DATA VISUALIZATION WITH GGPLOT2

Scatter Plots

37 Geometries

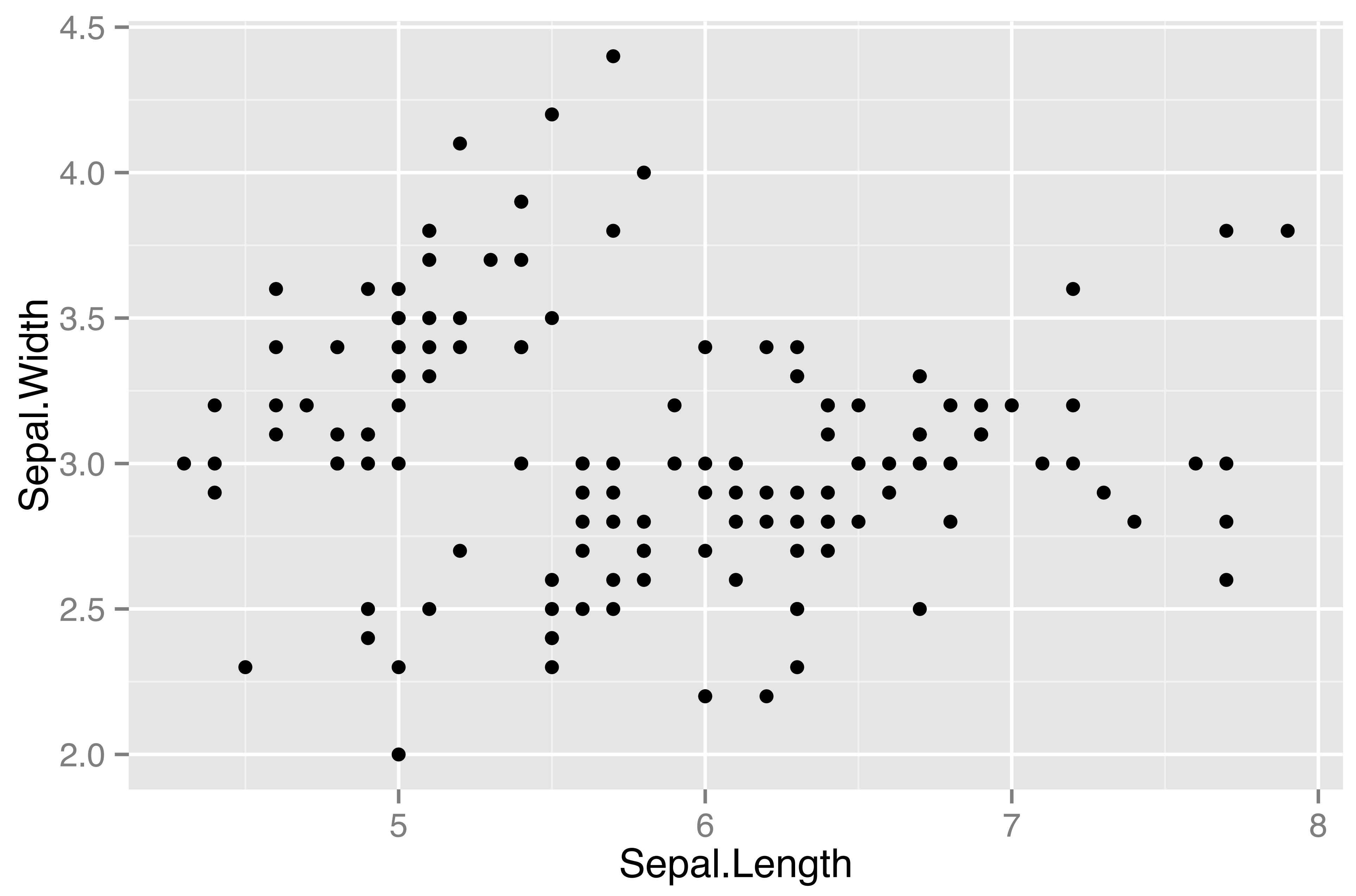
abline	density2d	line	rect	vline
area	dotplot	linerange	ribbon	
bar	errorbar	map	rug	
bin2d	errorbarh	path	segment	
blank	freqpoly	point	smooth	
boxplot	hex	pointrange	step	
contour	histogram	polygon	text	
crossbar	hline	quantile	tile	
density	jitter	raster	violin	

Common plot types

- Scatter plots
  - points, jitter, abline
- Bar plots
  - histogram, bar, errorbar
- Line plots
  - line

Scatter Plot

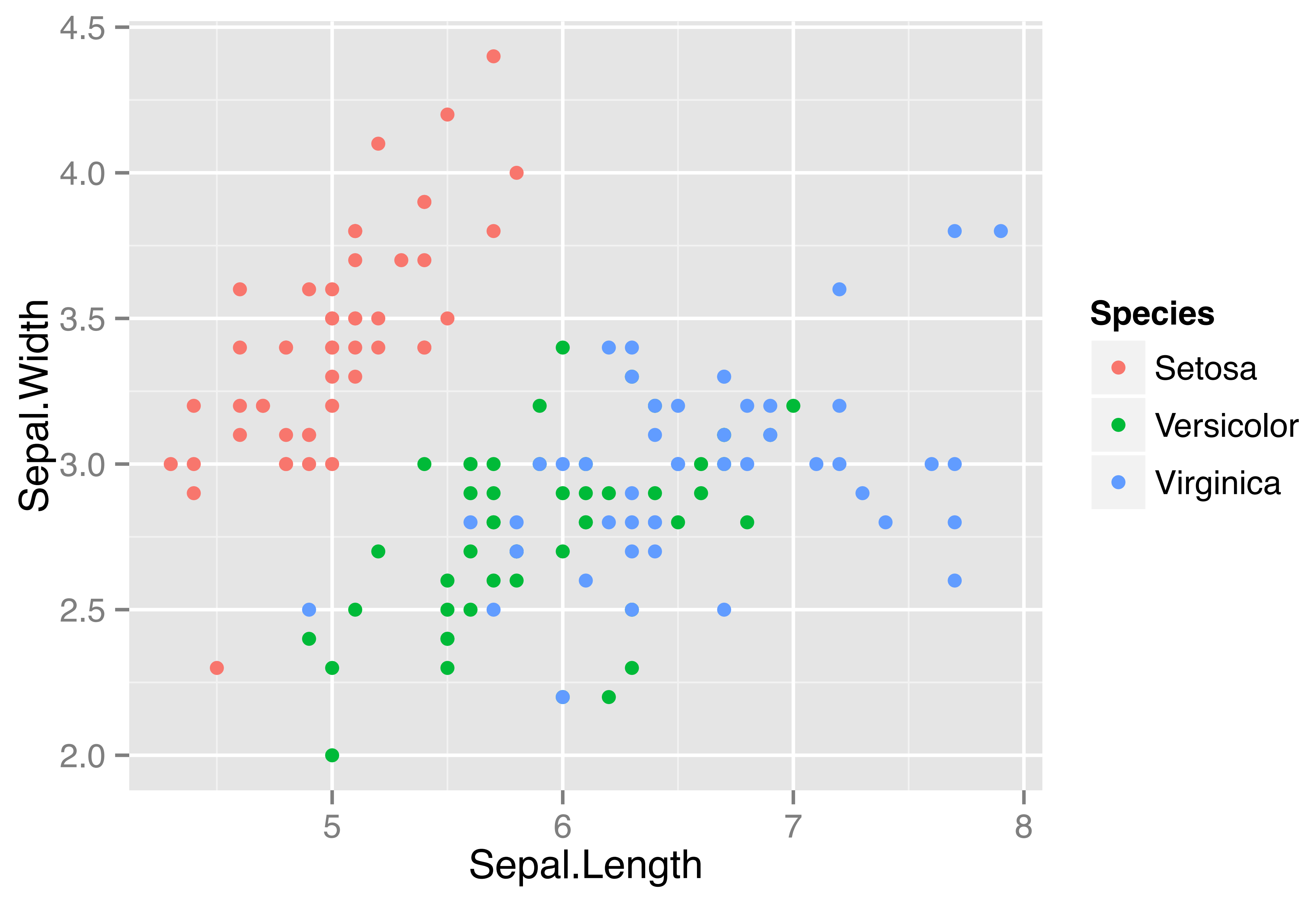
- Each geom has specific aesthetic mappings
- geom_point()
    - Essential: x, y



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

Scatter Plot

- Each geom has specific aesthetic mappings
- geom_point()
  - Essential: x, y
  - Optional: alpha, colour, fill, shape, size

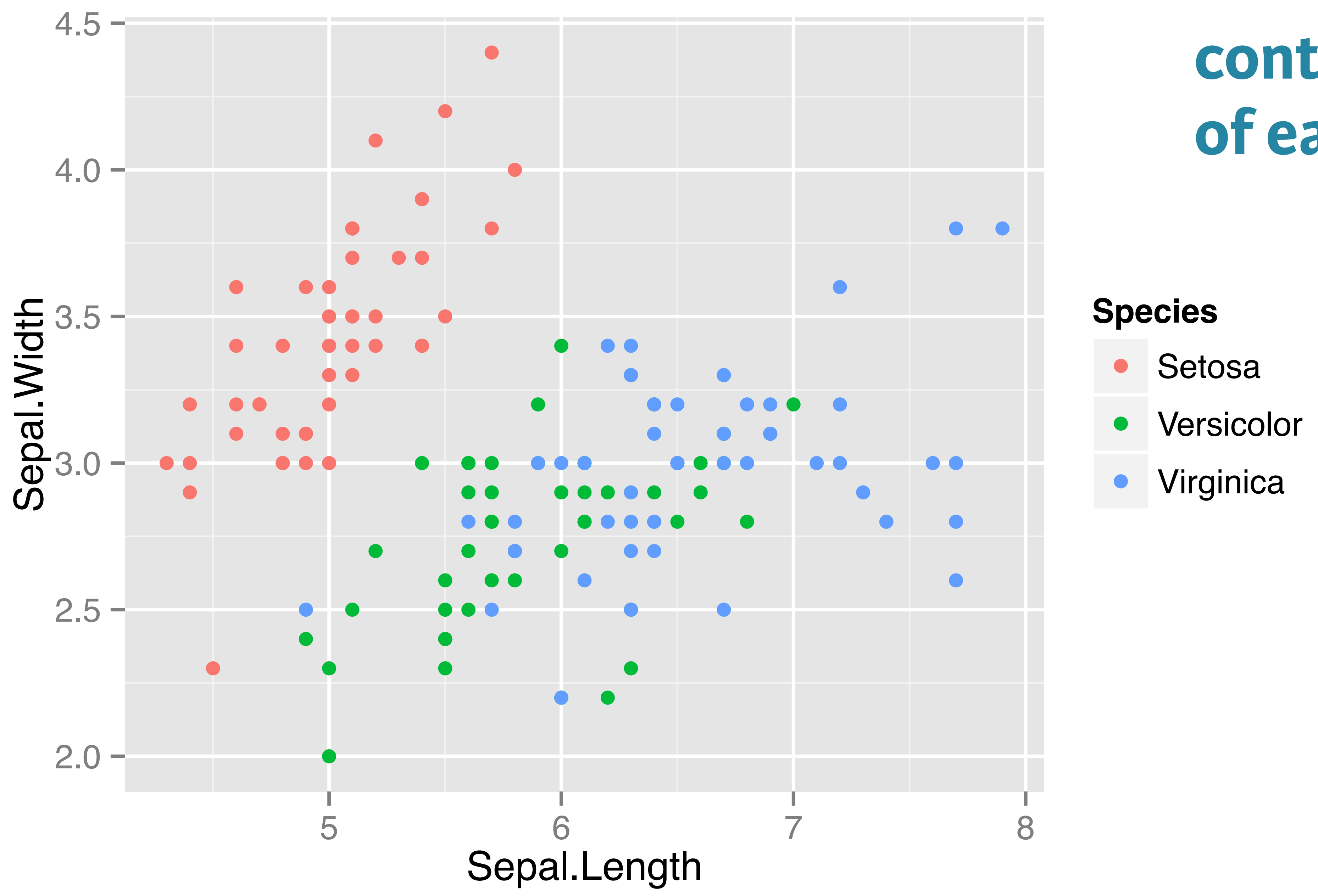


```
> ggplot(iris, aes(x = Sepal.Length, y = Sepal.Width, col = Species)) +
  geom_point()
```

aes() inside geom_()

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

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

same plot results!



control aesthetic mappings of each layer independently

Summary Statistics

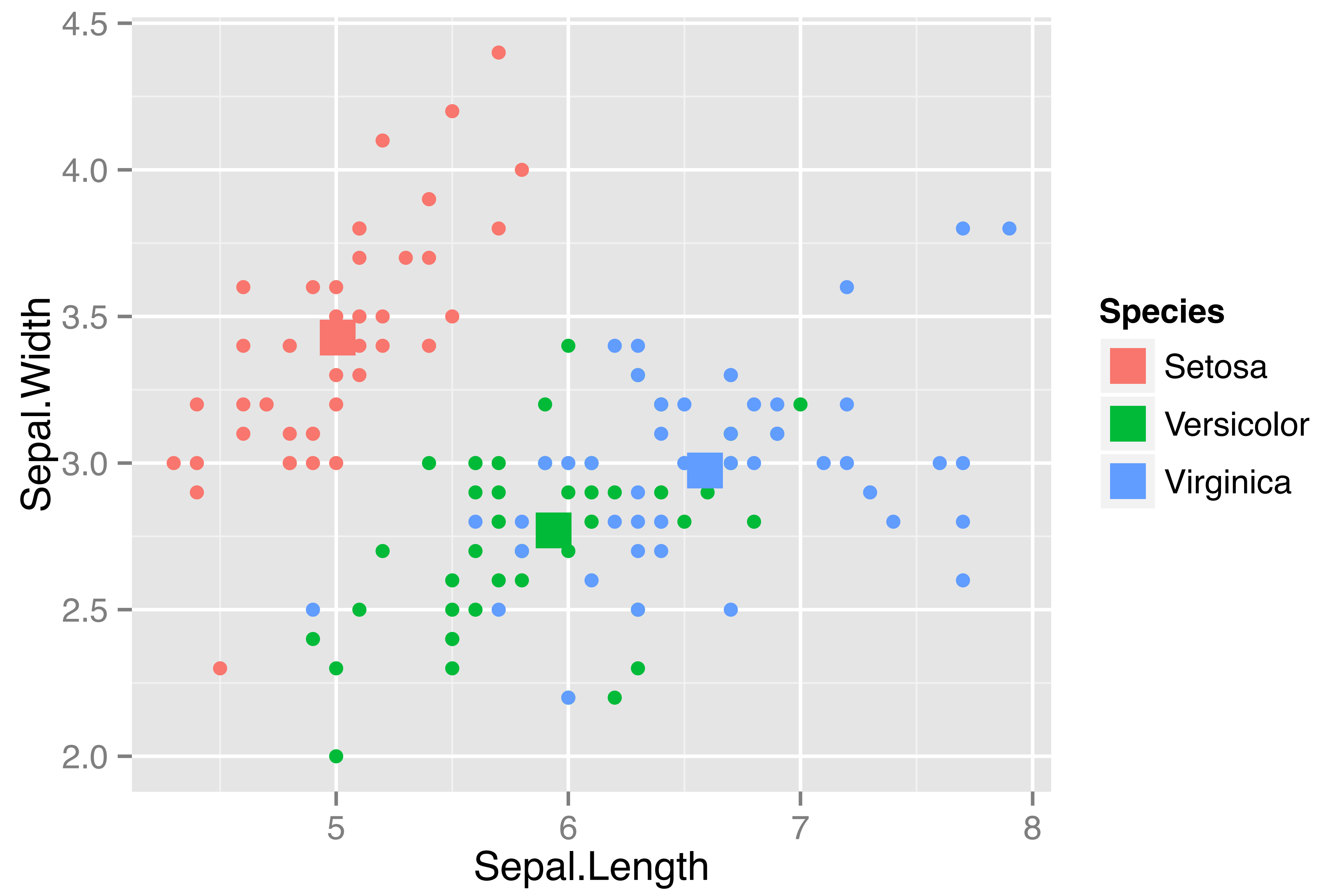
```
> head(iris)
  Species Sepal.Length Sepal.Width Petal.Length Petal.Width
1  Setosa          5.1         3.5          1.4         0.2
2  Setosa          4.9         3.0          1.4         0.2
3  Setosa          4.7         3.2          1.3         0.2
4  Setosa          4.6         3.1          1.5         0.2
5  Setosa          5.0         3.6          1.4         0.2
6  Setosa          5.4         3.9          1.7         0.4

> iris.summary <- aggregate(iris[2:5], list(iris$Species), mean)
> names(iris.summary)[1] <- "Species"
> iris.summary
     Species Sepal.Length Sepal.Width Petal.Length Petal.Width
1     Setosa        5.006       3.428        1.462       0.246
2 Versicolor        5.936       2.770        4.260       1.326
3  Virginica        6.588       2.974        5.552       2.026
```

Add Layers

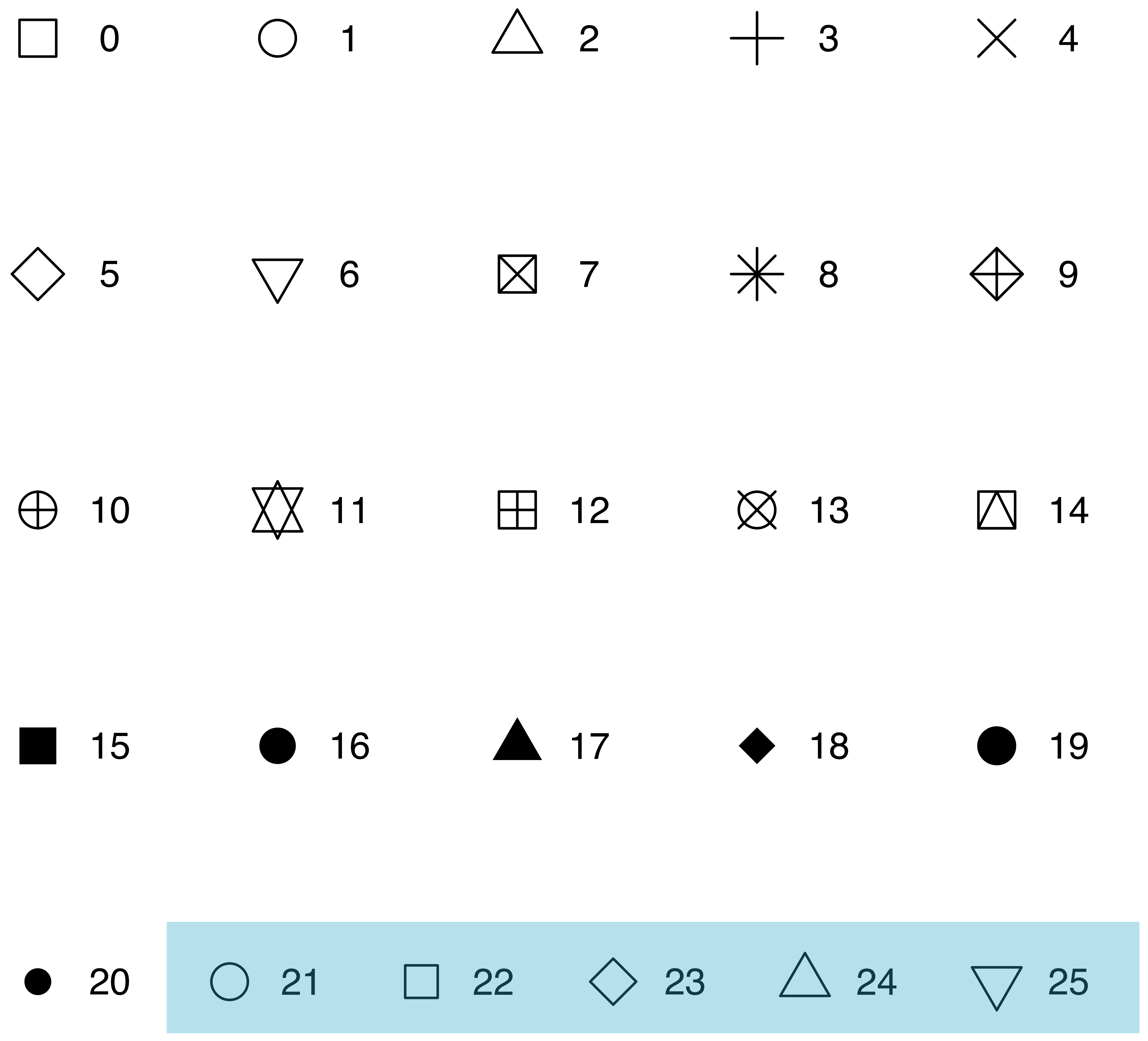
```
> ggplot(iris, aes(x = Sepal.Length, y = Sepal.Width, col = Species)) +
    geom_point() +
    geom_point(data = iris.summary, shape = 15, size = 5)
```

inherits data and aes from ggplot()

different data inherits aes



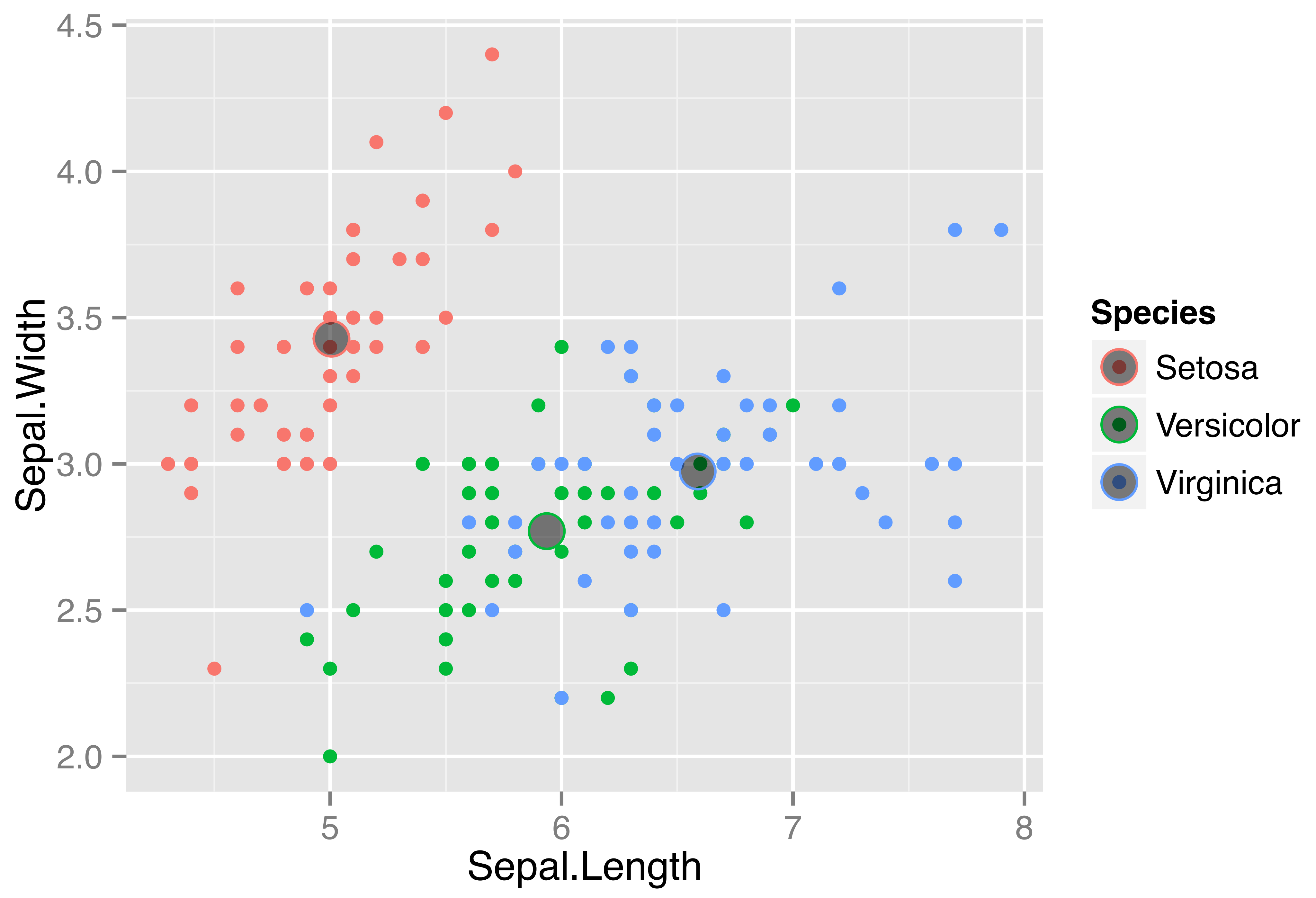
shape ~ pch



both fill and color

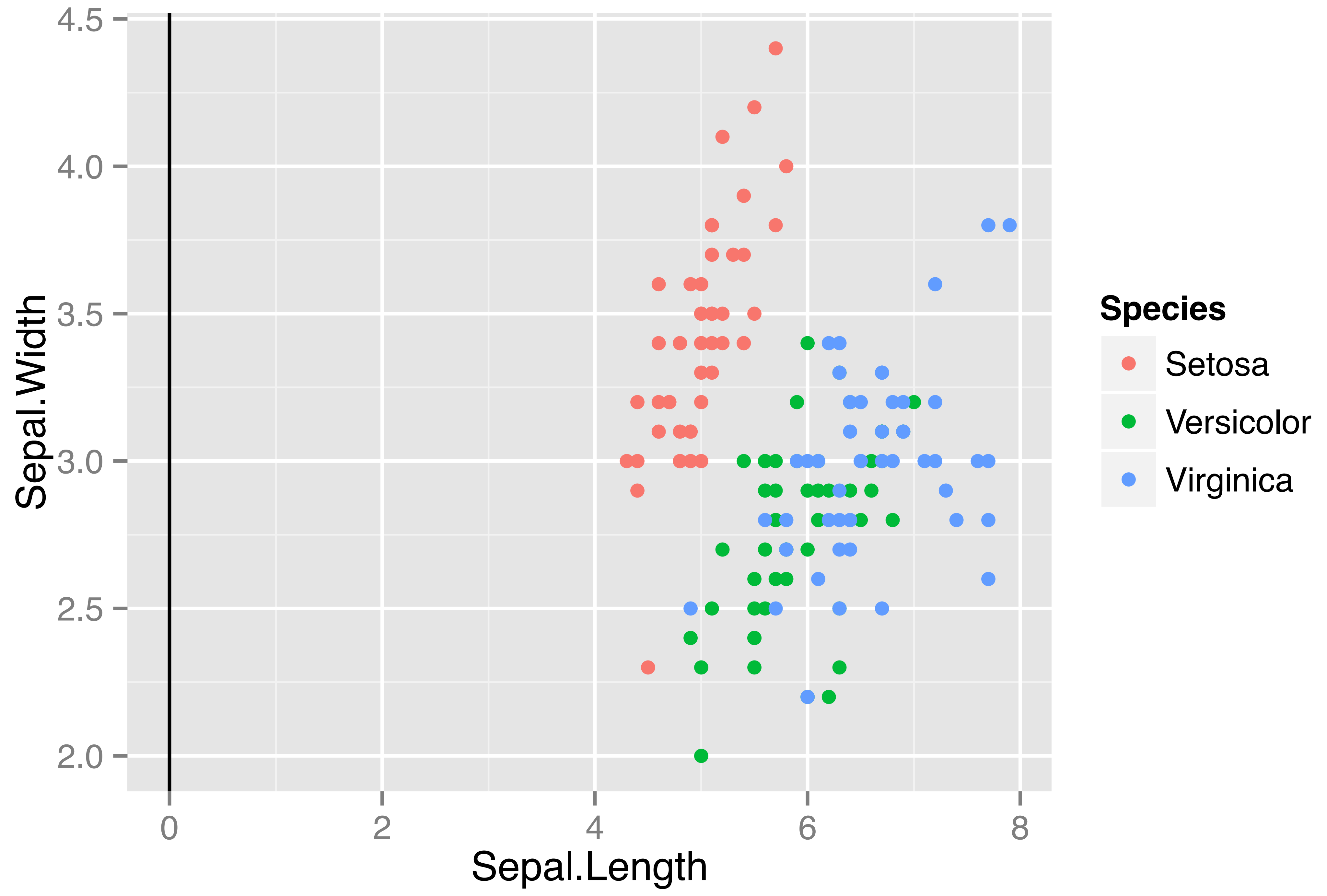
Example

```
> ggplot(iris, aes(x = Sepal.Length, y = Sepal.Width, col = Species)) +
    geom_point() +
    geom_point(data = iris.summary,
               shape = 21, size = 5, fill = "#00000080")
```



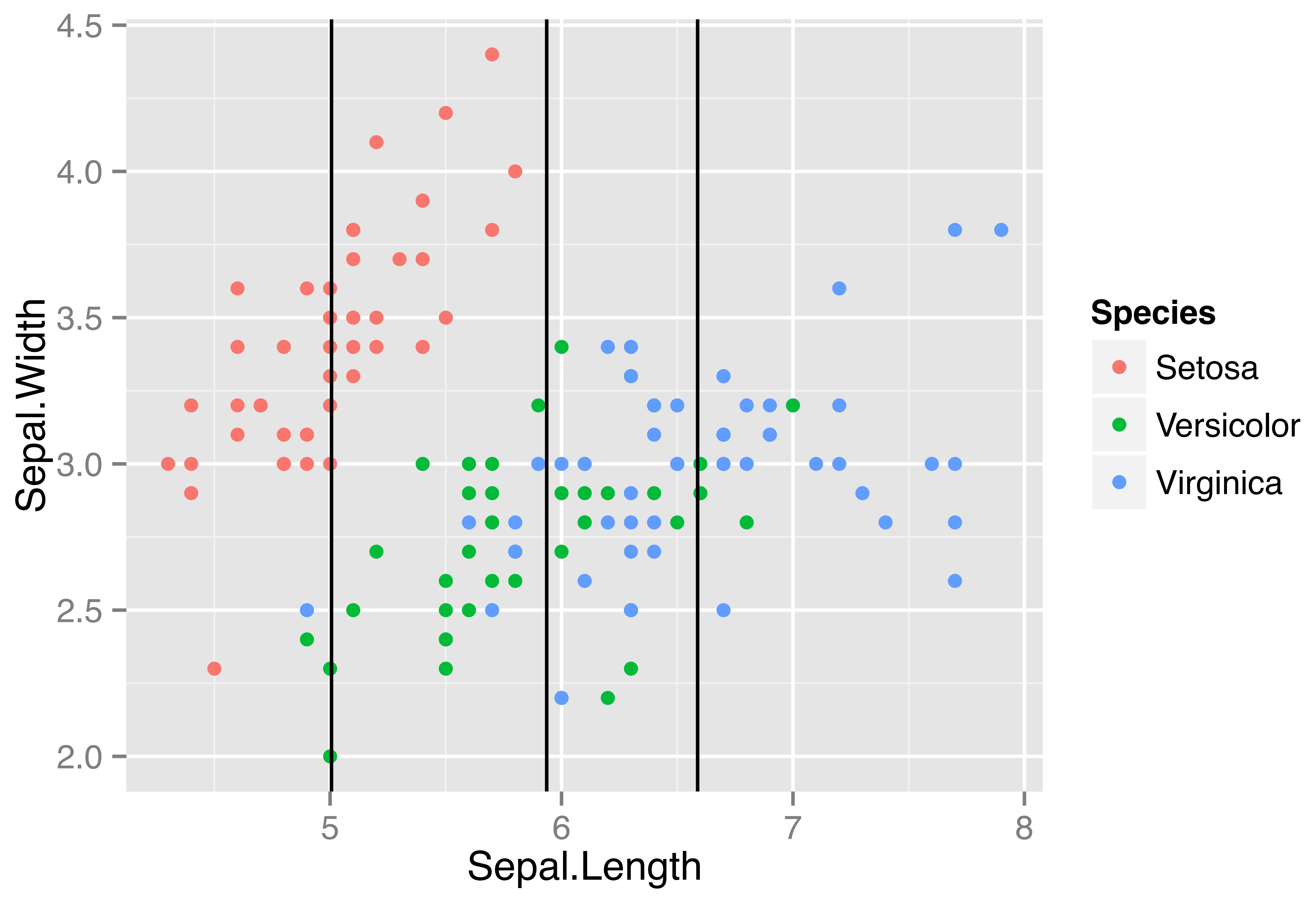
Crosshairs

```
> ggplot(iris, aes(x = Sepal.Length, y = Sepal.Width, col = Species)) +
    geom_point() +
    geom_vline(data = iris.summary)
```



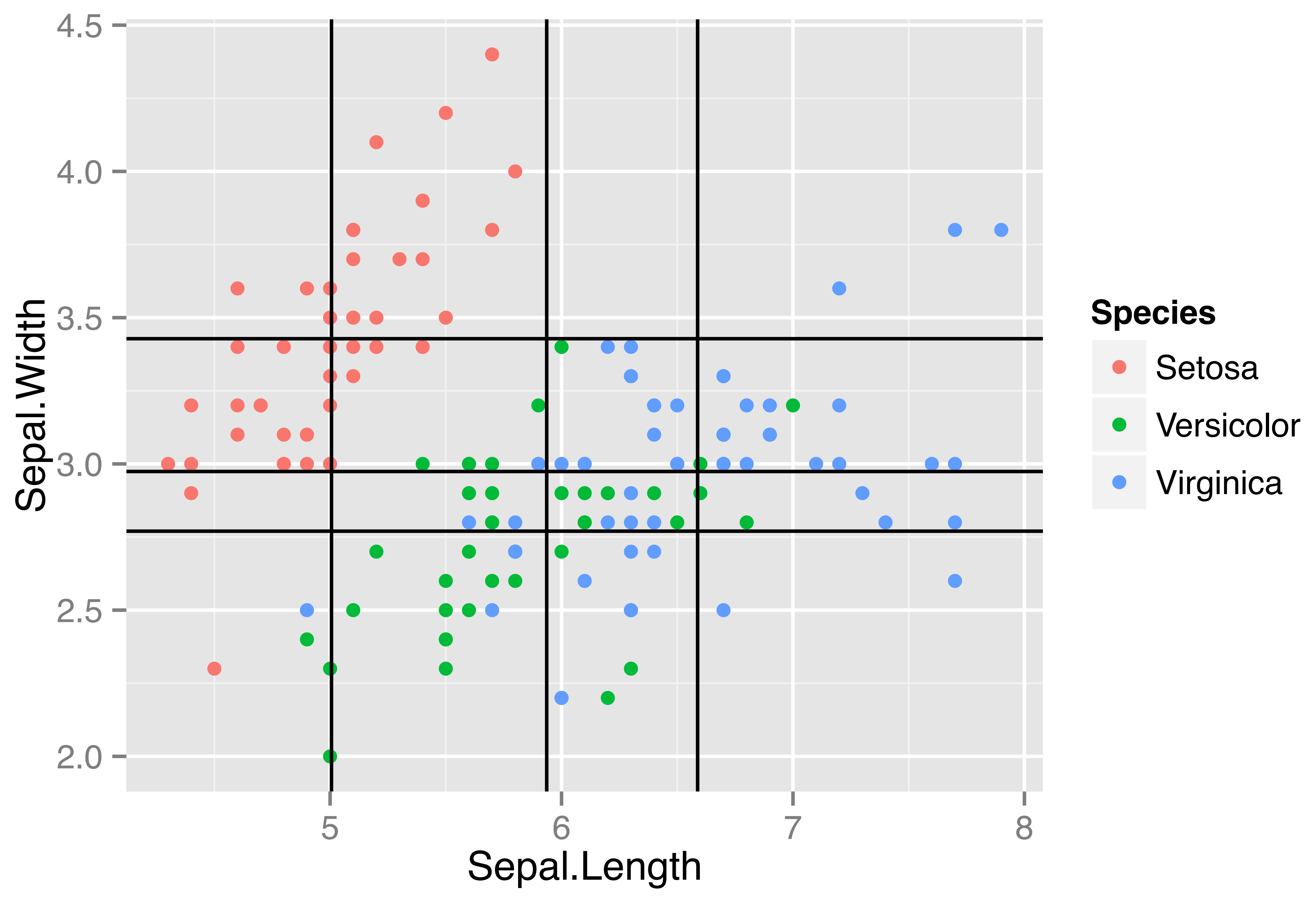
Crosshairs

```
> ggplot(iris, aes(x = Sepal.Length, y = Sepal.Width, col = Species)) +
    geom_point() +
    geom_vline(data = iris.summary, aes(xintercept = Sepal.Length))
```



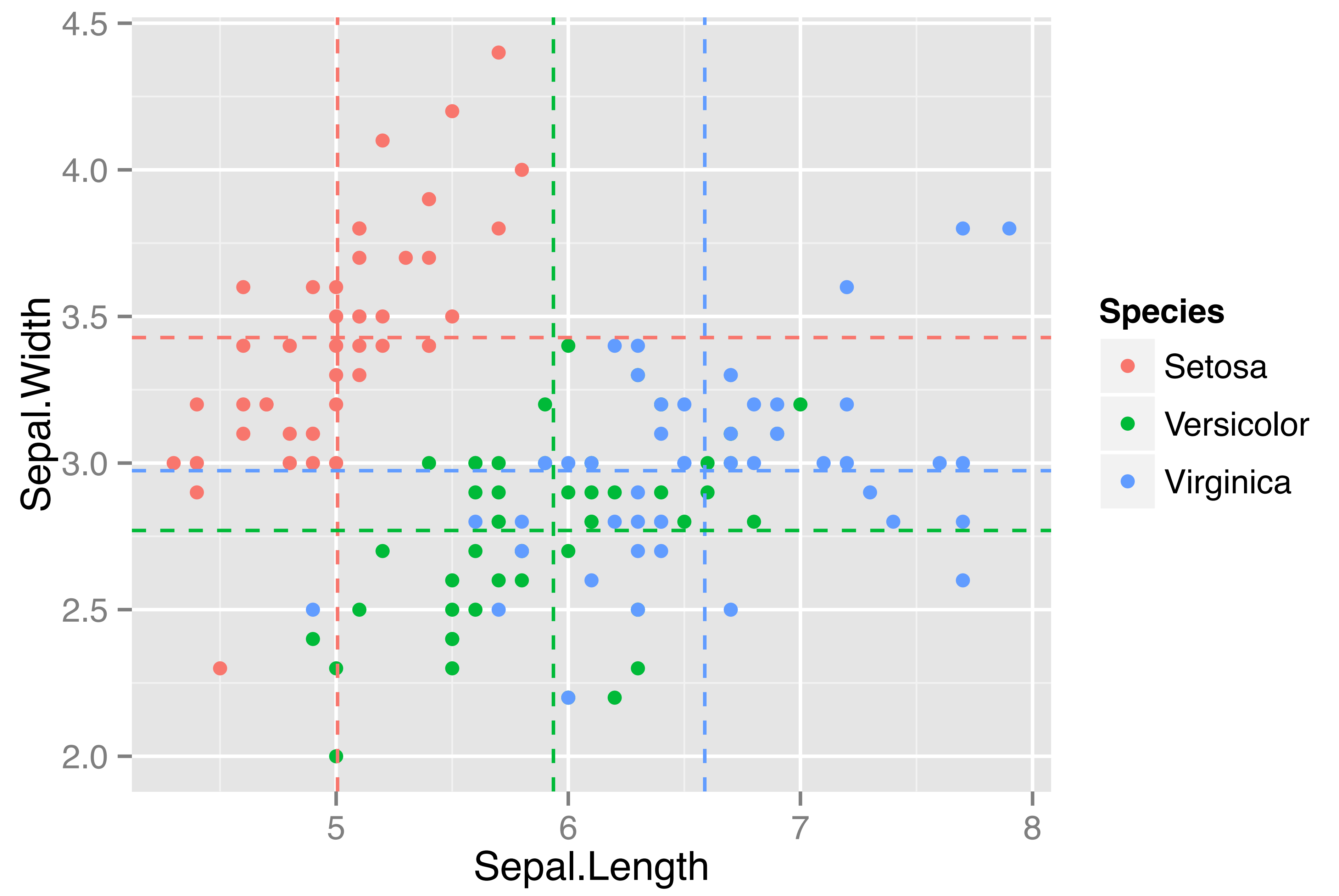
Crosshairs

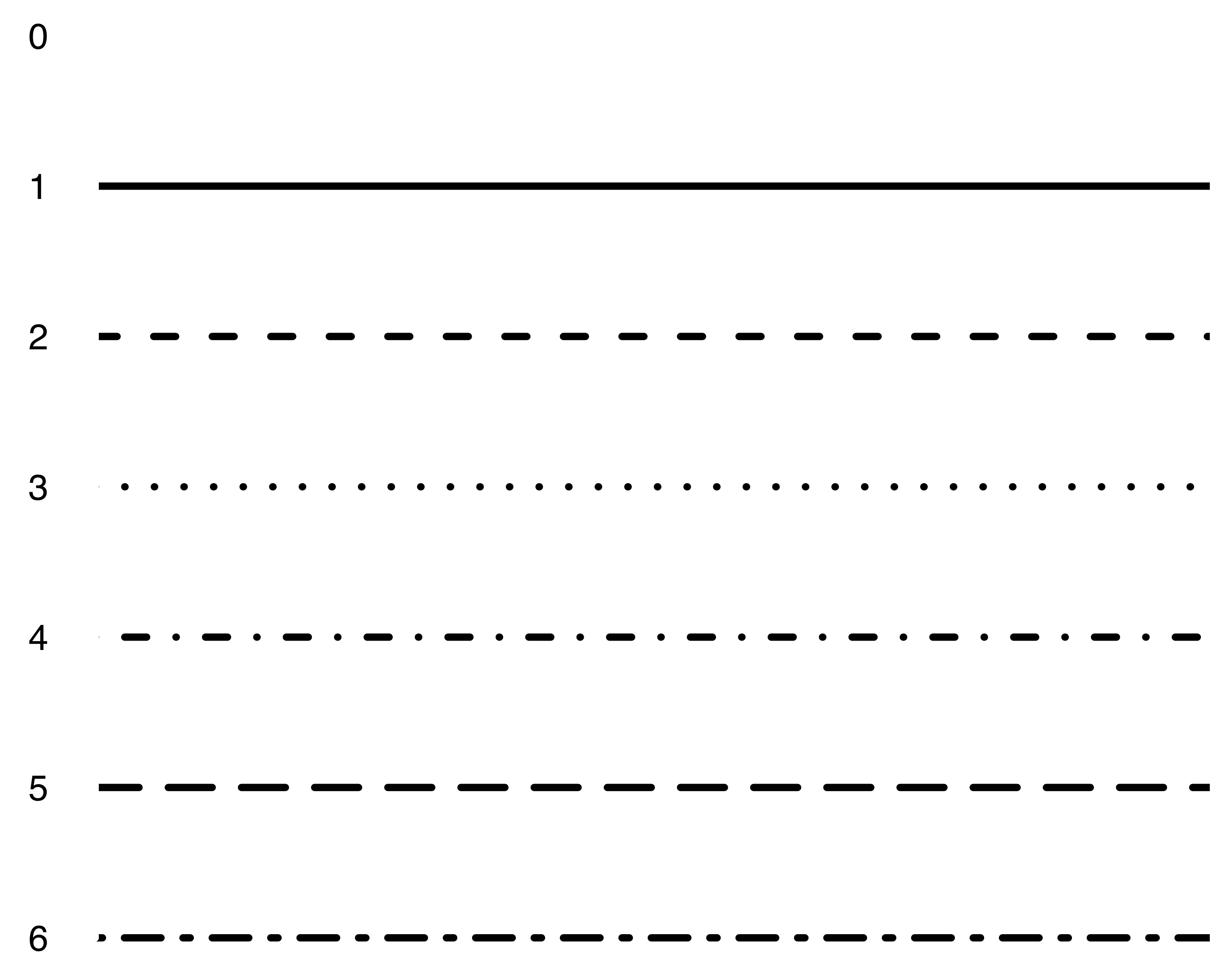
```
> ggplot(iris, aes(x = Sepal.Length, y = Sepal.Width, col = Species)) +
    geom_point() +
    geom_vline(data = iris.summary, aes(xintercept = Sepal.Length)) +
    geom_hline(data = iris.summary, aes(yintercept = Sepal.Width))
```



Crosshairs

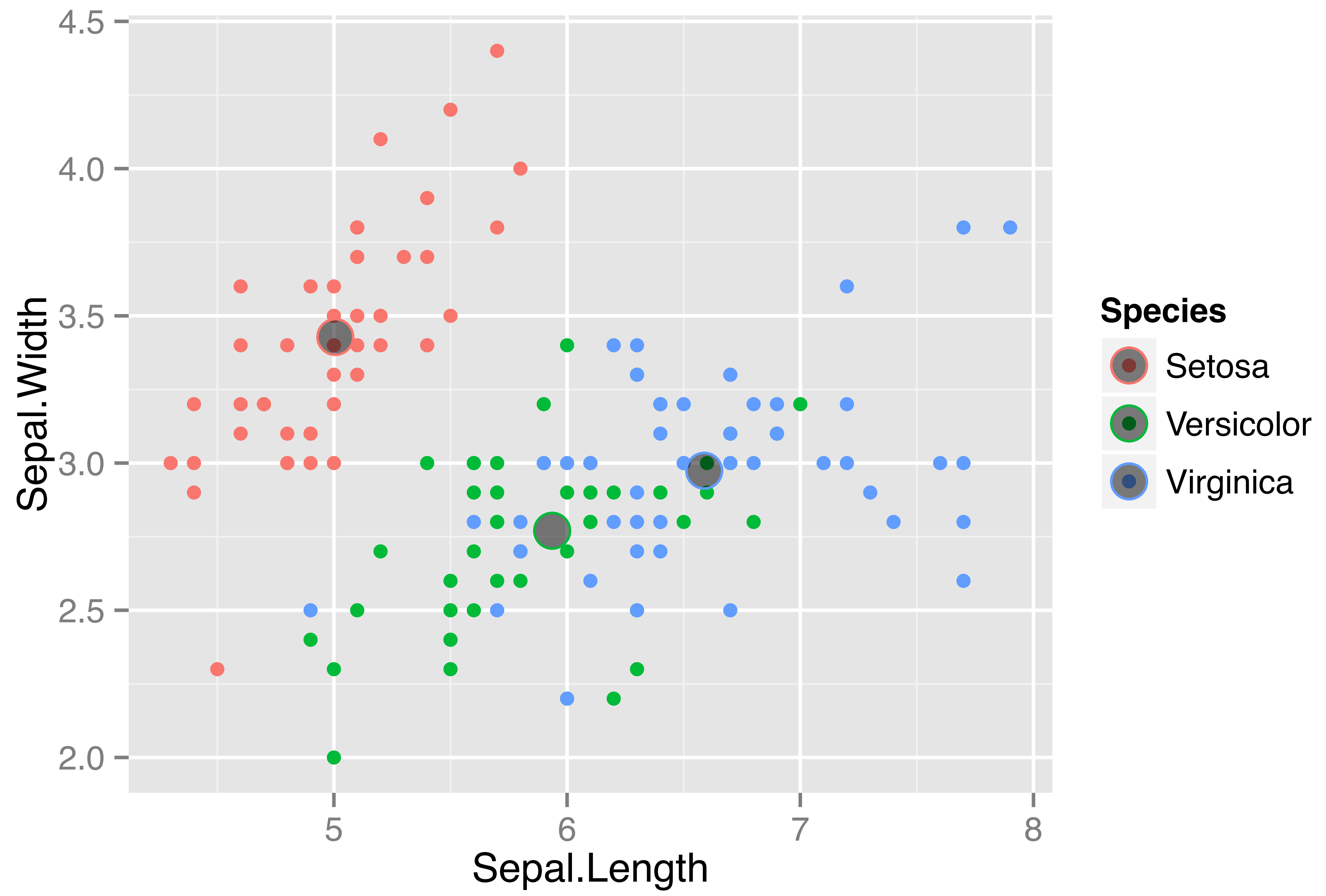
```
ggplot(iris, aes(x = Sepal.Length, y = Sepal.Width, col = Species)) +
  geom_point() +
  geom_vline(data = iris.summary, linetype = 2,
             aes(xintercept = Sepal.Length, col = Species)) +
  geom_hline(data = iris.summary, linetype = 2),
             aes(yintercept = Sepal.Width, col = Species),
```





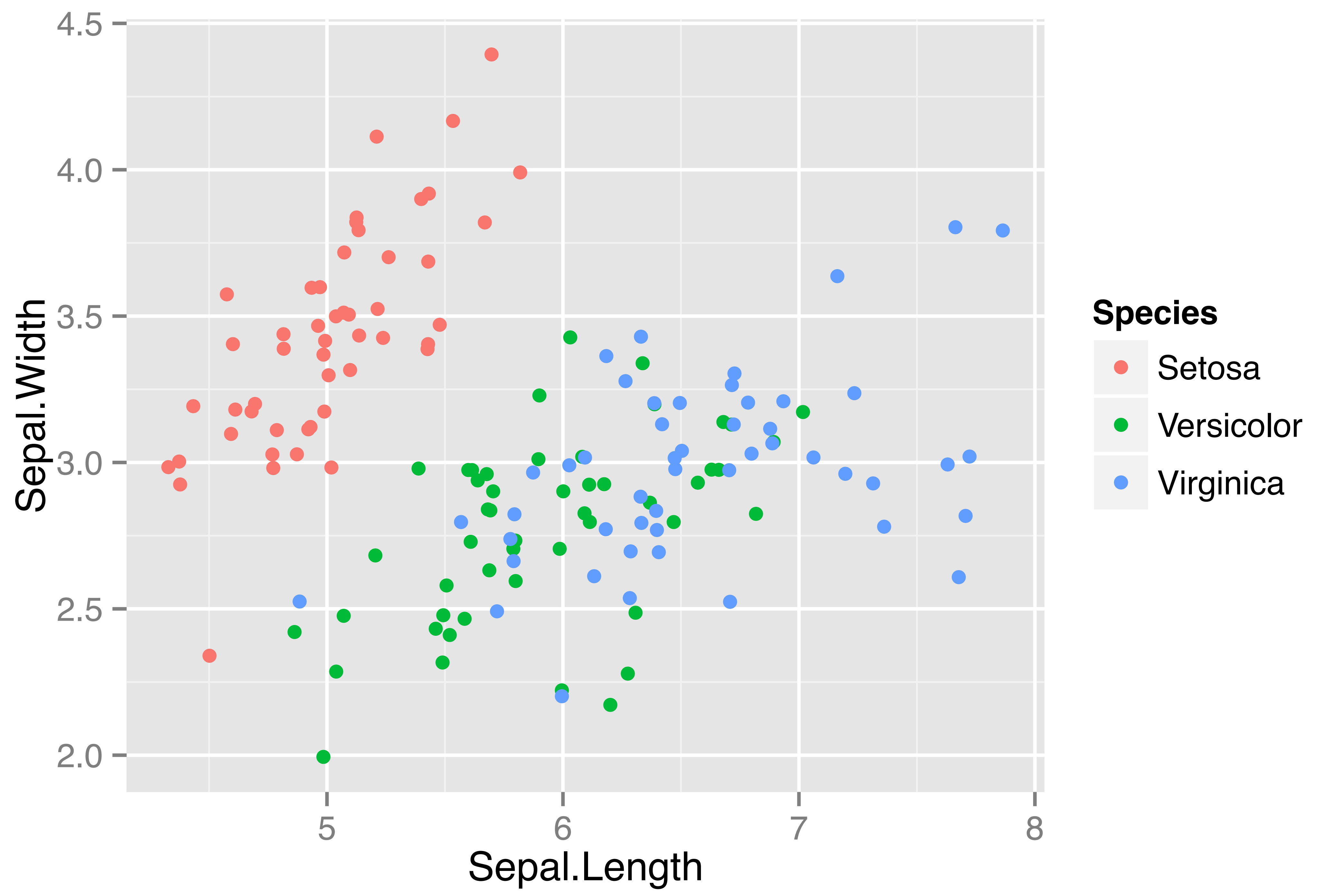
Remarks

- ggplot2 can also calculate statistics
- mean only = no good



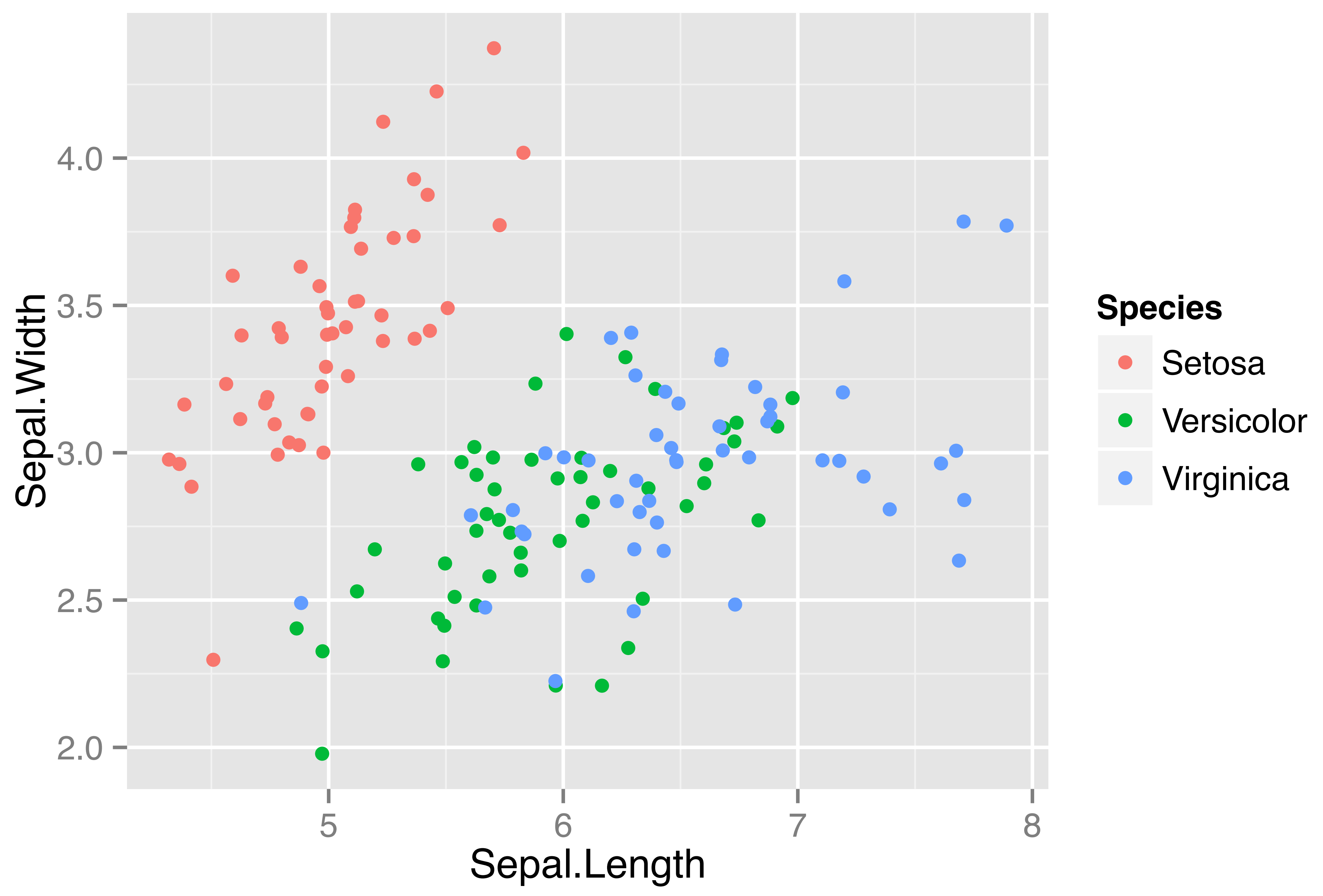
Jitter

```
> ggplot(iris, aes(x = Sepal.Length, y = Sepal.Width, col = Species)) +
    geom_point(position = "jitter")
```



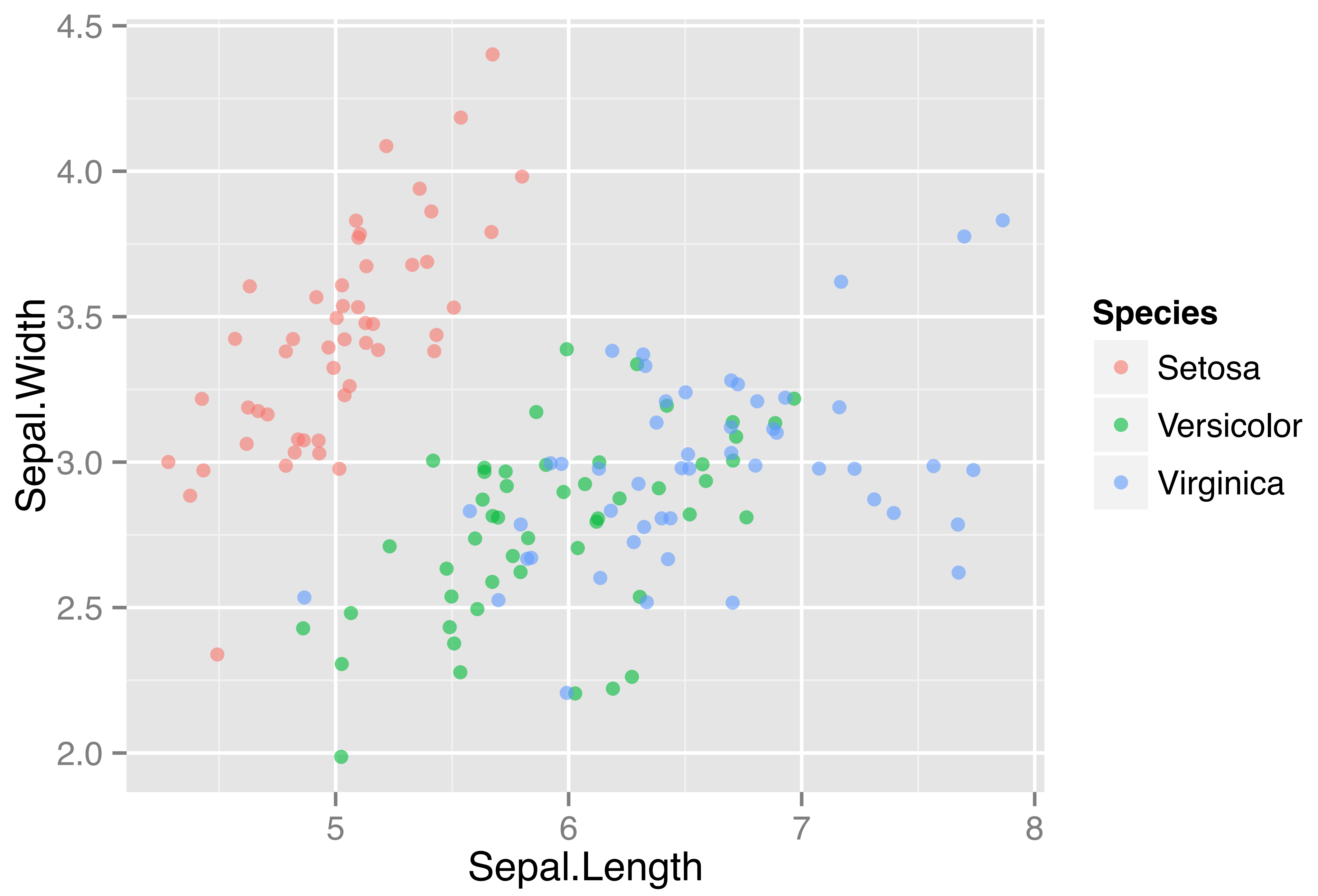
Jitter

```
> ggplot(iris, aes(x = Sepal.Length, y = Sepal.Width, col = Species)) +
    geom_jitter()
```



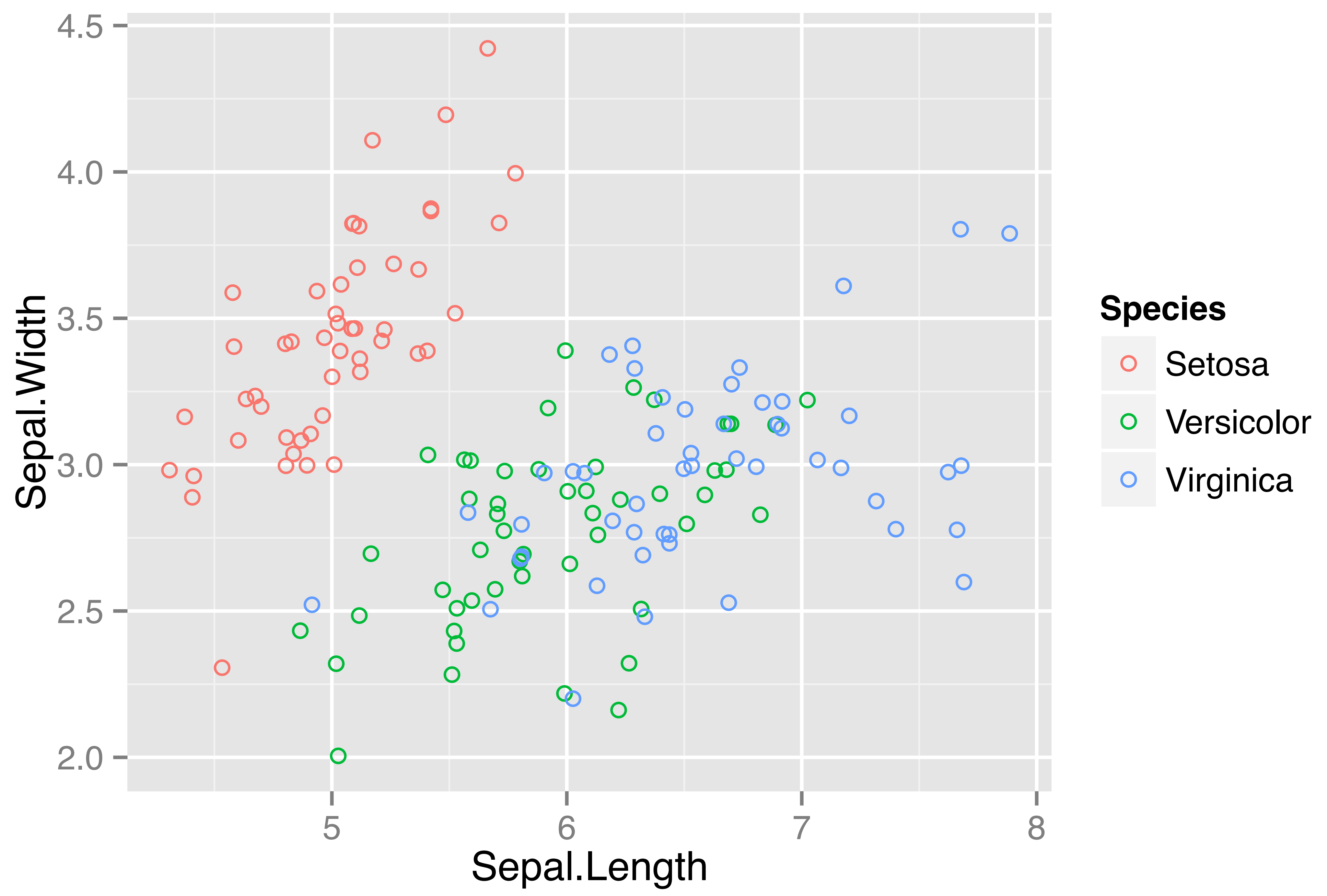
Jitter

```
> ggplot(iris, aes(x = Sepal.Length, y = Sepal.Width, col = Species)) +
    geom_jitter(alpha = 0.6)
```



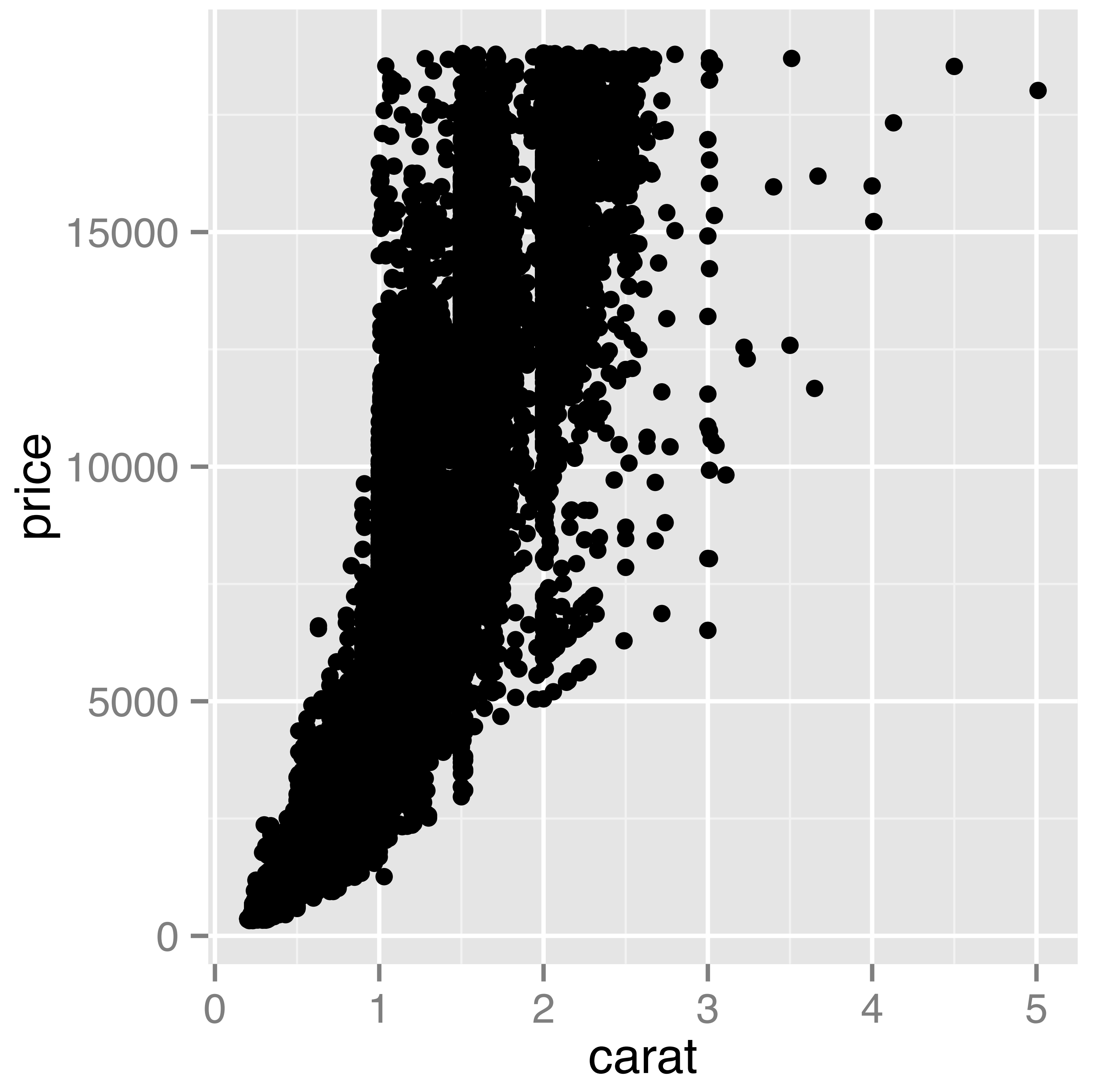
Jitter

```
> ggplot(iris, aes(x = Sepal.Length, y = Sepal.Width, col = Species)) +
    geom_jitter(shape = 1)
```



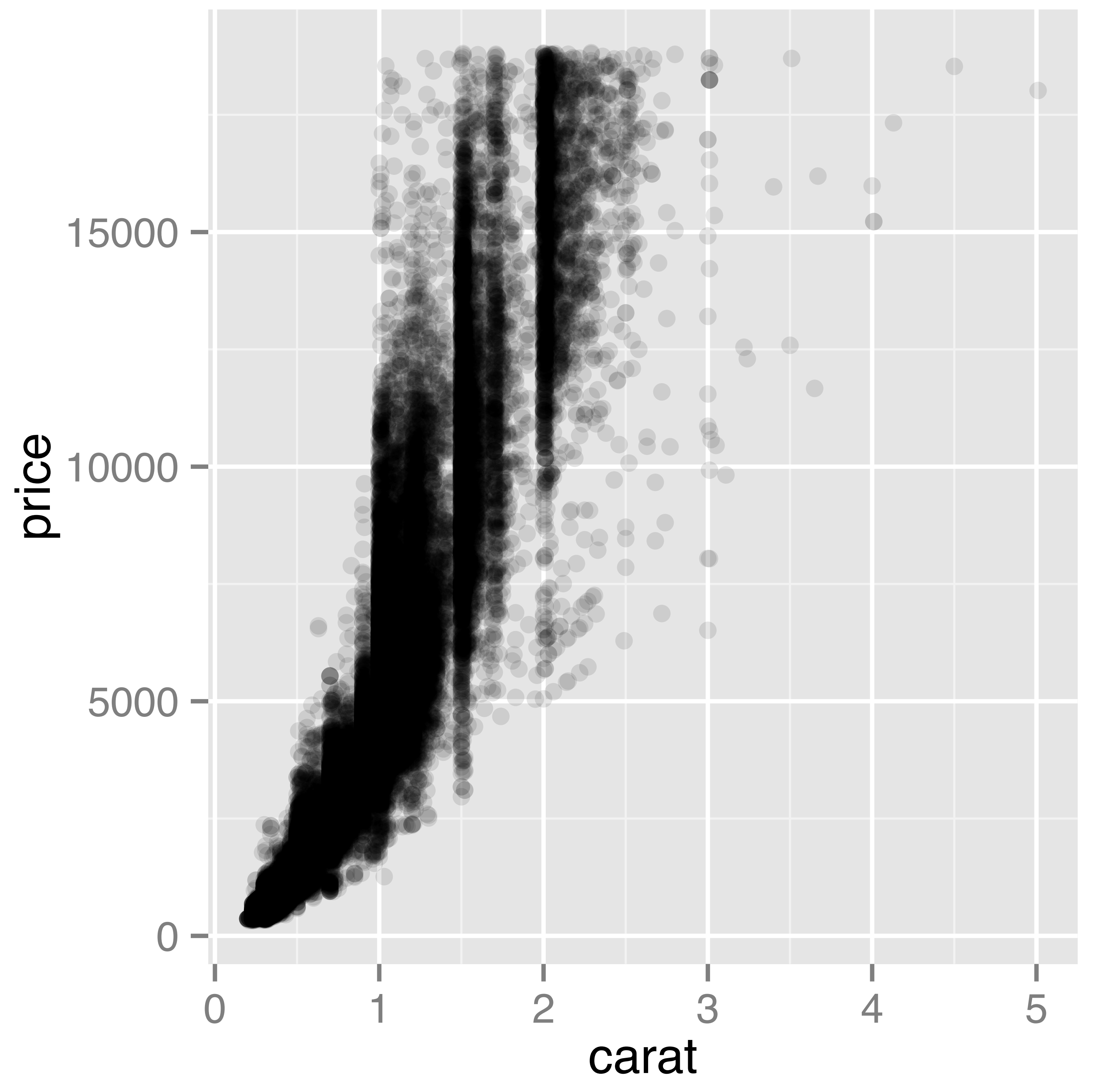
Diamonds

```
> ggplot(diamonds, aes(carat, price)) +
    geom_point()
```



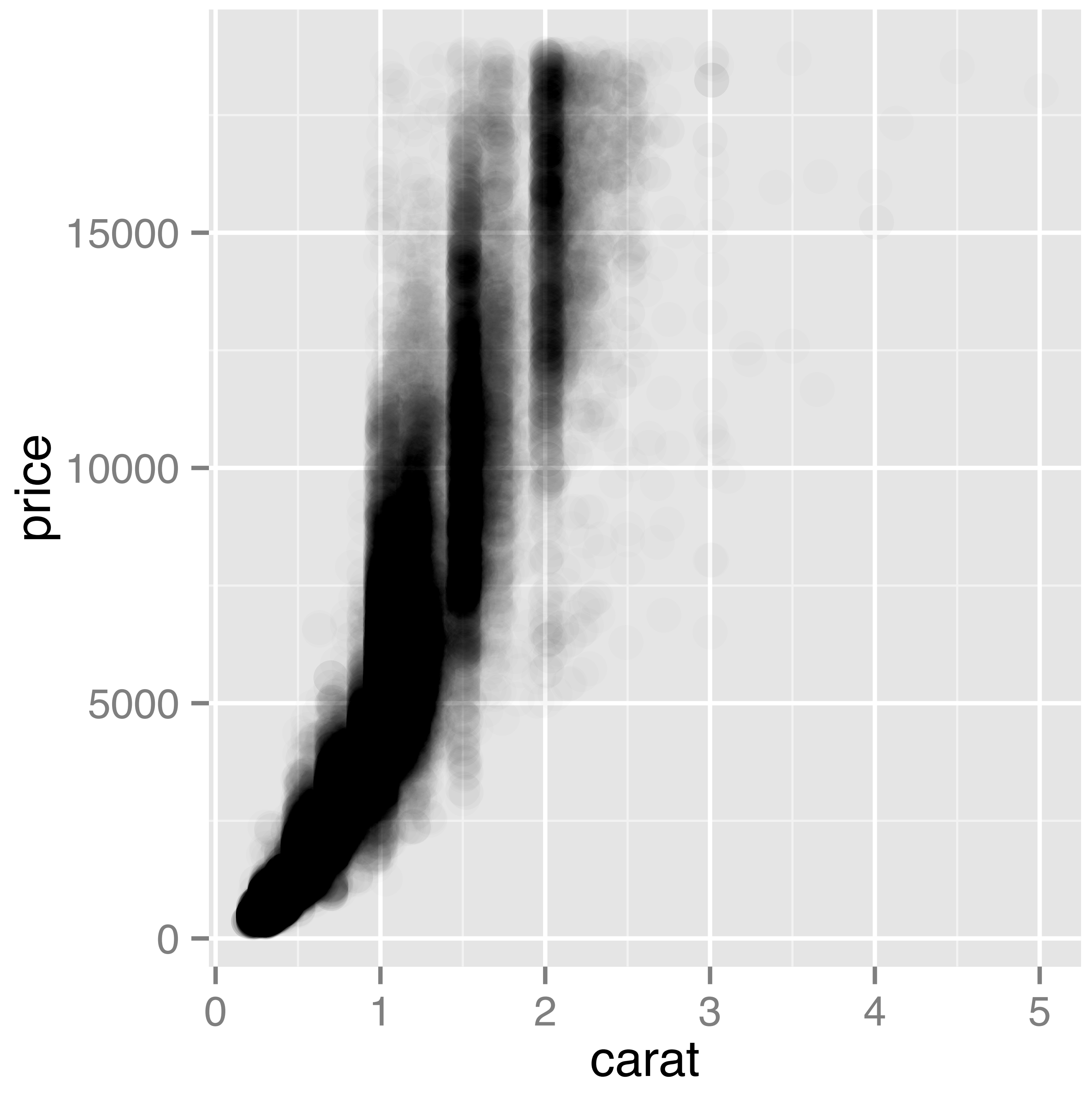
Diamonds

```
> ggplot(diamonds, aes(carat, price)) +
    geom_point(alpha = 0.1)
```



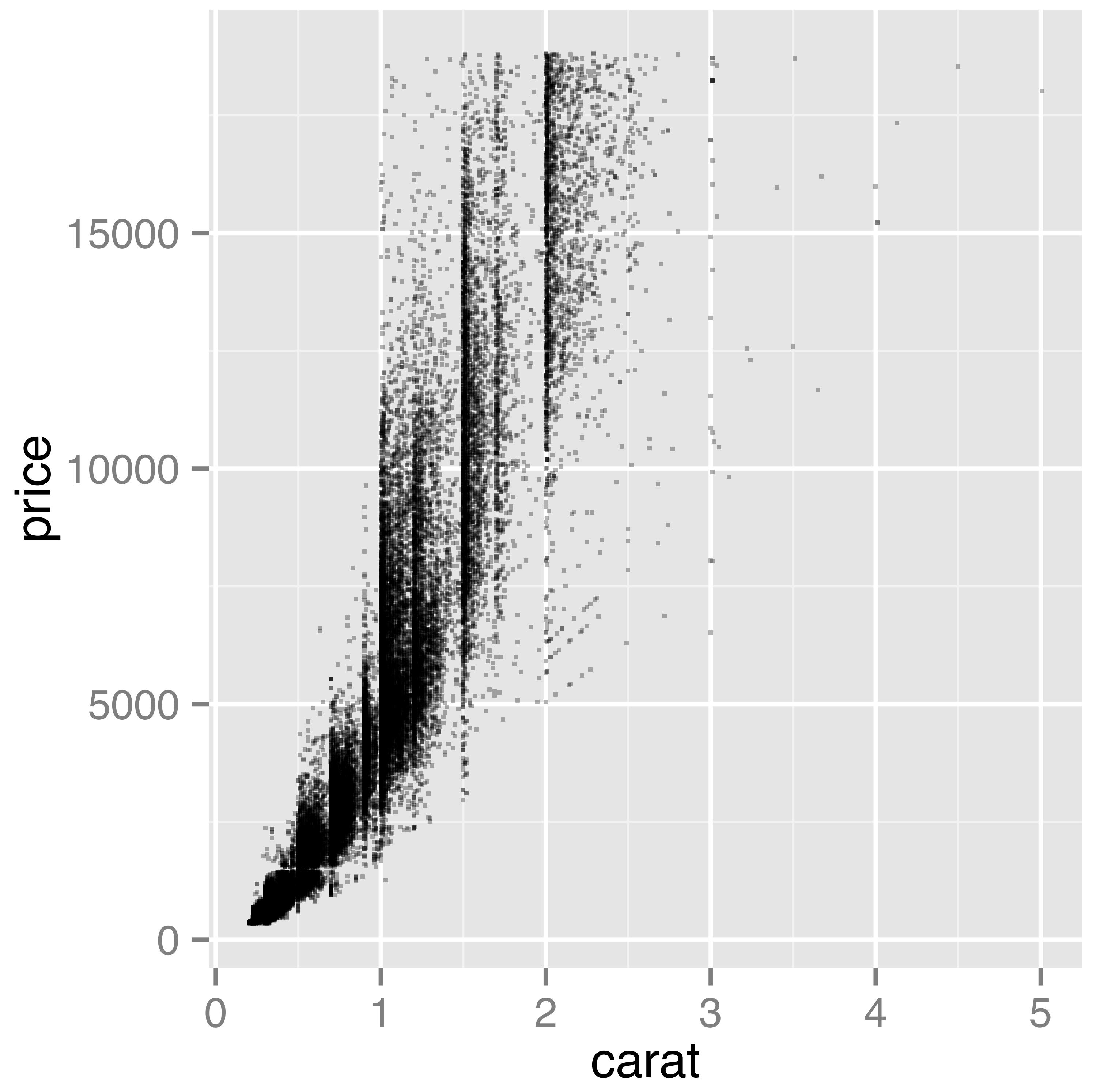
Diamonds

```
> ggplot(diamonds, aes(carat, price)) +
    geom_point(alpha = 0.01, size = 4)
```



Diamonds

```
> ggplot(diamonds, aes(carat, price)) +
    geom_point(alpha = 0.3, shape = ".")
```



Let's practice!

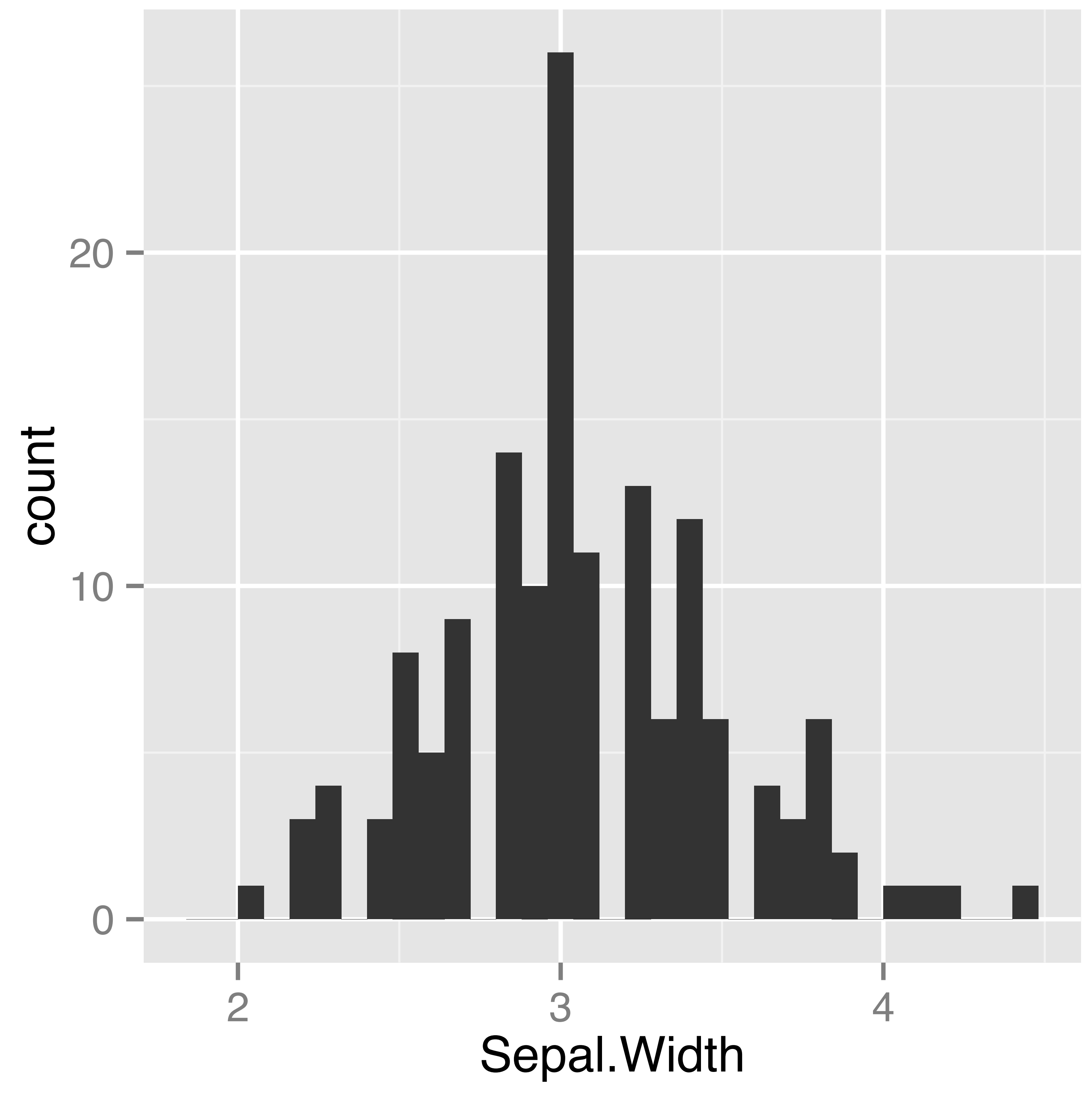
Bar Plots

Common plot types

- Scatter plots
    - points, jitter, abline
- **Bar plots**
    - **histogram, bar, errorbar**
- Line plots
    - line

Histogram

```
> ggplot(iris, aes(x = Sepal.Width)) +
    geom_histogram()
stat_bin: binwidth defaulted to range/30. Use 'binwidth = x' to
adjust this.

> diff(range(iris$Sepal.Width)) / 30
[1] 0.08
```



Plot of statistical function

Slightly different binning algorithm

Histogram

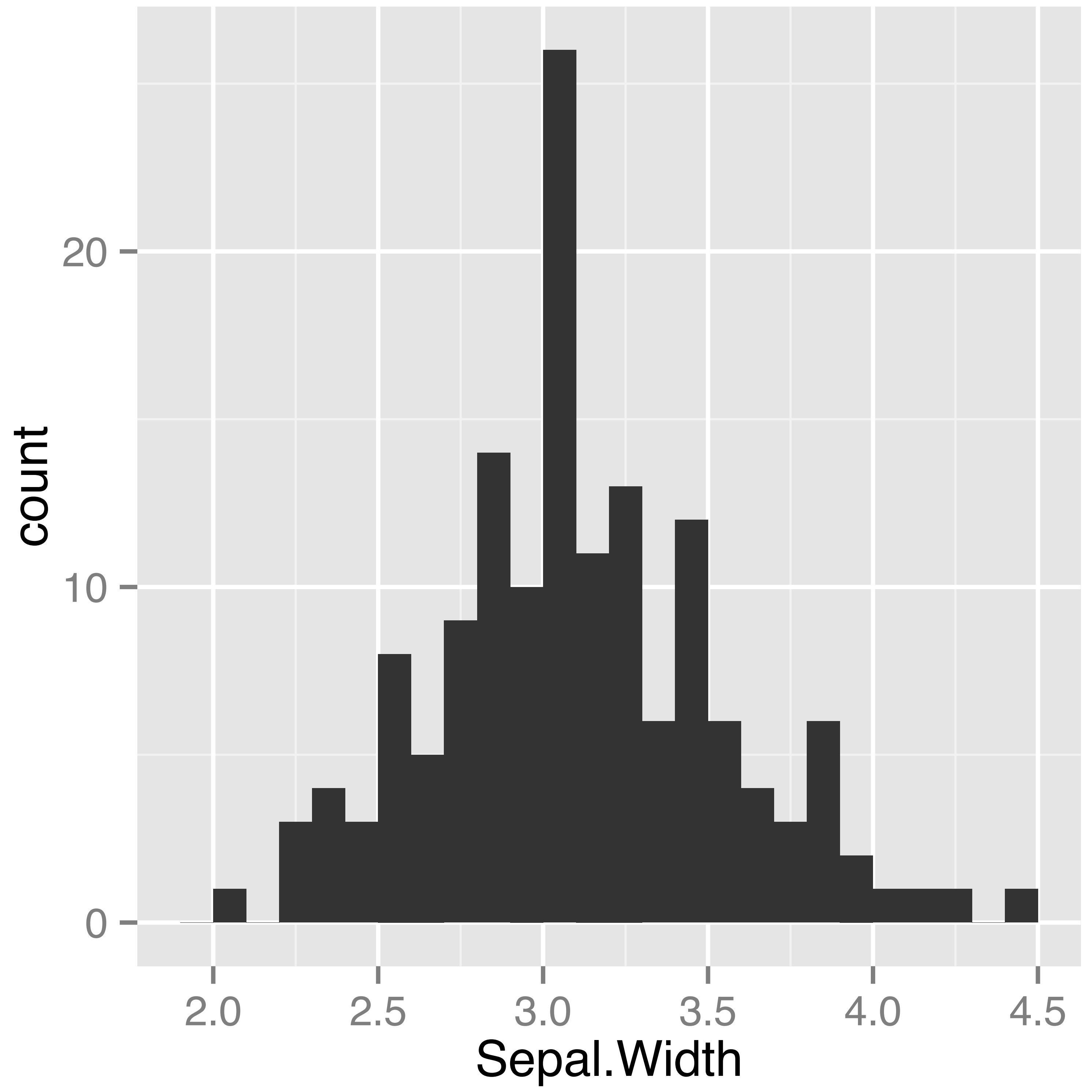
```
> ggplot(iris, aes(x = Sepal.Width)) +
    geom_histogram(binwidth = 0.1)
```

Many ways to do binning

No space between bars

x axis labels between the bars

Summary table in the background

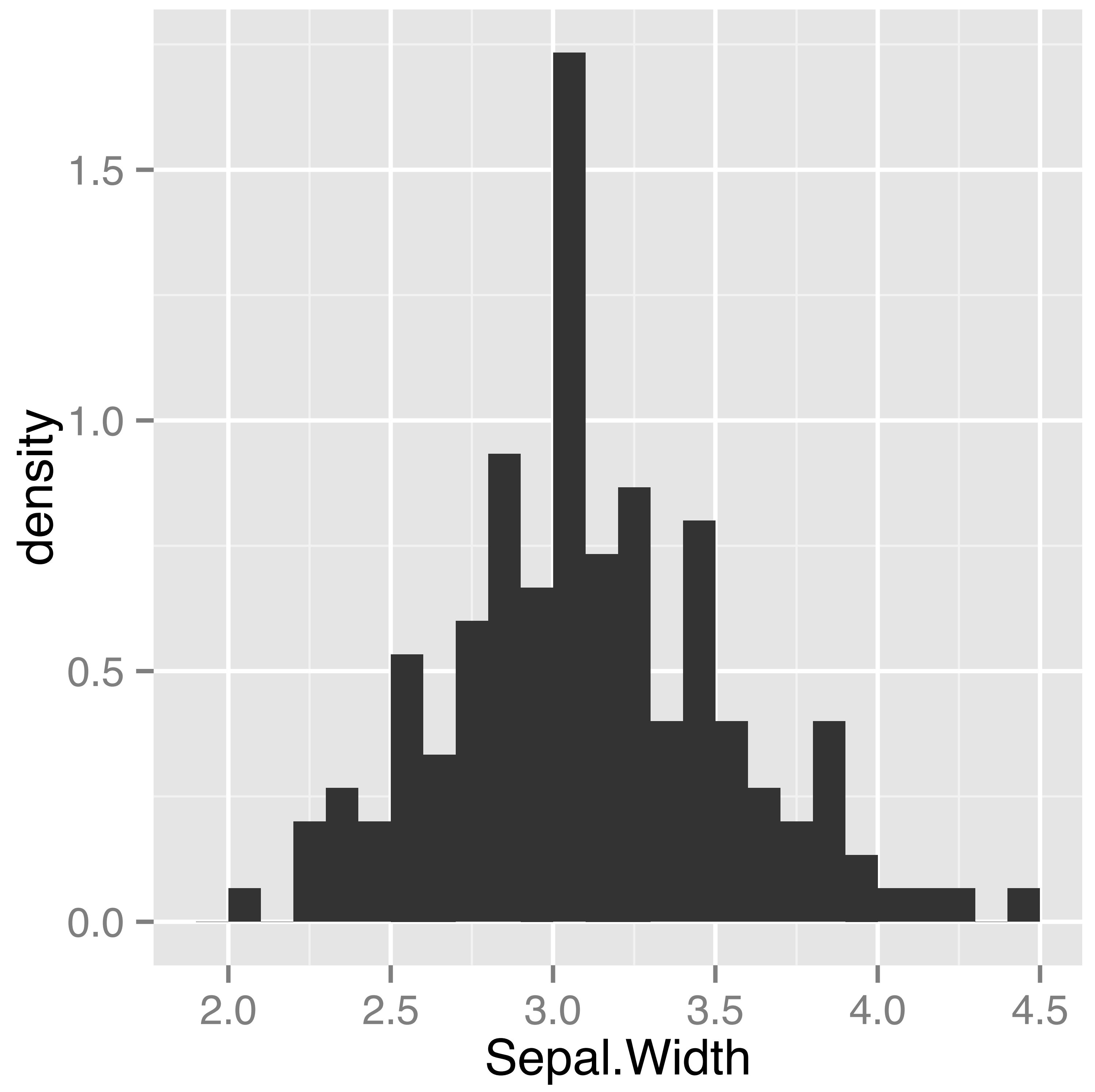


Histogram

```
> ggplot(iris, aes(x = Sepal.Width)) +
    geom_histogram(aes(y = ..density..), binwidth = 0.1)
```

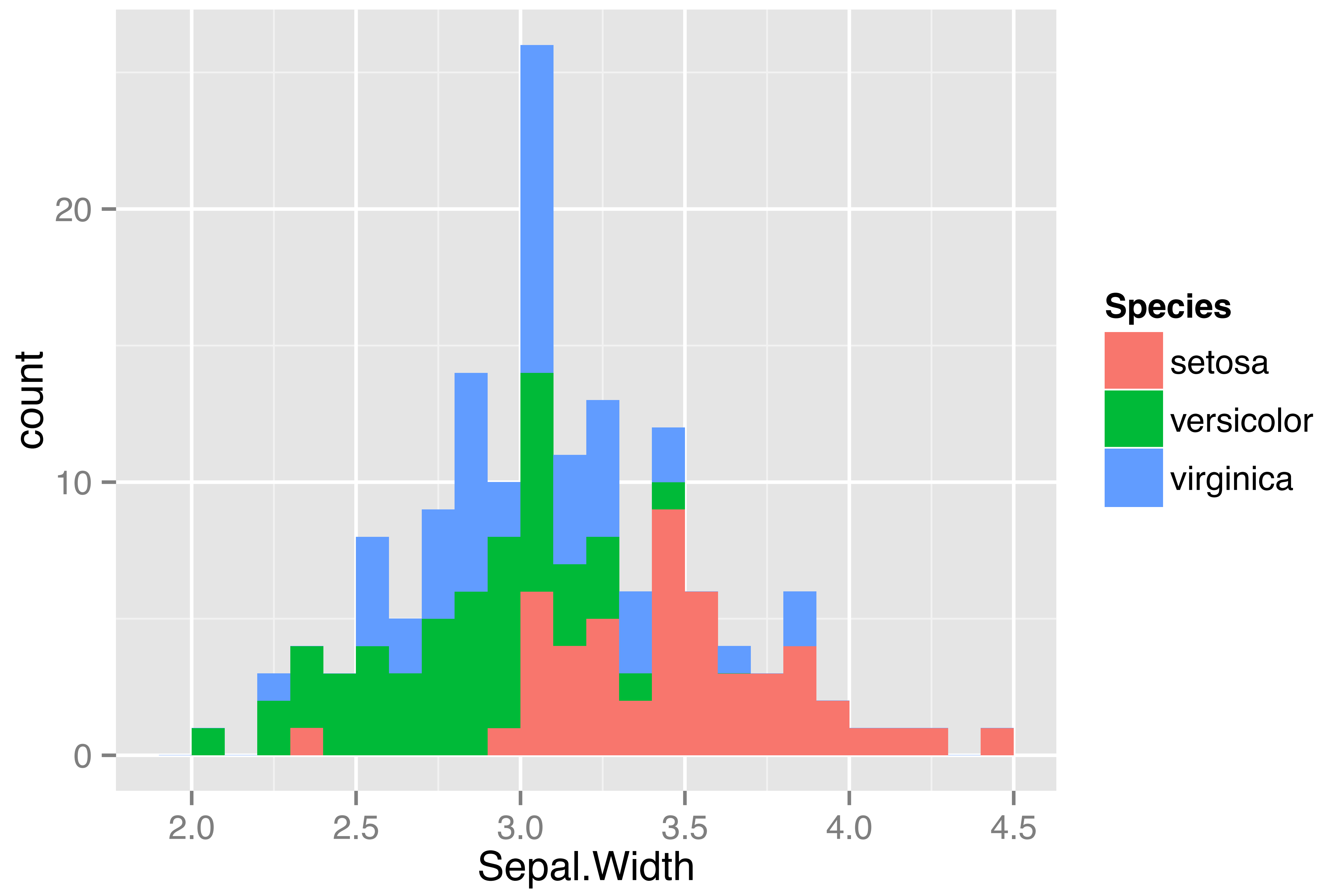
internal data frame

y axis re-labeled



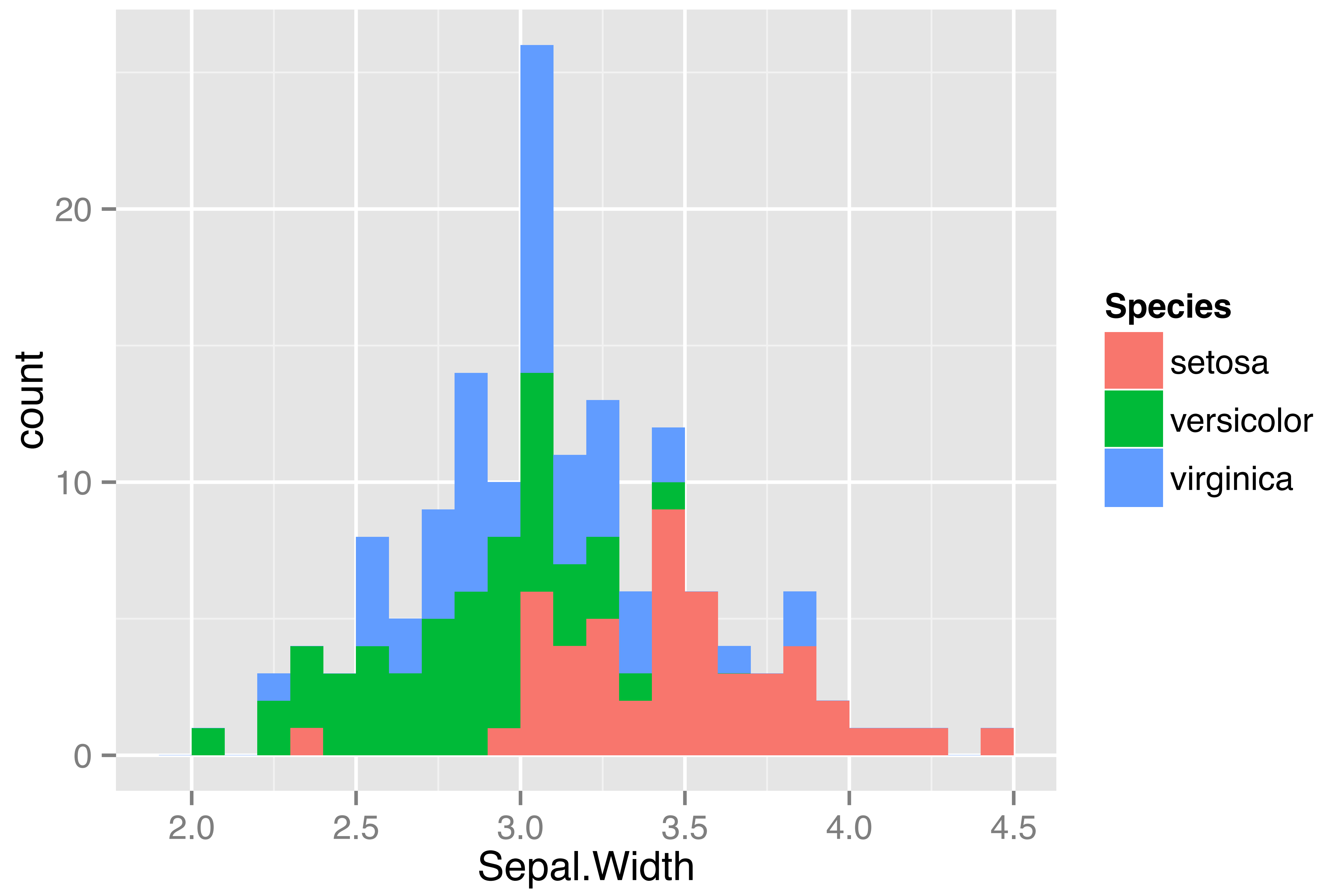
Different Species

```
> ggplot(iris, aes(x = Sepal.Width, fill = Species)) +
    geom_histogram(binwidth = 0.1)
```



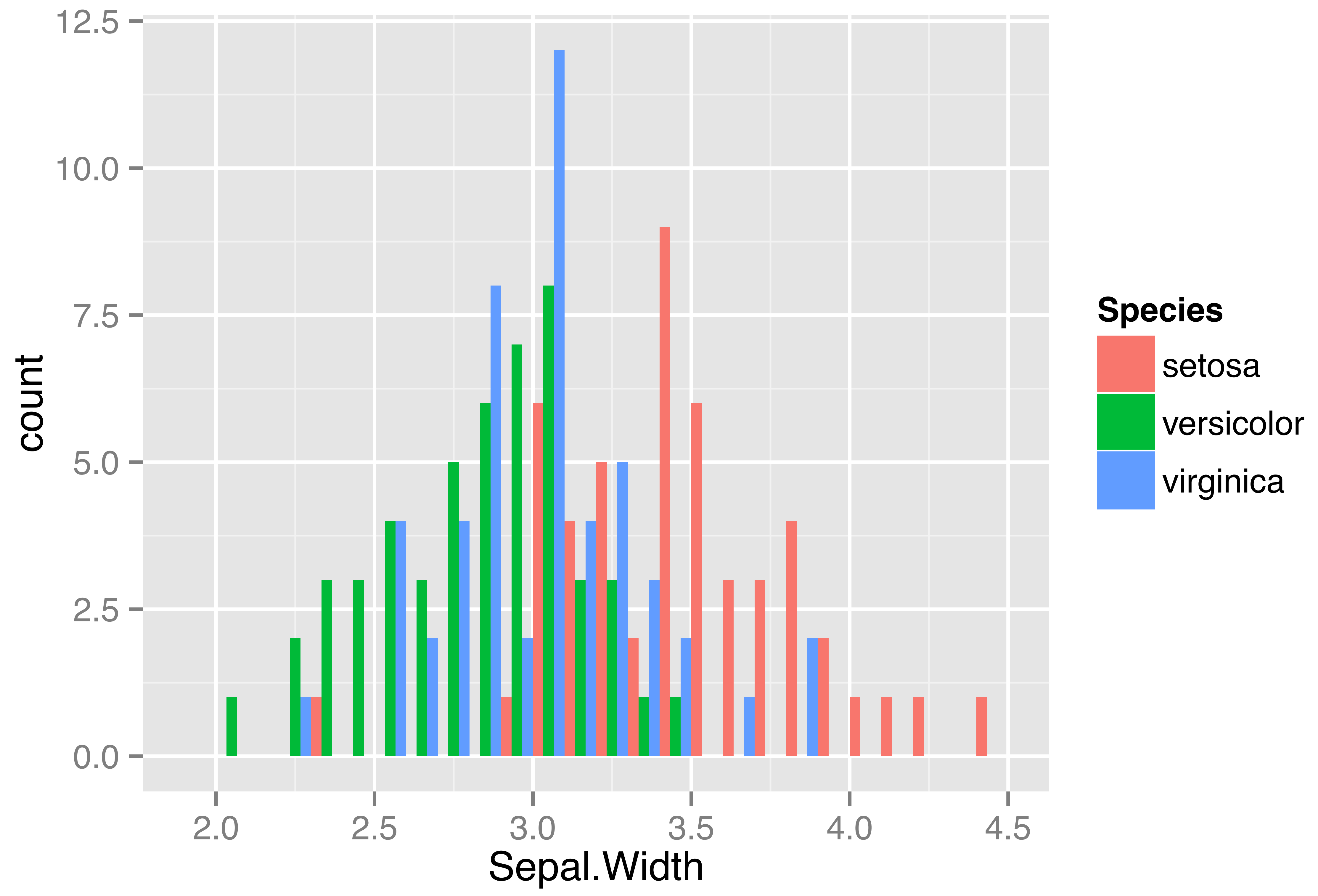
Different Species

```
> ggplot(iris, aes(x = Sepal.Width, fill = Species)) +
    geom_histogram(binwidth = 0.1, position = "stack")
```



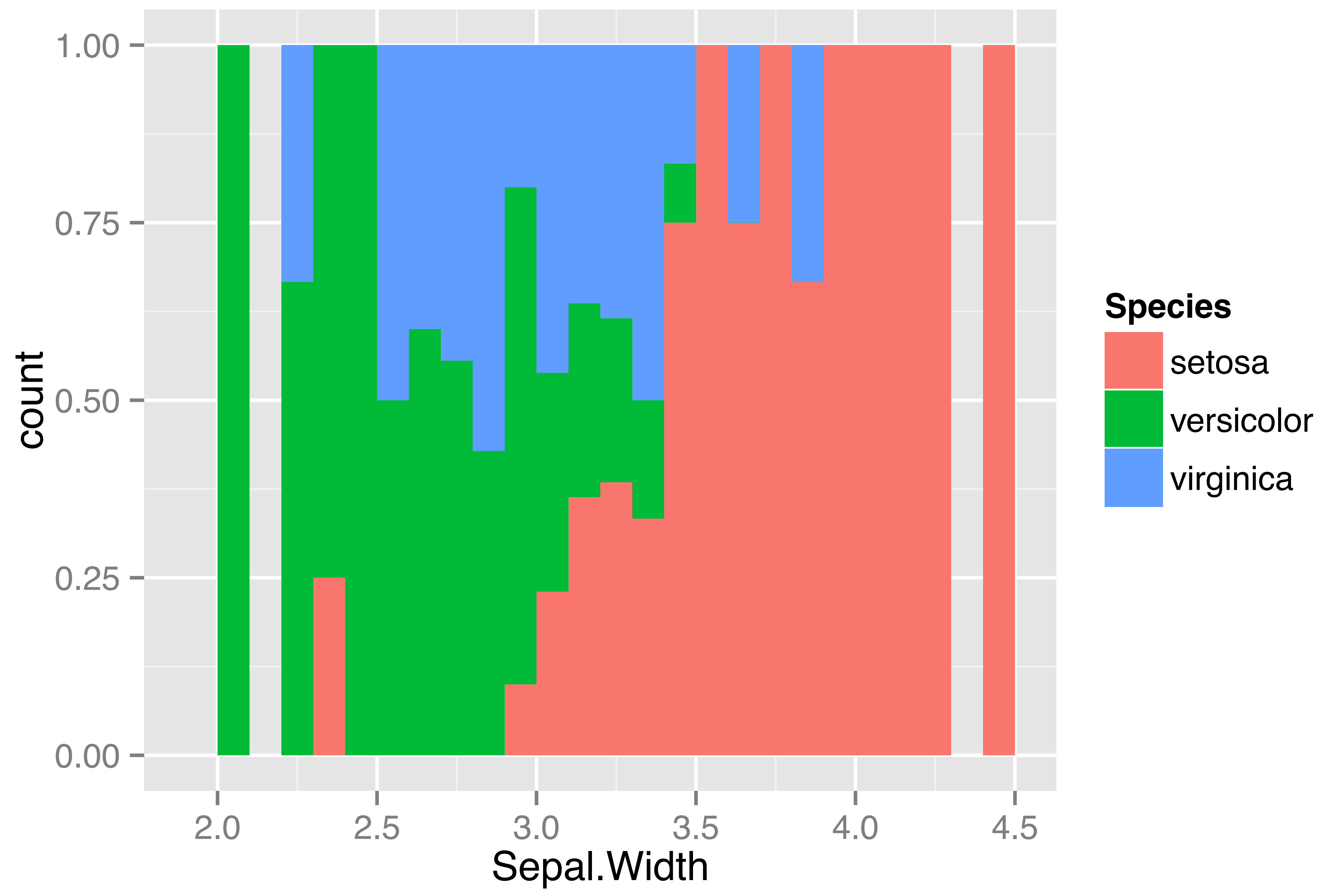
dodge

```
> ggplot(iris, aes(x = Sepal.Width, fill = Species)) +
    geom_histogram(binwidth = 0.1, position = "dodge")
```



fill

```
> ggplot(iris, aes(x = Sepal.Width, fill = Species)) +
    geom_histogram(binwidth = 0.1, position = "fill")
```

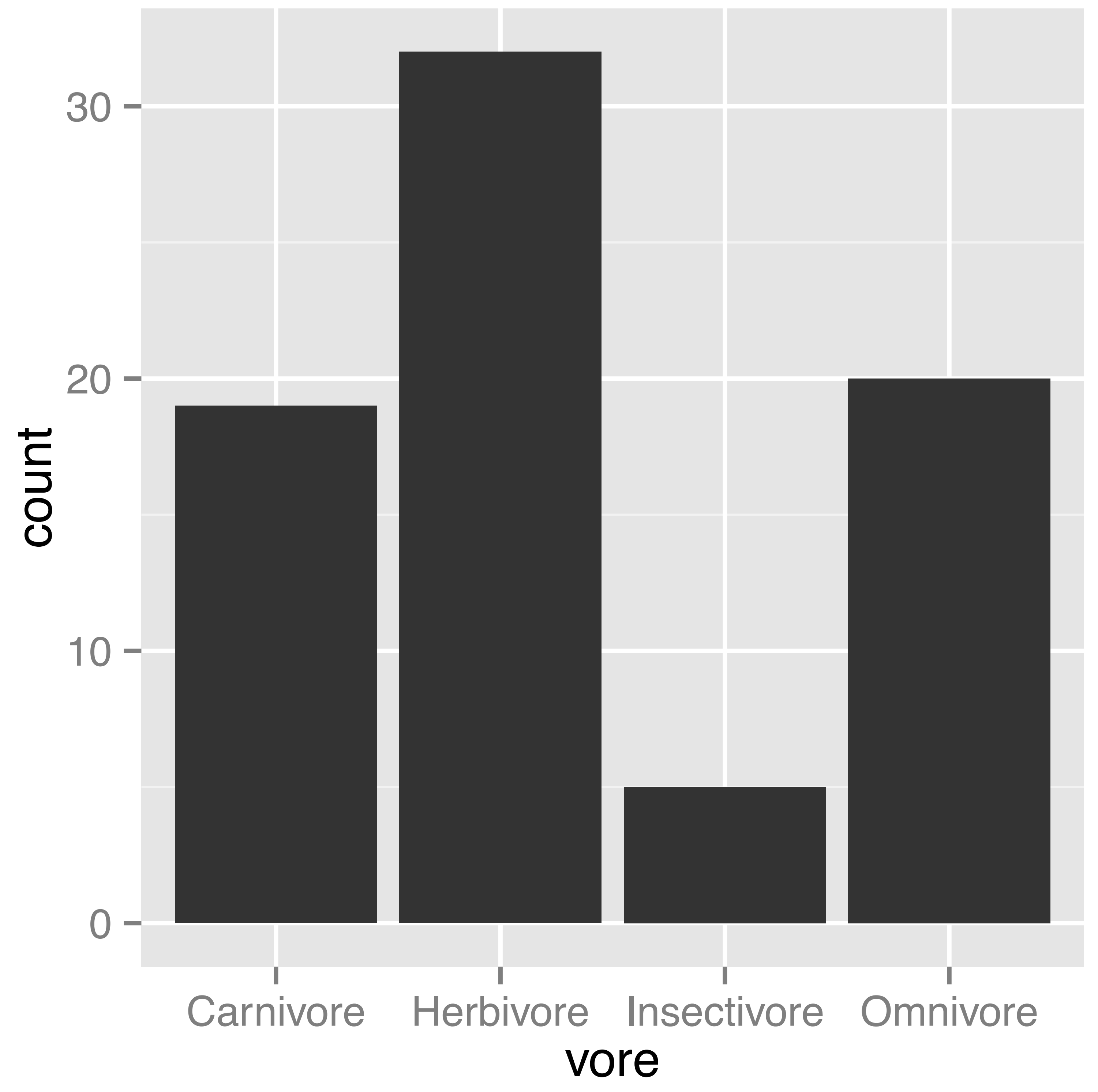


Bar Plot

- `geom_bar()`
- All positions from before available
- Two types
    - Absolute counts
    - Distributions

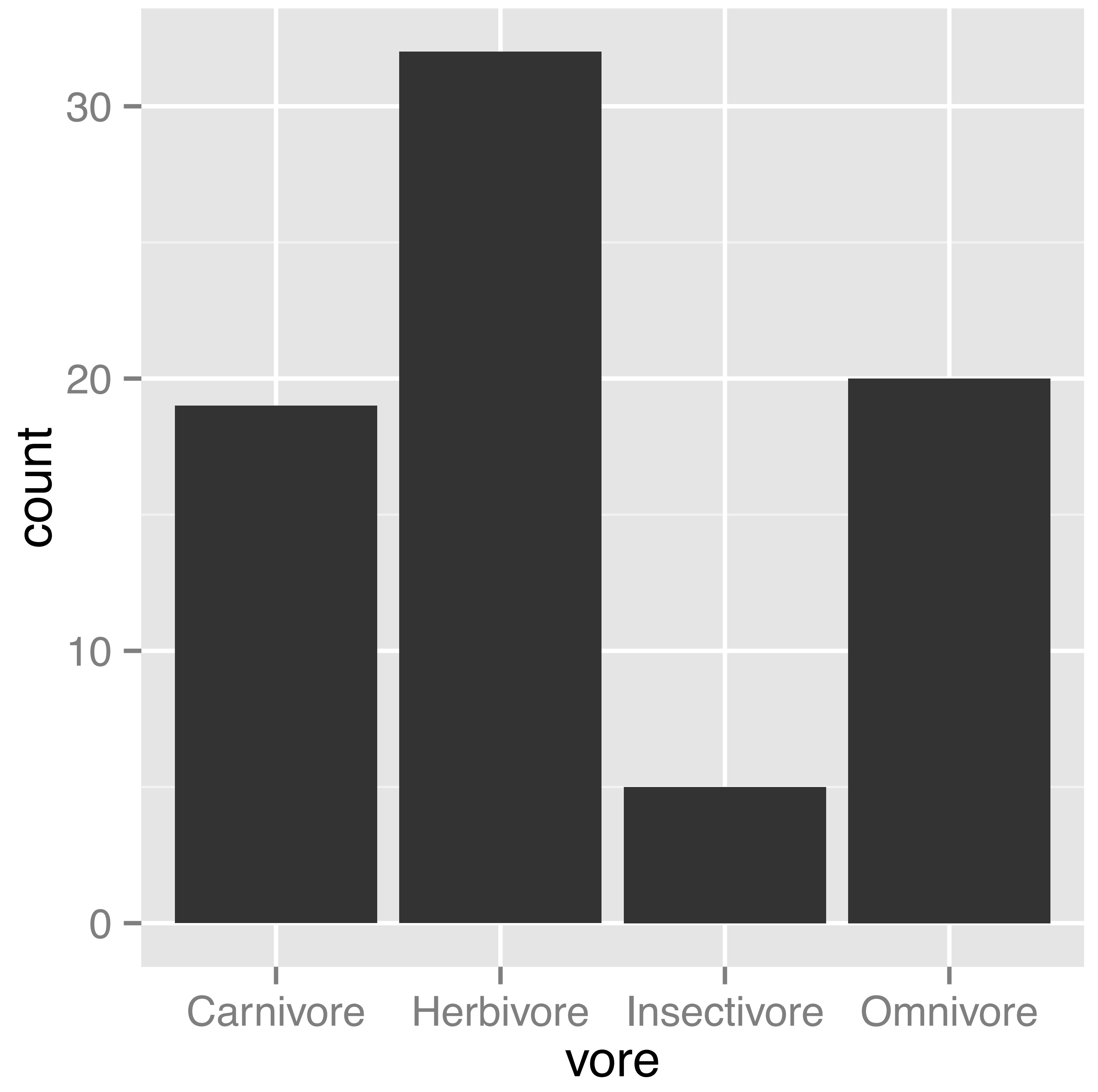
Habits of Mammals

```
> # Cleaning up of sleep left out

> ggplot(sleep, aes(vore)) + geom_bar()

> str(sleep)
'data.frame':76 obs. of  3 variables:
 $ vore : Factor w/ 4 levels "Carnivore","Herbivore",..
 $ total: num  12.1 17 14.4 14.9 4 14.4 8.7 10.1 3 5.3 ...
 $ rem  : num  NA 1.8 2.4 2.3 0.7 2.2 1.4 2.9 NA 0.6 ...
```



Habits of Mammals

```
> # Cleaning up of sleep left out

> ggplot(sleep, aes(vore)) + geom_bar(stat = "bin")

> str(sleep)
'data.frame':76 obs. of  3 variables:
 $ vore : Factor w/ 4 levels "Carnivore","Herbivore",..
 $ total: num  12.1 17 14.4 14.9 4 14.4 8.7 10.1 3 5.3 ...
 $ rem  : num  NA 1.8 2.4 2.3 0.7 2.2 1.4 2.9 NA 0.6 ...
```

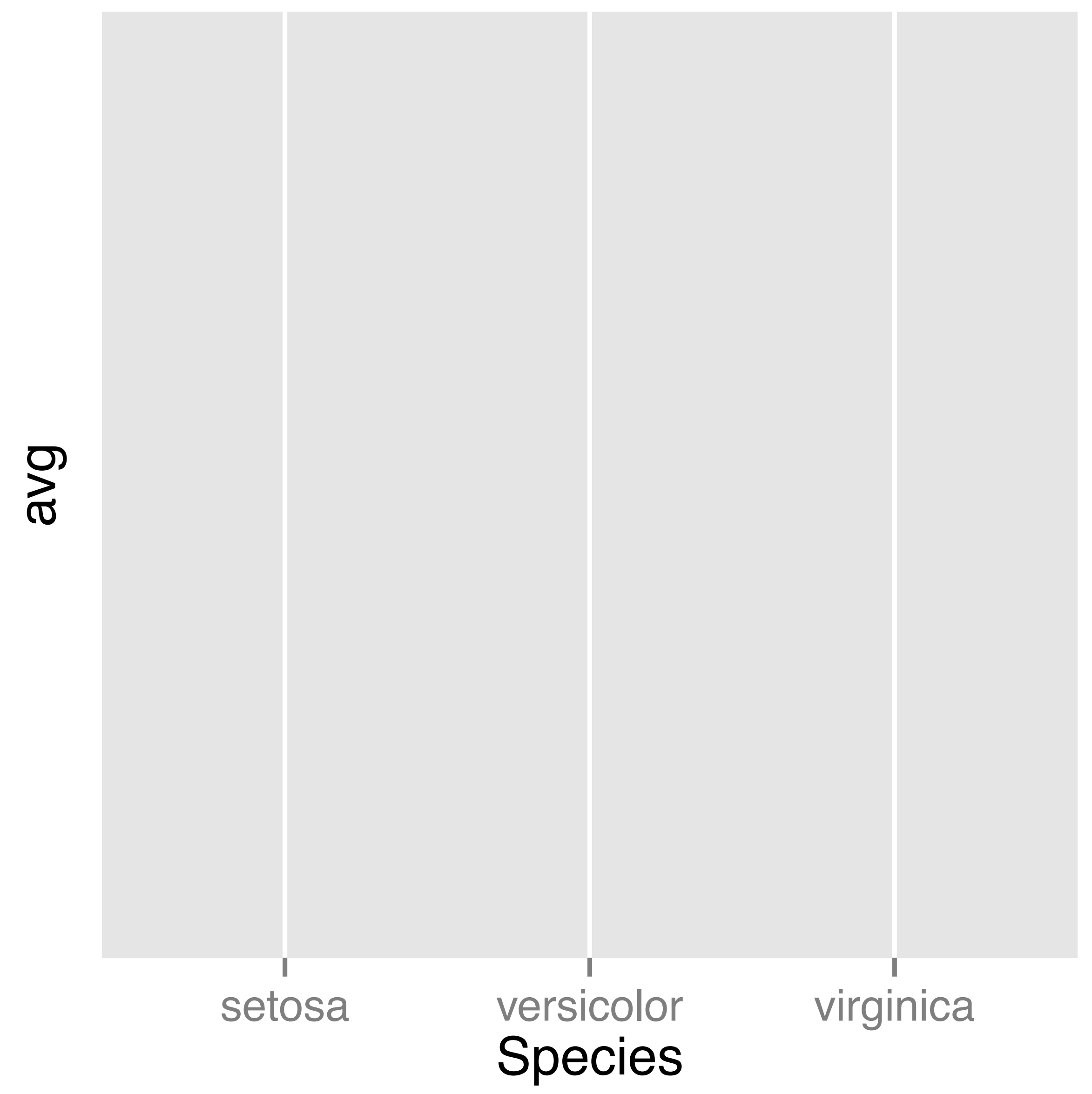


Distribution Bar Plots

```
> library(plyr)
> iris_melted <- melt(iris, value.name = "Value",
                      variable.name = "Measure")
> iris_summ <- ddply(iris_melted[iris_melted$Measure == "Sepal.Width",],
                     "Species", summarise, avg = mean(Value),
                     stdev = sd(Value))

> str(iris_summ)
'data.frame':3 obs. of  3 variables:
 $ Species: Factor w/ 3 levels "setosa","versicolor",..: 1 2 3
 $ avg    : num  3.43 2.77 2.97
 $ stdev  : num  0.379 0.314 0.322
```

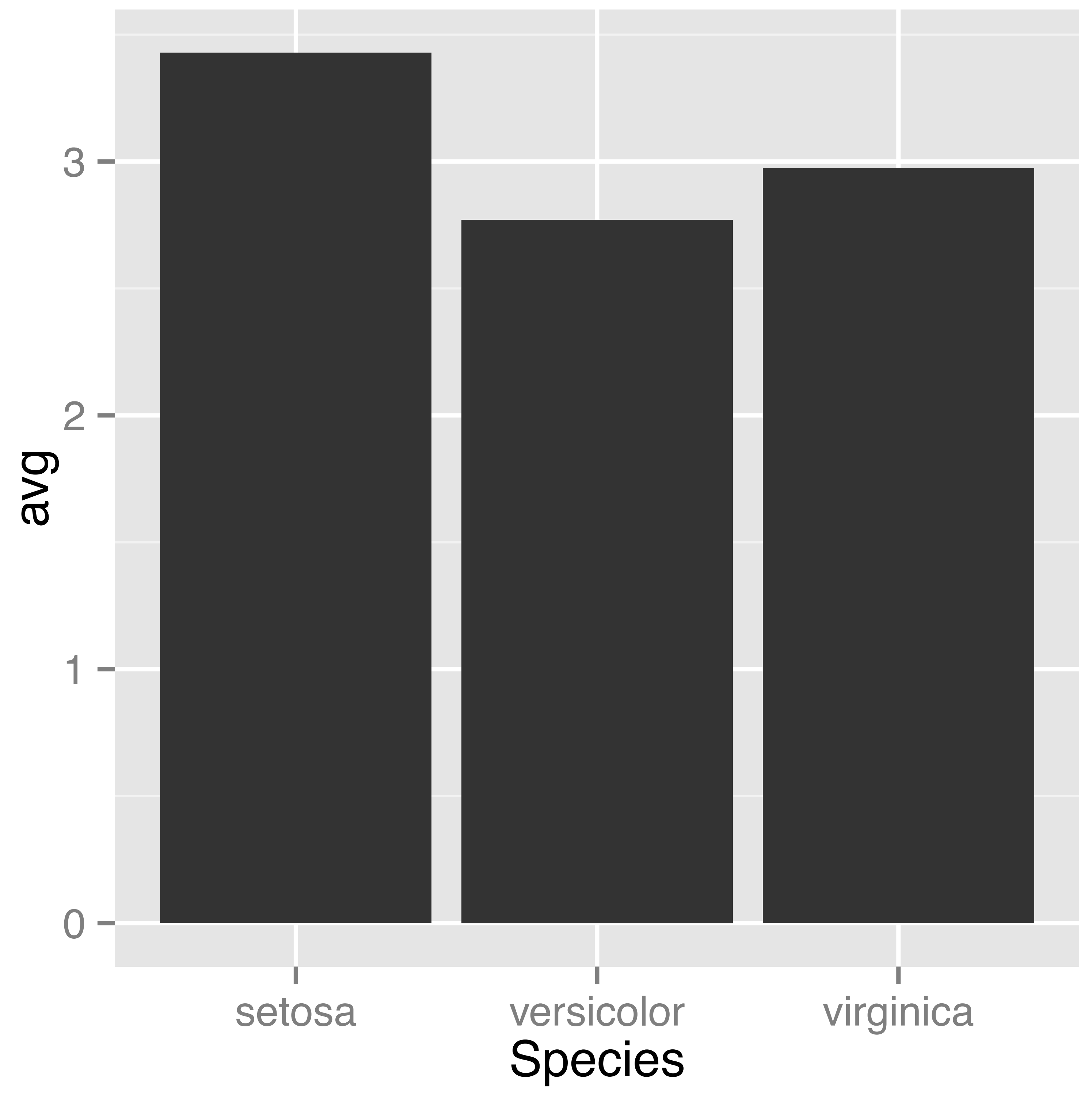
First try

```
> ggplot(iris_summ, aes(x = Species, y = avg)) +
    geom_bar()
Error : Mapping a variable to y and also using stat="bin".
  With stat="bin", it will attempt to set the y value to the count of cases in each group.
  This can result in unexpected behavior and will not be allowed in a future version of ggplot2.
  If you want y to represent counts of cases, use stat="bin" and don't map a variable to y.
  If you want y to represent values in the data, use stat="identity".
  See ?geom_bar for examples. (Defunct; last used in version 0.9.2)
```



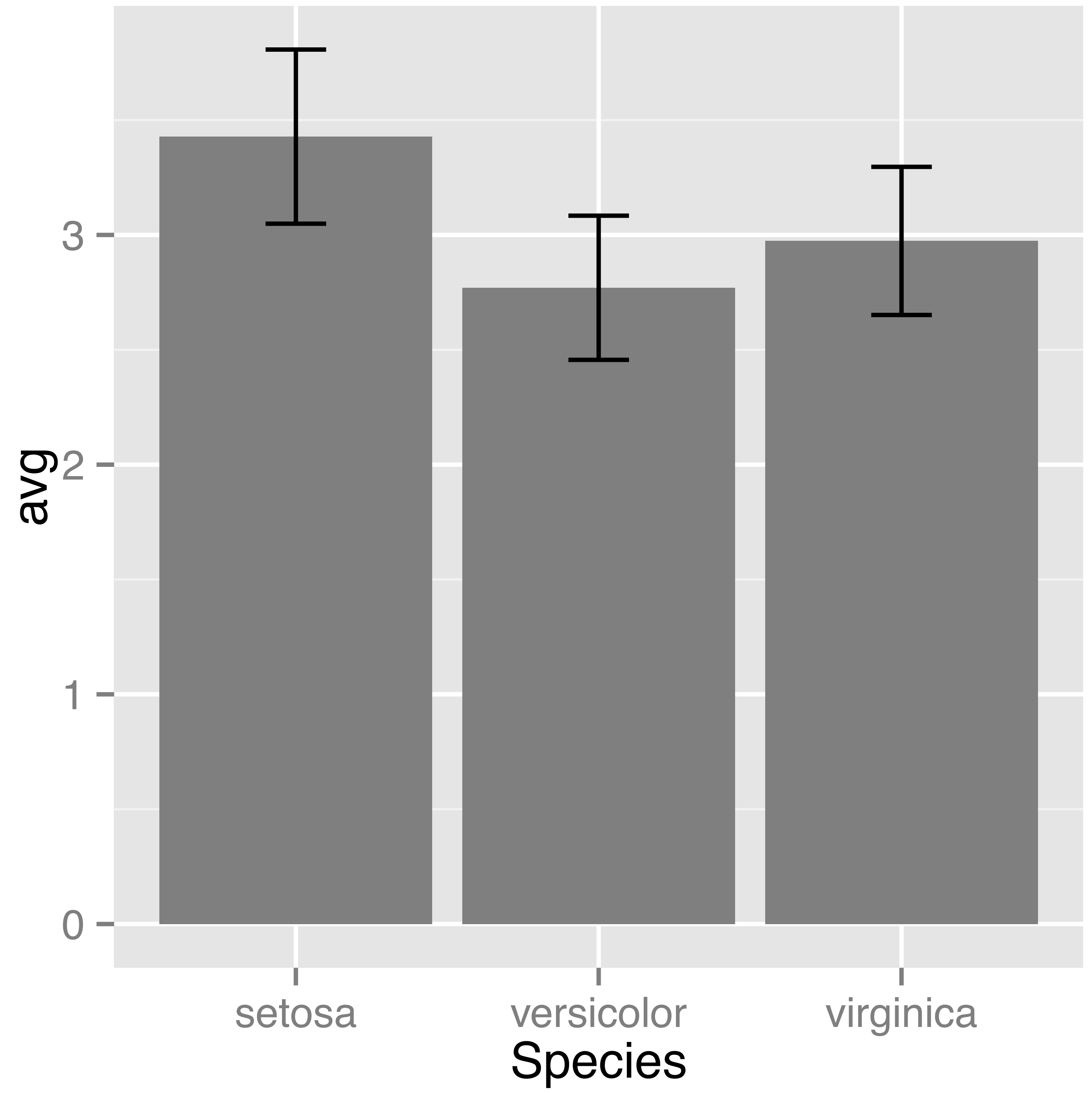
identity

```
> ggplot(iris_summ, aes(x = Species, y = avg)) +
    geom_bar(stat = "identity")
```



geom_errorbar

```
> ggplot(iris_summ, aes(x = Species, y = avg)) +
    geom_bar(stat = "identity", fill = "grey50") +
    geom_errorbar(aes(ymin = avg - stdev, ymax = avg + stdev),
                  width = 0.2)
```



Dynamite Plot

Let's practice!

Line Plots
Time Series

Common plot types

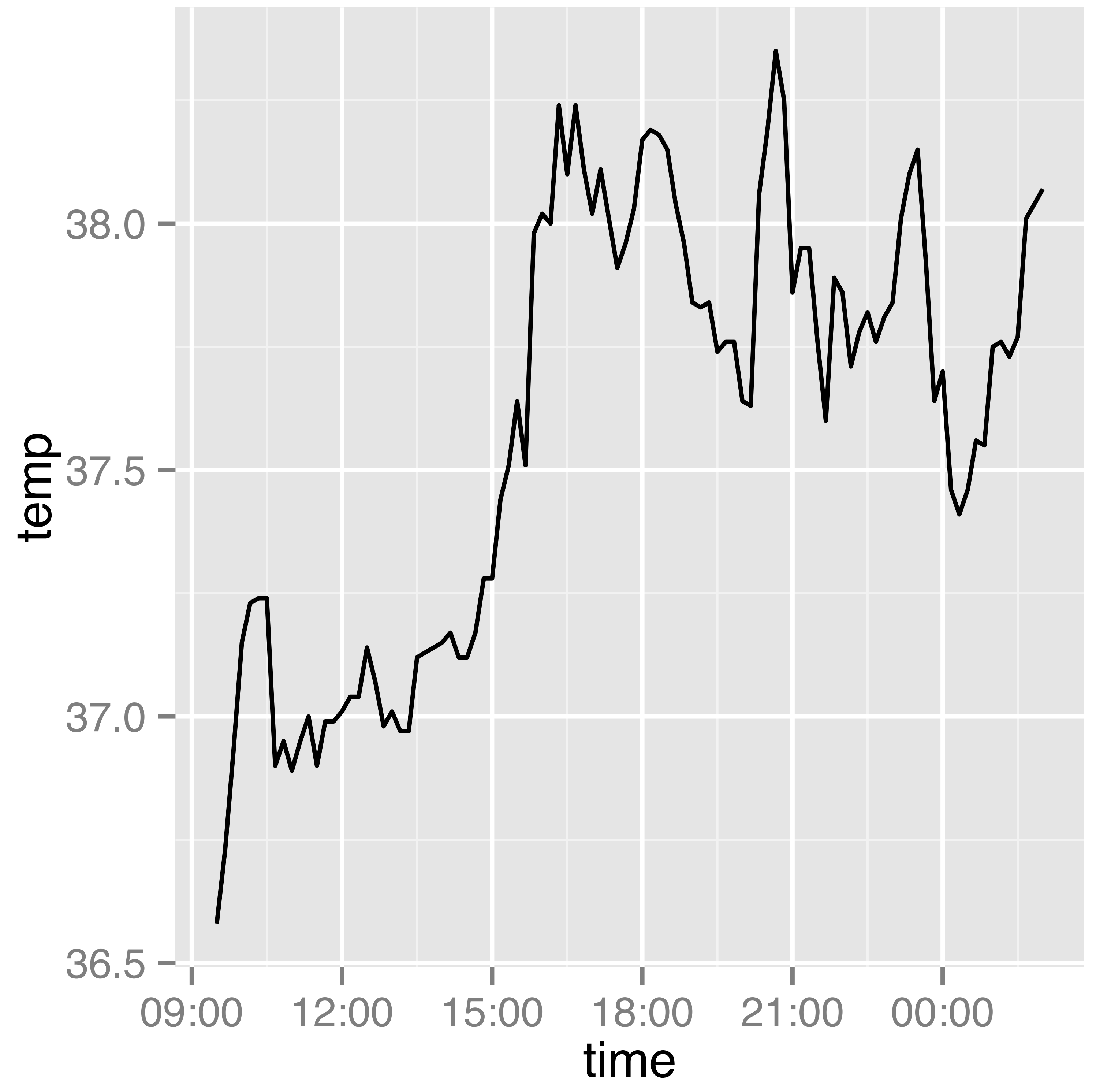
- Scatter plots
    - points, jitter, abline
- Bar plots
    - histogram, bar, errorbar
- **Line plots**
    - **line**

Beaver

```
> str(beaver)
'data.frame':101 obs. of  3 variables:
 $ time  : POSIXct, format: "2000-01-01 09:30:00" "2000-01-01 ...
 $ temp  : num  36.6 36.7 36.9 37.1 37.2 ...
 $ active: Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
```

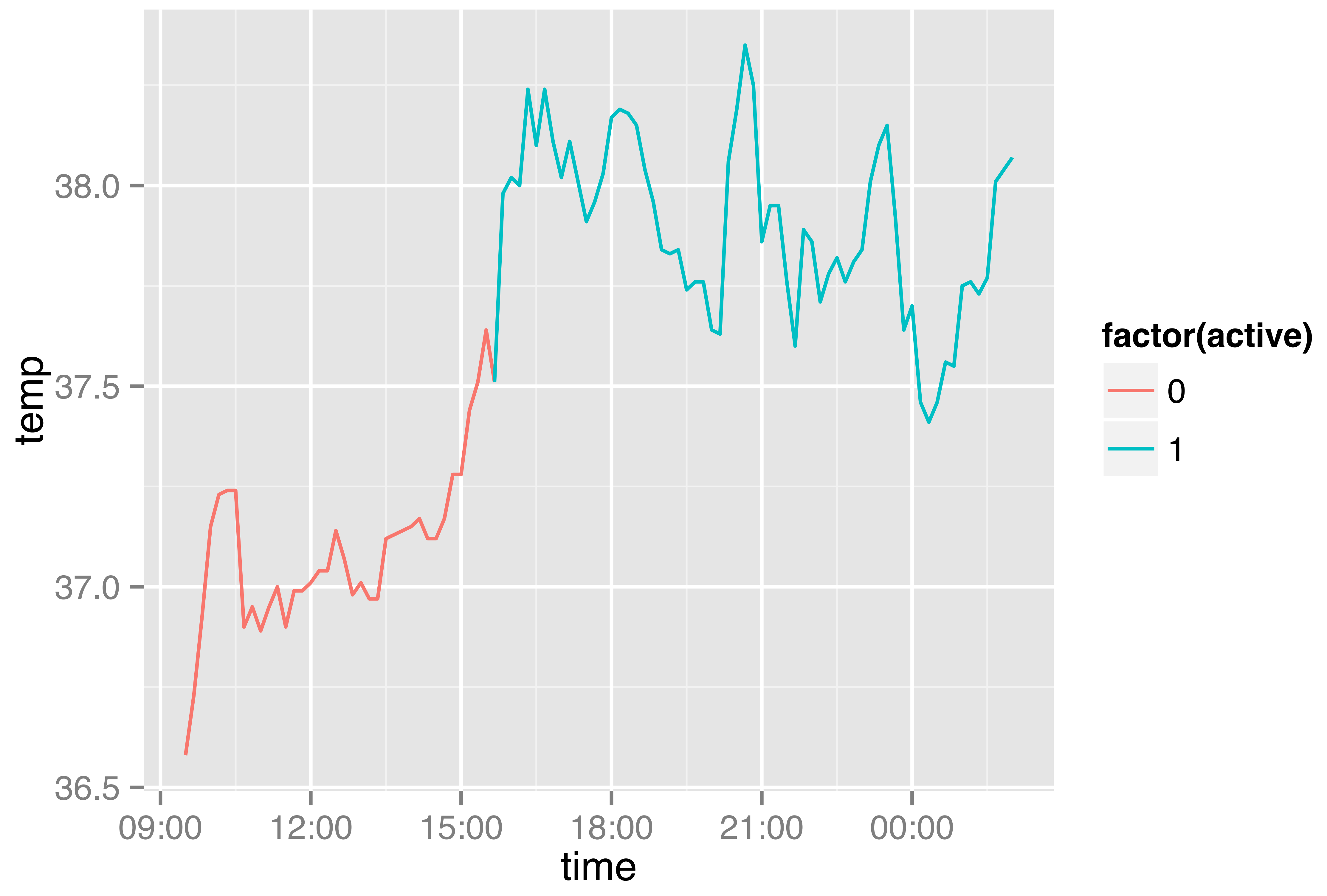
Beaver

```
> ggplot(beaver, aes(x = time, y = temp)) +
    geom_line()
```

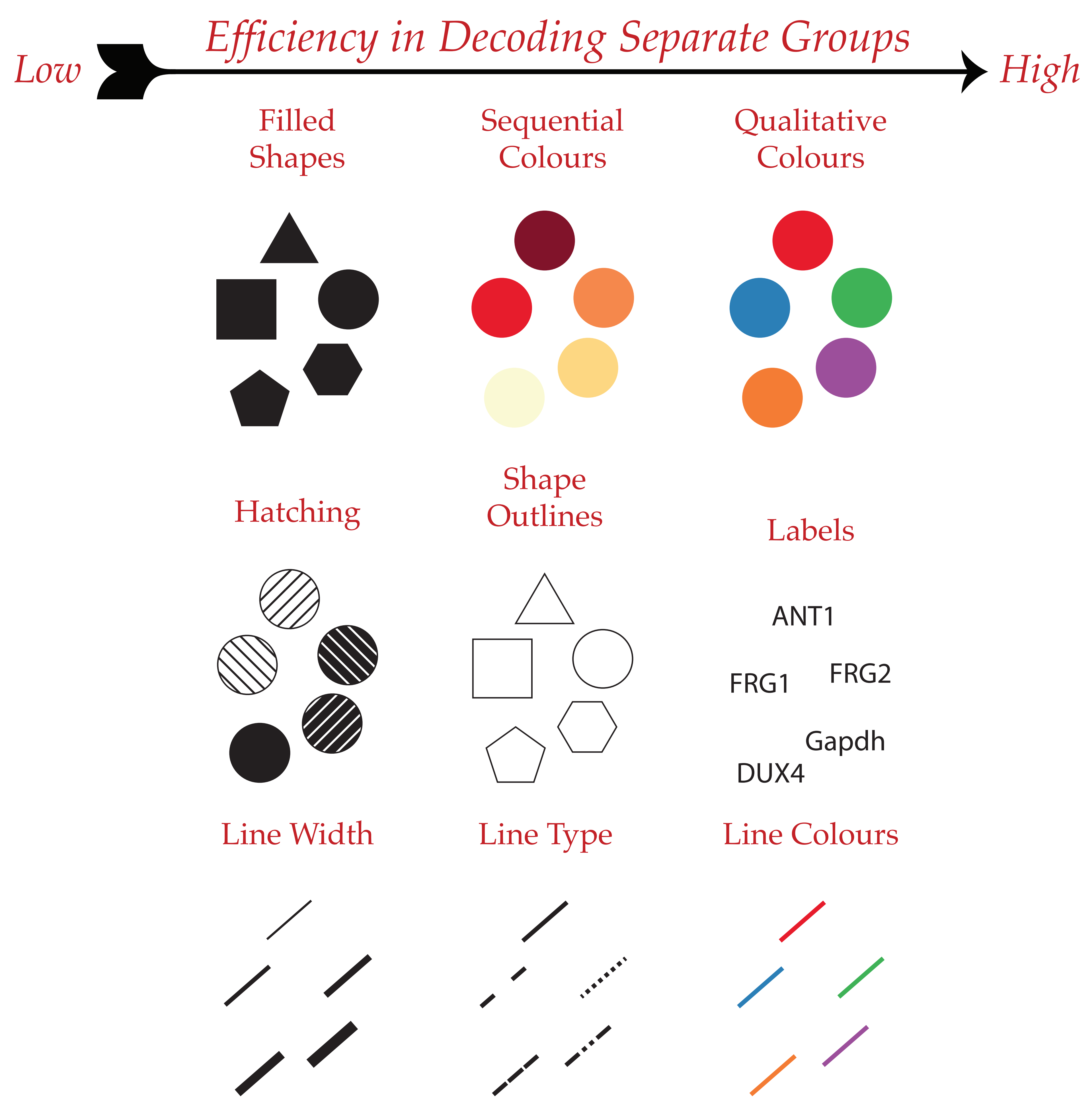


Beaver

```
> ggplot(beaver, aes(x = time, y = temp, col = factor(active))) +
    geom_line()
```

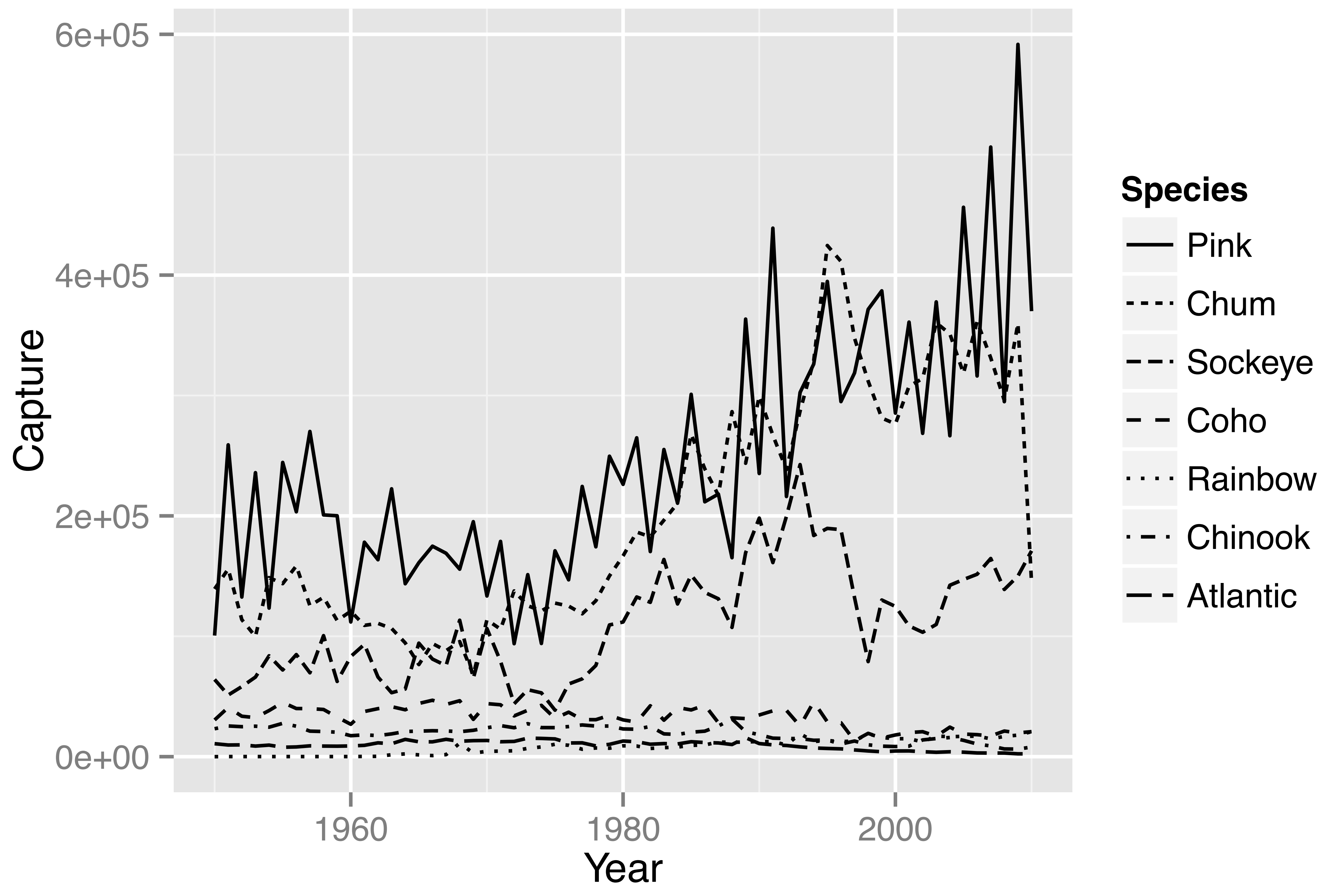


Aesthetics - categorical variables



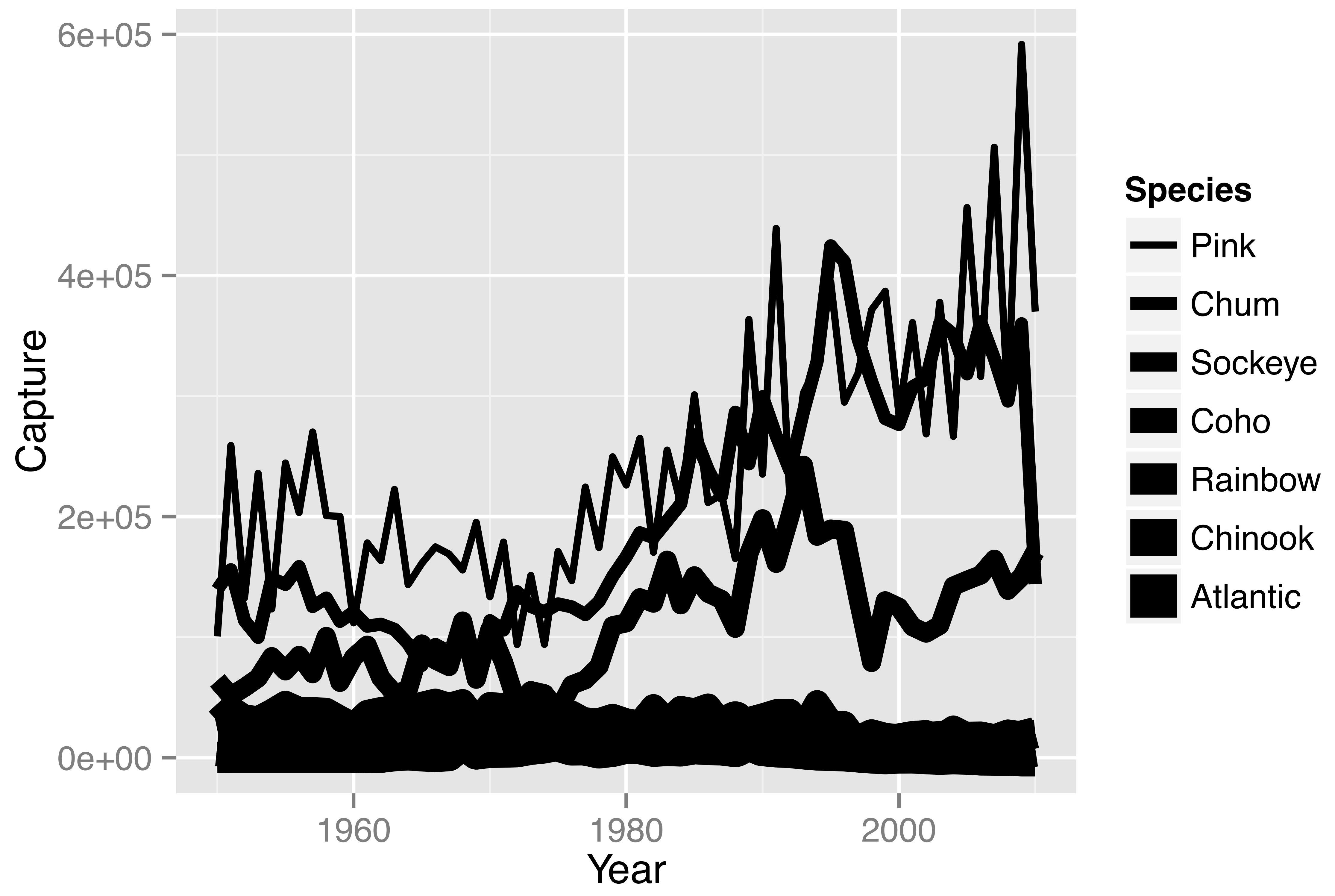
linetype

```
> ggplot(fish, aes(x = Year, y = Capture, linetype = Species)) +
    geom_line()
> names(fish)
[1] "Species" "Year"    "Capture"
```



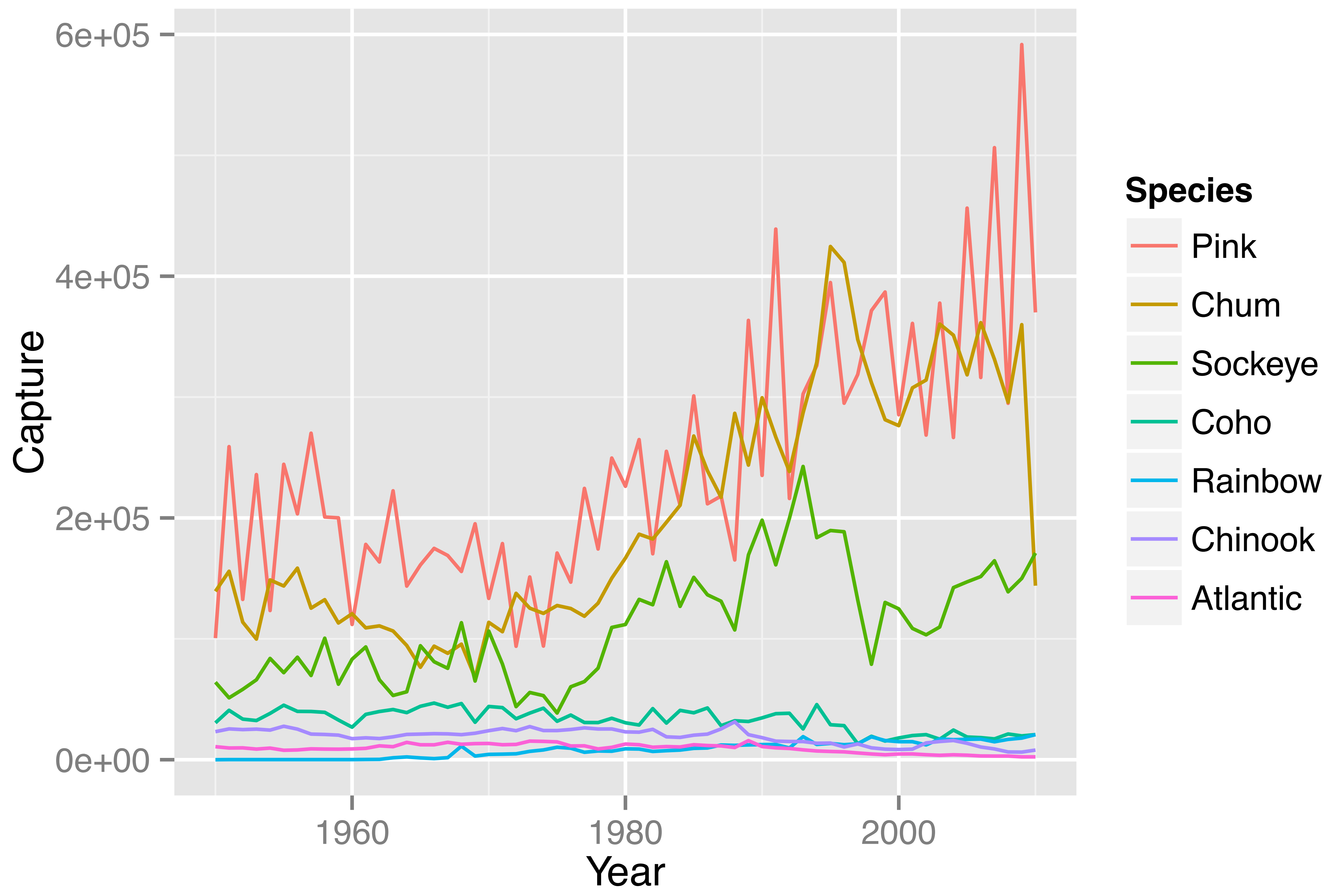
size

```
> ggplot(fish, aes(x = Year, y = Capture, size = Species)) +
    geom_line()
```



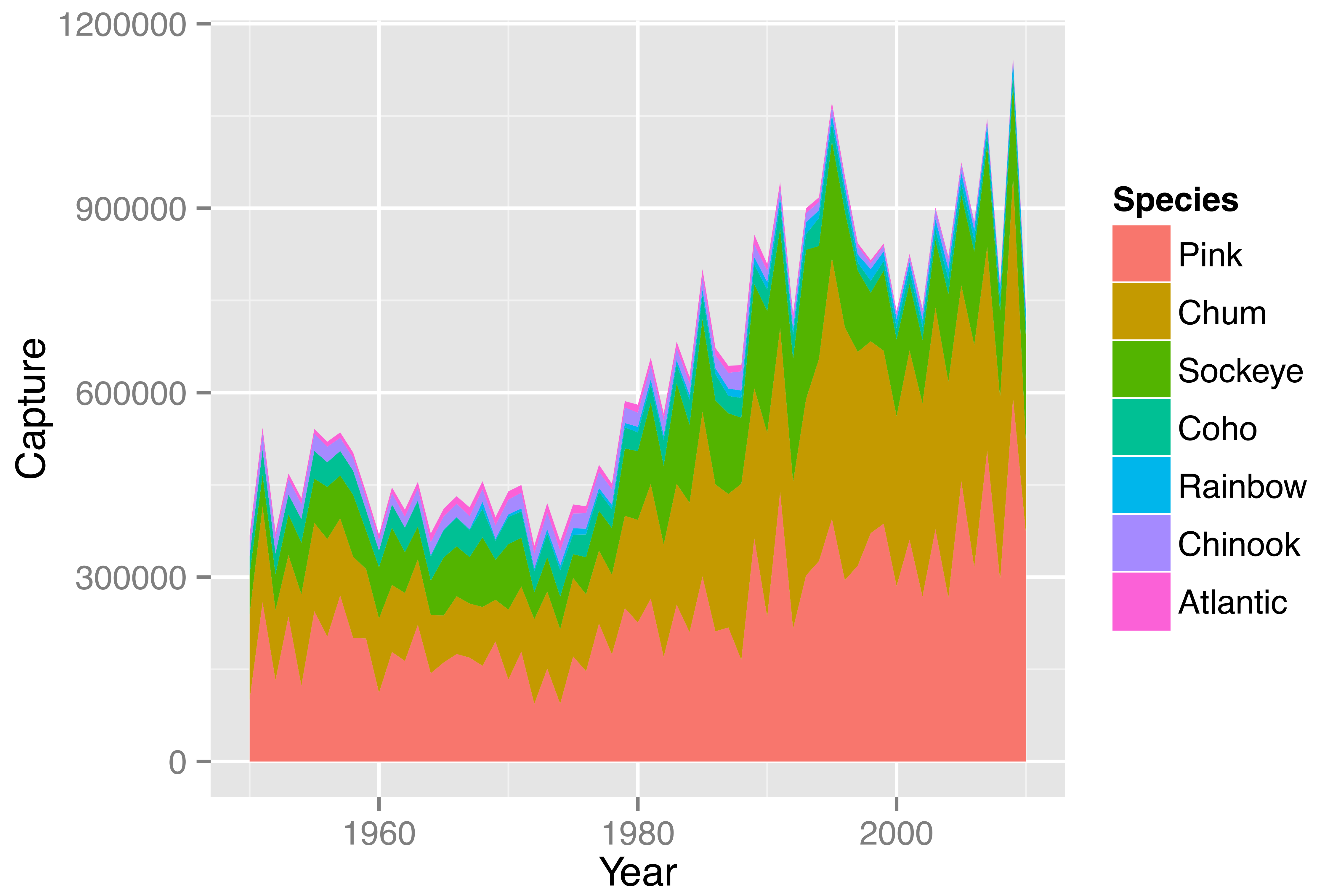
color

```
> ggplot(fish, aes(x = Year, y = Capture, color = Species)) +
    geom_line()
```



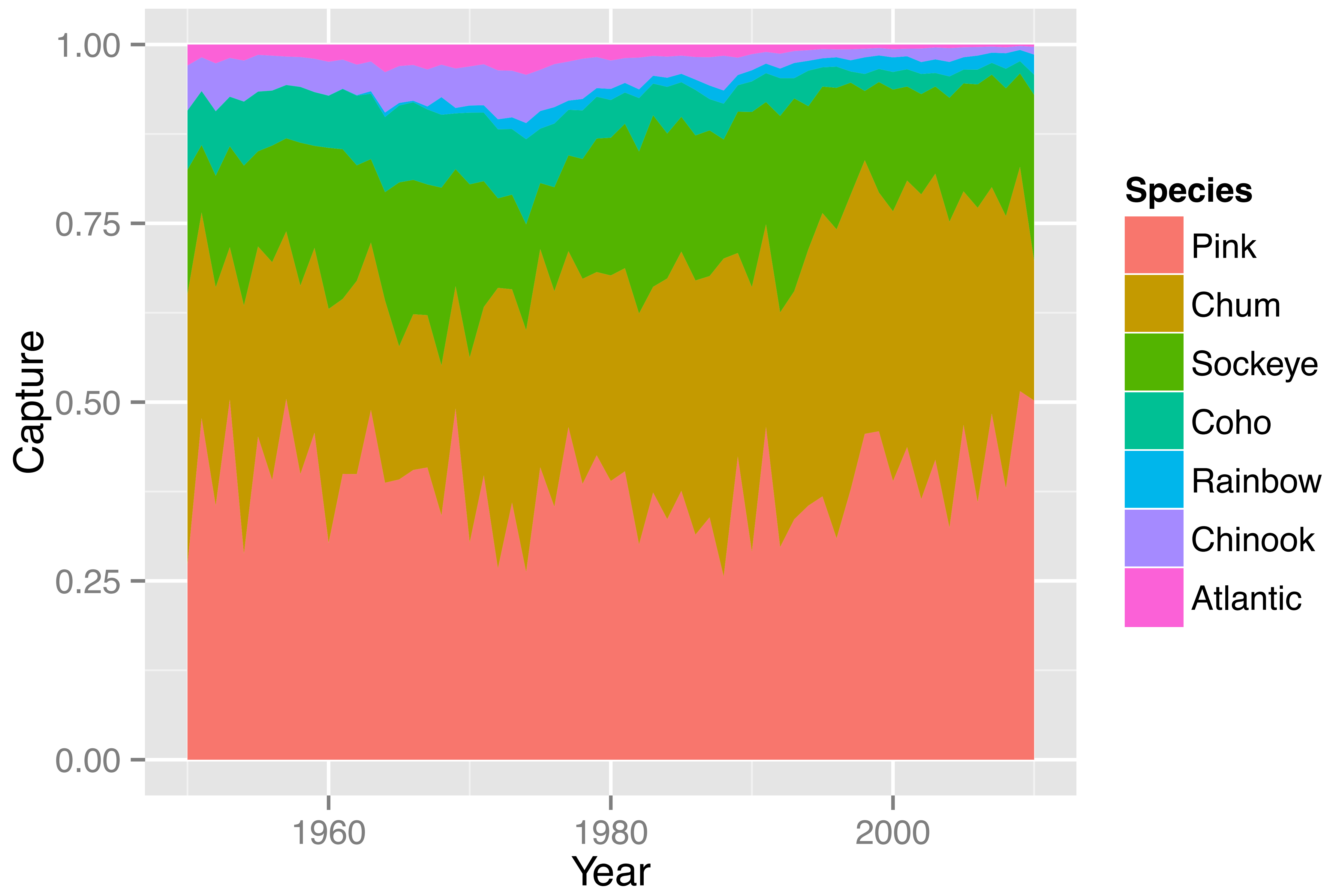
fill

```
> ggplot(fish, aes(x = Year, y = Capture, fill = Species)) +
    geom_area()
```



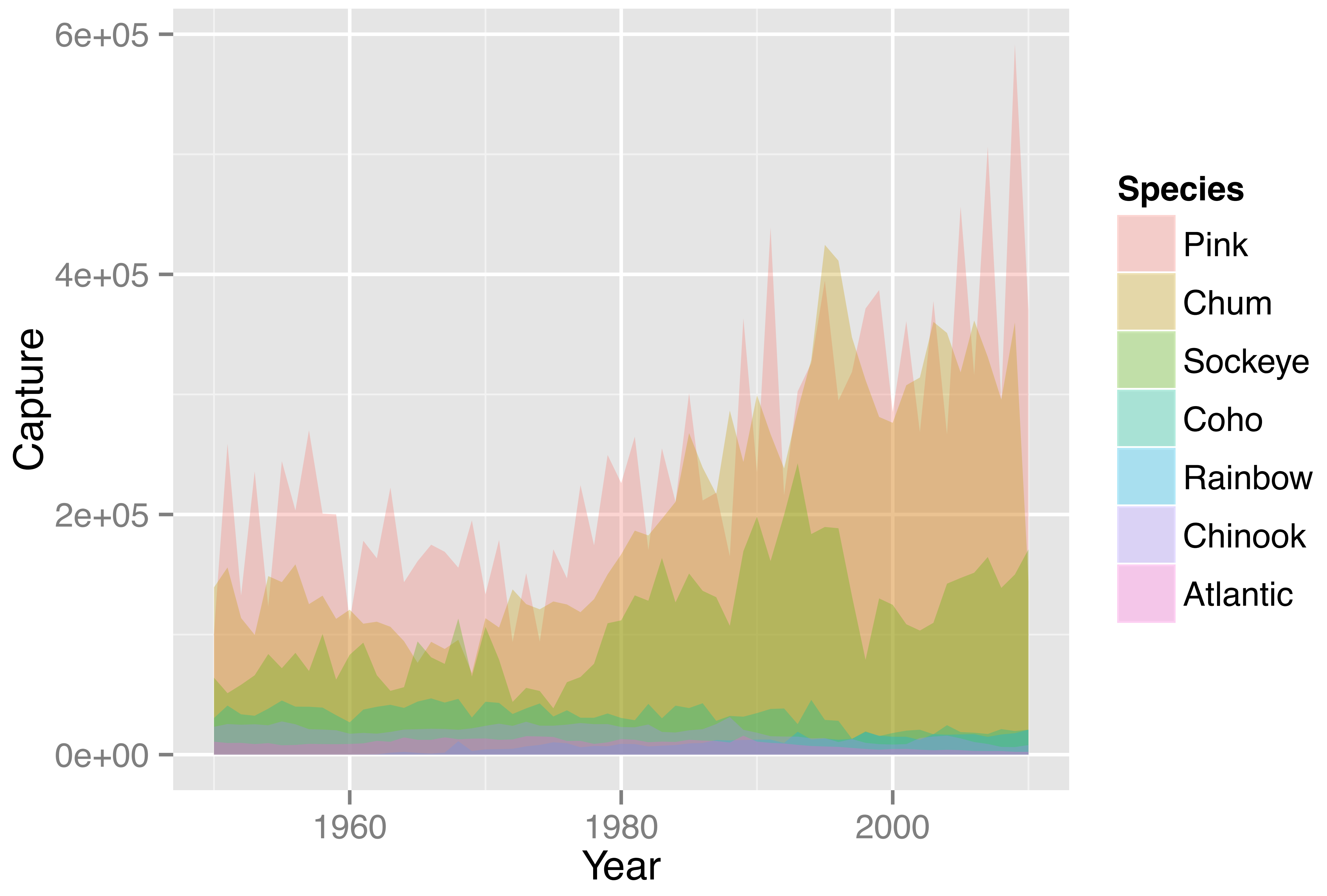
fill (2)

```
> ggplot(fish, aes(x = Year, y = Capture, fill = Species)) +
    geom_area(position = "fill")
```



geom_ribbon

```
> ggplot(fish, aes(x = Year, y = Capture, fill = Species)) +
    geom_ribbon(aes(ymax = Capture, ymin = 0), alpha = 0.3)
```



Let's practice!