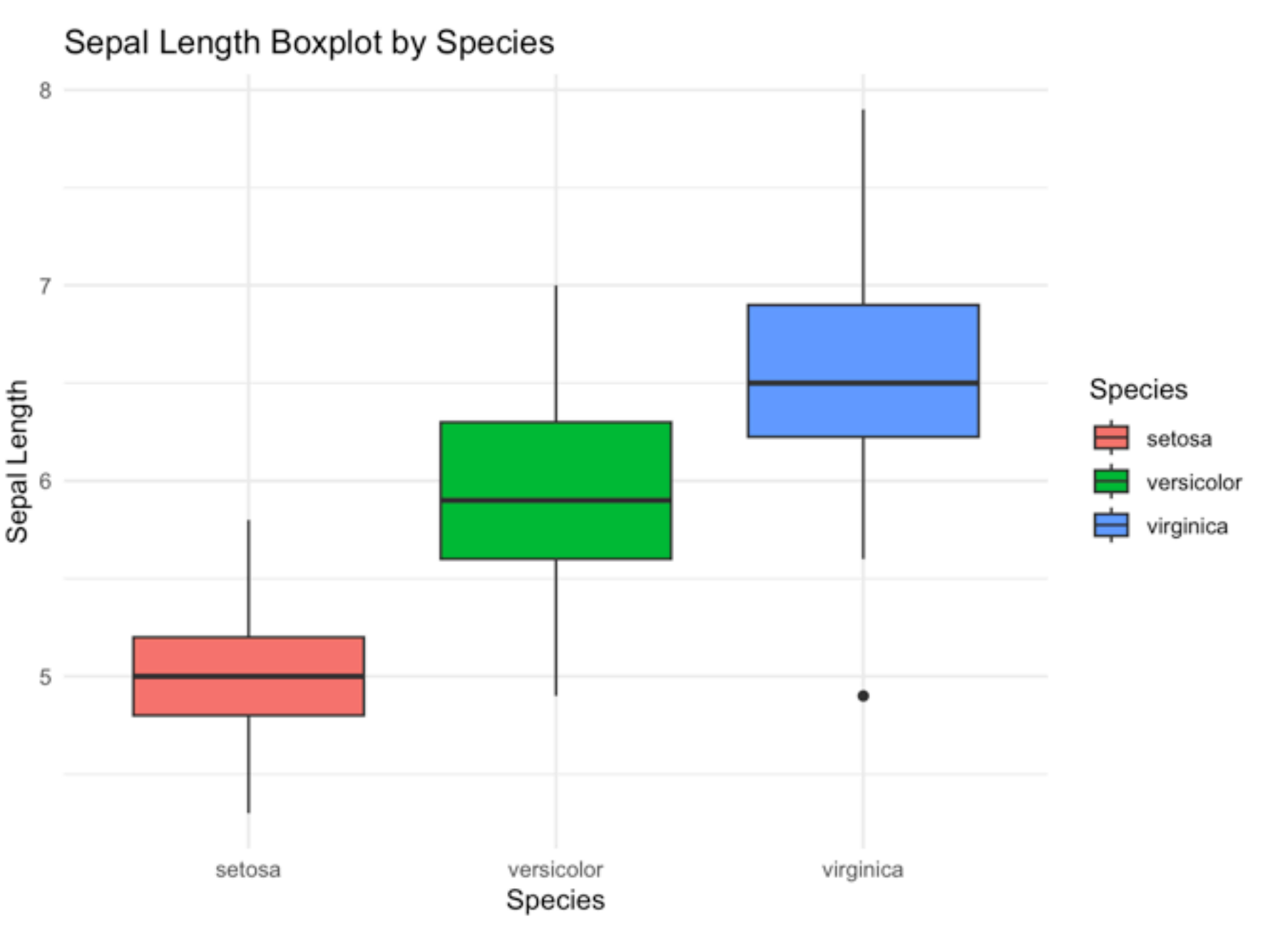
Boxplots are easy to create using ggplot2. We will use the **iris** dataset to create boxplots based on the iris species.

```
data(iris)
ggplot(data = iris, aes(x = Species, y = Petal.Length)) + geom_boxplot()
```



We can improve the aesthetics of the plot by executing the following command:

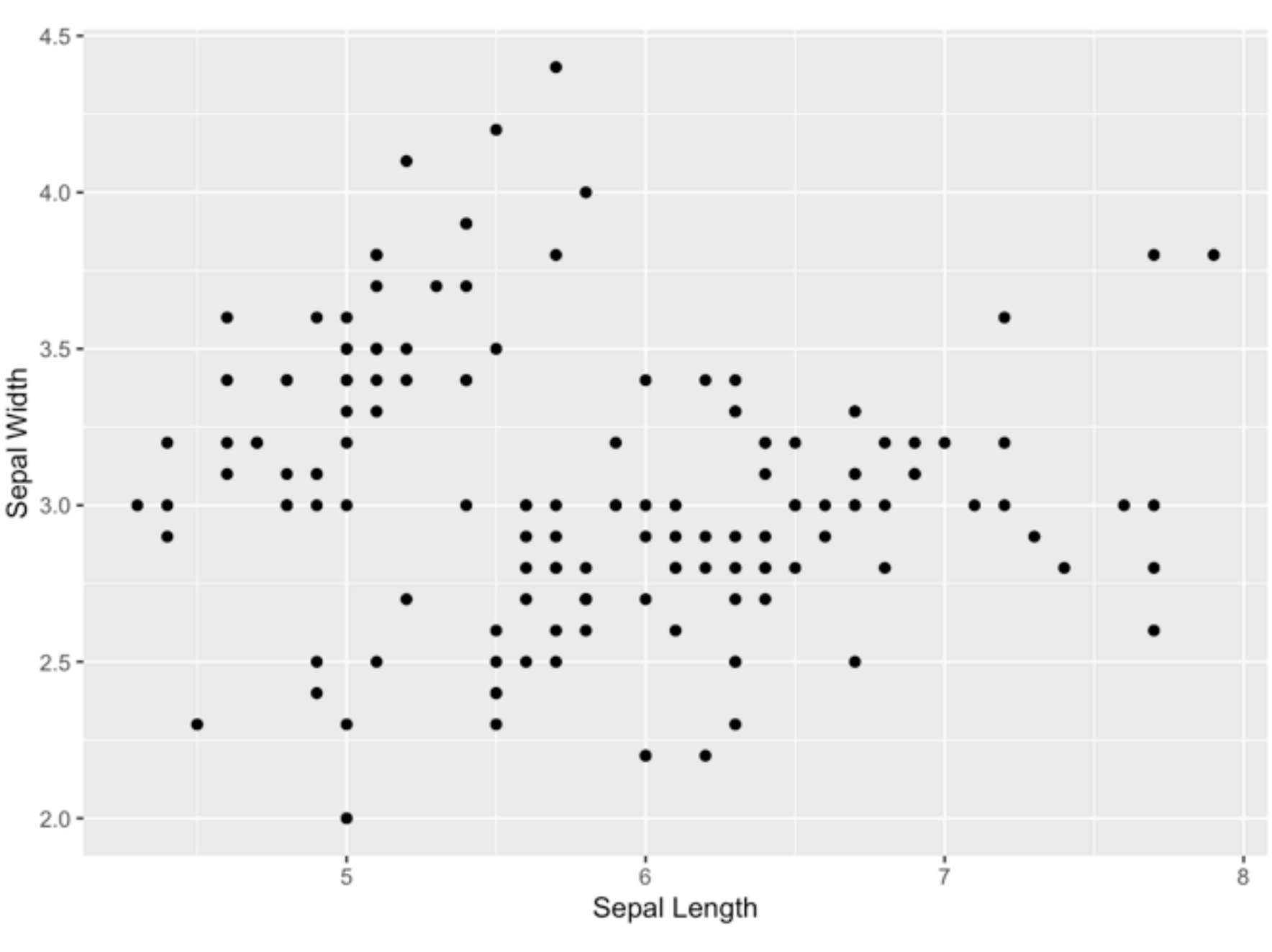
```
ggplot(iris, aes(x = Species, y = Sepal.Length, fill = Species)) +
  geom_boxplot() +
  labs(title = "Sepal Length Boxplot by Species",
       x = "Species", y = "Sepal Length") +
  theme_minimal()
```



Scatter plots

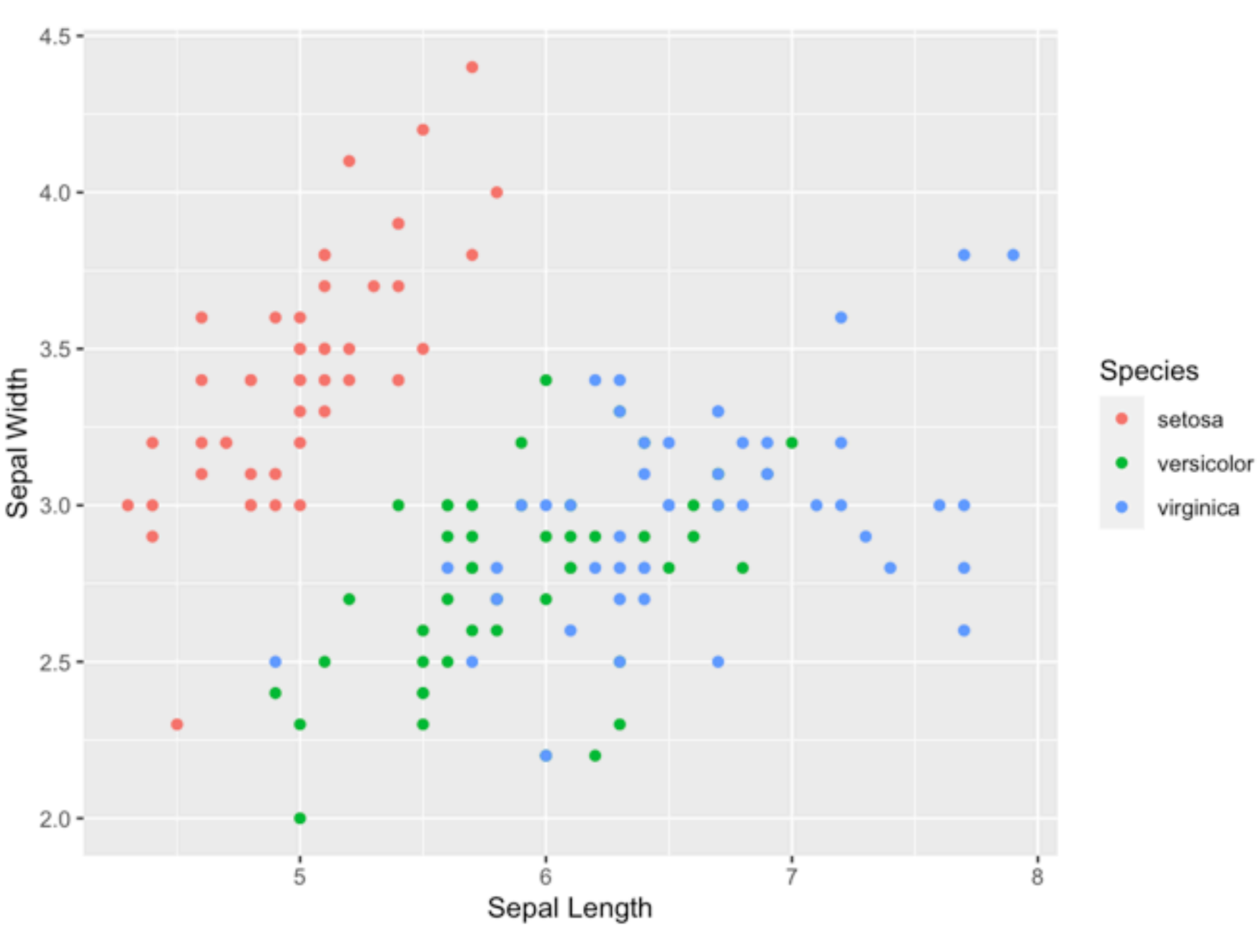
Still using the **iris** dataset, we will create and customize line plots using the *ggplot2 **package. We will use the** geom_point** argument to specify that we will be creating a scatter plot.

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



Just like what we did in the boxplots, we can color the data points based on the iris species.

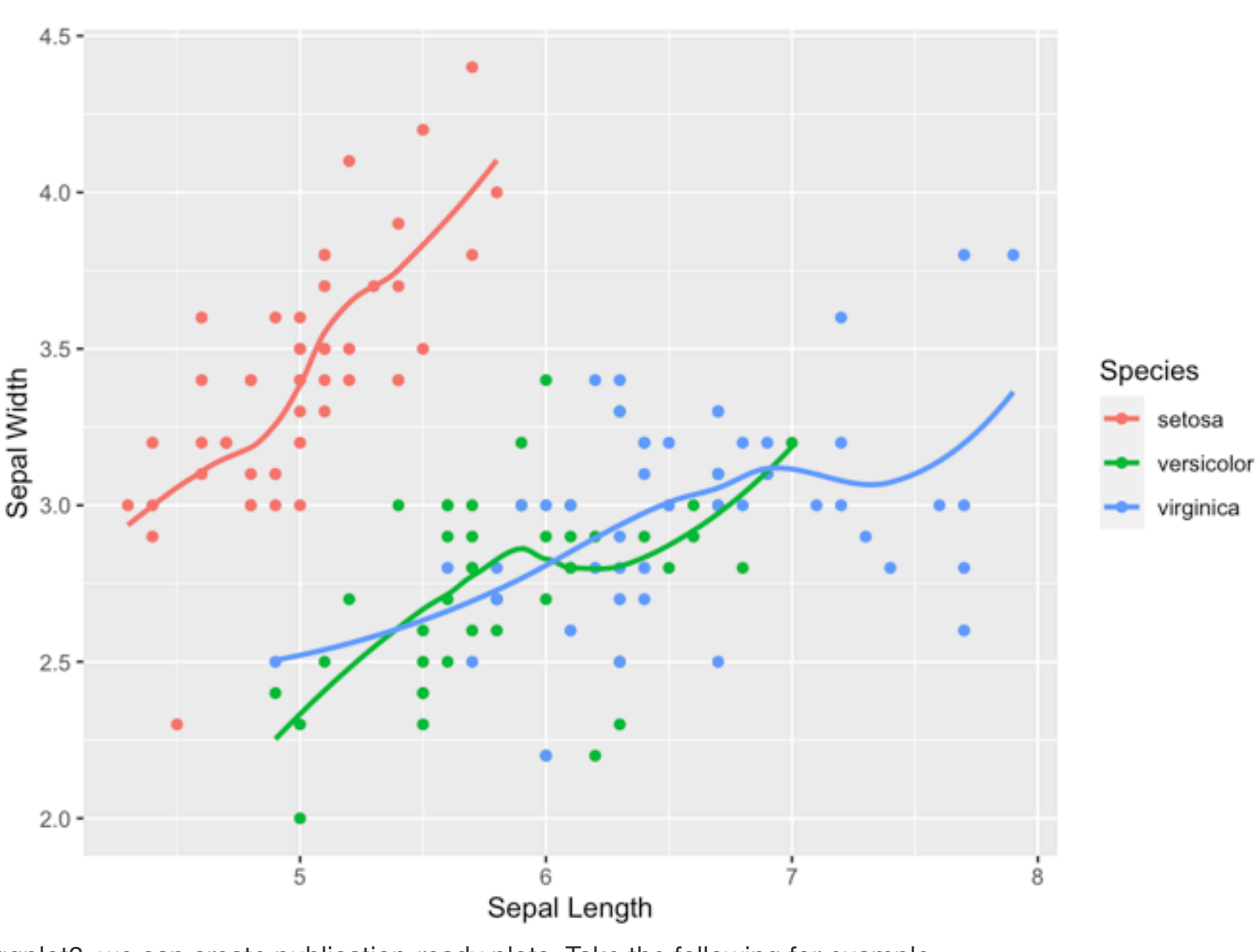
```
ggplot(data = iris, aes(x = Sepal.Length, y = Sepal.Width, color = Species)) + geom_point() + labs(x = "Sepal Length", y = "Sepal Width") + geom_point()
```



We can add a trendline by adding the **geom_smooth()** argument.

```
ggplot(data = iris, aes(x = Sepal.Length, y = Sepal.Width, color = Species)) + geom_point() + labs(x = "Sepal Length", y = "Sepal Width") + geom_point() + geom_smooth(se = F)
```

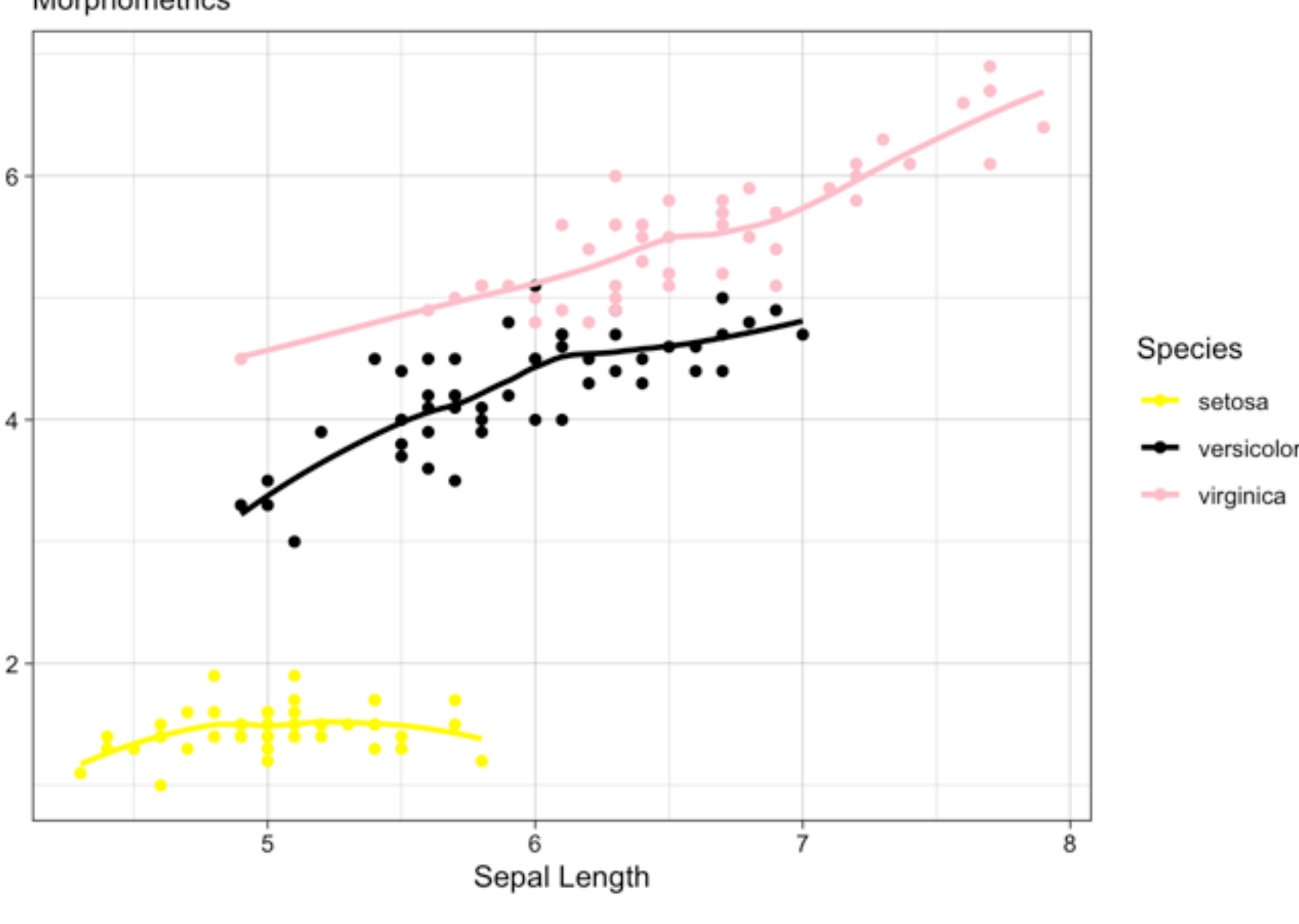
```
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
```



To fully appreciate the power of ggplot2, we can create publication-ready plots. Take the following for example.

```
ggplot(data = iris, aes(x = Sepal.Length, y = Petal.Length, color = Species)) + geom_point() + labs(x = "Sepal Length", y = "Petal Length", title = "Exploring the iris dataset", subtitle = "Morphometrics")+theme(plot.title = element_text(hjust = 1, size = 16, face = "bold.italic")) + scale_color_manual(breaks = c("setosa", "versicolor", "virginica"), values = c("yellow", "black", "pink")) + geom_smooth(se = FALSE) + theme_linedraw()
```

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



That concludes our activity introducing you to the **ggplot2** package. Refer to the package documentation and other online resources to maximize the potential of this package in data visualization.

Exercises

Create the plots which will be able to answer the following questions. Make your plots as visually appealing as possible. Refer to the ggplot2 documentation for more information on plot customization.

1. Using the **mtcars** dataset, is there a significant difference in fuel efficiency based on transmission type?
2. Using the **InsectSprays** dataset, which brand appears to be the most effective?
3. Using the **trees** dataset, which of the two variables exhibit a linear relationship?

This document is for the exclusive use of students enrolled in the course Data Science for Life Scientists at De La Salle University.