Code

R Notebook

Data Visualization in ggplot

Penn SAS Data Driven Discovery Summer Hangouts 2023

Brynn Sherman (brynns@sas.upenn.edu (mailto:brynns@sas.upenn.edu))

DDDI postdoctoral fellow | Department of Psychology

Load in the relevant packages

Hide

```
library(tidyverse)
library(datasets)
```

Check out the Iris dataset

Hide

```
?iris
```

Hide

```
iris
```

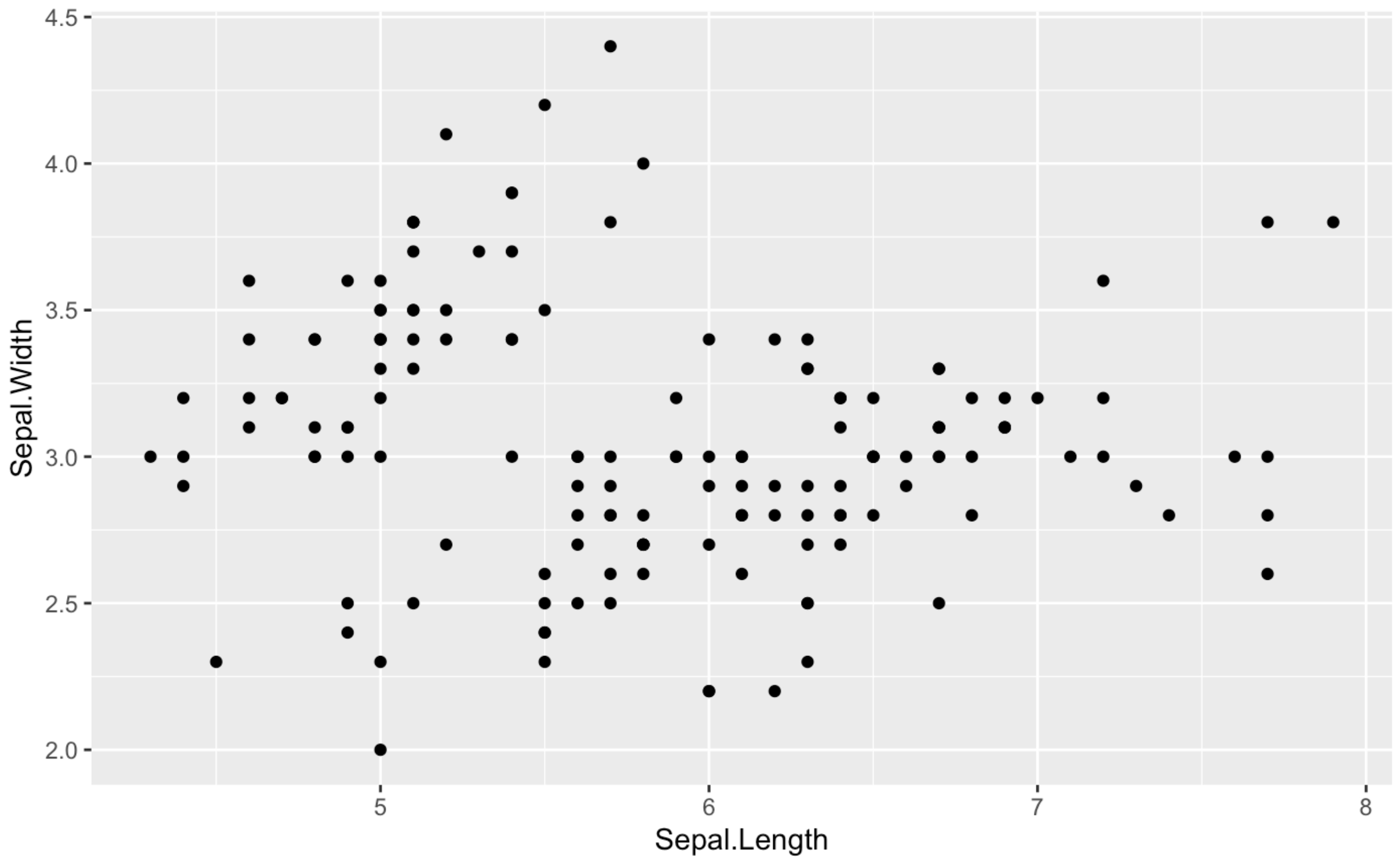
Sepal.Length <dbl>	Sepal.Width <dbl>	Petal.Length <dbl>	Petal.Width <dbl>	Species <fctr>
5.1	3.5	1.4	0.2	setosa
4.9	3.0	1.4	0.2	setosa
4.7	3.2	1.3	0.2	setosa
4.6	3.1	1.5	0.2	setosa
5.0	3.6	1.4	0.2	setosa
5.4	3.9	1.7	0.4	setosa
4.6	3.4	1.4	0.3	setosa
5.0	3.4	1.5	0.2	setosa
4.4	2.9	1.4	0.2	setosa
4.9	3.1	1.5	0.1	setosa

1-10 of 150 rows   Previous **1** 2 3 4 5 6 ... 15 Next

Plot sepal length and sepal width against one another

Hide

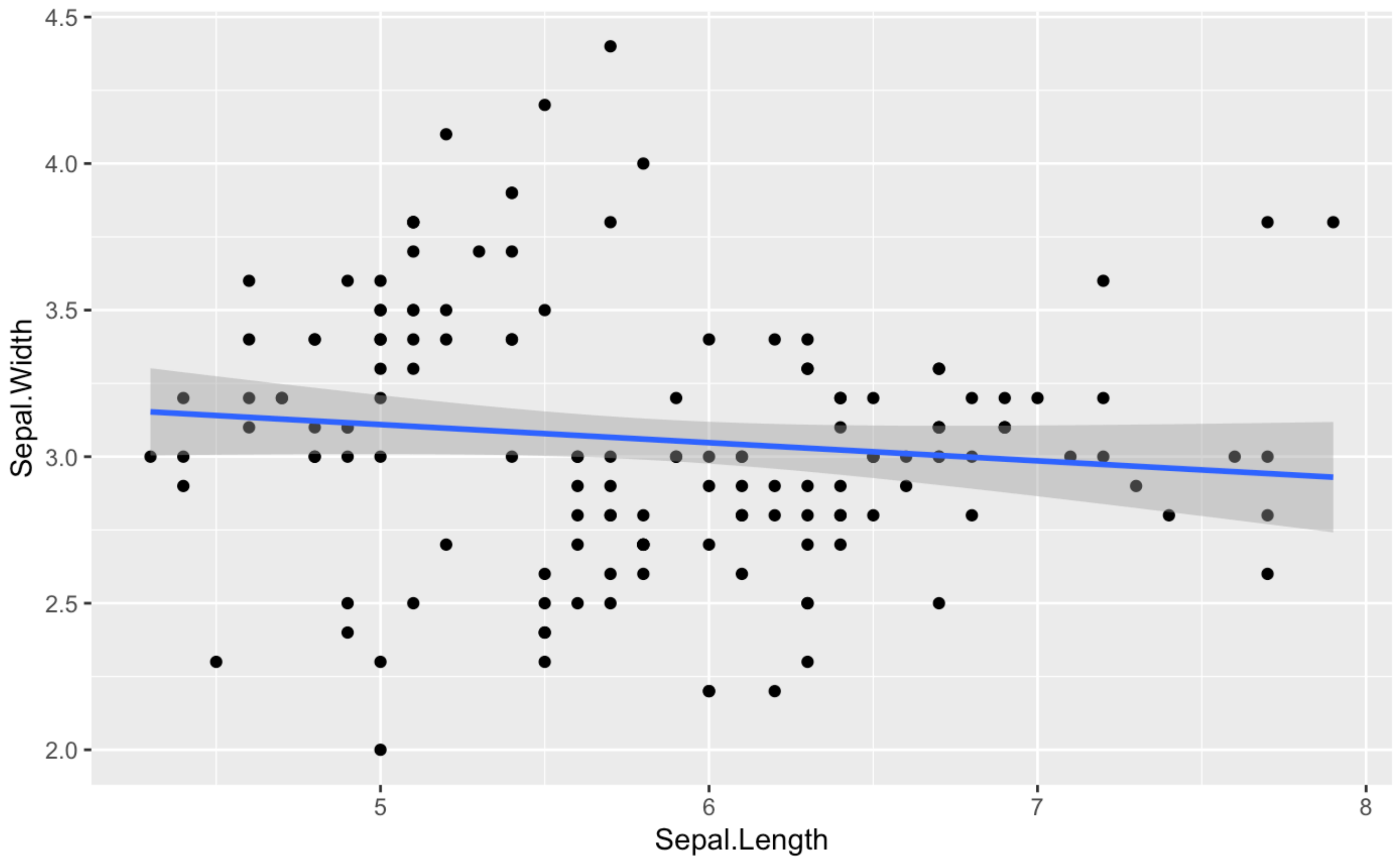
```
ggplot(data = iris, aes(x = Sepal.Length, y = Sepal.Width)) + geom_point()
```



Add a line of best fit

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

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



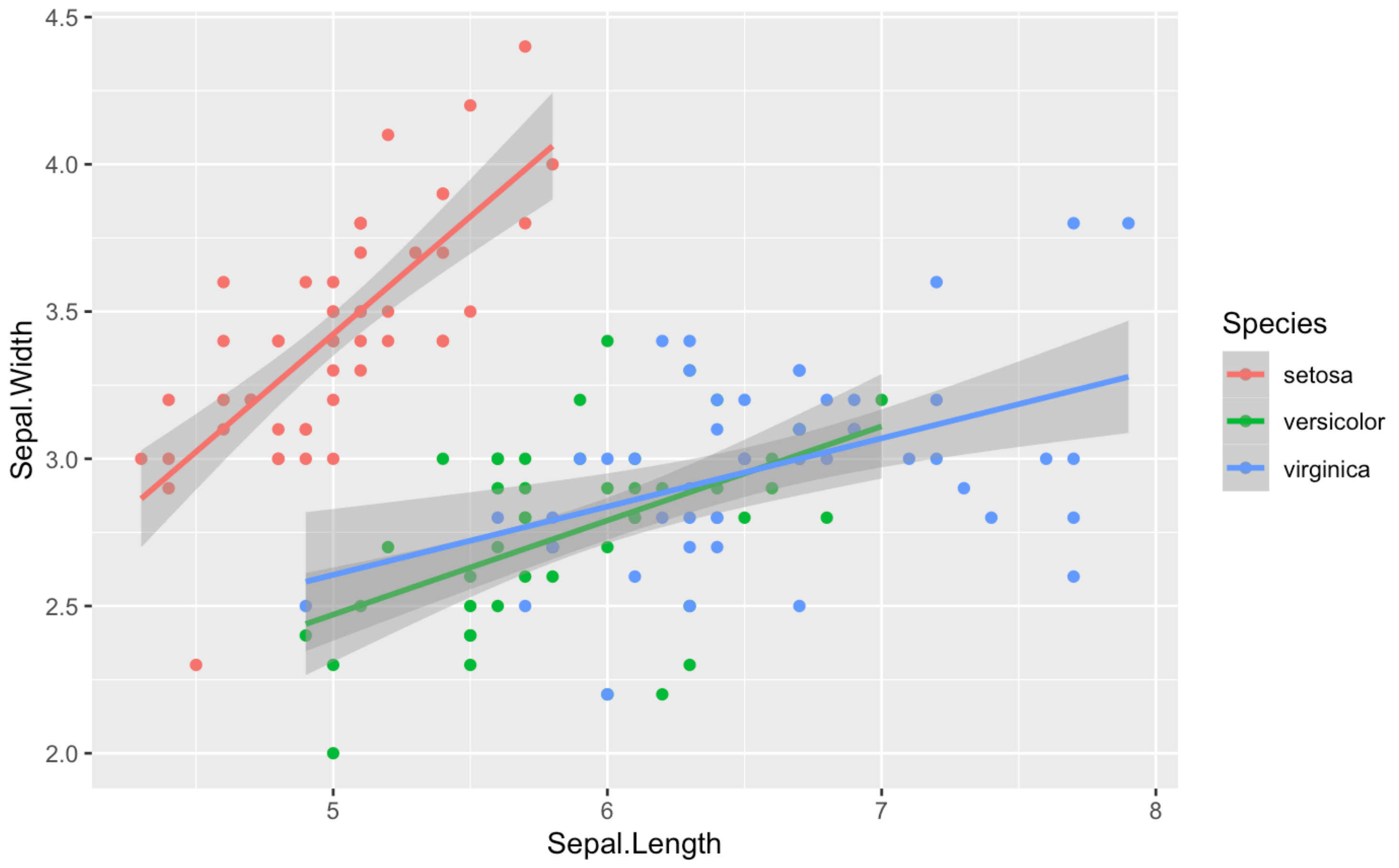
Let’s separate out based on species

First, by color:

Hide

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

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

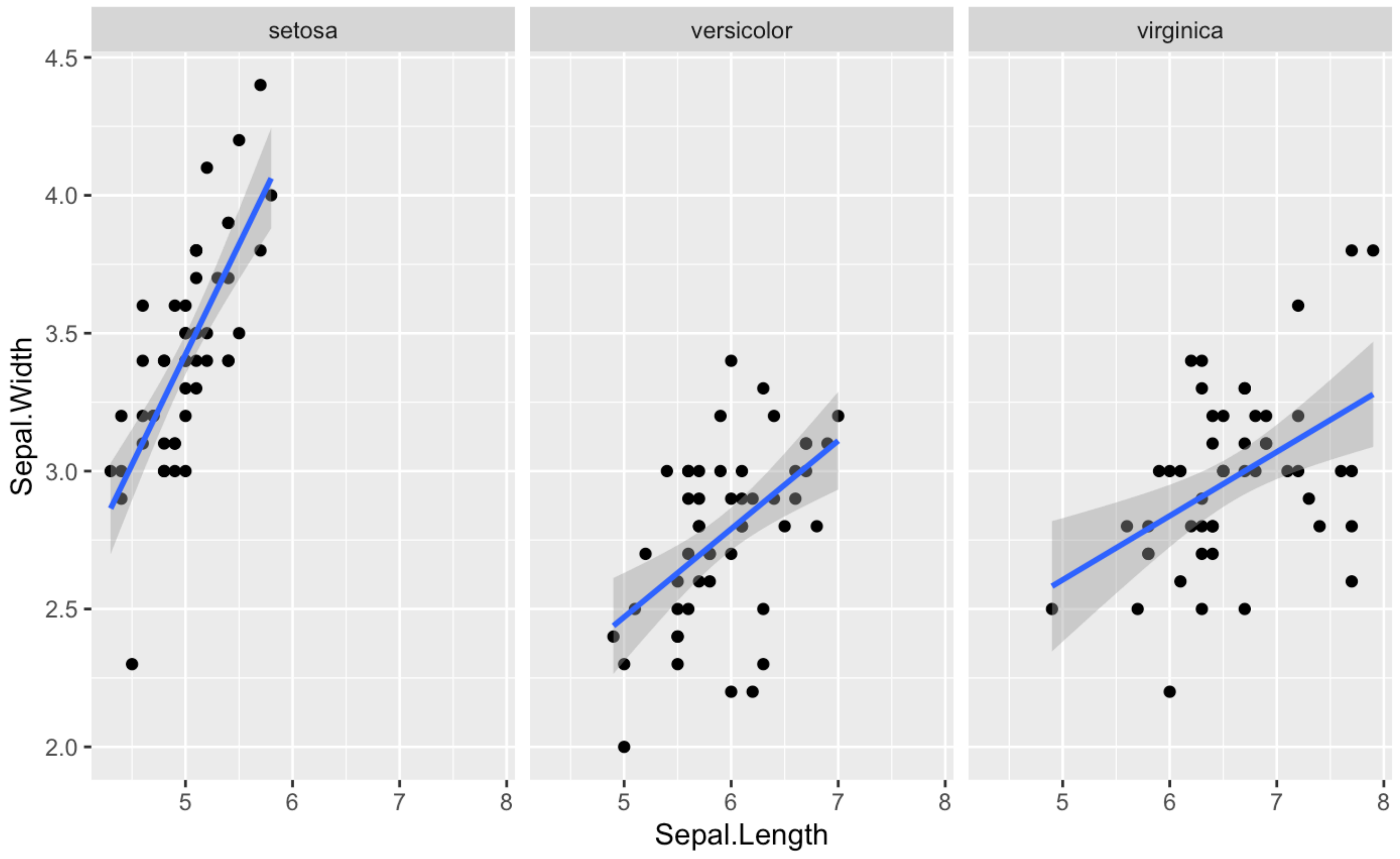


Next, by facets

Hide

```
ggplot(data = iris, aes(x = Sepal.Length, y = Sepal.Width)) + geom_point() + geom_smooth(method = "lm") +
facet_wrap(~Species)
```

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

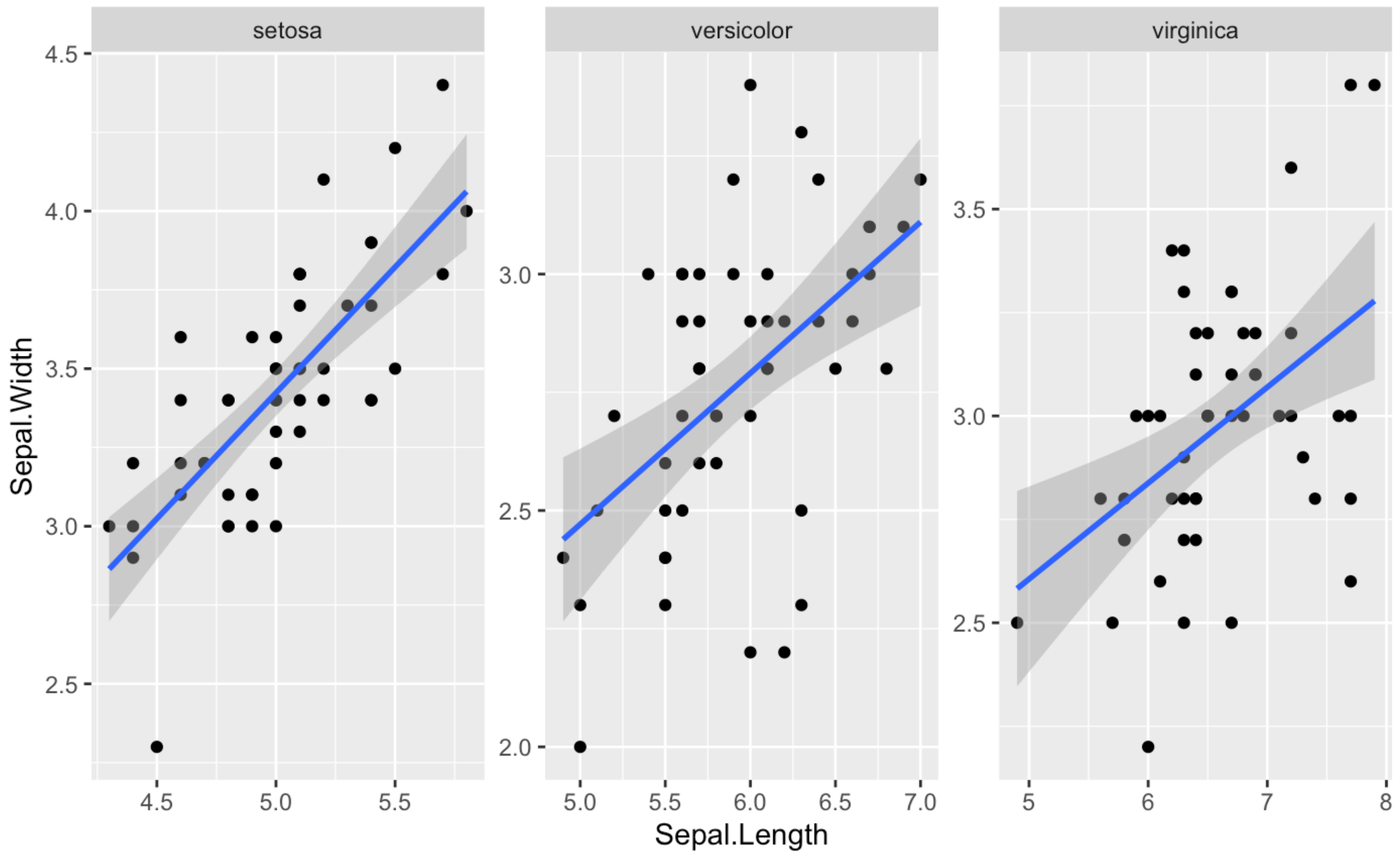


Note that by default, R has the scale of the three subplots as the same. How can we change that?

Hide

```
ggplot(data = iris, aes(x = Sepal.Length, y = Sepal.Width)) + geom_point() + geom_smooth(method = "lm") +
facet_wrap(~Species, scales = "free")
```

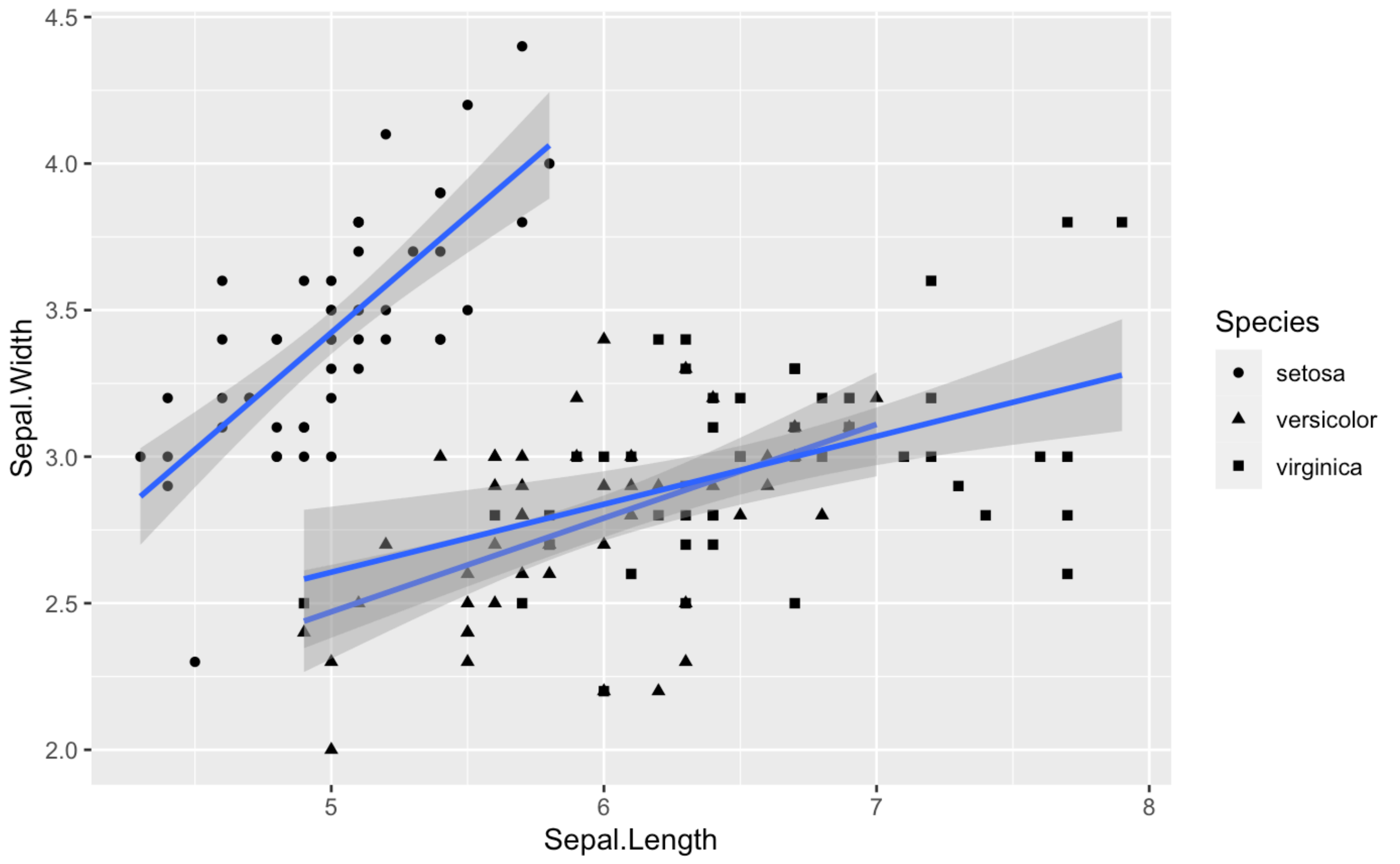
```
`geom_smooth()` using formula 'y ~ x'
```



We can also change the shape of the points associated with the three species

```
ggplot(data = iris, aes(x = Sepal.Length, y = Sepal.Width, shape = Species)) + geom_point() + geom_smooth(
method = "lm")
```

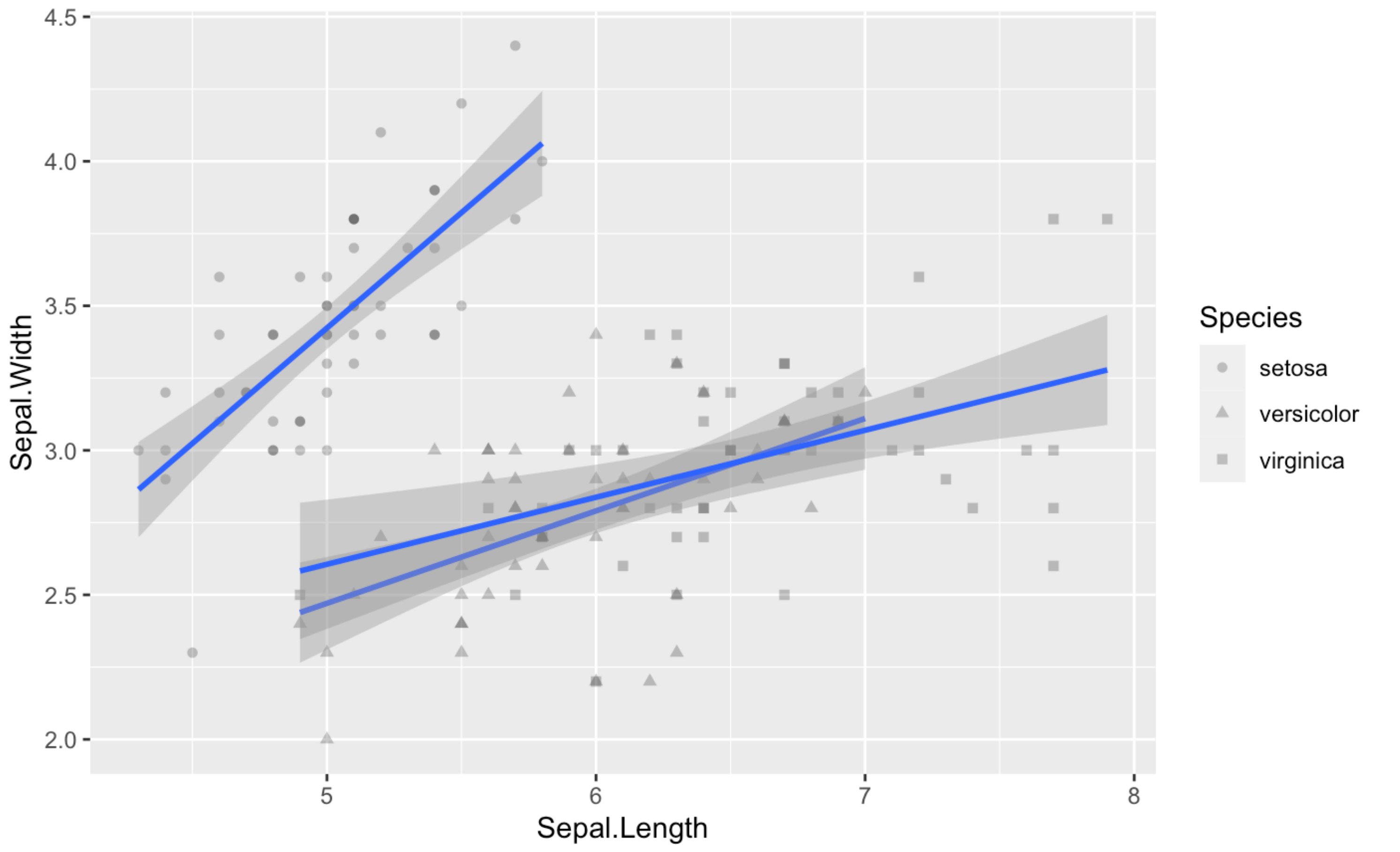
```
`geom_smooth()` using formula 'y ~ x'
```



Hide

```
ggplot(data = iris, aes(x = Sepal.Length, y = Sepal.Width, shape = Species)) + geom_point(alpha = .25) + g
eom_smooth(method = "lm")
```

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



Plotting the average petal length for each species

Hide

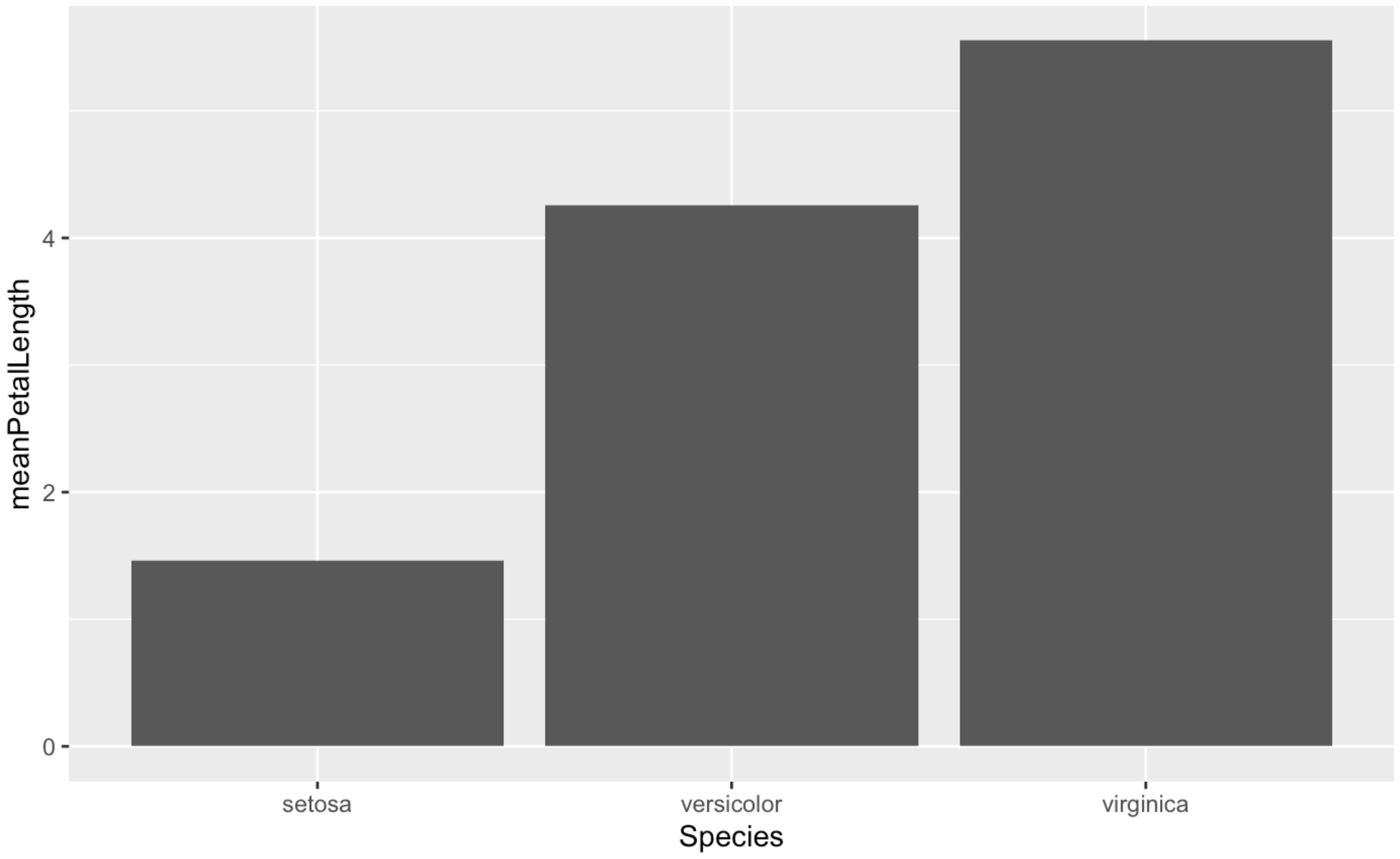
```
groupedData = group_by(iris, Species) %>% summarise(meanPetalLength = mean(Petal.Length))
groupedData
```

Species <fctr>	meanPetalLength <dbl>
setosa	1.462
versicolor	4.260
virginica	5.552

3 rows

Hide

```
ggplot(data = groupedData,aes(x = Species, y = meanPetalLength)) + geom_bar(stat = "identity")
```



The bars represent the means, which isn’t the most useful. Ideally, we’d also like a measure of variance.

One way to do this is to add error bars (in this case, standard error of the mean)

Hide

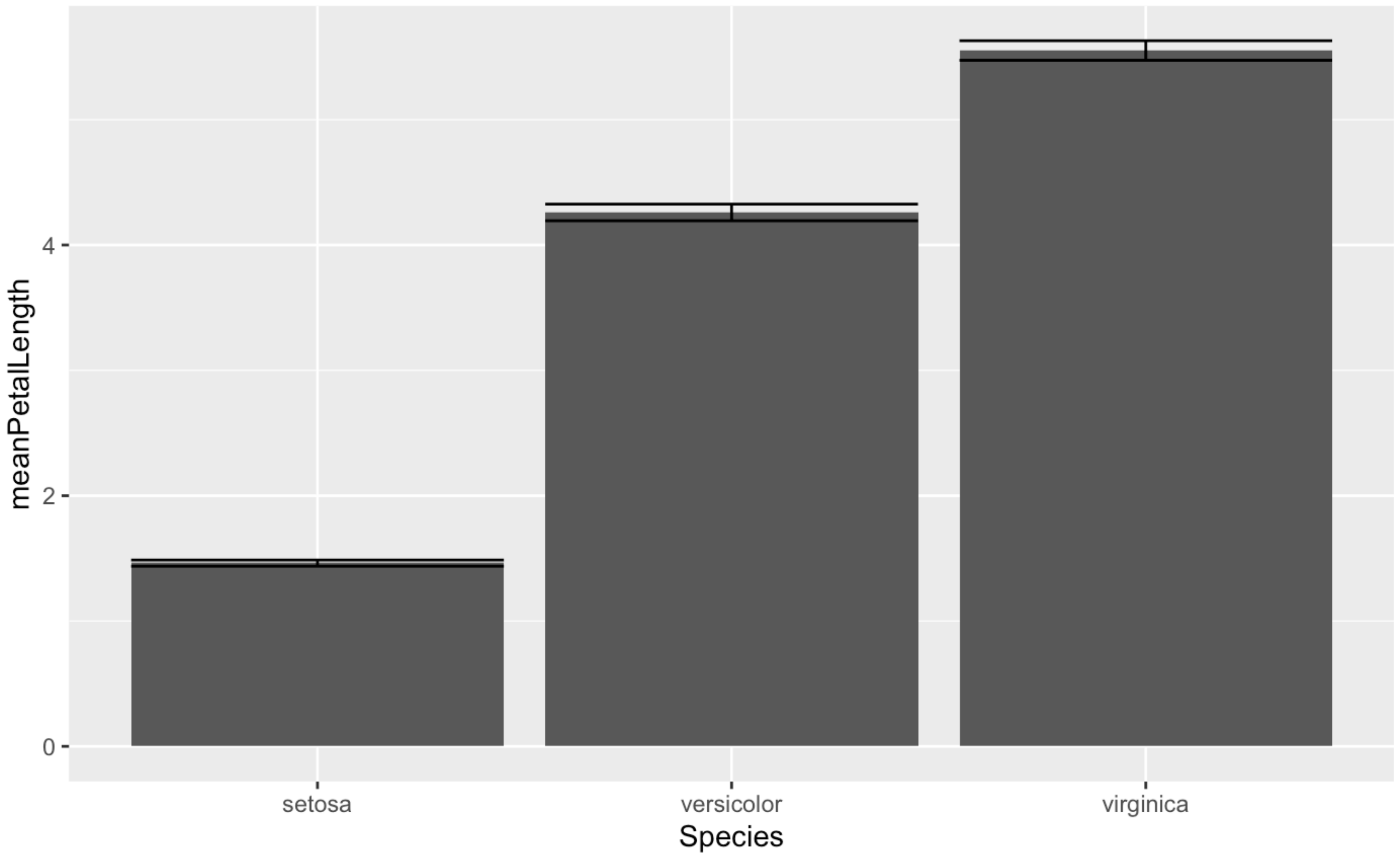
```
groupedData = group_by(iris, Species) %>% summarise(meanPetalLength = mean(Petal.Length),sdPetalLength = s
d(Petal.Length)/sqrt(n()))
groupedData
```

Species	**meanPetalLength**	**sdPetalLength**
<fctr>	<dbl>	<dbl>
setosa	1.462	0.02455980
versicolor	4.260	0.06645545
virginica	5.552	0.07804970

3 rows

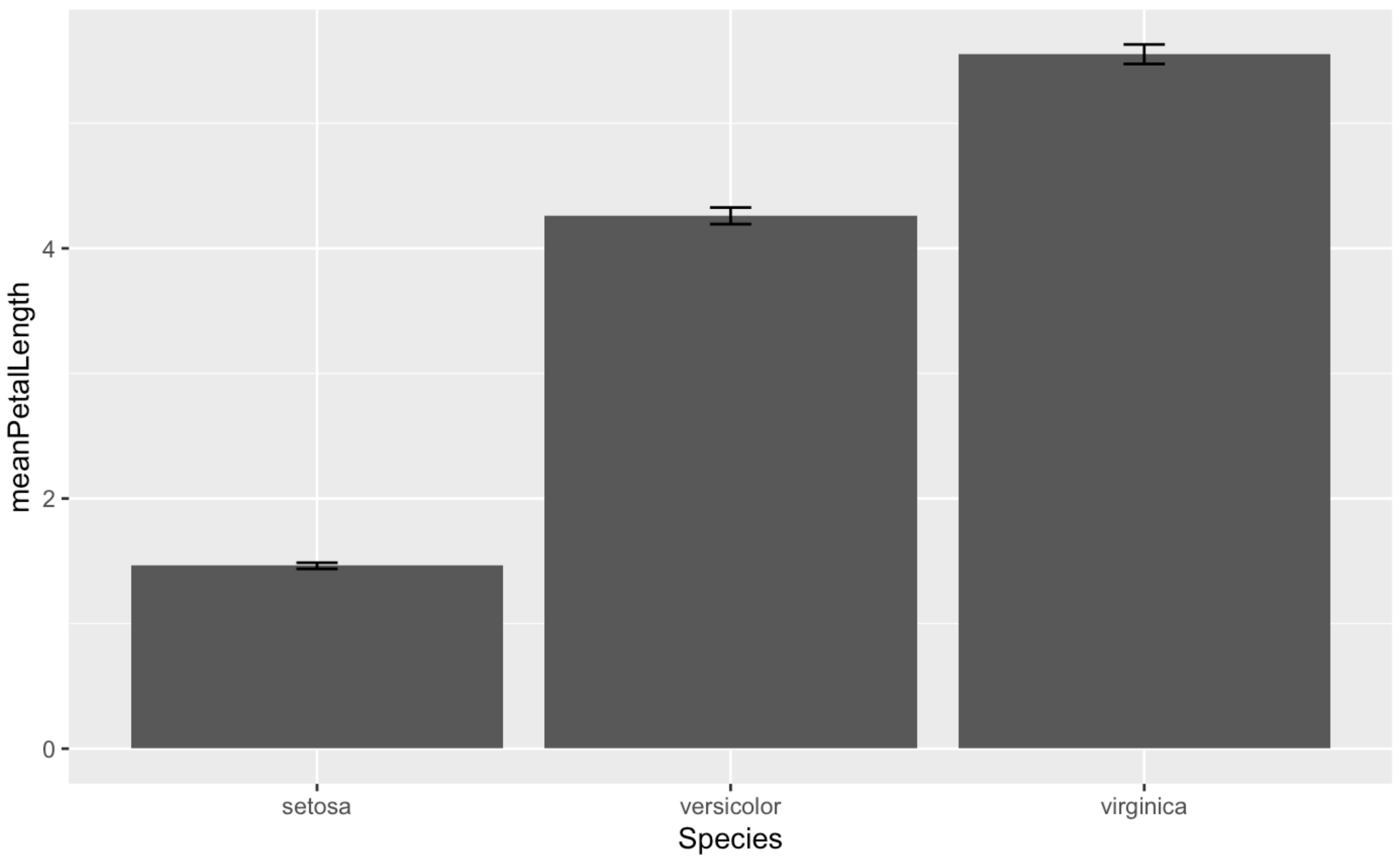
Hide

```
ggplot(data = groupedData,aes(x = Species, y = meanPetalLength)) + geom_bar(stat = "identity") + geom_erro
rbar(data = groupedData,aes(x = Species, ymin = meanPetalLength - sdPetalLength, ymax = meanPetalLength +
sdPetalLength))
```



Hide

```
ggplot(data = groupedData,aes(x = Species, y = meanPetalLength)) + geom_bar(stat = "identity") + geom_erro
rbar(data = groupedData,aes(x = Species, ymin = meanPetalLength - sdPetalLength, ymax = meanPetalLength +
sdPetalLength),width=.1)
```

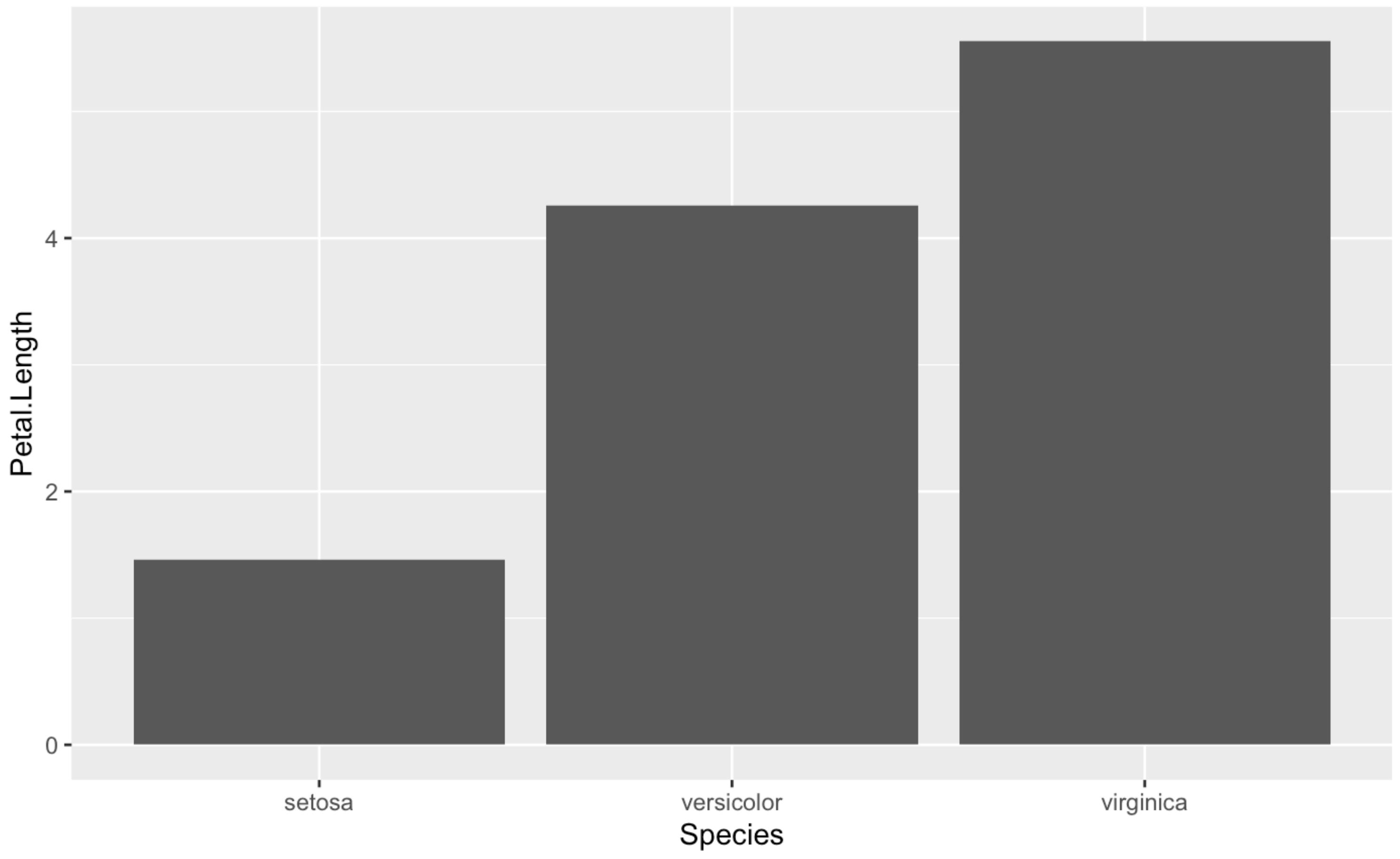


Aside: Another way to plot the mean more directly (without creating a new dataframe)

Hide

```
ggplot(data = iris,aes(x = Species, y = Petal.Length)) + geom_bar(stat = "summary", fun.y = "mean")
```

```
Warning: Ignoring unknown parameters: fun.y
No summary function supplied, defaulting to `mean_se()`
```



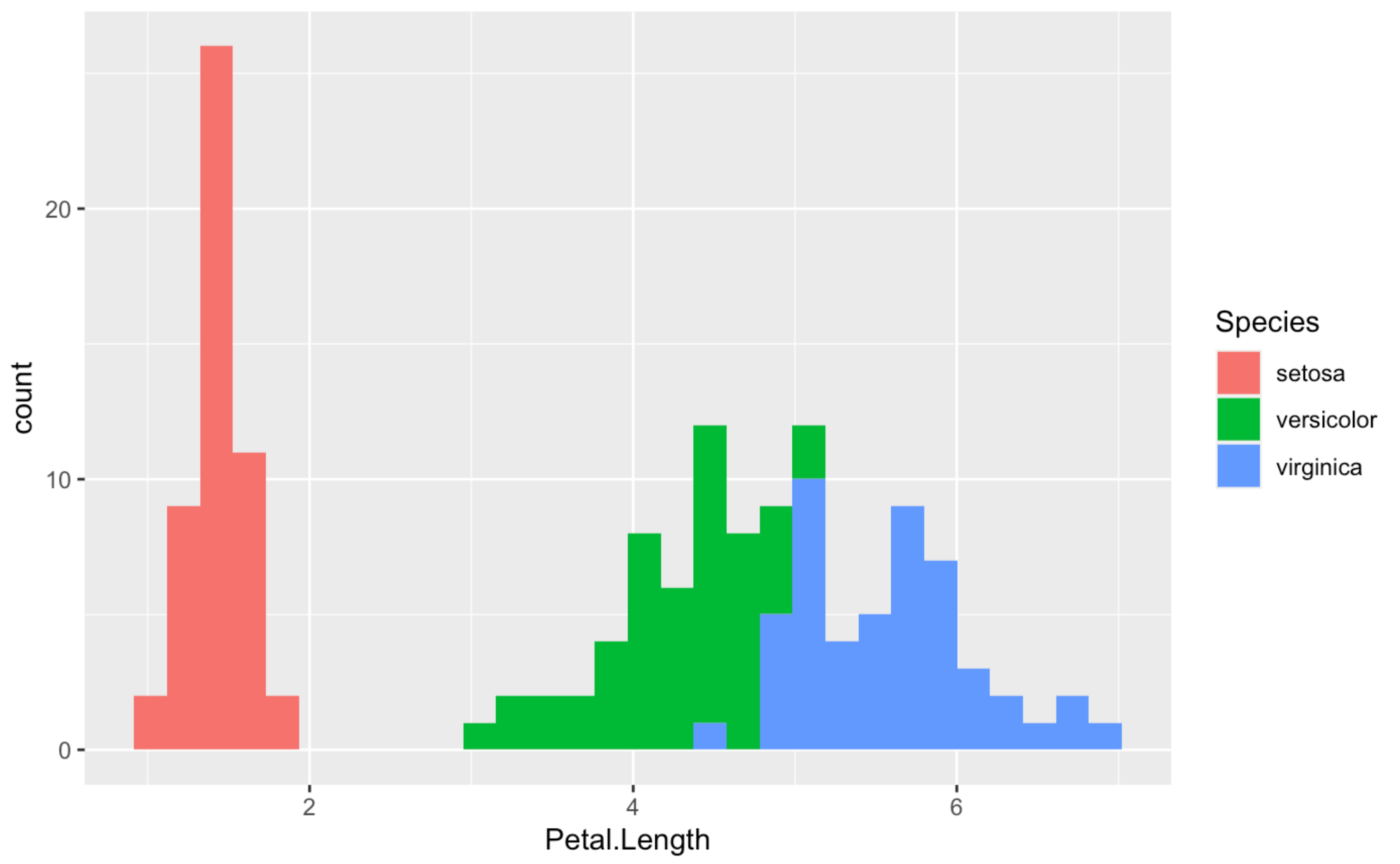
What if we want to get a better sense of the distribution of petal lengths for each species (not just the mean/sd)?

Histograms

Hide

```
ggplot(data = iris, aes(x = Petal.Length,fill = Species)) + geom_histogram()
```

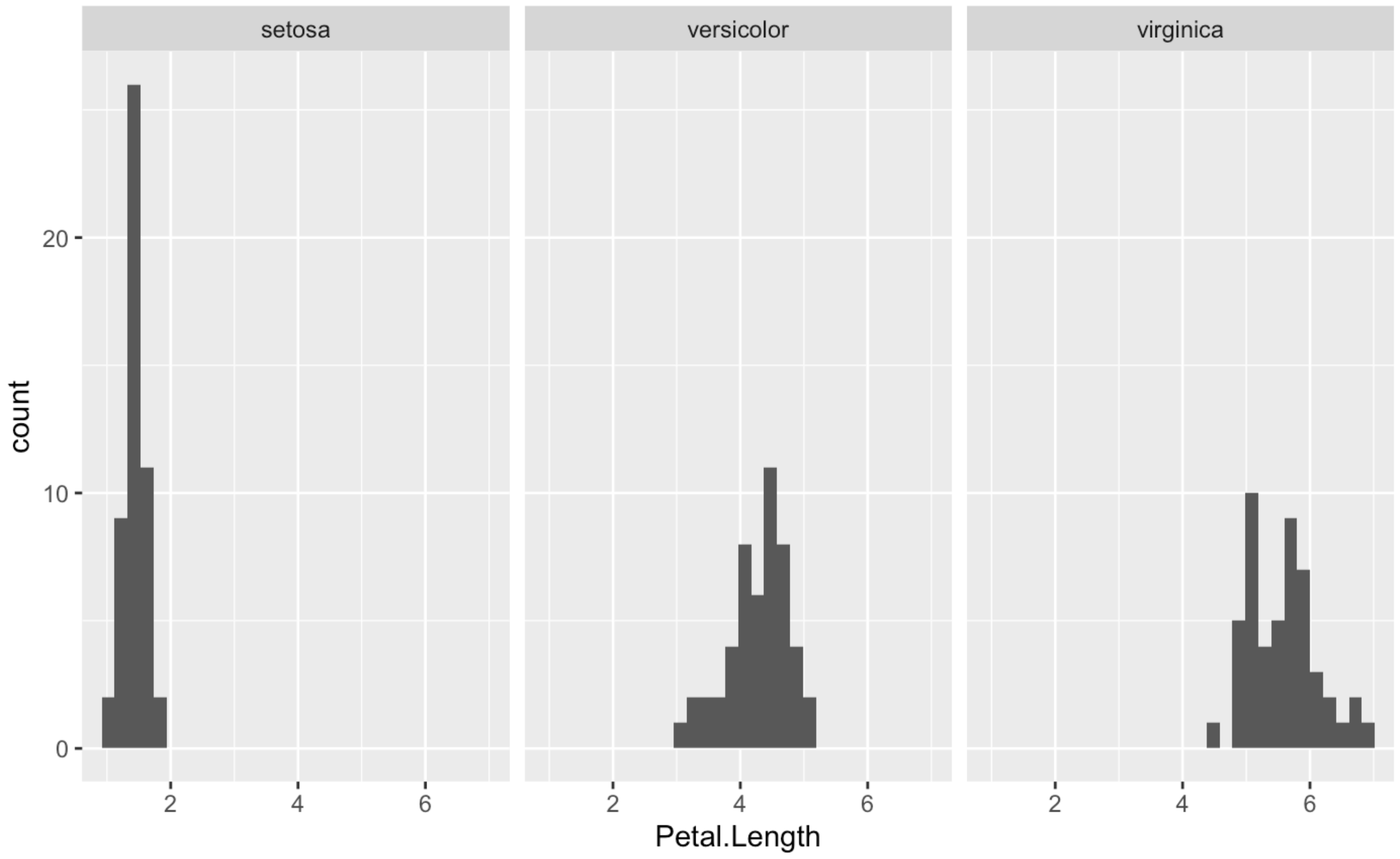
```
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Hide

```
ggplot(data = iris, aes(x = Petal.Length)) + geom_histogram() + facet_wrap(~Species)
```

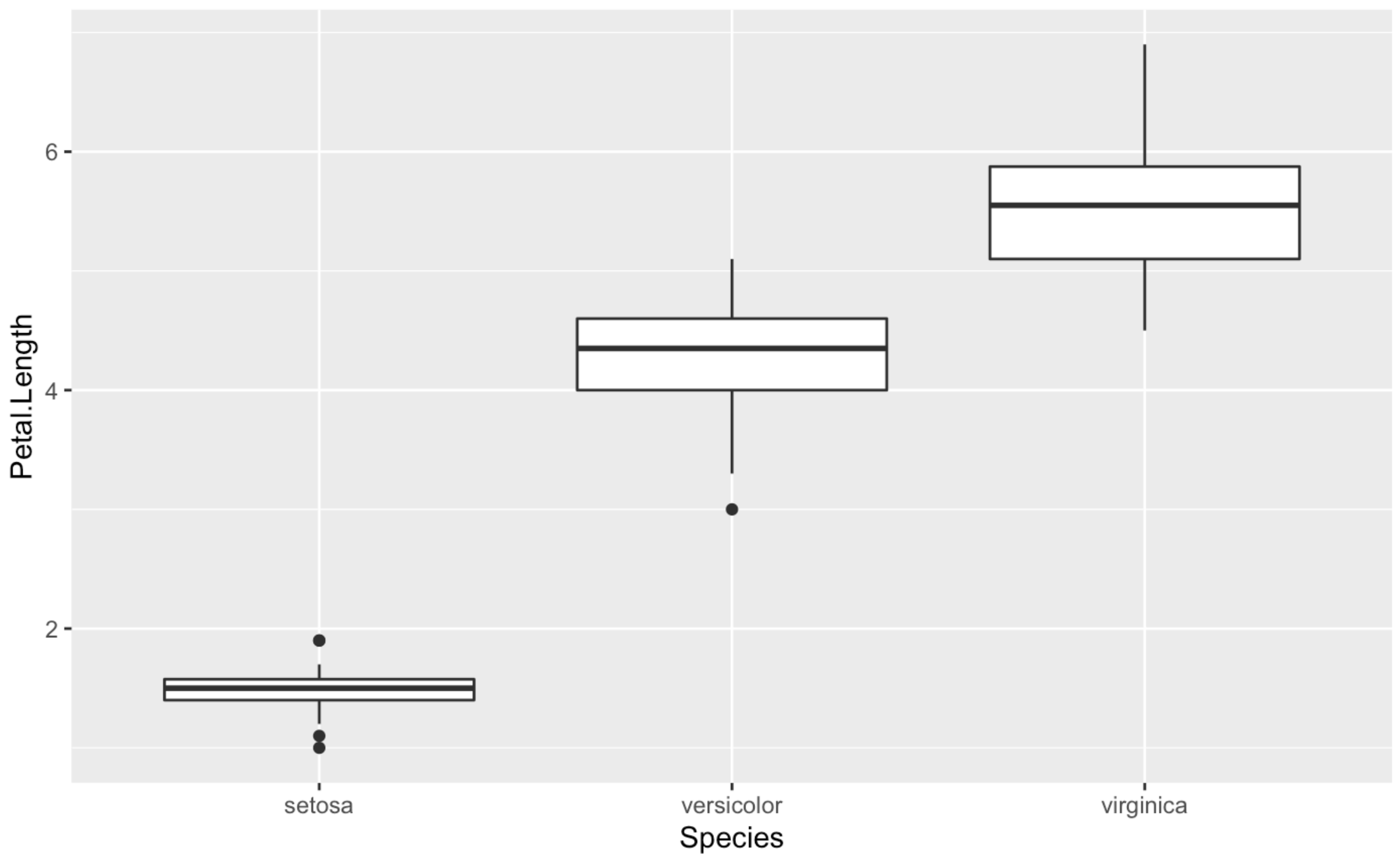
```
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Boxplot

Hide

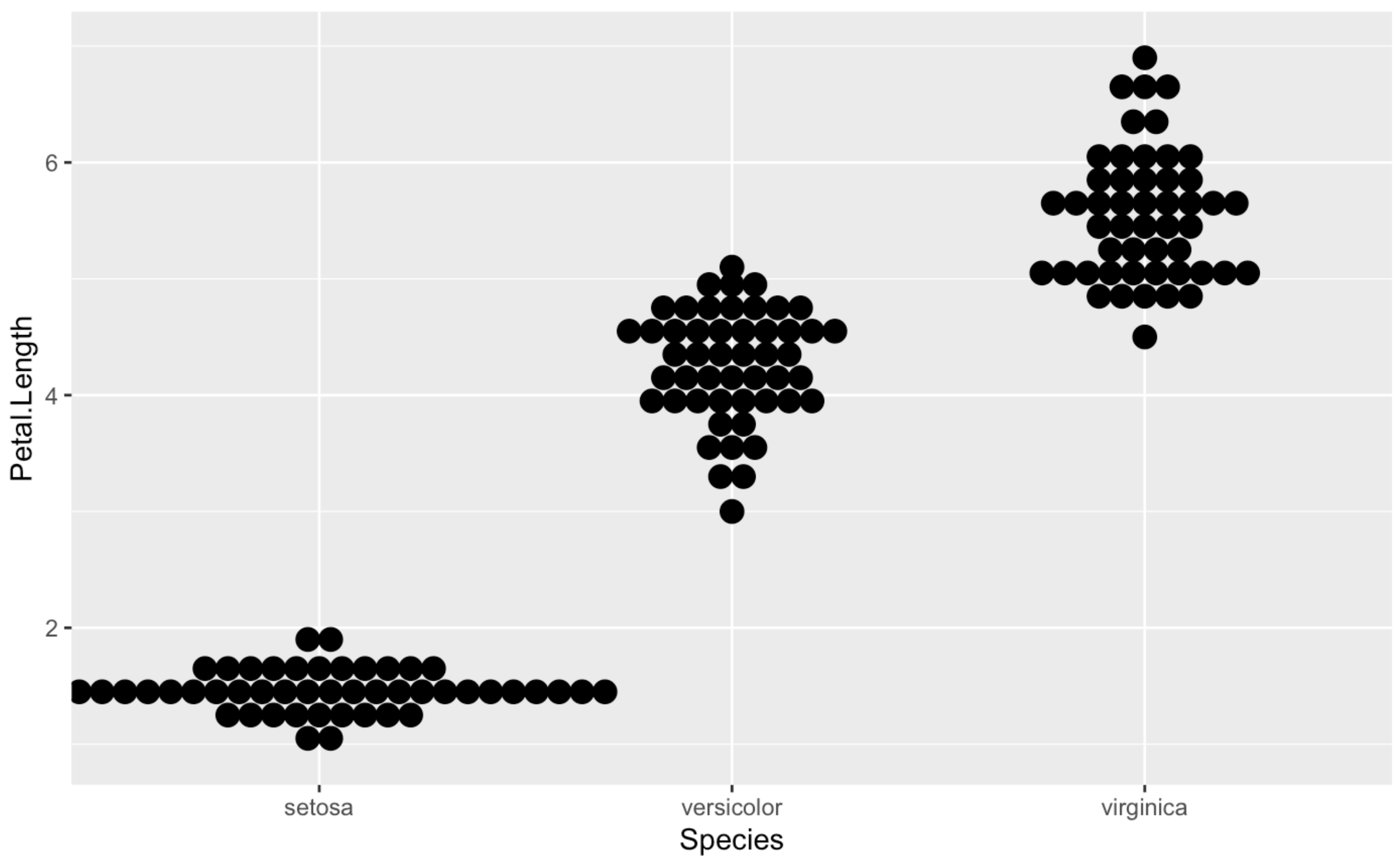
```
ggplot(data = iris, aes(x = Species, y = Petal.Length)) + geom_boxplot()
```



Dotplot

```
ggplot(data = iris, aes(x = Species, y = Petal.Length)) + geom_dotplot(binaxis = "y",stackdir = "center")
```

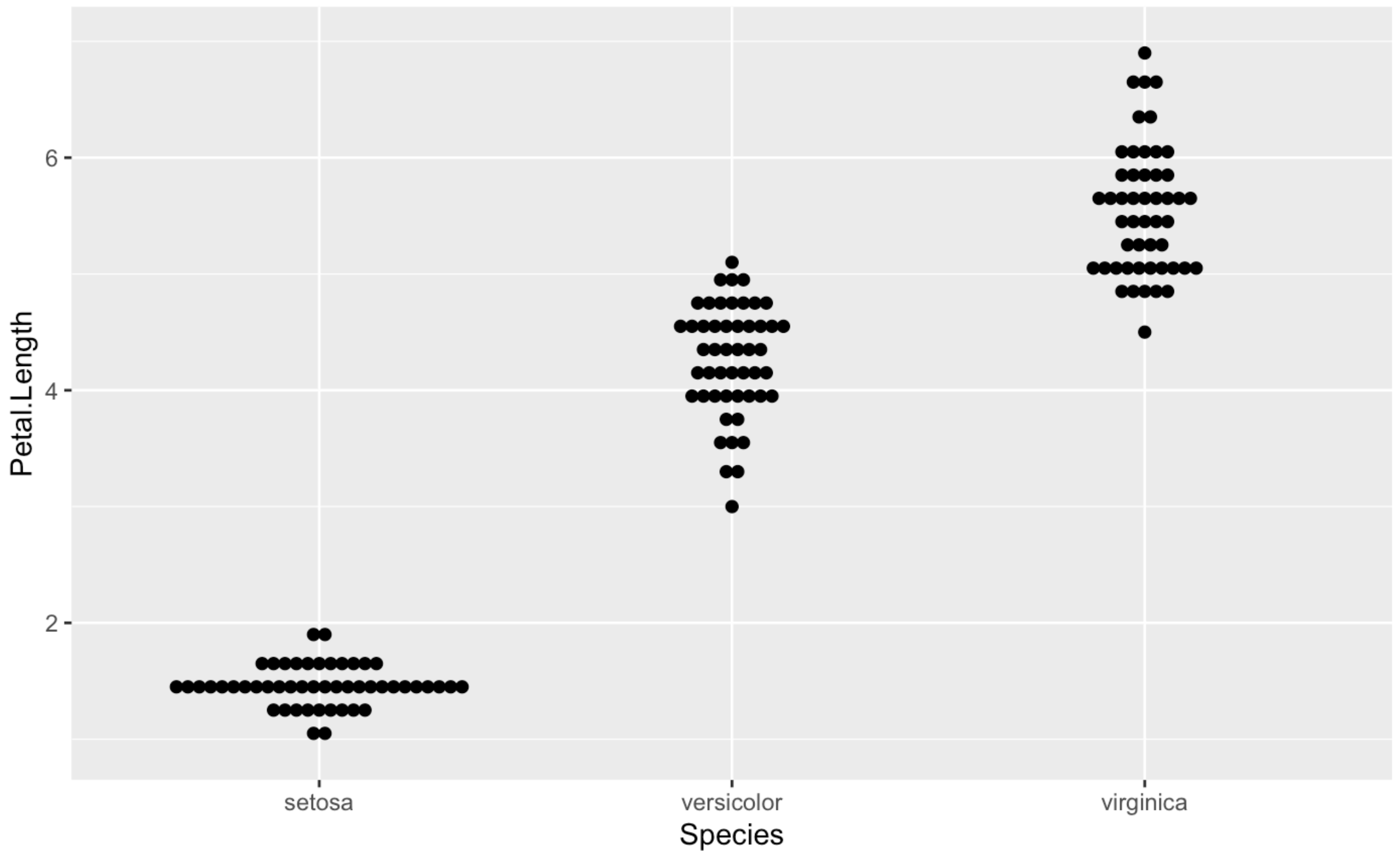
```
Bin width defaults to 1/30 of the range of the data. Pick better value with `binwidth`.
```



Hide

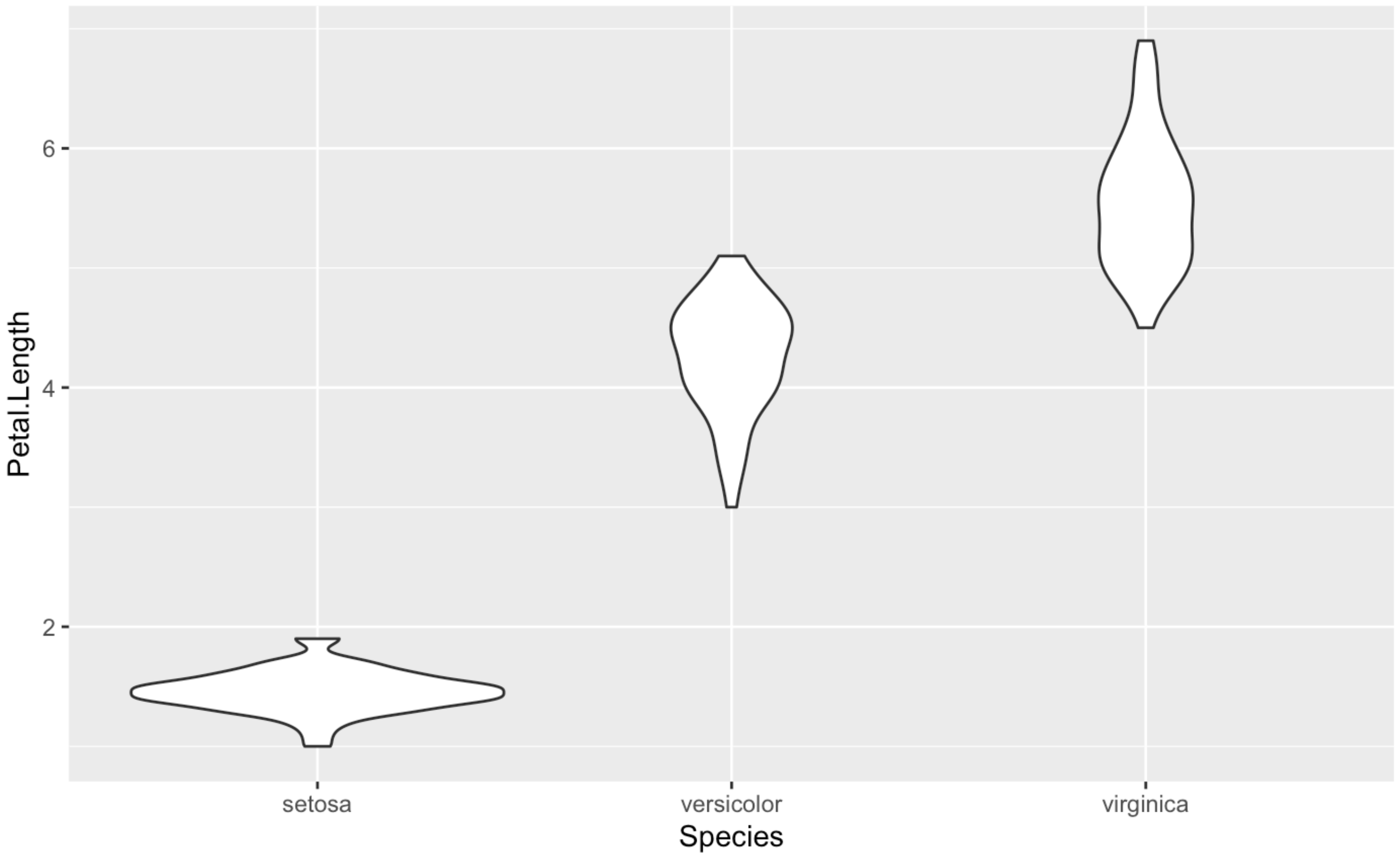
```
ggplot(data = iris, aes(x = Species, y = Petal.Length)) + geom_dotplot(binaxis = "y",stackdir = "center",d
otsize=.5)
```

```
Bin width defaults to 1/30 of the range of the data. Pick better value with `binwidth`.
```



Violin plot

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

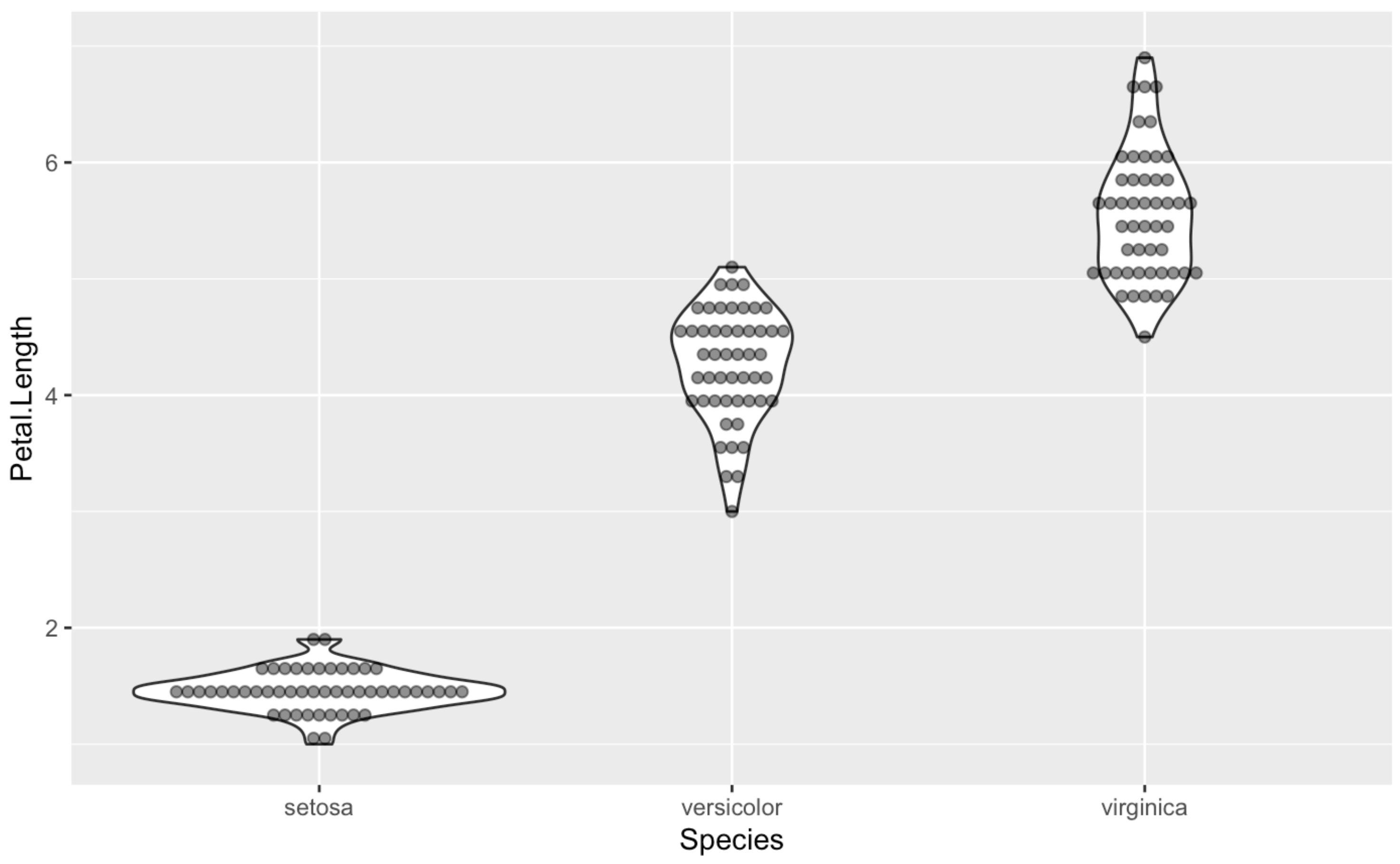


Violin + boxplot

Hide

```
ggplot(data = iris, aes(x = Species, y = Petal.Length)) + geom_violin() + geom_dotplot(binaxis = "y",stack
dir = "center",dotsize=.5,alpha=.5)
```

```
Bin width defaults to 1/30 of the range of the data. Pick better value with `binwidth`.
```



Plotting the individual data points on the mean bars

Hide

```
groupedData
```

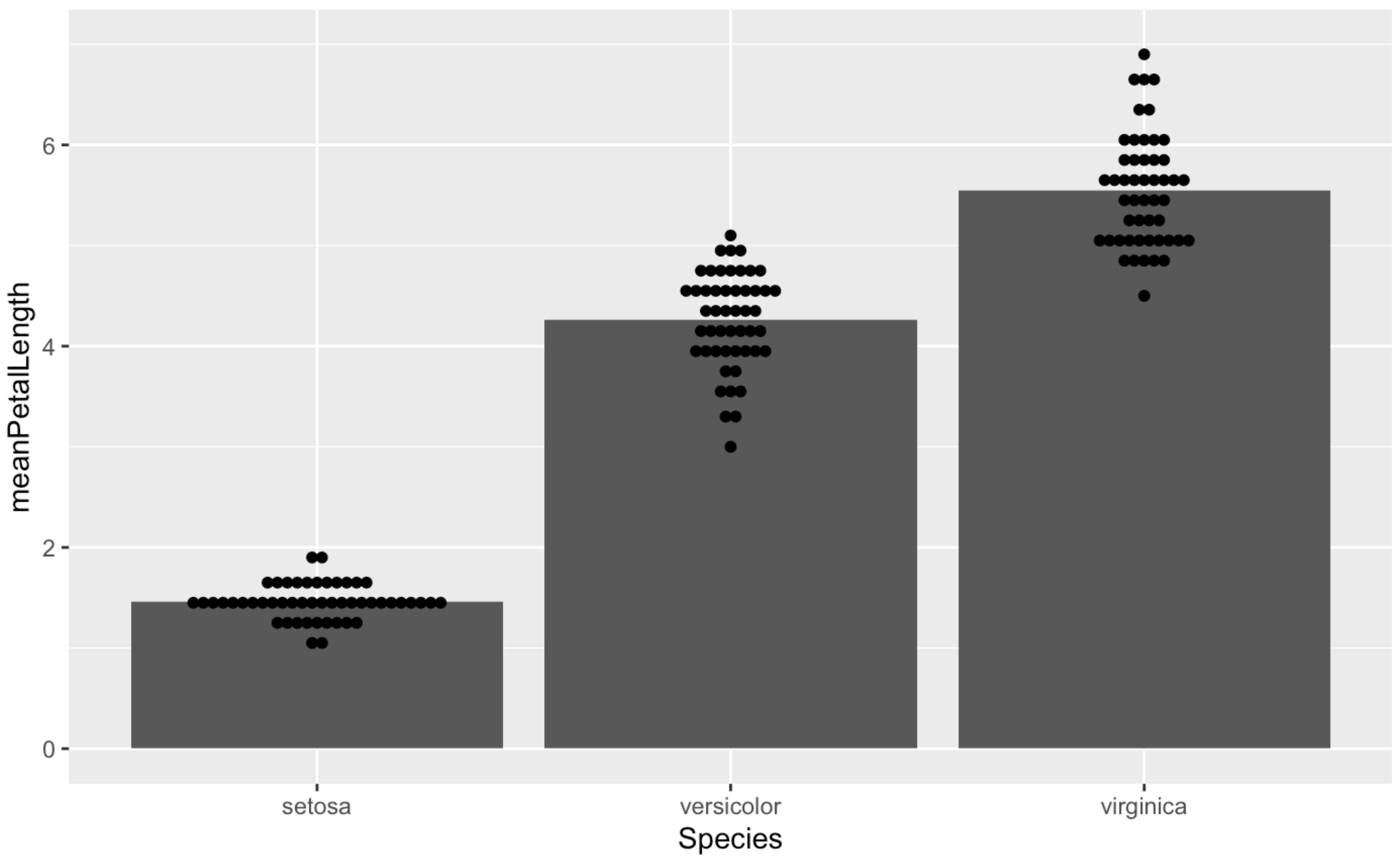
Species <fctr>	meanPetalLength <dbl>	sdPetalLength <dbl>
setosa	1.462	0.02455980
versicolor	4.260	0.06645545
virginica	5.552	0.07804970

3 rows

Hide

```
ggplot(data = groupedData,aes(x = Species, y = meanPetalLength)) + geom_bar(stat = "identity") + geom_dotp
lot(data = iris, aes(x = Species, y = Petal.Length), binaxis = "y", stackdir = "center",dotsize = .5)
```

```
Bin width defaults to 1/30 of the range of the data. Pick better value with `binwidth`.
```



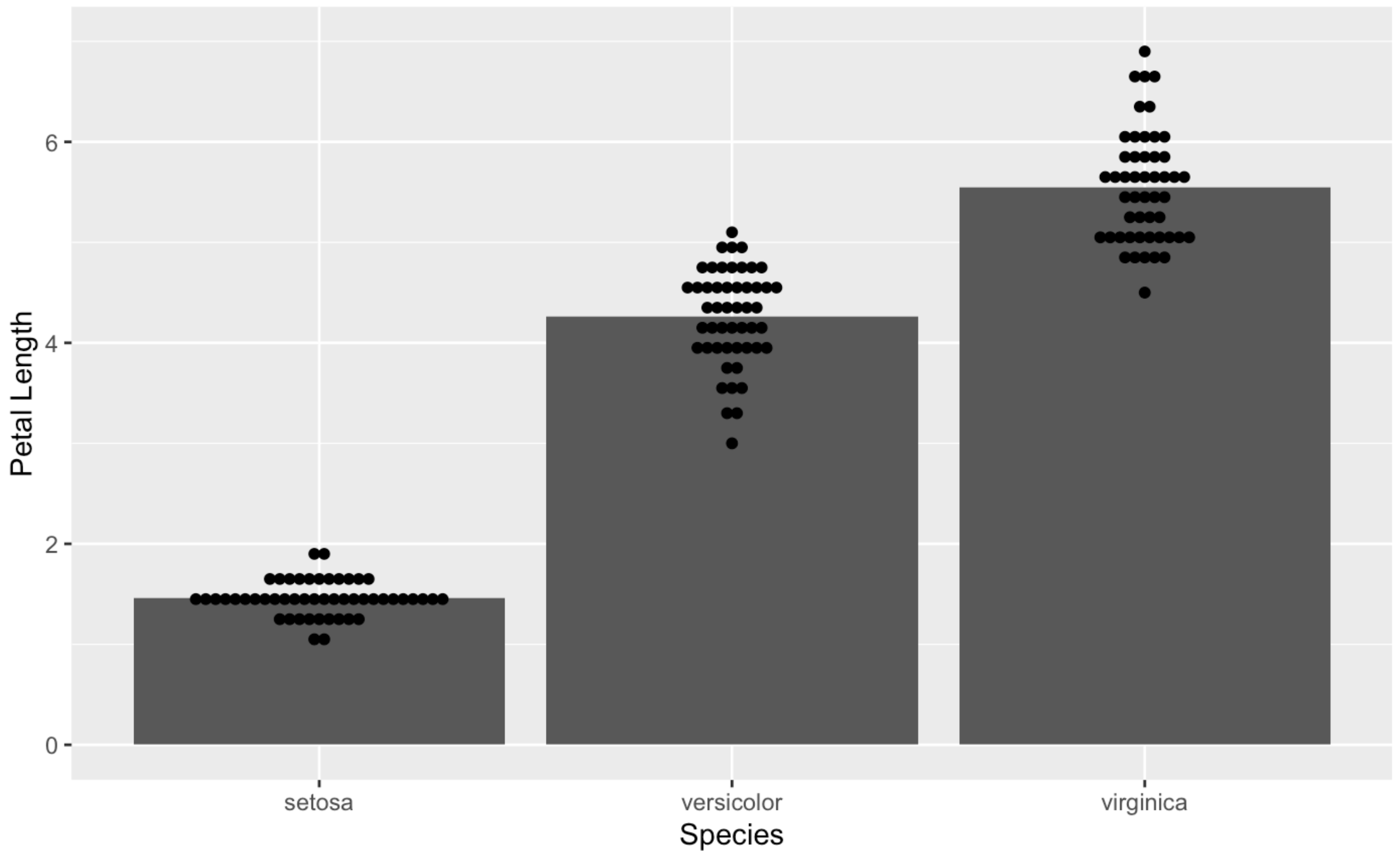
Now let’s spruce up the graph

First fix the y axis label

Hide

```
ggplot(data = groupedData,aes(x = Species, y = meanPetalLength)) + geom_bar(stat = "identity") + geom_dotp
lot(data = iris, aes(x = Species, y = Petal.Length), binaxis = "y", stackdir = "center",dotsize = .5) + yl
ab('Petal Length')
```

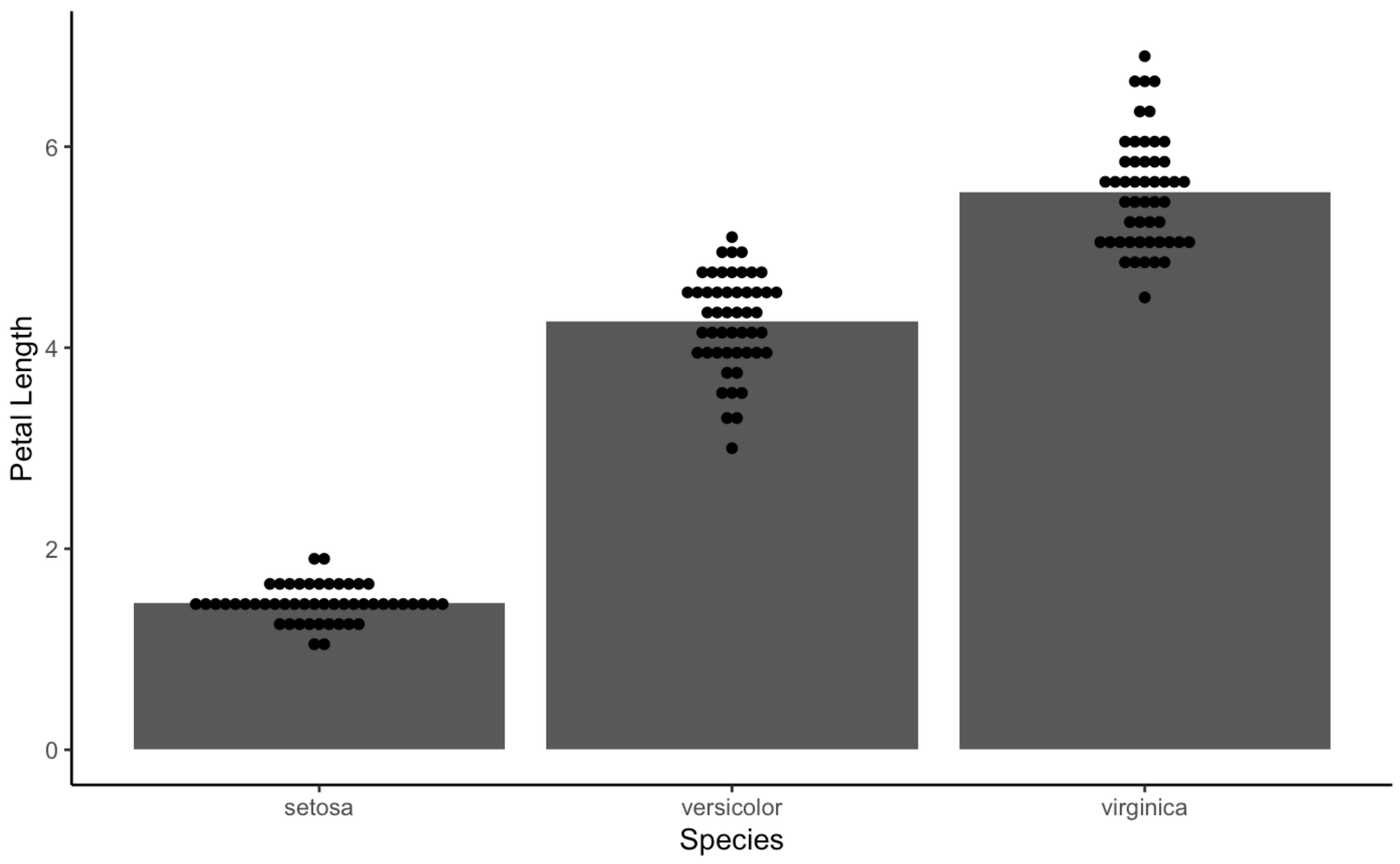
```
Bin width defaults to 1/30 of the range of the data. Pick better value with `binwidth`.
```



Change the theme

```
ggplot(data = groupedData,aes(x = Species, y = meanPetalLength)) + geom_bar(stat = "identity") + geom_dotp
lot(data = iris, aes(x = Species, y = Petal.Length), binaxis = "y", stackdir = "center",dotsize = .5) + yl
ab('Petal Length') + theme_classic()
```

```
Bin width defaults to 1/30 of the range of the data. Pick better value with `binwidth`.
```

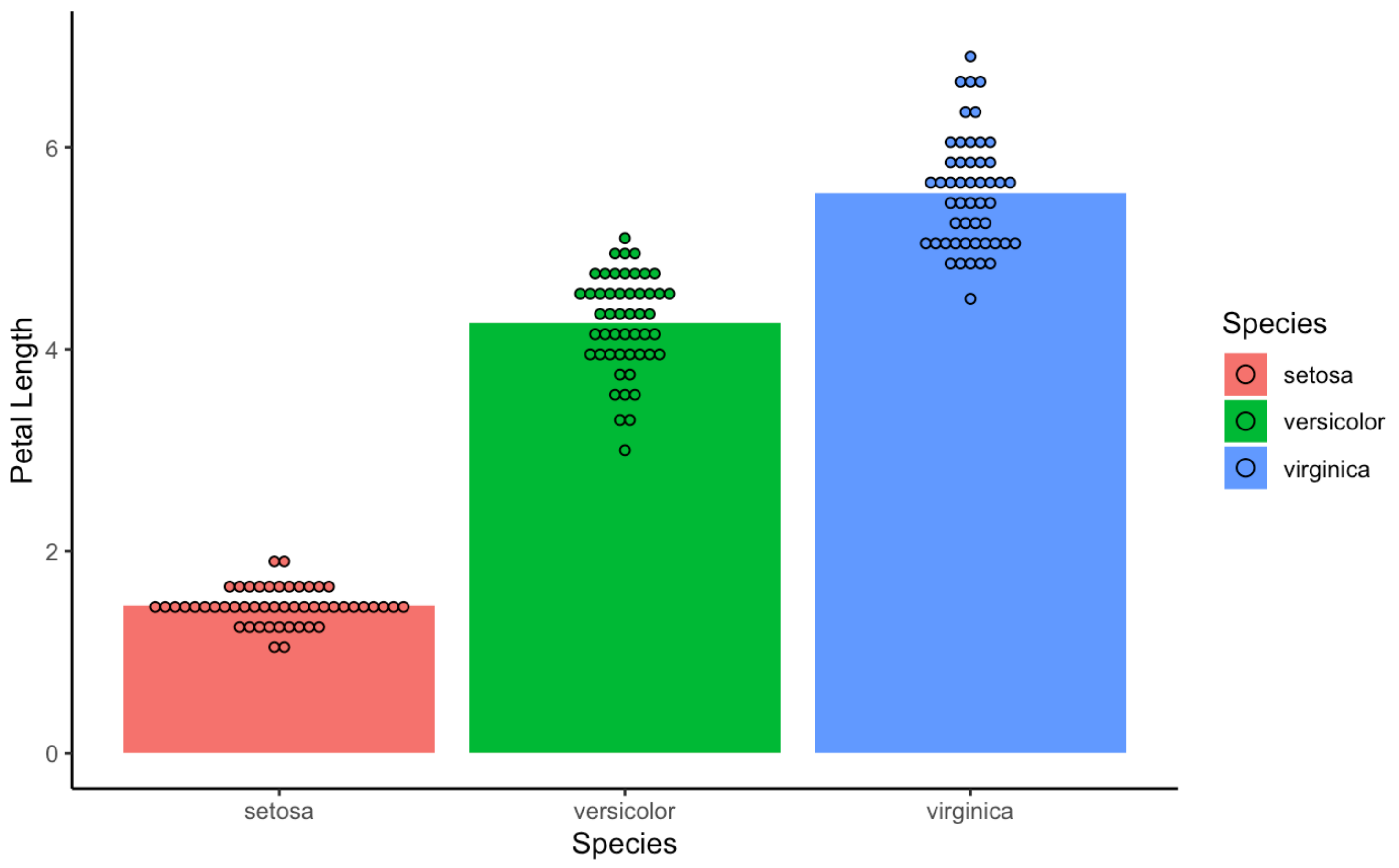


Color by species

Hide

```
ggplot(data = groupedData,aes(x = Species, y = meanPetalLength, fill = Species)) + geom_bar(stat = "identi
ty") + geom_dotplot(data = iris, aes(x = Species, y = Petal.Length), binaxis = "y", stackdir = "center",do
tsize = .5) + ylab('Petal Length') + theme_classic()
```

```
Bin width defaults to 1/30 of the range of the data. Pick better value with `binwidth`.
```



Change the color scheme

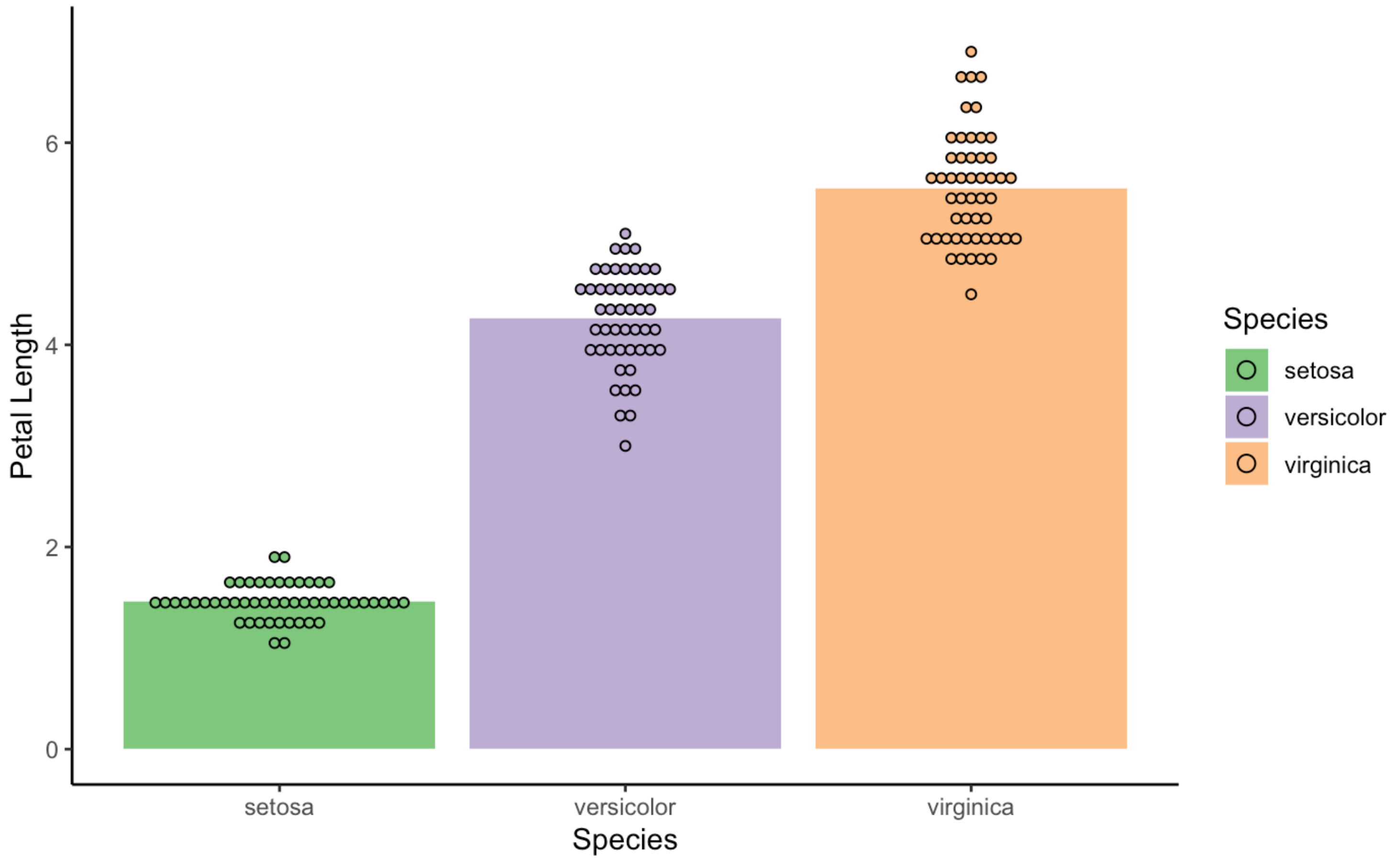
Hide

```
library(RColorBrewer)
```

Hide

```
ggplot(data = groupedData,aes(x = Species, y = meanPetalLength, fill = Species)) + geom_bar(stat = "identi
ty") + geom_dotplot(data = iris, aes(x = Species, y = Petal.Length), binaxis = "y", stackdir = "center",do
tsize = .5) + ylab('Petal Length') + theme_classic() + scale_fill_brewer(palette = "Accent")
```

```
Bin width defaults to 1/30 of the range of the data. Pick better value with `binwidth`.
```



Change the font, font size

Hide

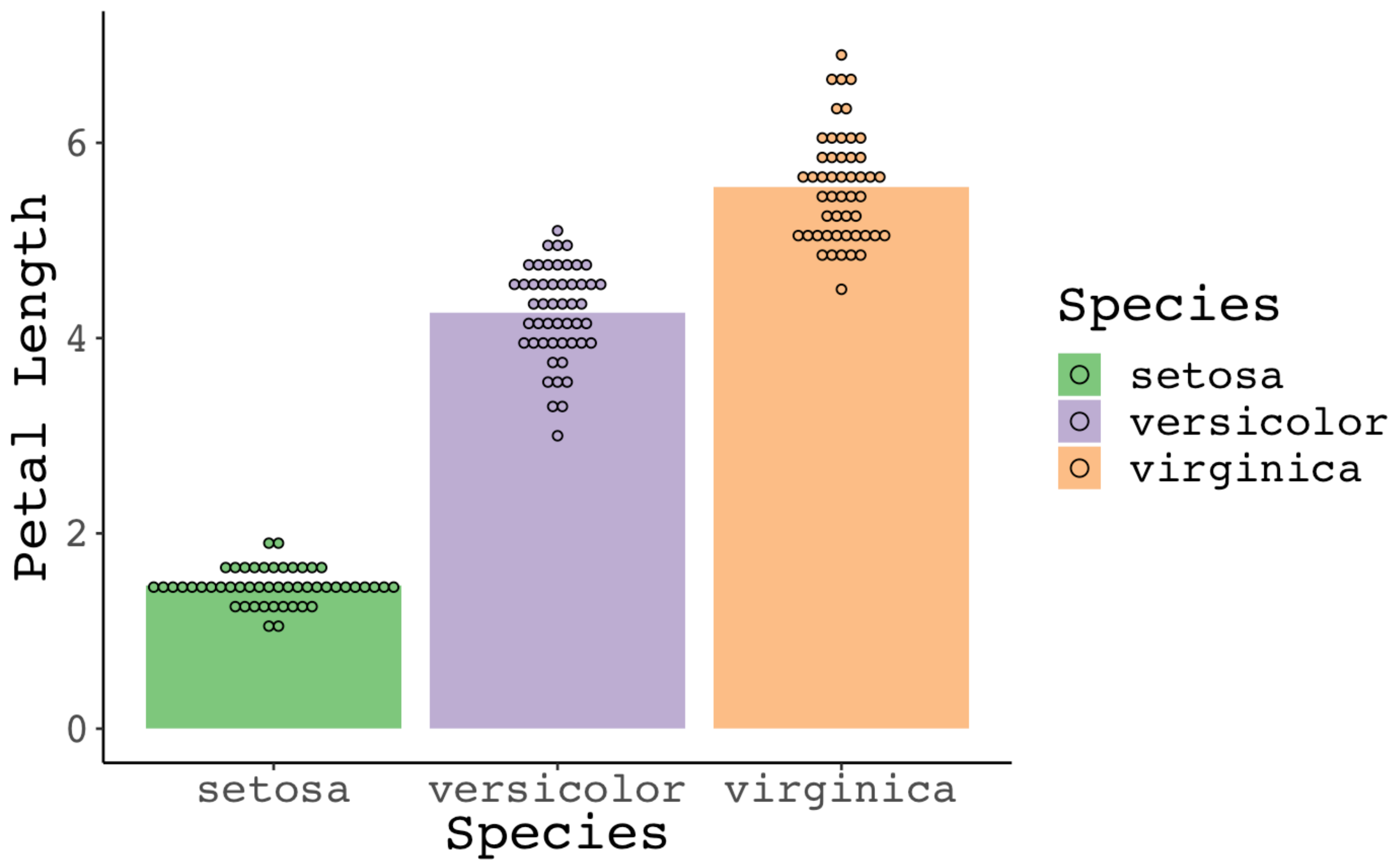
```
ggplot(data = groupedData,aes(x = Species, y = meanPetalLength, fill = Species)) + geom_bar(stat = "identi
ty") + geom_dotplot(data = iris, aes(x = Species, y = Petal.Length), binaxis = "y", stackdir = "center",do
tsize = .5) + ylab('Petal Length') + theme_classic() + scale_fill_brewer(palette = "Accent") + theme(text
= element_text(size = 20,family = "mono"))
```

```
Bin width defaults to 1/30 of the range of the data. Pick better value with `binwidth`.
```

Hide

```
# save the plot
ggsave('petal_means.pdf',width=5,height=5)
```

```
Bin width defaults to 1/30 of the range of the data. Pick better value with `binwidth`.
```



Change the size of the plot so that it's not cut off when it saves

Hide

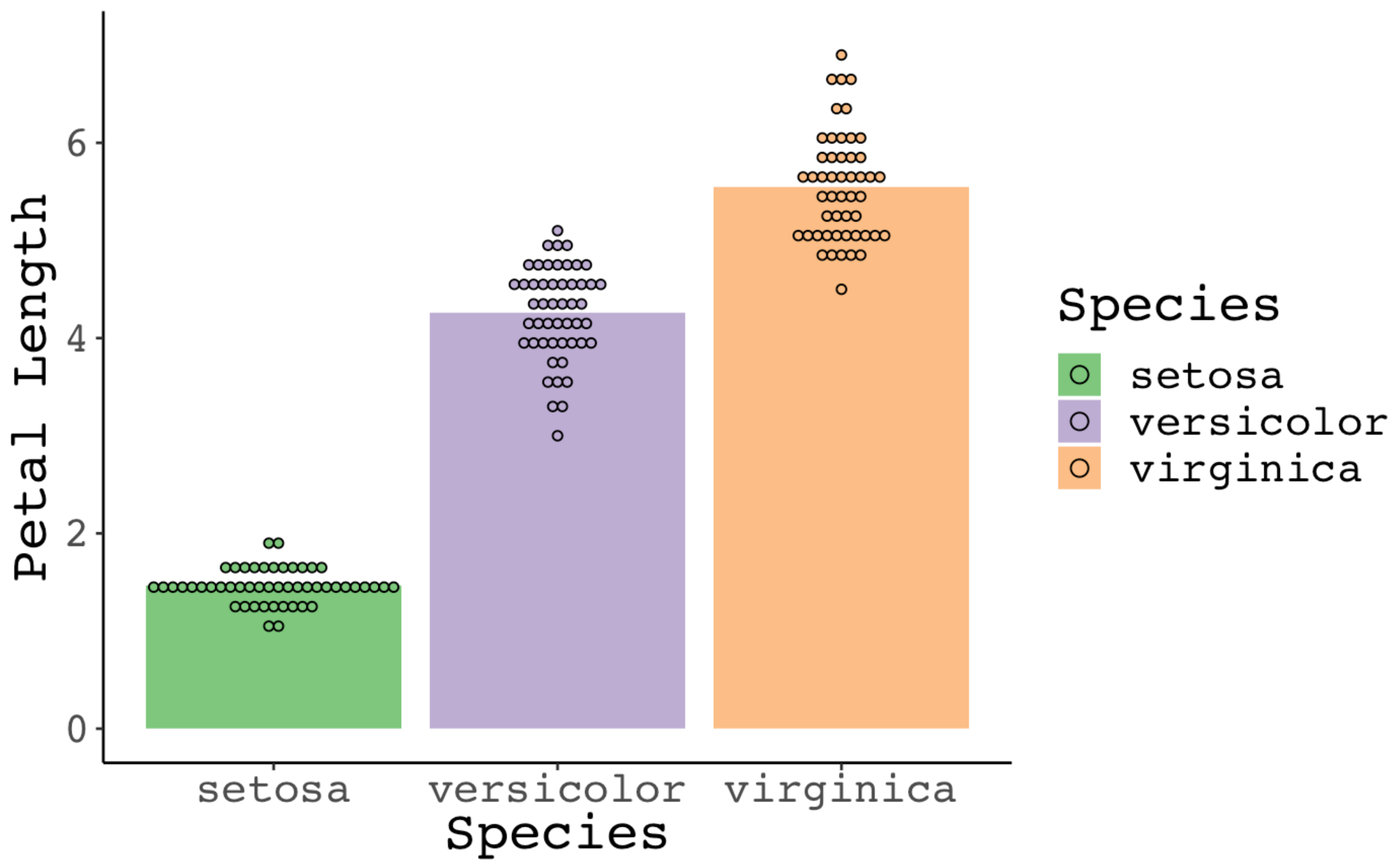
```
ggplot(data = groupedData,aes(x = Species, y = meanPetalLength, fill = Species)) + geom_bar(stat = "identi
ty") + geom_dotplot(data = iris, aes(x = Species, y = Petal.Length), binaxis = "y", stackdir = "center",do
tsize = .5) + ylab('Petal Length') + theme_classic() + scale_fill_brewer(palette = "Accent") + theme(text
= element_text(size = 20,family = "mono"))
```

```
Bin width defaults to 1/30 of the range of the data. Pick better value with `binwidth`.
```

Hide

```
# save the plot
ggsave('petal_means_wide.pdf',width=7,height=5)
```

```
Bin width defaults to 1/30 of the range of the data. Pick better value with `binwidth`.
```



Alternatively: is a legend really necessary in this plot?

Hide

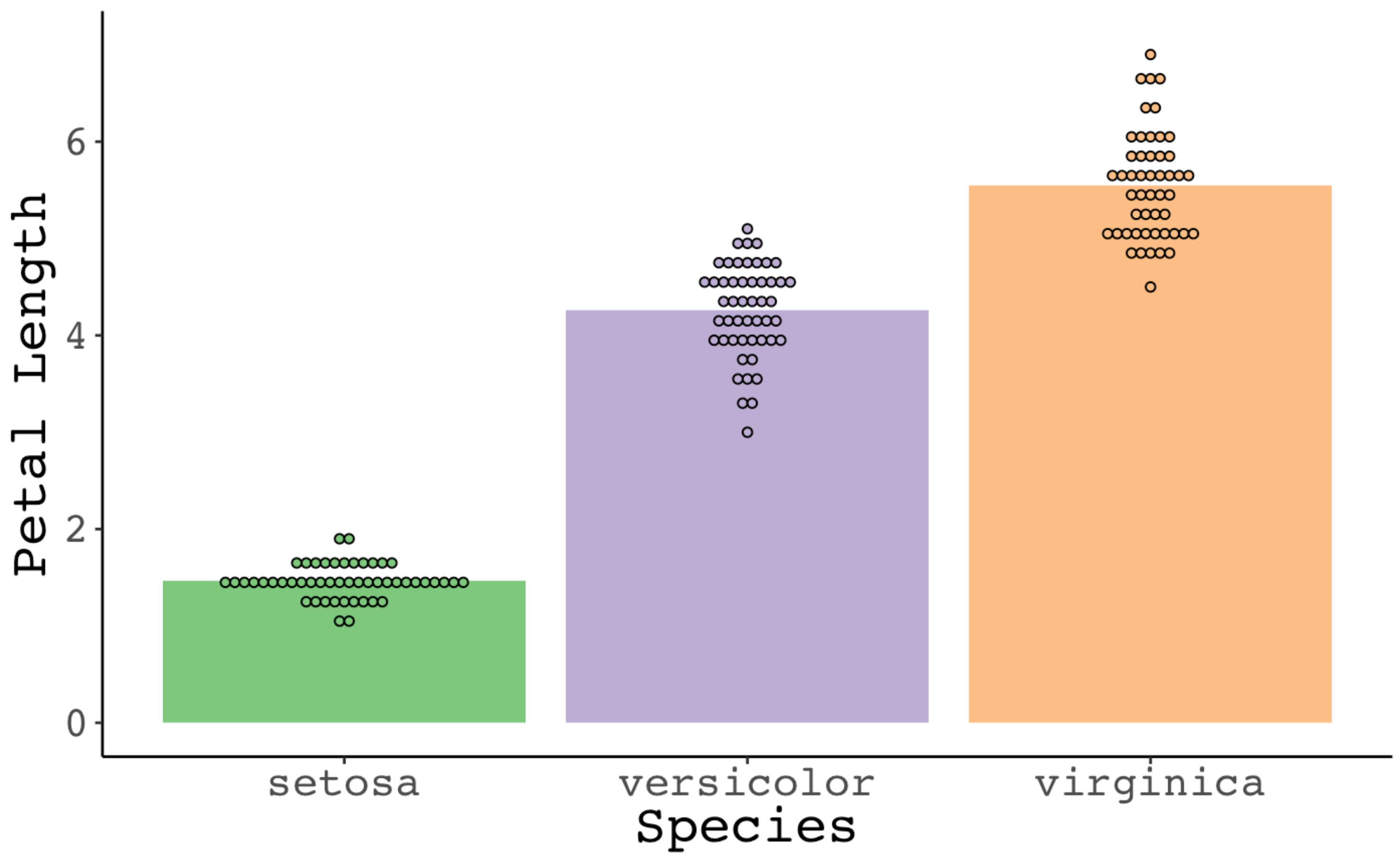
```
ggplot(data = groupedData,aes(x = Species, y = meanPetalLength, fill = Species)) + geom_bar(stat = "identi
ty") + geom_dotplot(data = iris, aes(x = Species, y = Petal.Length), binaxis = "y", stackdir = "center",do
tsize = .5) + ylab('Petal Length') + theme_classic() + scale_fill_brewer(palette = "Accent") + theme(text
= element_text(size = 20,family = "mono")) + theme(legend.position = "none")
```

```
Bin width defaults to 1/30 of the range of the data. Pick better value with `binwidth`.
```

Hide

```
# save the plot
ggsave('petal_means_noLegend.pdf',width=5,height=5)
```

```
Bin width defaults to 1/30 of the range of the data. Pick better value with `binwidth`.
```



Load in a different dataset (mpg) to illustrate a few other specific instances that come up often

Hide

```
?mpg
```

Hide

```
mpg
```

manufacturer <chr>	model <chr>	displ <dbl>	year <int>	cyl <int>	trans <chr>	drv <chr>	cty <int>	hwy <int>	fl <chr>
audi	a4	1.8	1999	4	auto(l5)	f	18	29	p
audi	a4	1.8	1999	4	manual(m5)	f	21	29	p
audi	a4	2.0	2008	4	manual(m6)	f	20	31	p
audi	a4	2.0	2008	4	auto(av)	f	21	30	p
audi	a4	2.8	1999	6	auto(l5)	f	16	26	p
audi	a4	2.8	1999	6	manual(m5)	f	18	26	p
audi	a4	3.1	2008	6	auto(av)	f	18	27	p
audi	a4 quattro	1.8	1999	4	manual(m5)	4	18	26	p
audi	a4 quattro	1.8	1999	4	auto(l5)	4	16	25	p
audi	a4 quattro	2.0	2008	4	manual(m6)	4	20	28	p

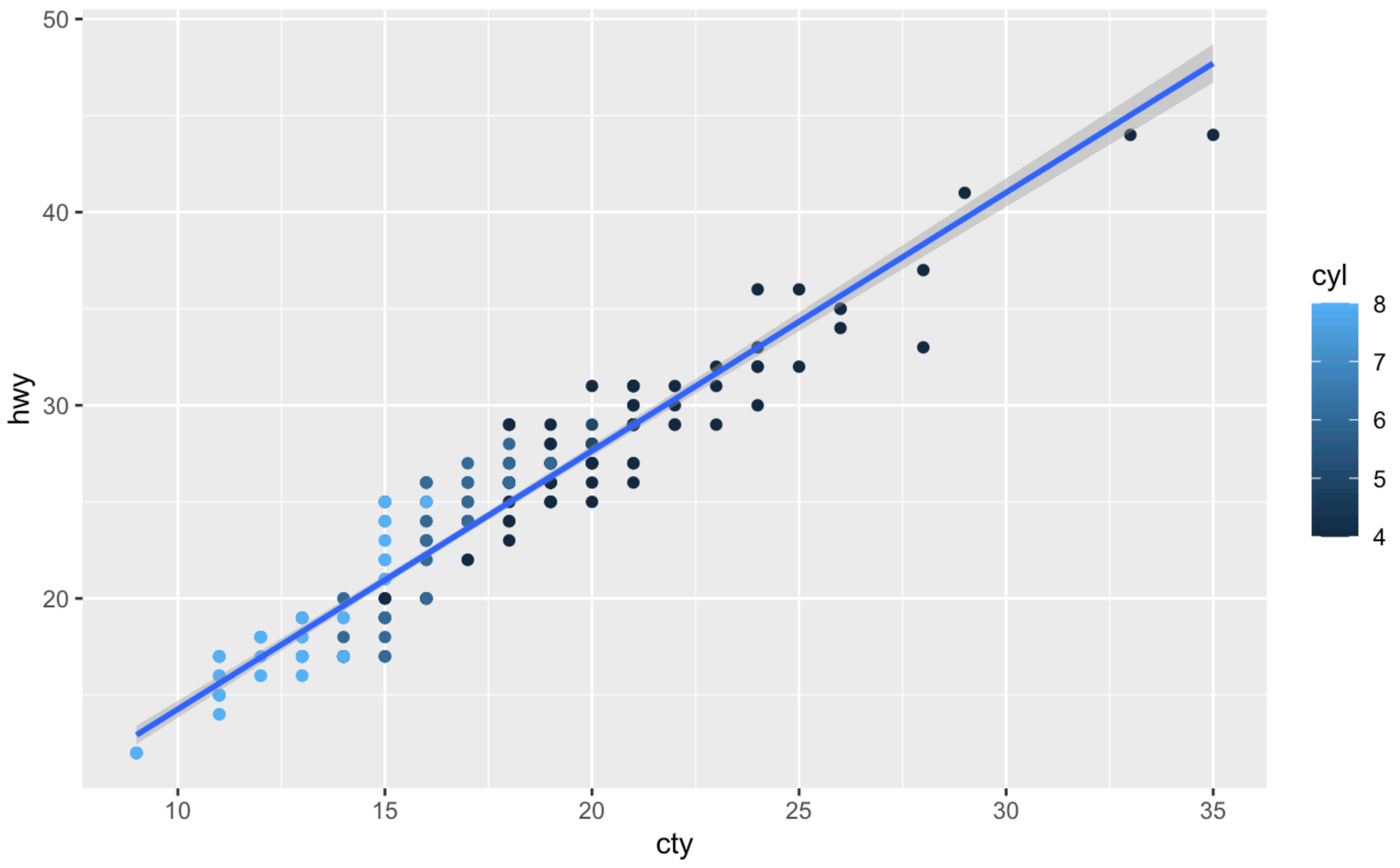
1-10 of 234 rows | 1-10 of 11 columns   Previous **1** 2 3 4 5 6 ... 24 Next

How do highway and city mpg relate to one another, and is there an effect of the model year?

Hide

```
ggplot(data = mpg,aes(x = cty, y = hwy, color = cyl)) + geom_point() + geom_smooth(method = "lm")
```

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

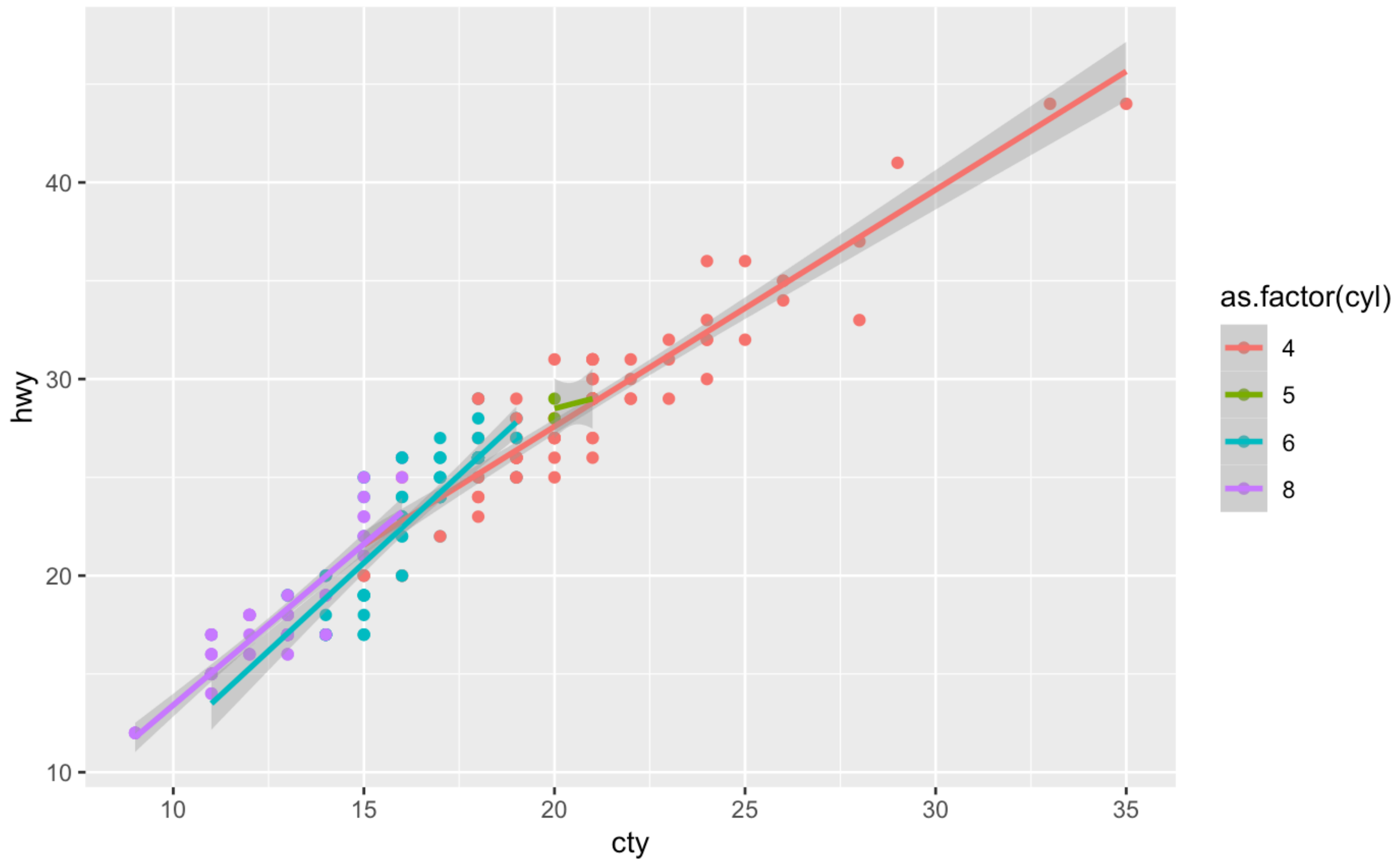


The above graph treats cylinder as a continuous variable (which is probably okay in this case). But what if it is discrete variable?

Hide

```
ggplot(data = mpg,aes(x = cty, y = hwy, color = as.factor(cyl))) + geom_point() + geom_smooth(method = "lm
")
```

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



Hide

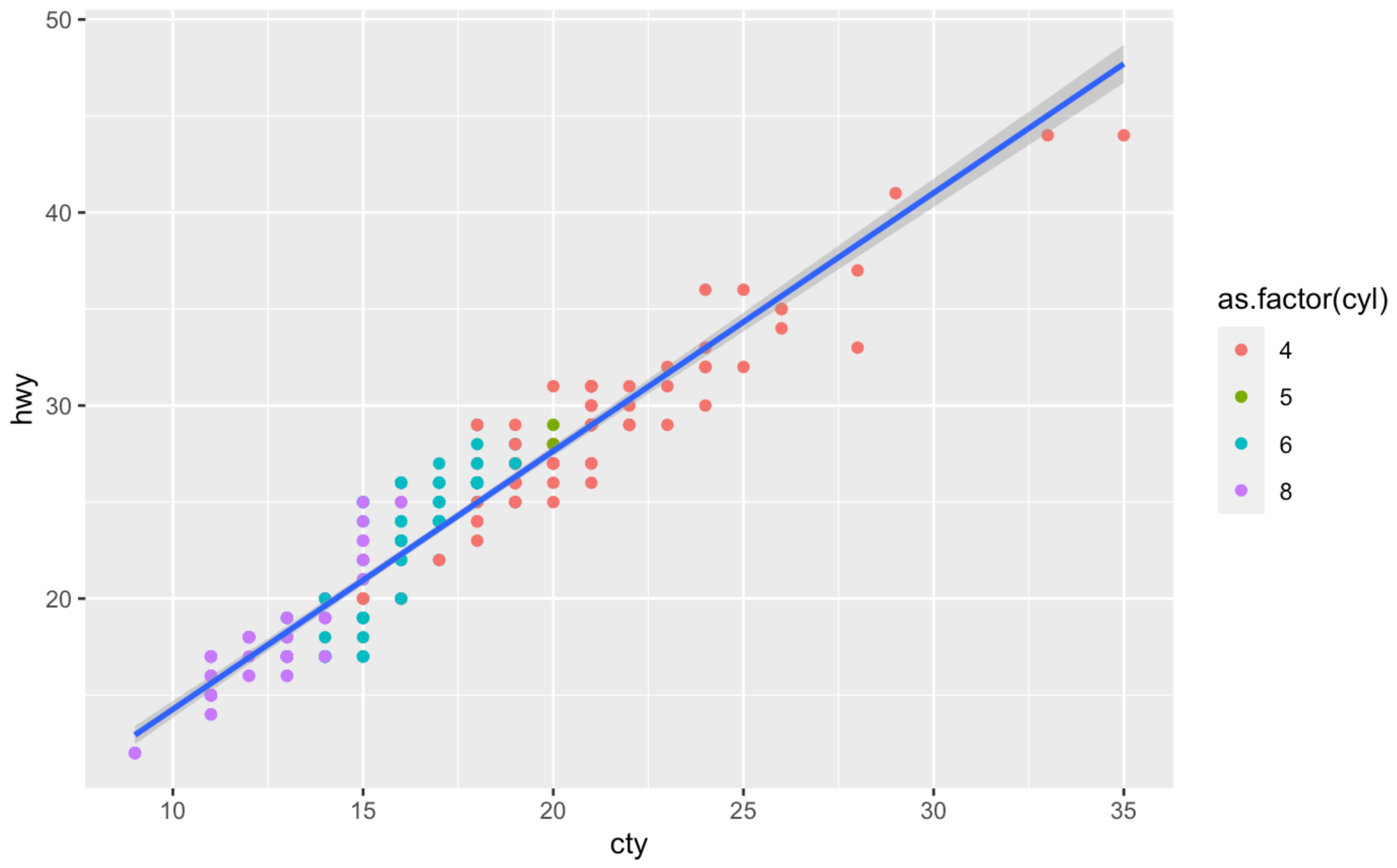
```
# alternatively, could also change it within the dataframe:
# mpg$cyl = factor(mpg$cyl)
```

Changing it to a factor means that we get individual lines of best fit for each level of the variable. Here’s a work-around to avoid that

Hide

```
ggplot(data = mpg,aes(x = cty, y = hwy)) + geom_point(aes(color = as.factor(cyl))) + geom_smooth(method =
"lm")
```

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



Hide

```
mpg
```

manufacturer <chr>	model <chr>	displ <dbl>	year <int>	cyl <int>	trans <chr>	drv <chr>	cty <int>	hwy <int>	fl <chr>
audi	a4	1.8	1999	4	auto(l5)	f	18	29	p
audi	a4	1.8	1999	4	manual(m5)	f	21	29	p
audi	a4	2.0	2008	4	manual(m6)	f	20	31	p
audi	a4	2.0	2008	4	auto(av)	f	21	30	p
audi	a4	2.8	1999	6	auto(l5)	f	16	26	p
audi	a4	2.8	1999	6	manual(m5)	f	18	26	p
audi	a4	3.1	2008	6	auto(av)	f	18	27	p
audi	a4 quattro	1.8	1999	4	manual(m5)	4	18	26	p
audi	a4 quattro	1.8	1999	4	auto(l5)	4	16	25	p
audi	a4 quattro	2.0	2008	4	manual(m6)	4	20	28	p

1-10 of 234 rows | 1-10 of 11 columns Previous **1** 2 3 4 5 6 ... 24 Next

What is the average highway mpg by year for each manufacturer?

Hide

```
groupedData = group_by(mpg, manufacturer, year) %>% summarise(meanMPG = mean(hwy),sdMPG = sd(hwy)/sqrt(n()
))
```

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

Hide

```
groupedData
```

manufacturer <chr>	year <int>	meanMPG <dbl>	sdMPG <dbl>
audi	1999	26.11111	0.5879447
audi	2008	26.77778	0.8624541
chevrolet	1999	21.57143	1.9254825
chevrolet	2008	22.08333	1.5396691
dodge	1999	18.43750	0.8513164
dodge	2008	17.57143	0.8152459
ford	1999	18.60000	0.8663003
ford	2008	20.50000	0.9803627
honda	1999	31.60000	0.6782330
honda	2008	33.75000	1.6520190

1-10 of 30 rows Previous **1** 2 3 Next

For the sake of space, let's only plot Audi, Chevy, Dodge, Ford, and Honda

Hide

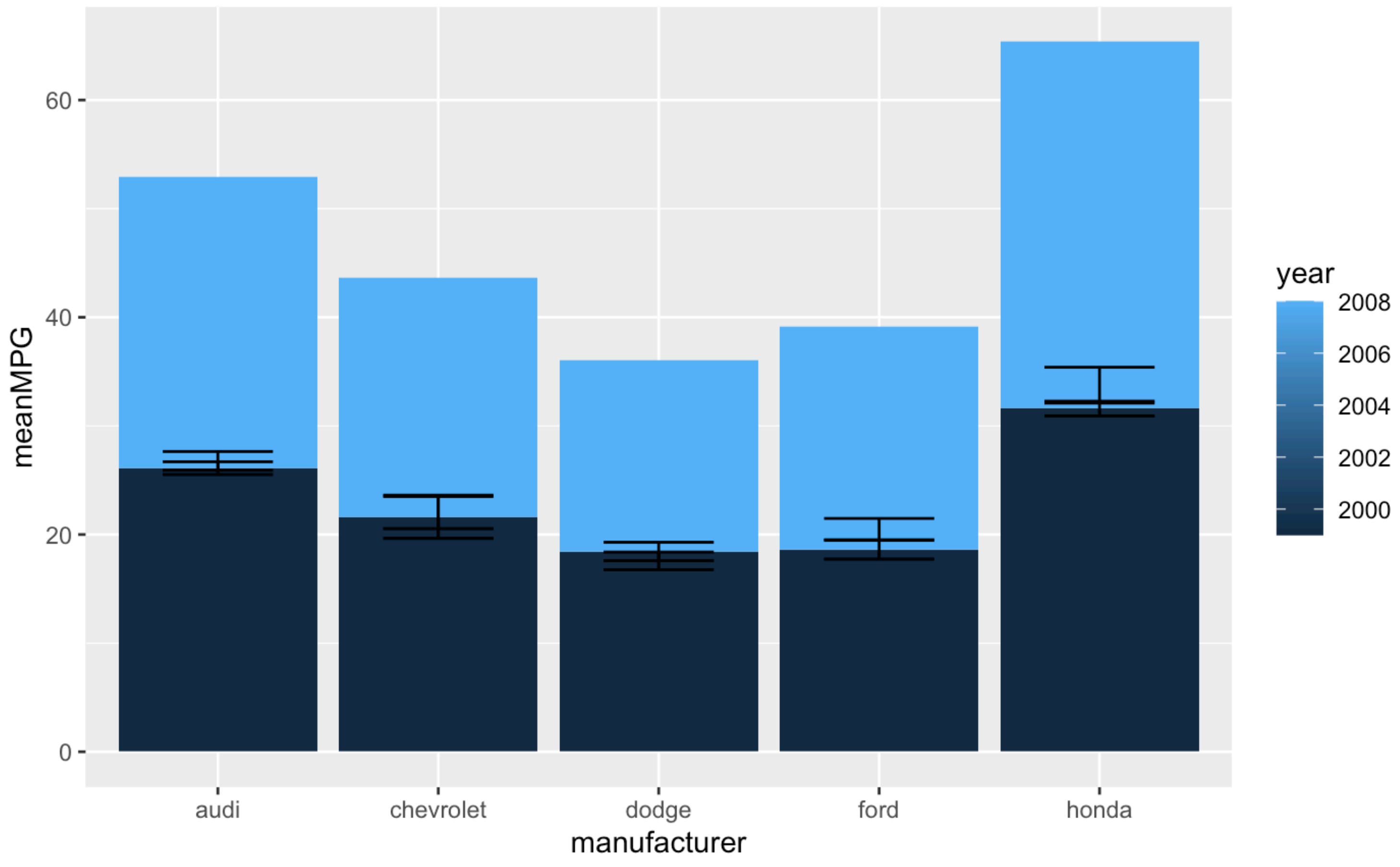
```
filteredData = filter(groupedData,manufacturer %in% c("audi","chevrolet","dodge","ford","honda"))
filteredData
```

manufacturer <chr>	year <int>	meanMPG <dbl>	sdMPG <dbl>
audi	1999	26.11111	0.5879447
audi	2008	26.77778	0.8624541
chevrolet	1999	21.57143	1.9254825
chevrolet	2008	22.08333	1.5396691
dodge	1999	18.43750	0.8513164
dodge	2008	17.57143	0.8152459
ford	1999	18.60000	0.8663003
ford	2008	20.50000	0.9803627
honda	1999	31.60000	0.6782330
honda	2008	33.75000	1.6520190

1-10 of 10 rows

Hide

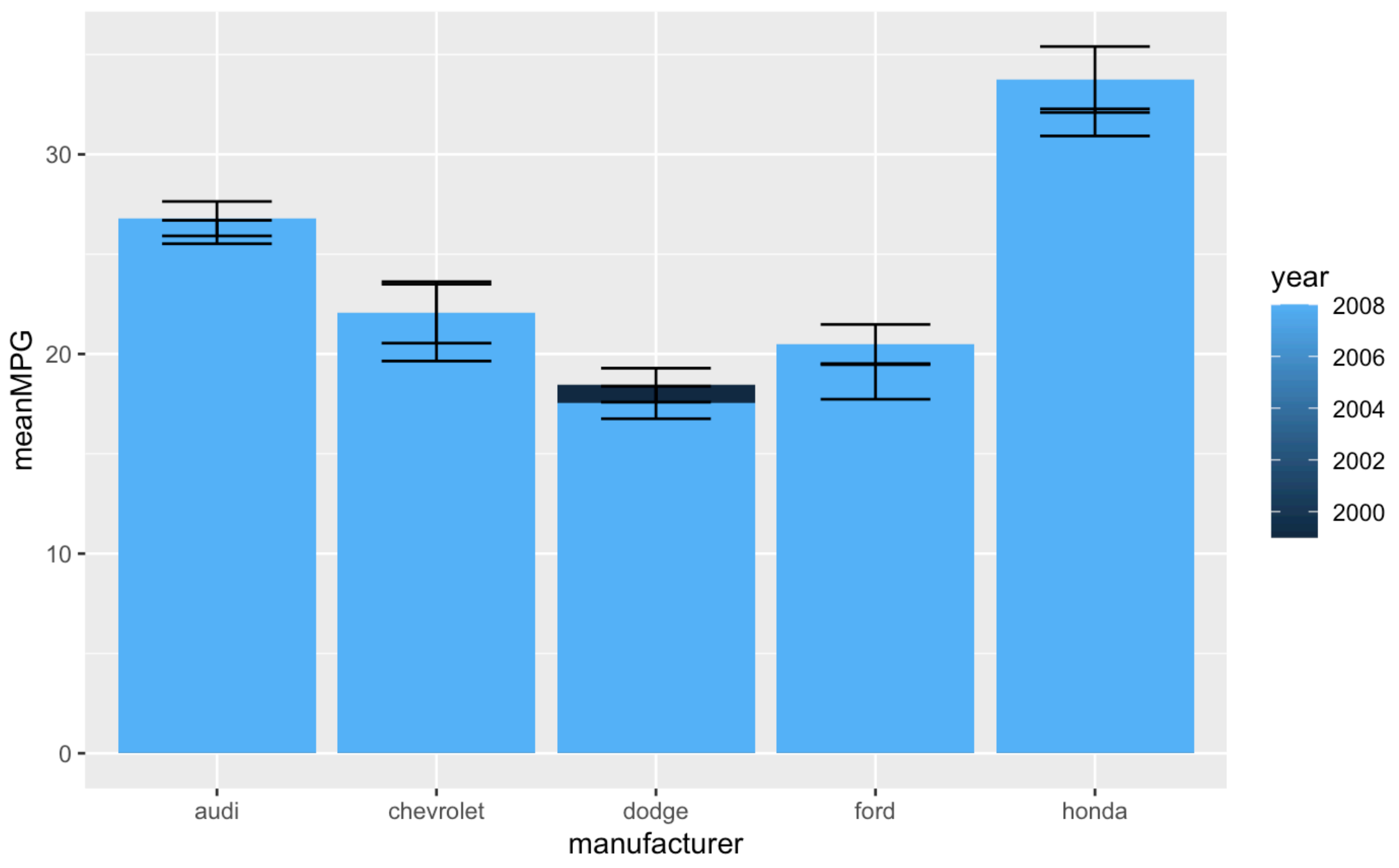
```
ggplot(data = filteredData,aes(x = manufacturer, y = meanMPG, fill = year)) + geom_bar(stat = "identity")
+ geom_errorbar(data = filteredData,aes(x = manufacturer, ymin = meanMPG - sdMPG, ymax = meanMPG + sdMPG),
width=.5)
```



Unstack the bars

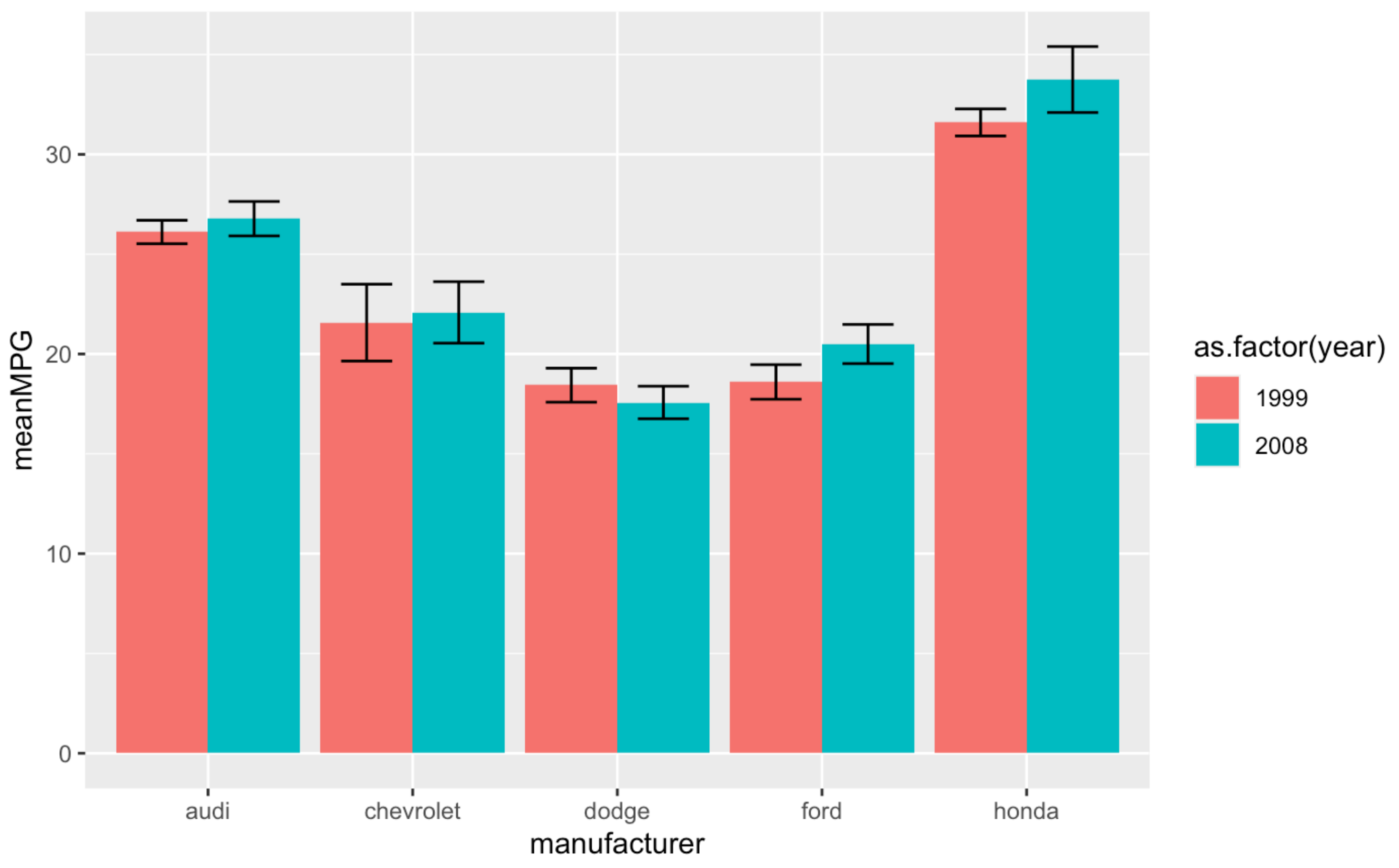
Hide

```
ggplot(data = filteredData,aes(x = manufacturer, y = meanMPG, fill = year)) + geom_bar(stat = "identity",
position = "dodge") + geom_errorbar(data = filteredData,aes(x = manufacturer, ymin = meanMPG - sdMPG, ymax
= meanMPG + sdMPG),width=.5, position = position_dodge(.9))
```



Why did that not work?

```
ggplot(data = filteredData,aes(x = manufacturer, y = meanMPG, fill = as.factor(year))) + geom_bar(stat = "
identity", position = "dodge") + geom_errorbar(data = filteredData,aes(x = manufacturer, ymin = meanMPG -
sdMPG, ymax = meanMPG + sdMPG),width=.5, position = position_dodge(.9))
```



Make generative art! Courtesy of https://r-graph-gallery.com/137-spring-shapes-data-art.html (https://urldefense.com/v3/__https://r-graph-gallery.com/137-spring-shapes-data-art.html__;!!IBzWLUs!TpBr_vtWaMdvzFRYPUXCHpwR9wrd560zf3bU_FNBSVYHI79rBF0WyD_euGnF0gyksBBD25IFGGY88Wv_fwf46TGl$)

Hide

```
set.seed(567)
ngroup=30
names=paste("G_",seq(1,ngroup),sep="")
DAT=data.frame()

for(i in seq(1:30)){
    data=data.frame( matrix(0, ngroup , 3))
    data[,1]=i
    data[,2]=sample(names, nrow(data))
    data[,3]=prop.table(sample( c(rep(0,100),c(1:ngroup)) ,nrow(data)))
    DAT=rbind(DAT,data)
    }
colnames(DAT)=c("Year","Group","Value")
DAT=DAT[order( DAT$Year, DAT$Group) , ]


coul = brewer.pal(12, "Paired")
coul = colorRampPalette(coul)(ngroup)
coul=coul[sample(c(1:length(coul)) , size=length(coul) ) ]

ggplot(DAT, aes(x=Year, y=Value, fill=Group )) +
    geom_area(alpha=1  )+
    theme_bw() +
    #scale_fill_brewer(colour="red", breaks=rev(levels(DAT$Group)))+
    scale_fill_manual(values = coul)+
     theme(
        text = element_blank(),
        line = element_blank(),
        title = element_blank(),
        legend.position="none",
        panel.border = element_blank(),
        panel.background = element_blank())
ggsave('generative_art.pdf',width=7,height=5)
```