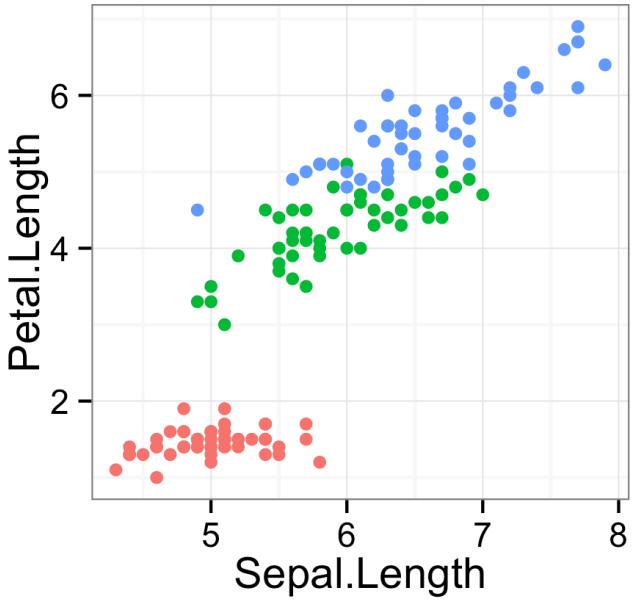
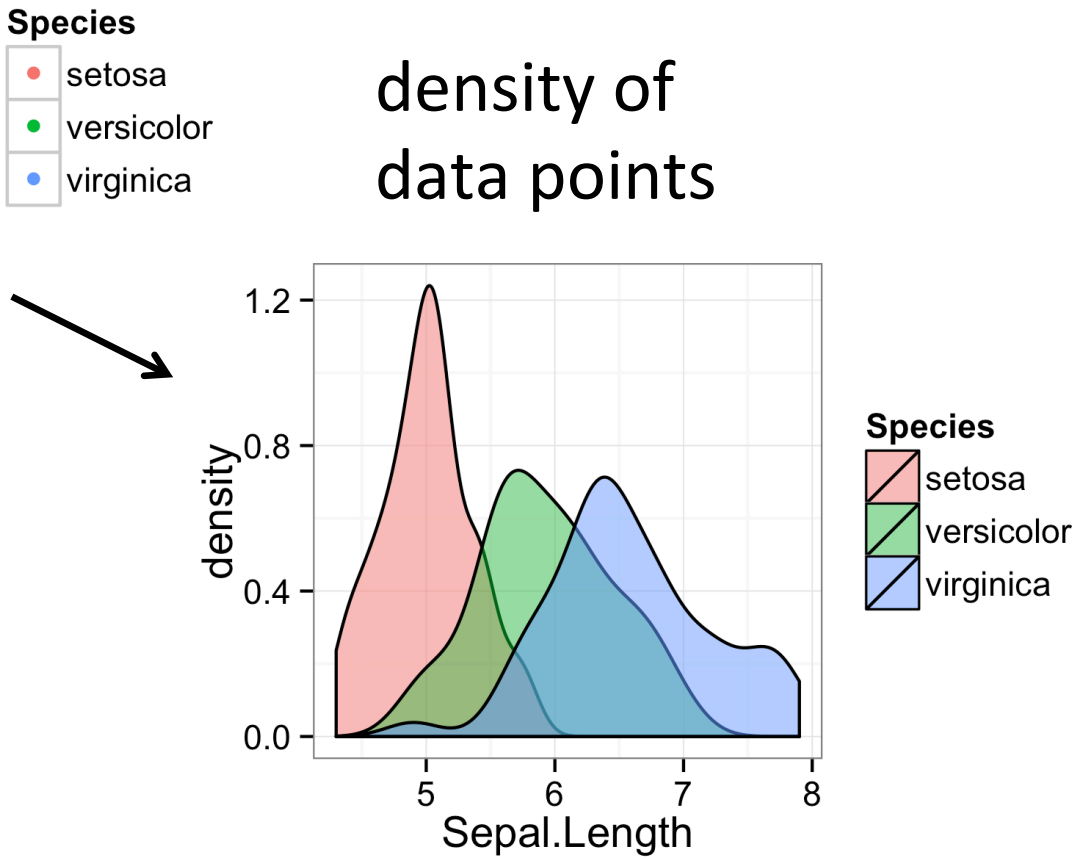
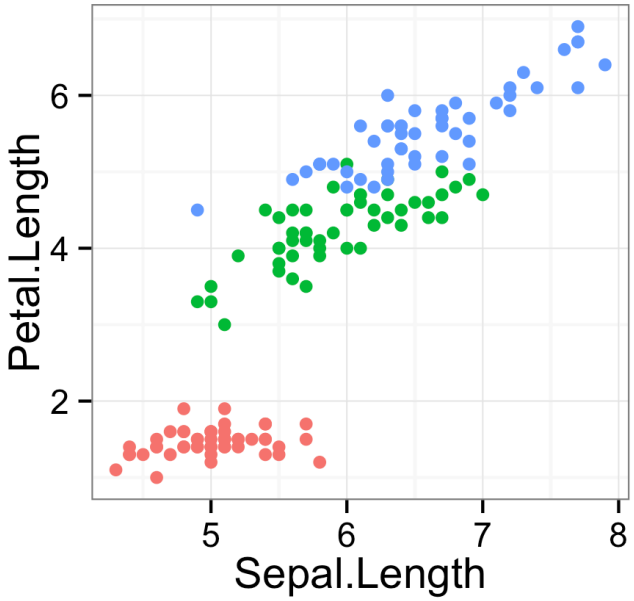
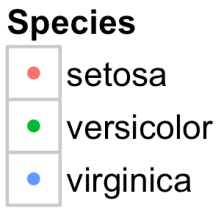
We often need to do statistical transformations before plotting



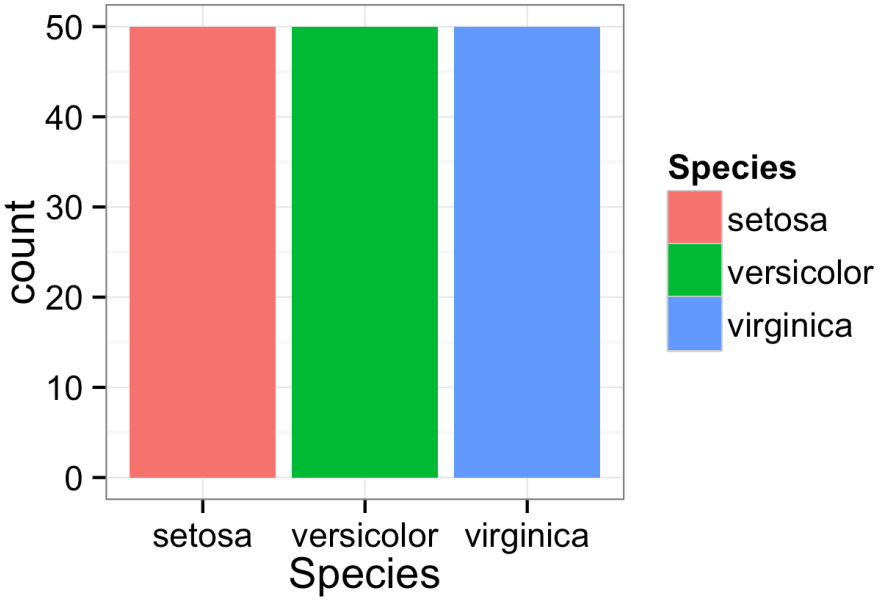


We often need to do statistical transformations before plotting

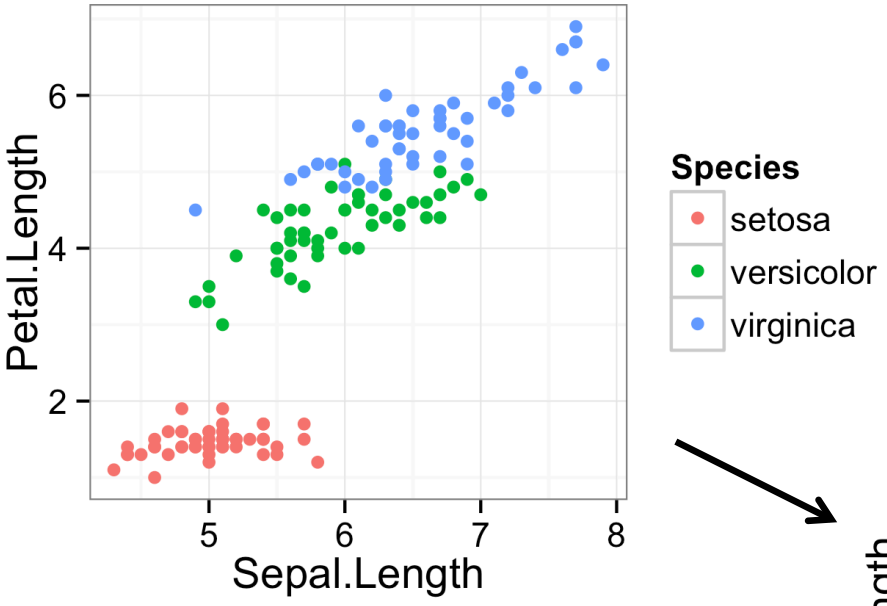




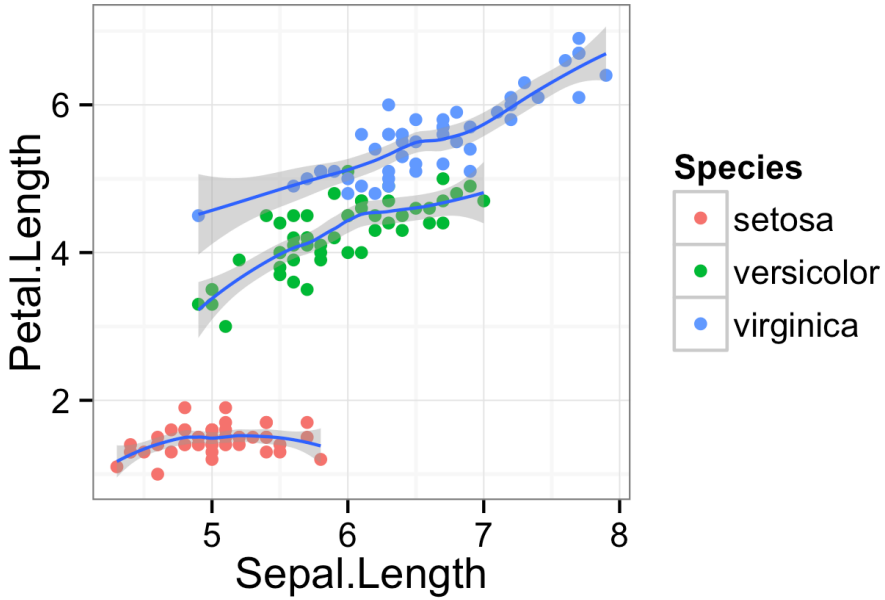
count of number of different types



We often need to do statistical transformations before plotting



statistical smoothing/ trend lines



In ggplot2, these transformations are done with stats

- `stat_bin`
  Bin data.

- `stat_bin2d`
  Count number of observation in rectangular bins.
- `stat_bindot`
  Bin data for dot plot.

- `stat_binhex`
  Bin 2d plane into hexagons.
- `stat_boxplot`
  Calculate components of box and whisker plot.

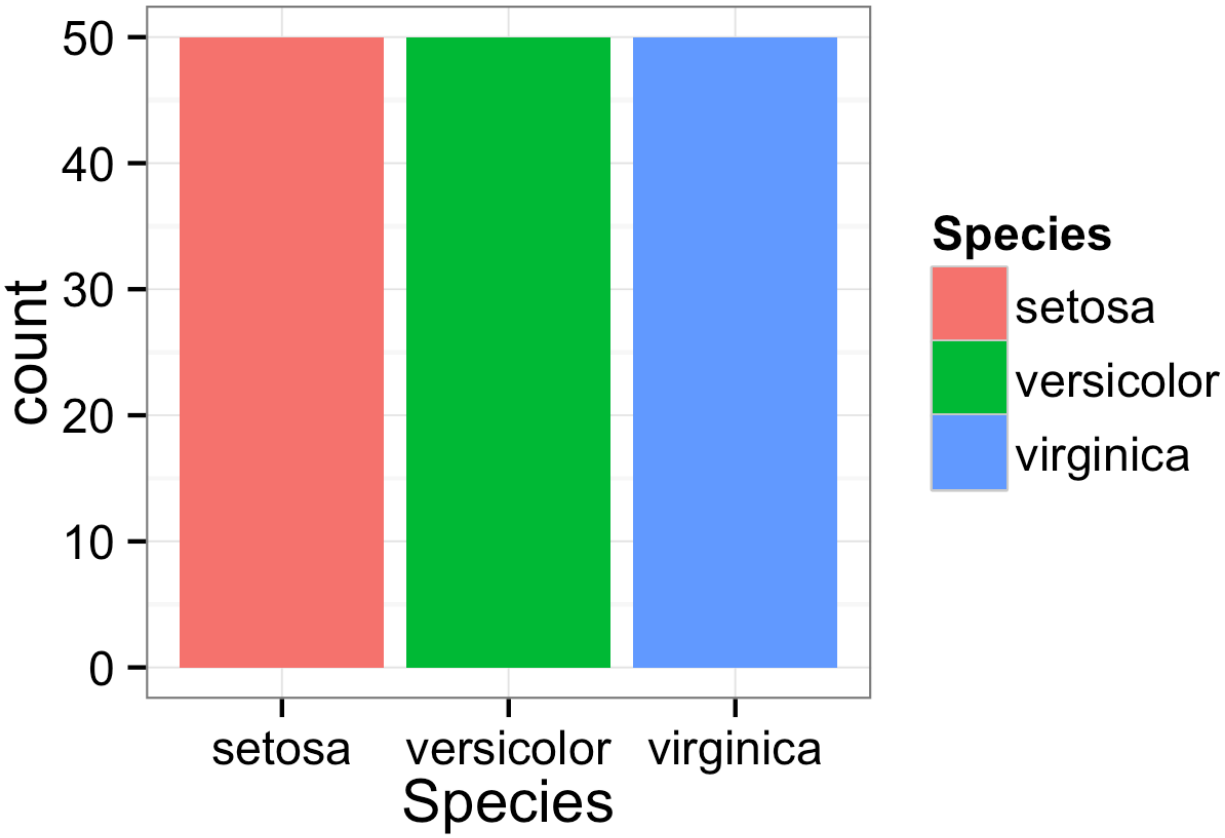
- `stat_contour`
  Calculate contours of 3d data.

- `stat_density`
  1d kernel density estimate.

- `stat_density2d`

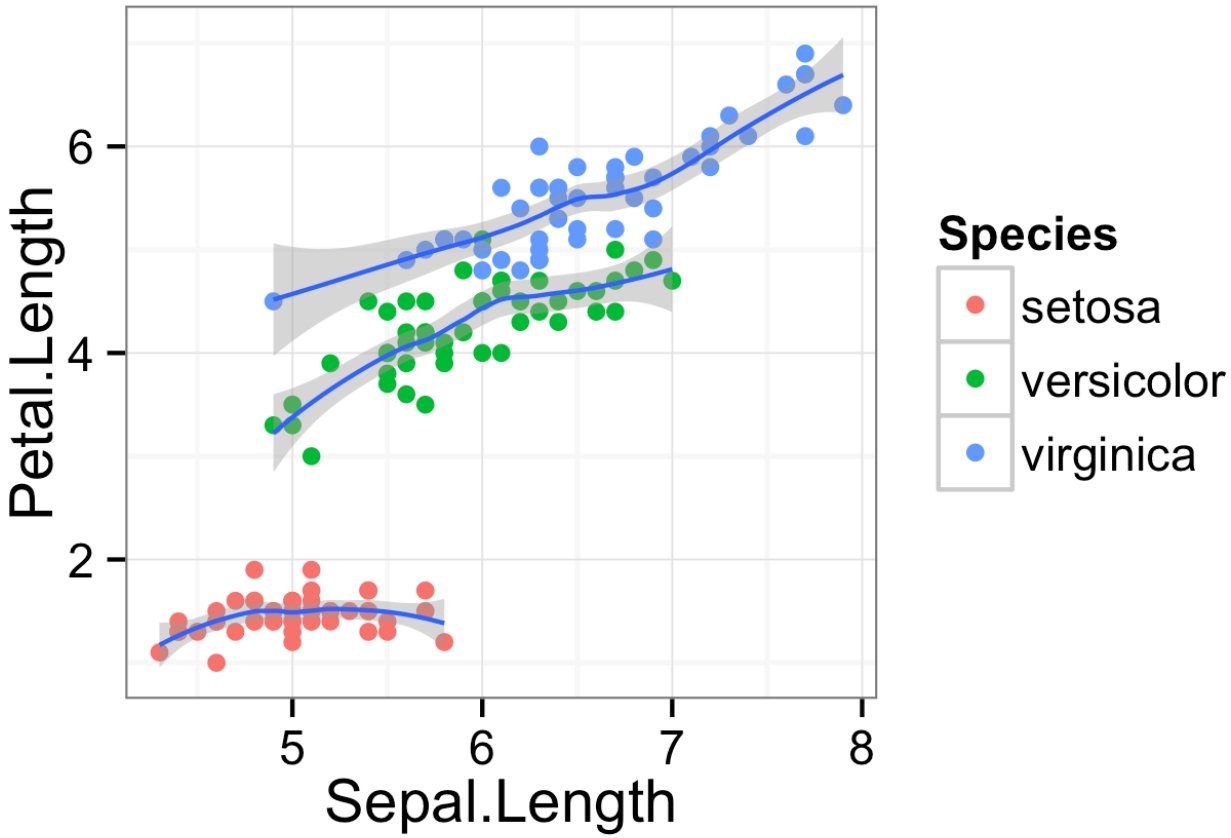
In most cases we just need to call the appropriate geom and it calls a stat

```
ggplot(iris, aes(x=Species, fill=Species)) +
                                    geom_bar()
```



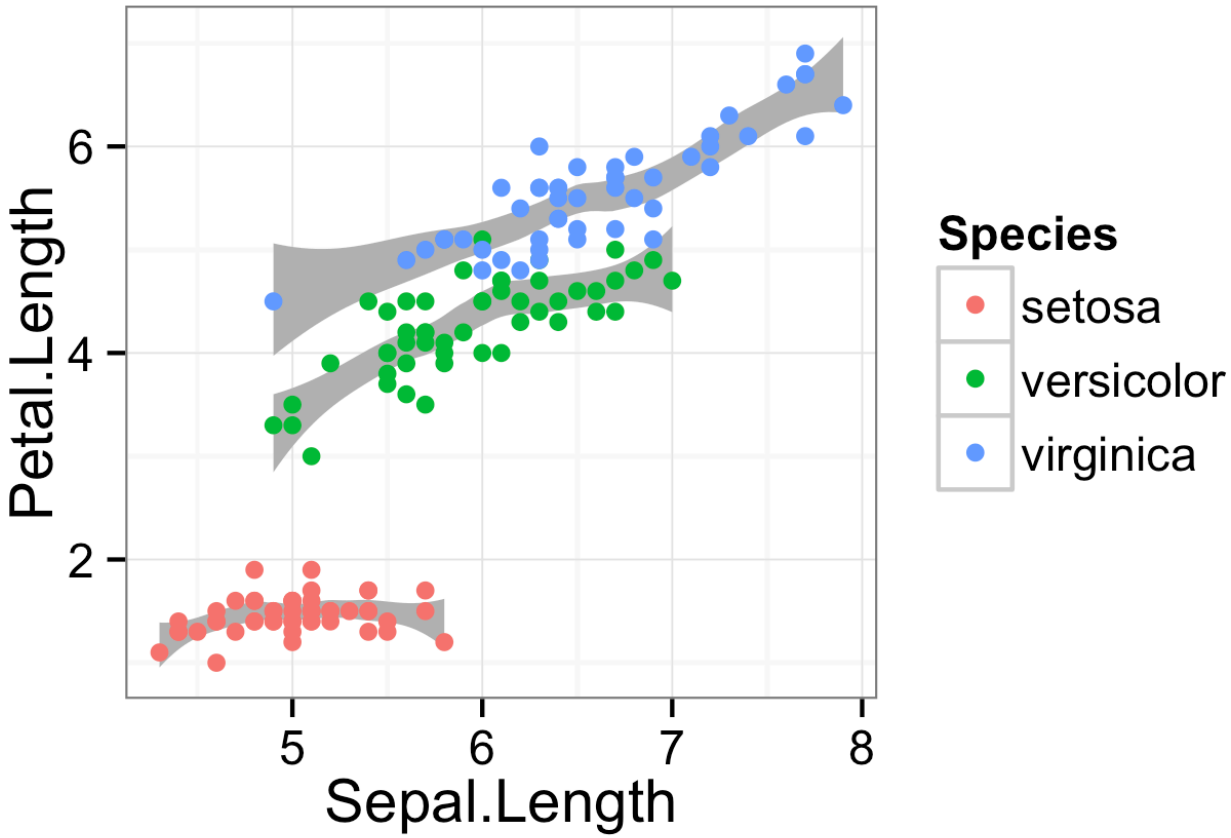
In most cases we just need to call the appropriate geom and it calls a stat

```
ggplot(iris, aes(x=Sepal.Length, y=Petal.Length)) +
                    geom_point(aes(color=Species)) +
                    geom_smooth(aes(group=Species))
```



However, sometimes it can be helpful to call the stat directly

```
ggplot(iris, aes(x=Sepal.Length, y=Petal.Length)) +
stat_smooth(aes(group=Species), geom="ribbon", fill='gray70') +
                                    geom_point(aes(color=Species))
```



Scales define how to map data onto aesthetics

- `scale_x_continuous` (scale_x_log10, scale_x_reverse, scale_x_sqrt, scale_y_continuous, scale_y_log10, scale_y_reverse, scale_y_sqrt)
  Continuous position scales (x & y).
- `scale_x_date` (scale_y_date)
  Position scale, date
- `scale_x_datetime` (scale_y_datetime)
  Position scale, date
- `scale_x_discrete` (scale_y_discrete)
  Discrete position.

Scales define how to map data onto aesthetics

- `scale_colour_brewer` (scale_color_brewer, scale_fill_brewer)
  Sequential, diverging and qualitative colour scales from colorbrewer.org

- `scale_colour_gradient` (scale_color_continuous, scale_color_gradient, scale_colour_continuous, scale_fill_continuous, scale_fill_gradient)
  Smooth gradient between two colours

- `scale_colour_gradient2` (scale_color_gradient2, scale_fill_gradient2)
  Diverging colour gradient

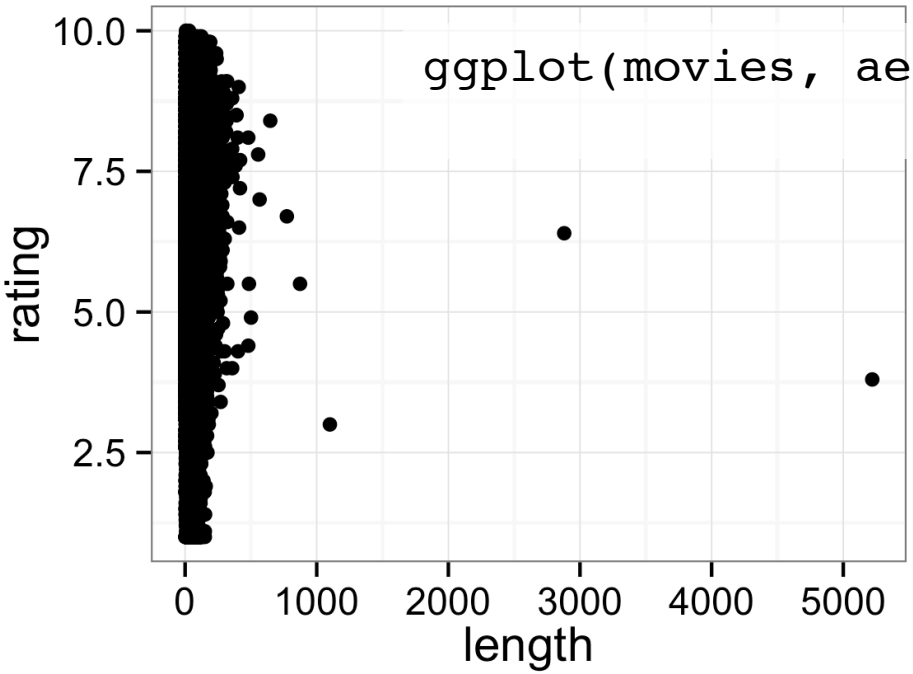
- `scale_colour_gradientn` (scale_color_gradientn, scale_fill_gradientn)
  Smooth colour gradient between n colours

- `scale_colour_grey` (scale_color_grey, scale_fill_grey)
  Sequential grey colour scale.

Scales define how to map data onto aesthetics

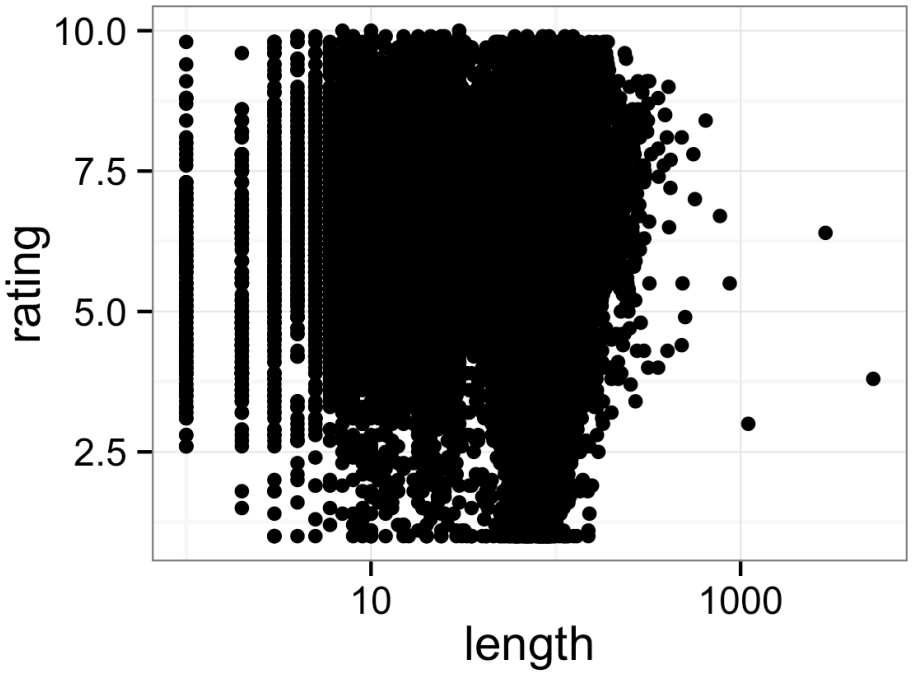
- `scale_linetype` (scale_linetype_continuous, scale_linetype_discrete)
  Scale for line patterns.
- `scale_shape` (scale_shape_continuous, scale_shape_discrete)
  Scale for shapes, aka glyphs.
- `scale_size` (scale_size_continuous, scale_size_discrete)
  Size scale.

Example 1: Change scaling of x axis

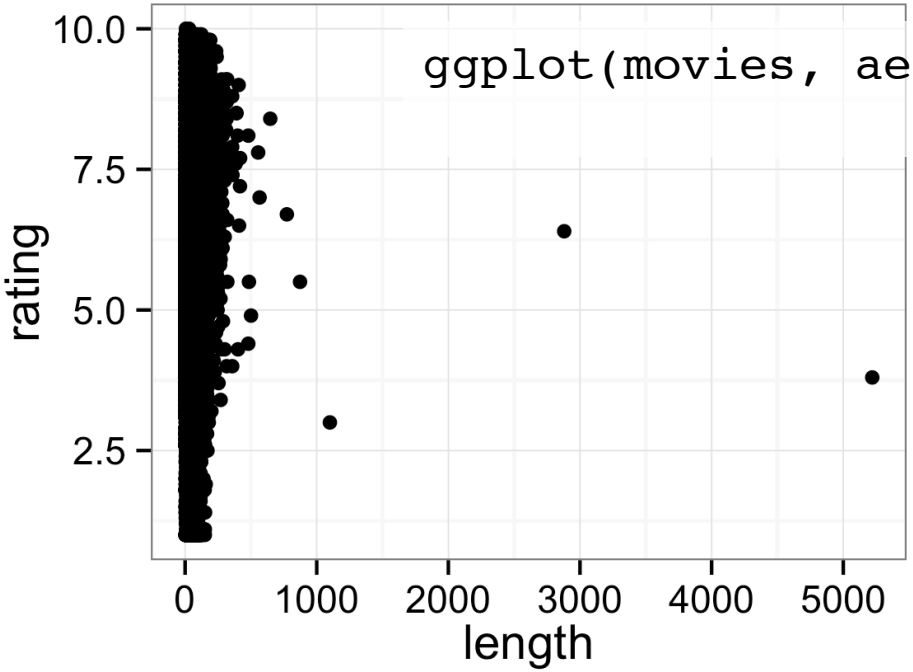



```
ggplot(movies, aes(x=length, y=rating)) +
                                    geom_point()
```

```
+ scale_x_log10()
```

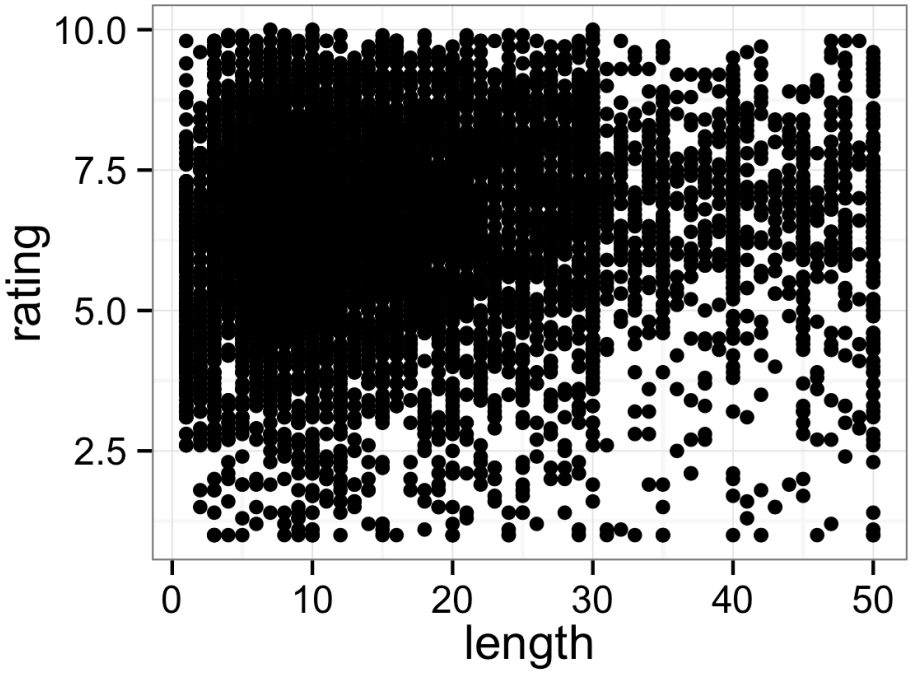


Example 1: Change scaling of x axis

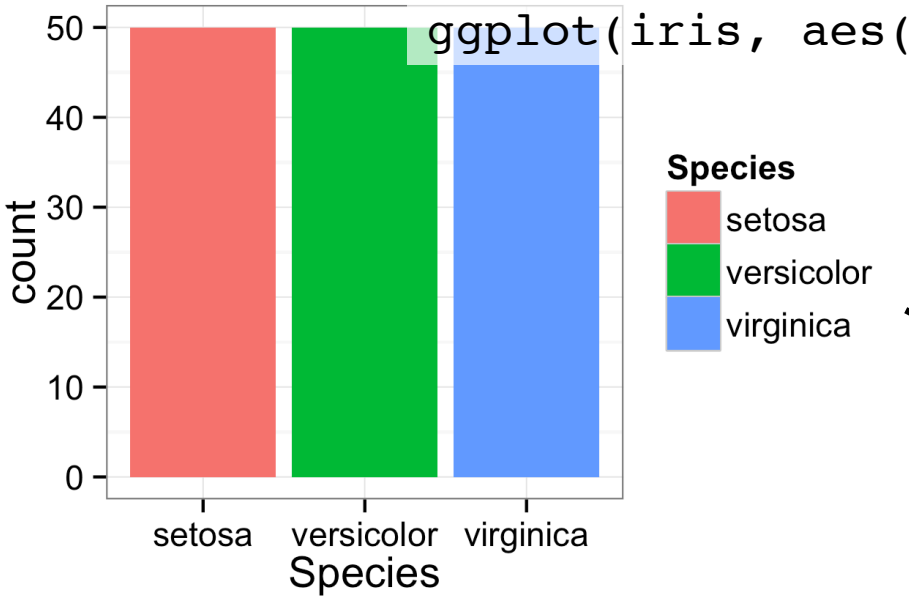



```
ggplot(movies, aes(x=length, y=rating)) +
                                    geom_point()
```

```
+ xlim(1, 50)
```

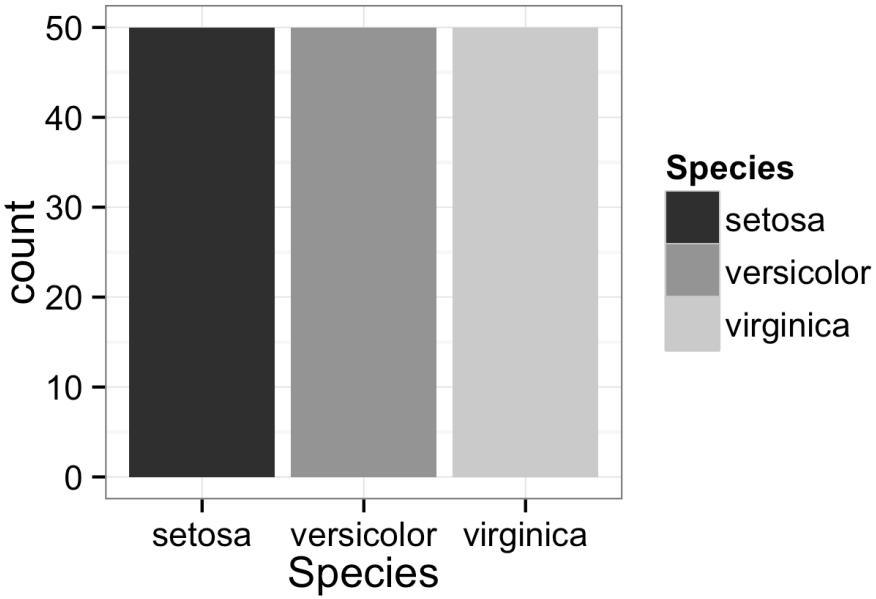


Example 2: Change color scaling

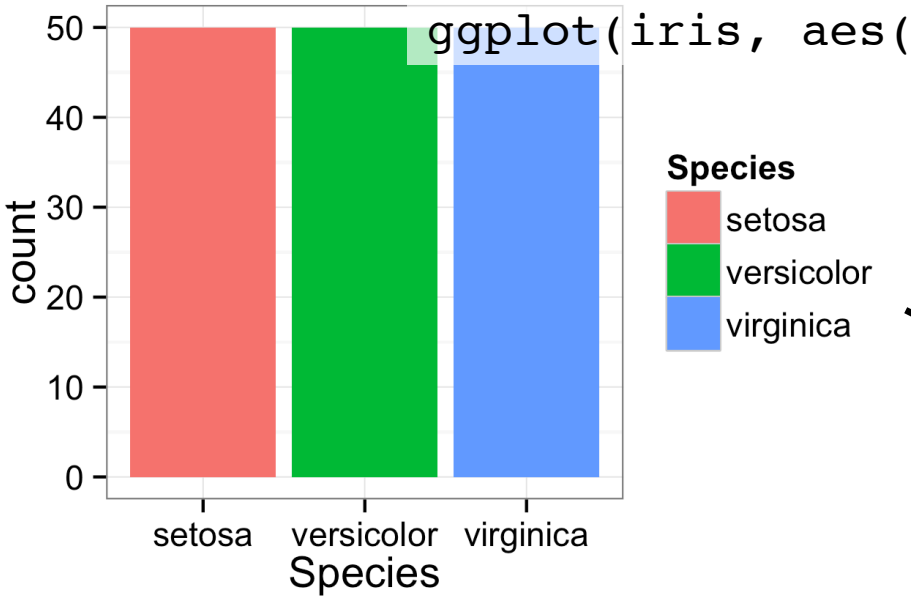



```
ggplot(iris, aes(x=Species, fill=Species)) +
                                        geom_bar()
```

```
+ scale_fill_grey()
```



Example 2: Change color scaling




```
ggplot(iris, aes(x=Species, fill=Species)) +
                                        geom_bar()
```

```
+ scale_fill_brewer()
```