Visualization Notes - Chapter 2 and 3

Key Components of a Visual - Yau's Taxonomy

- Visual cues
- Coordinate systems
- Scale
- Context

GGplot2's Grammar of Graphics

Based on Wilkinson's grammar of graphics, very similar to Yau's Taxonomy

- Layer - which consists of data and a mapping (required), and may contain a statistical transformation (stat), a geometric object (geom), and a position adjustment (position)
- Coordinate system (coord)
- Scale
- Faceting

Context is included by adding axis labels, titles, legends, etc. (e.g. labs)

Data

FIFA game from 2019 - soccer - these are player statistics from within the game, combined with some real-life information (demographics, team ID, etc.). We will consider players with an Overall ability score in the game over 85.

There are 40 variables in the data set with 77 observations (down from over 18000 players in the original).

```
glimpse(Fifasmall)
```

```
## Rows: 77
## Columns: 40
## $ Name            <chr> "L. Messi", "Cristiano Ronaldo", "Neymar Jr", "De Gea"~
## $ Age             <dbl> 31, 33, 26, 27, 27, 27, 32, 31, 32, 25, 29, 28, 32, 32~
## $ Overall         <dbl> 94, 94, 92, 91, 91, 91, 91, 91, 91, 90, 90, 90, 90, 90~
## $ Club            <chr> "FC Barcelona", "Juventus", "Paris Saint-Germain", "Ma~
## $ PreferredFoot   <chr> "Left", "Right", "Right", "Right", "Right", "Right", "~
## $ Position        <chr> "RF", "ST", "LW", "GK", "RCM", "LF", "RCM", "RS", "RCB~
## $ Crossing        <dbl> 84, 84, 79, 17, 93, 81, 86, 77, 66, 13, 62, 88, 55, 84~
## $ Finishing       <dbl> 95, 94, 87, 13, 82, 84, 72, 93, 60, 11, 91, 76, 42, 76~
## $ HeadingAccuracy <dbl> 70, 89, 62, 21, 55, 61, 55, 77, 91, 15, 85, 54, 92, 54~
## $ ShortPassing    <dbl> 90, 81, 84, 50, 92, 89, 93, 82, 78, 29, 83, 92, 79, 93~
## $ Volleys         <dbl> 86, 87, 84, 13, 82, 80, 76, 88, 66, 13, 89, 82, 47, 82~
## $ Dribbling       <dbl> 97, 88, 96, 18, 86, 95, 90, 87, 63, 12, 85, 81, 53, 89~
## $ Curve           <dbl> 93, 81, 88, 21, 85, 83, 85, 86, 74, 13, 77, 86, 49, 82~
## $ FKAccuracy      <dbl> 94, 76, 87, 19, 83, 79, 78, 84, 72, 14, 86, 84, 51, 77~
## $ LongPassing     <dbl> 87, 77, 78, 51, 91, 83, 88, 64, 77, 26, 65, 93, 70, 87~
## $ BallControl     <dbl> 96, 94, 95, 42, 91, 94, 93, 90, 84, 16, 89, 90, 76, 94~
```

```
## $ Acceleration    <dbl> 91, 89, 94, 57, 78, 94, 80, 86, 76, 43, 77, 64, 68, 70~
## $ SprintSpeed     <dbl> 86, 91, 90, 58, 76, 88, 72, 75, 75, 60, 78, 62, 68, 64~
## $ Agility         <dbl> 91, 87, 96, 60, 79, 95, 93, 82, 78, 67, 78, 70, 58, 92~
## $ Reactions       <dbl> 95, 96, 94, 90, 91, 90, 90, 92, 85, 86, 90, 89, 85, 90~
## $ Balance         <dbl> 95, 70, 84, 43, 77, 94, 94, 83, 66, 49, 78, 71, 54, 90~
## $ ShotPower       <dbl> 85, 95, 80, 31, 91, 82, 79, 86, 79, 22, 88, 87, 67, 72~
## $ Jumping         <dbl> 68, 95, 61, 67, 63, 56, 68, 69, 93, 76, 84, 30, 91, 64~
## $ Stamina         <dbl> 72, 88, 81, 43, 90, 83, 89, 90, 84, 41, 78, 75, 66, 78~
## $ Strength        <dbl> 59, 79, 49, 64, 75, 66, 58, 83, 83, 78, 84, 73, 88, 52~
## $ LongShots       <dbl> 94, 93, 82, 12, 91, 80, 82, 85, 59, 12, 84, 92, 43, 75~
## $ Aggression      <dbl> 48, 63, 56, 38, 76, 54, 62, 87, 88, 34, 80, 60, 89, 57~
## $ Interceptions   <dbl> 22, 29, 36, 30, 61, 41, 83, 41, 90, 19, 39, 82, 88, 50~
## $ Positioning     <dbl> 94, 95, 89, 12, 87, 87, 79, 92, 60, 11, 91, 79, 48, 89~
## $ Vision          <dbl> 94, 82, 87, 68, 94, 89, 92, 84, 63, 70, 77, 86, 52, 92~
## $ Penalties       <dbl> 75, 85, 81, 40, 79, 86, 82, 85, 75, 11, 88, 73, 50, 75~
## $ Composure       <dbl> 96, 95, 94, 68, 88, 91, 84, 85, 82, 70, 86, 85, 82, 93~
## $ Marking         <dbl> 33, 28, 27, 15, 68, 34, 60, 62, 87, 27, 34, 72, 90, 59~
## $ StandingTackle  <dbl> 28, 31, 24, 21, 58, 27, 76, 45, 92, 12, 42, 79, 89, 53~
## $ SlidingTackle   <dbl> 26, 23, 33, 13, 51, 22, 73, 38, 91, 18, 19, 69, 89, 29~
## $ GKDiving        <dbl> 6, 7, 9, 90, 15, 11, 13, 27, 11, 86, 15, 10, 6, 6, 15,~
## $ GKHandling      <dbl> 11, 11, 9, 85, 13, 12, 9, 25, 8, 92, 6, 11, 8, 15, 12,~
## $ GKKicking       <dbl> 15, 15, 15, 87, 5, 6, 7, 31, 9, 78, 12, 13, 15, 7, 10,~
## $ GKPositioning   <dbl> 14, 14, 15, 88, 10, 8, 14, 33, 7, 88, 8, 7, 5, 6, 7, 5~
## $ GKReflexes      <dbl> 8, 11, 11, 94, 13, 8, 9, 37, 11, 89, 10, 10, 15, 12, 1~
```

```
#Fifasmall %>%
  #group_by(Position) %>% summarize(count = n())

mosaic::tally(~ Position, data = Fifasmall)
```

```
## Registered S3 method overwritten by 'mosaic':
##   method                           from
##   fortify.SpatialPolygonsDataFrame ggplot2
```

```
## Position
## CAM  CB CDM  CM  GK LAM  LB LCB LCM LDM  LF  LM  LS  LW  RB RCB RCM RDM  RF  RM
##   5   3   4   1  10   1   3   5   4   1   3   4   2   5   1   6   3   1   2   2
##  RS  RW  ST
##   1   2   8
```

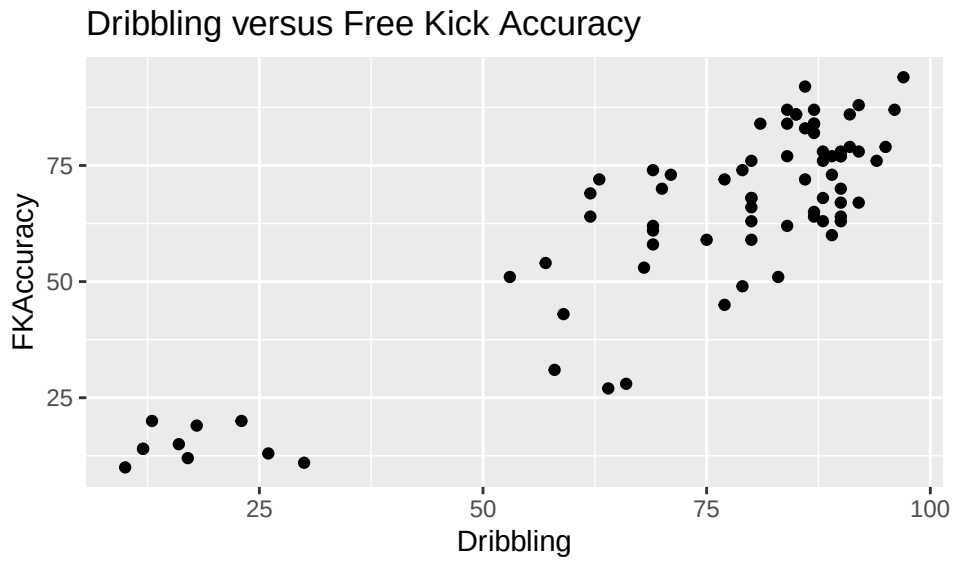
We'll learn how the filtering worked and how to wrangle Position next week. (You might argue it's easier to want to work with only 4 categories for Position - forwards, midfielders, defenders, and goalkeepers rather than this breakdown.)

Making Plots

The text demonstrates building graphics by literally *adding* to a saved plot object at each step. You can use this approach but you don't have to.

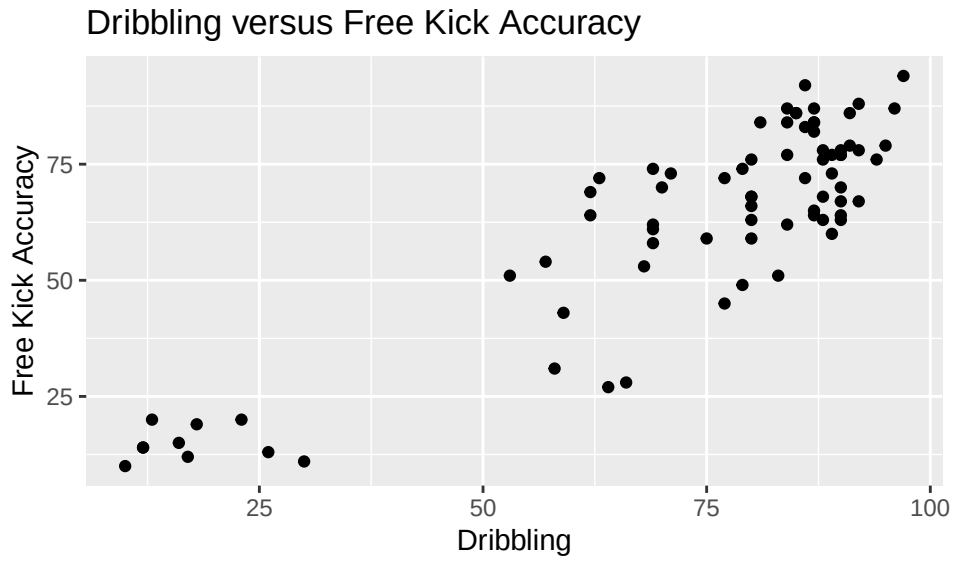
Example:

```
g <- ggplot(data = Fifasmall, mapping = aes(x = Dribbling, y = FKAccuracy))
g <- g + geom_point()
g <- g + labs(title = "Dribbling versus Free Kick Accuracy")
g
```




```
#is equivalent to

ggplot(data = Fifasmall, mapping = aes(x = Dribbling, y = FKAccuracy)) +
  geom_point() +
  labs(title = "Dribbling versus Free Kick Accuracy",
       y = "Free Kick Accuracy")
```



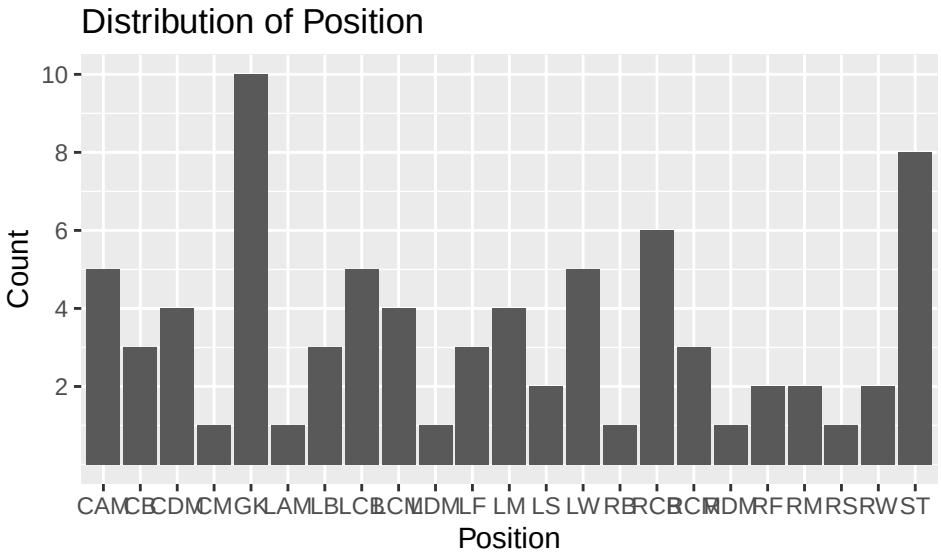
The former approach may be helpful as you learn, but if you are comfortable with the latter, feel free to use it.

Plots

Depending on your variable type, different plots are appropriate.

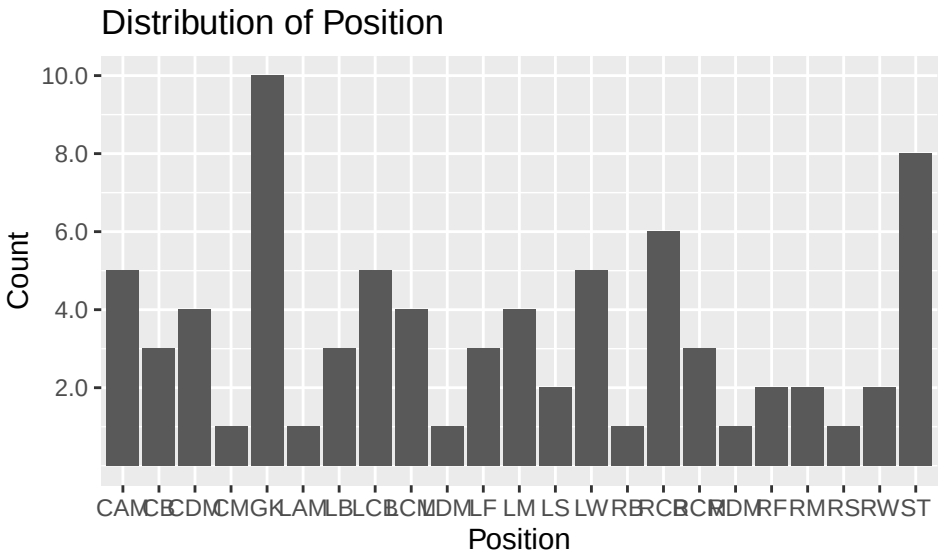
Drawing a bar chart for a categorical variable means you need the counts. We'll learn more about *group_by* and *summarize* next week, but that's how I got the counts here.

```
Poscounts <- Fifasmall %>%
  group_by(Position) %>% summarize(counts = n())
ggplot(data = Poscounts, mapping = aes(y = counts, x = Position)) +
  geom_col() +
  scale_y_continuous(breaks = c(2, 4, 6, 8, 10)) +
  labs(y = "Count",
       title = "Distribution of Position")
```



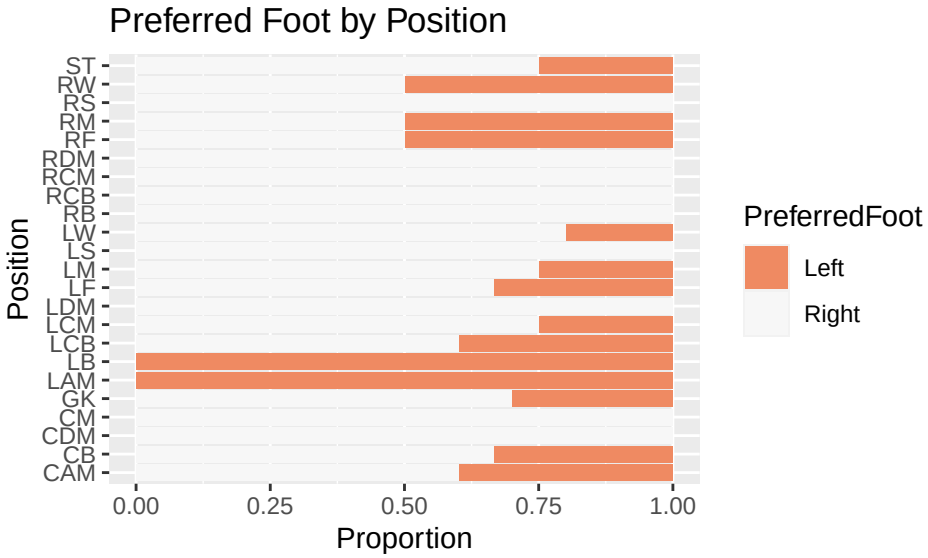
In class I was asked how to make the axis show 2.0 instead of 2, etc. Here's that version of the plot.

```
Poscounts <- Fifasmall %>%
  group_by(Position) %>% summarize(counts = n())
ggplot(data = Poscounts, mapping = aes(y = counts, x = Position)) +
  geom_col() +
  scale_y_continuous(breaks = c(2, 4, 6, 8, 10), label = scales::comma) +
  labs(y = "Count",
       title = "Distribution of Position")
```



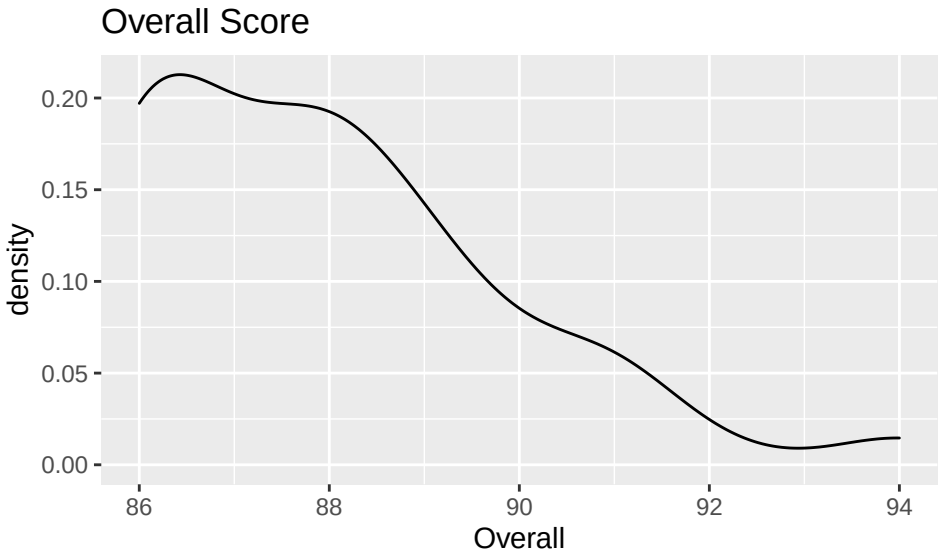
Stacked bar charts let you add a second categorical variable.

```
ggplot(data = Fifasmall, aes(x = Position)) +
  geom_bar(aes(fill = PreferredFoot), position = "fill") +
  scale_fill_brewer(palette = "RdBu") +  #try commenting this line out
  coord_flip() +
  labs(y = "Proportion",
       title = "Preferred Foot by Position") #note this is y before the flip!
```



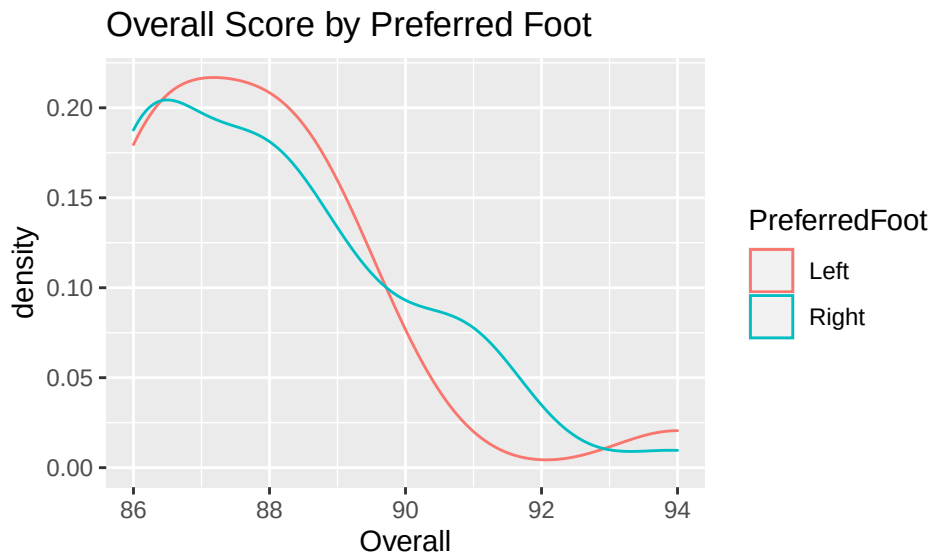
Working with quantitative variables is more common. For univariate analyses, I strongly encourage you to use density plots, rather than histograms.

```
ggplot(data = Fifasmall, aes(x = Overall)) +
  geom_density() +
  labs(title = "Overall Score")
```



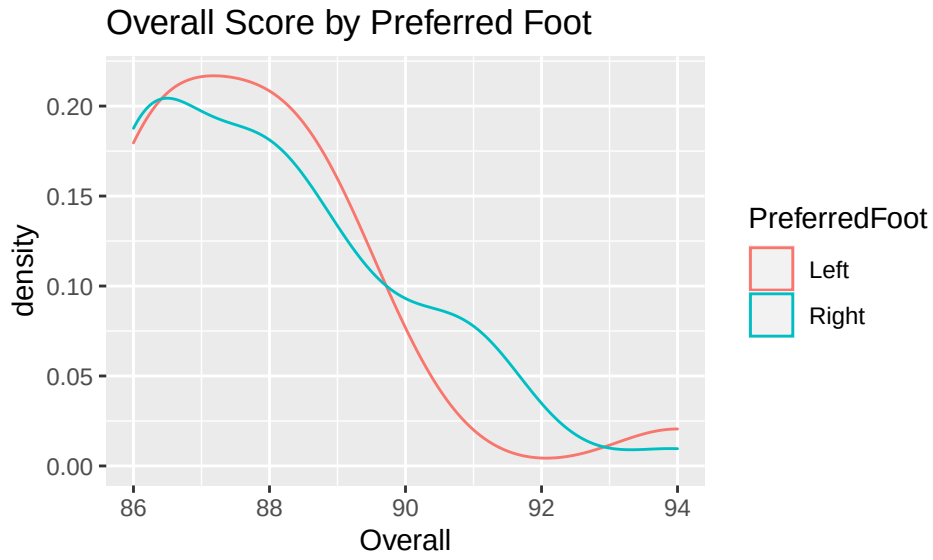
It's still easy to incorporate categorical variables to compare groups.

```
ggplot(data = Fifasmall, aes(x = Overall, color = PreferredFoot)) +
  geom_density() +
  labs(title = "Overall Score by Preferred Foot")
```



Note that here, we set the color in the aesthetic for the overall plot. We could also set it just within the geom_density() portion.

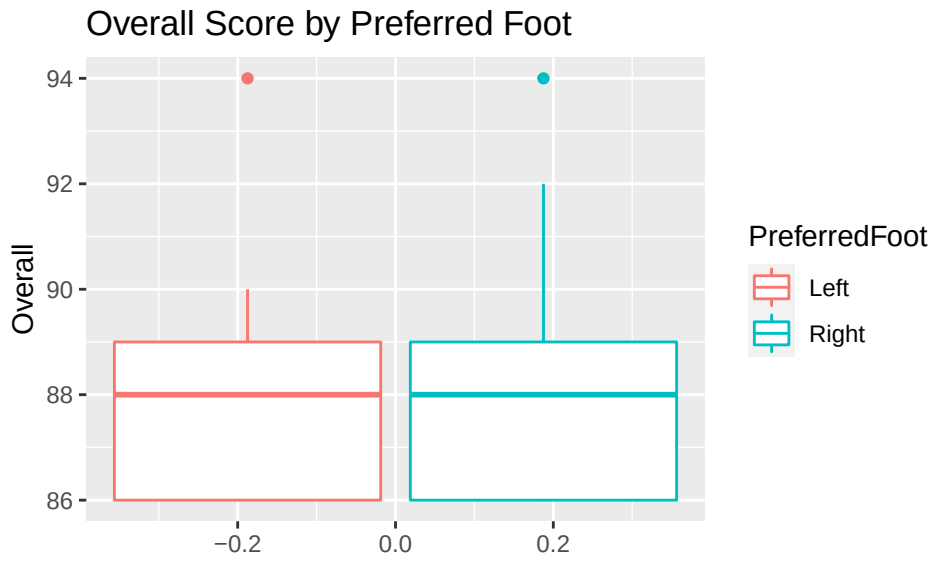
```
ggplot(data = Fifasmall, aes(x = Overall)) +
  geom_density(aes(color = PreferredFoot)) +
  labs(title = "Overall Score by Preferred Foot")
```



Sometimes boxplots can be better for comparing quantitative variables across groups.

```
ggplot(data = Fifasmall, aes(x = Overall, color = PreferredFoot)) +
  geom_boxplot() +
  labs(title = "Overall Score by Preferred Foot") +
  scale_fill_brewer(palette = "Spectral") +
```

```
  coord_flip()
```



The minimum and Q1 are the same, in case you are wondering why there is no lower whisker.

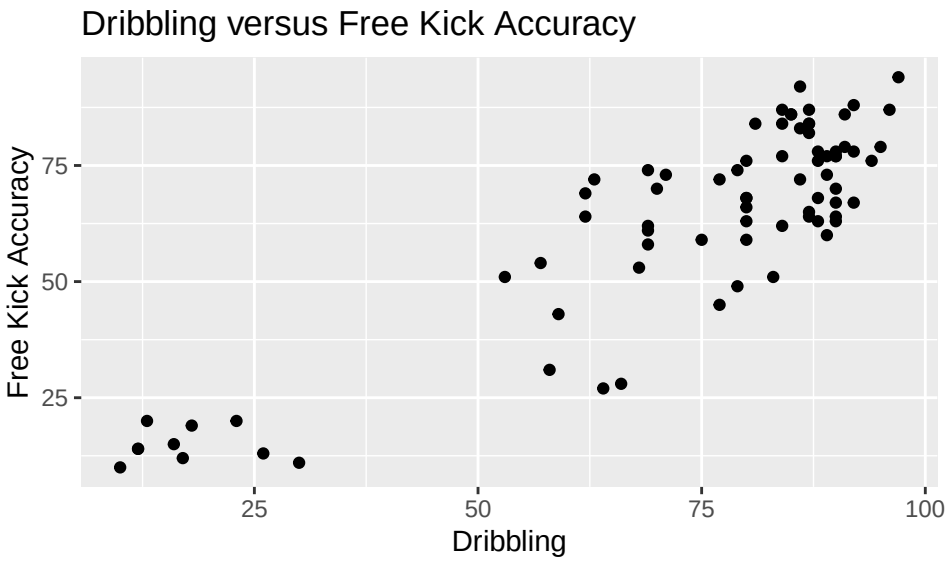
```
mosaic::favstats(Overall ~ PreferredFoot, data = Fifasmall)
```

```
##   PreferredFoot min Q1 median Q3 max     mean       sd  n missing
## 1          Left  86 86     88 89  94 87.80952 1.887301 21       0
## 2         Right  86 86     88 89  94 88.03571 1.916097 56       0
```

In fact, the five number summaries are the same for both PreferredFoot groups.

More commonly, we'll start with some form of a scatterplot to examine the relationship between two quantitative variables.

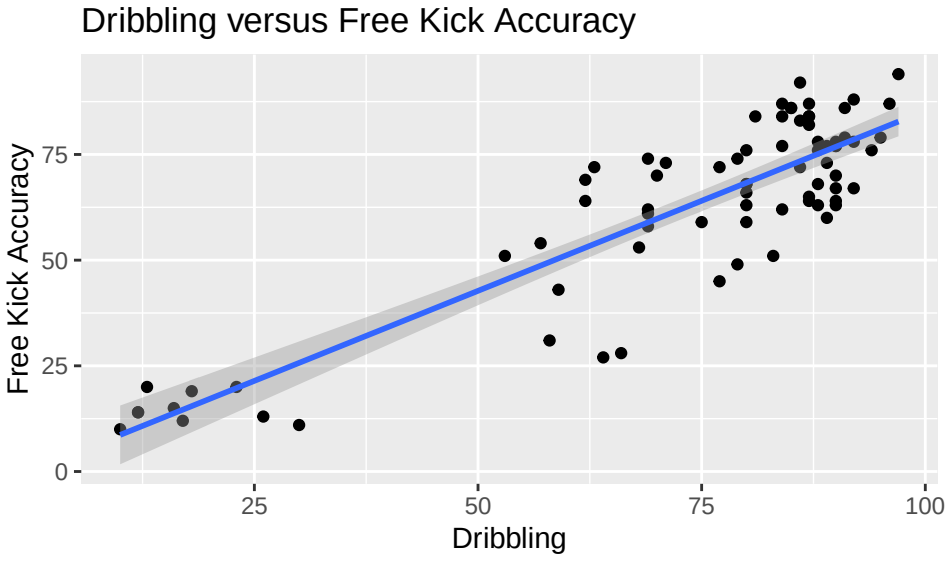
```
ggplot(data = Fifasmall, aes(x = Dribbling, y = FKAccuracy)) +
  geom_point() +
  labs(title = "Dribbling versus Free Kick Accuracy",
       y = "Free Kick Accuracy")
```



You can add smoothed lines or fitted regression lines with or without error margins.

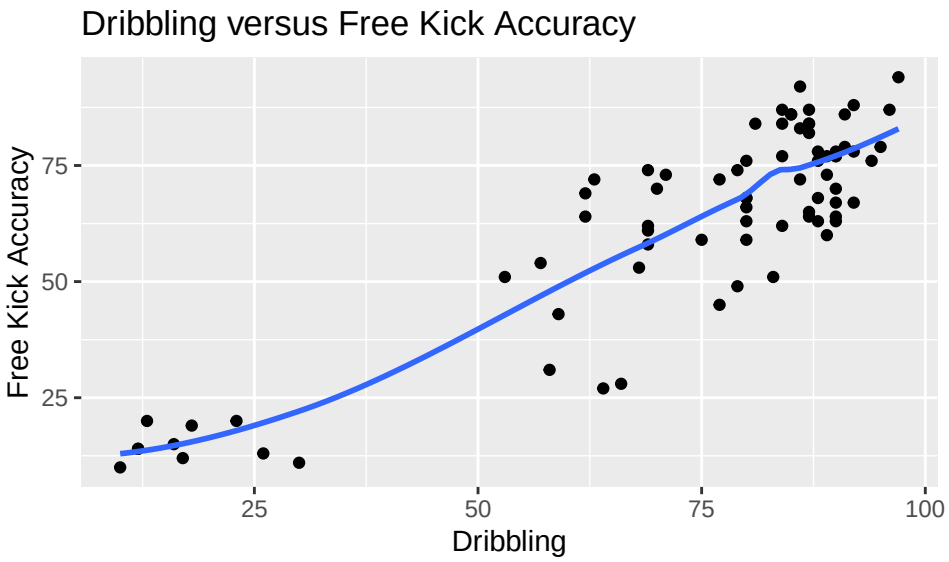
```
ggplot(data = Fifasmall, aes(x = Dribbling, y = FKAccuracy)) +
  geom_point() +
  geom_smooth(method = "lm", se = TRUE) +
  labs(title = "Dribbling versus Free Kick Accuracy",
       y = "Free Kick Accuracy")
```

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




```
ggplot(data = Fifasmall, aes(x = Dribbling, y = FKAccuracy)) +
  geom_point() +
  geom_smooth(se = FALSE) +
  labs(title = "Dribbling versus Free Kick Accuracy",
       y = "Free Kick Accuracy")
```

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

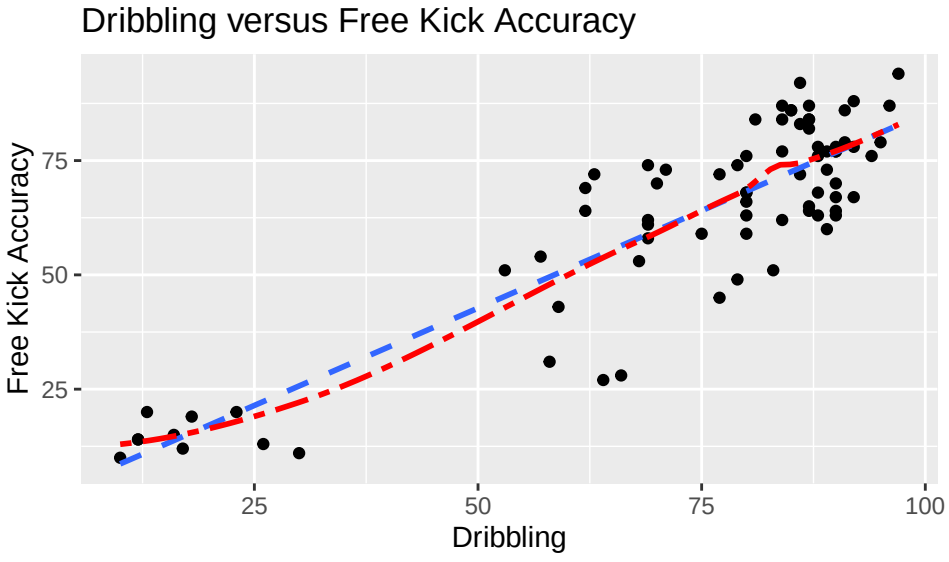


For that matter, you could combine these plots to compare the lines. You can adjust the linetypes and colors as desired.

```
ggplot(data = Fifasmall, aes(x = Dribbling, y = FKAccuracy)) +
  geom_point() +
  geom_smooth(method = "lm", se = FALSE, linetype = 2) +
  geom_smooth(se = FALSE, linetype = 6, color = "red") +
  labs(title = "Dribbling versus Free Kick Accuracy",
       y = "Free Kick Accuracy")
```

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

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

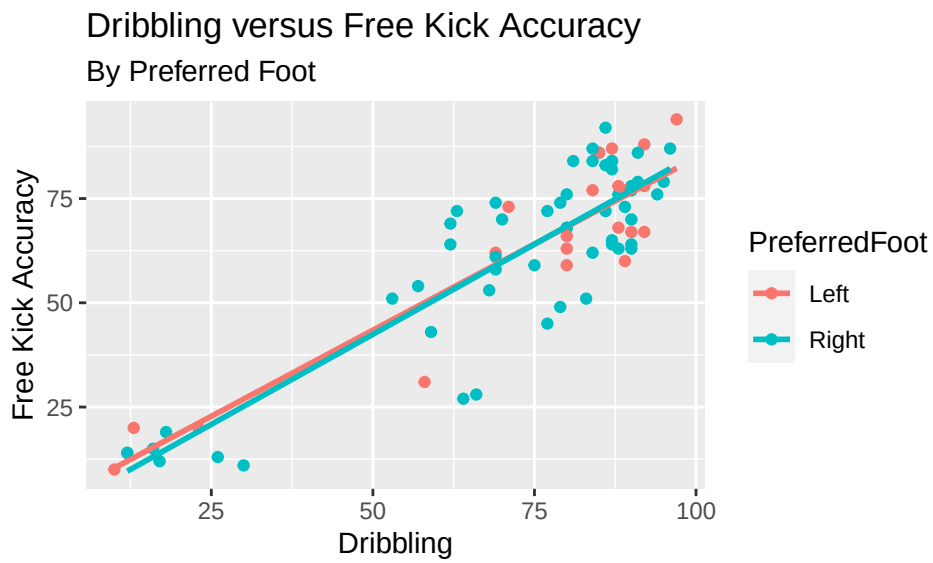


We can add another variable via color fairly easily.

```
ggplot(data = Fifasmall, aes(x = Dribbling, y = FKAccuracy, color = PreferredFoot)) +
  geom_point() +
```

```
  geom_smooth(method = "lm", se = FALSE) +
  labs(title = "Dribbling versus Free Kick Accuracy",
       subtitle = "By Preferred Foot",
       y = "Free Kick Accuracy")
```

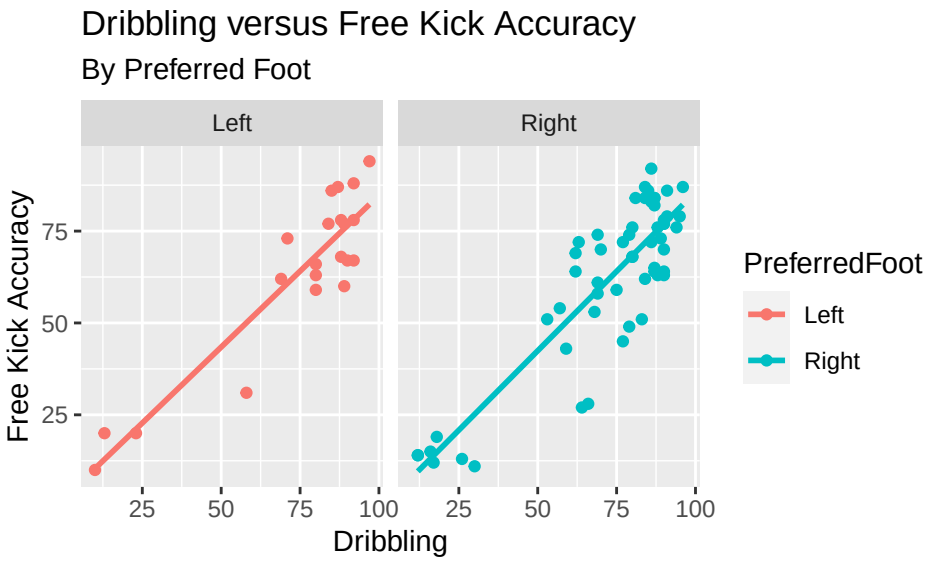
```
## `geom_smooth()` using formula 'y ~ x'
```



If you'd rather not have the plots overlaid, but would like them side by side, use facets.

```
ggplot(data = Fifasmall, aes(x = Dribbling, y = FKAccuracy, color = PreferredFoot)) +
  geom_point() +
  geom_smooth(method = "lm", se = FALSE) +
  facet_wrap(~ PreferredFoot) +
  labs(title = "Dribbling versus Free Kick Accuracy",
       subtitle = "By Preferred Foot",
       y = "Free Kick Accuracy")
```

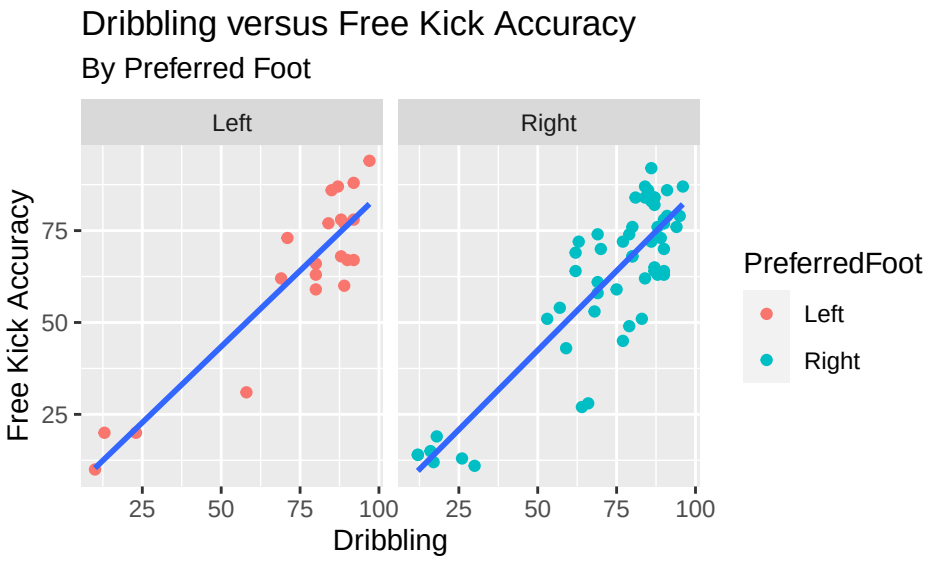
```
## `geom_smooth()` using formula 'y ~ x'
```



We have two geoms here. What happens if we set the color aesthetic within the call to geom_point here, instead of as part of the overall aesthetic?

```
ggplot(data = Fifasmall, aes(x = Dribbling, y = FKAccuracy)) +
  geom_point(aes(color = PreferredFoot)) +
  geom_smooth(method = "lm", se = FALSE) +
  facet_wrap(~ PreferredFoot) +
  labs(title = "Dribbling versus Free Kick Accuracy",
       subtitle = "By Preferred Foot",
       y = "Free Kick Accuracy")
```

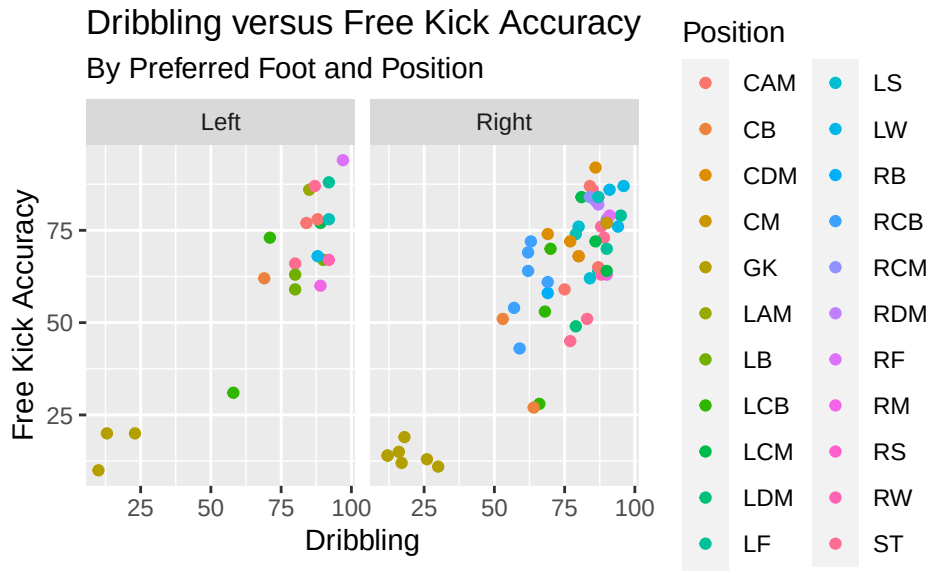
```
## `geom_smooth()` using formula 'y ~ x'
```



Which of these do you prefer?

Of course, nothing says the facet and color variables need to be the same. (I'm removing the regression lines for this, and setting color back in the overall aesthetic.)

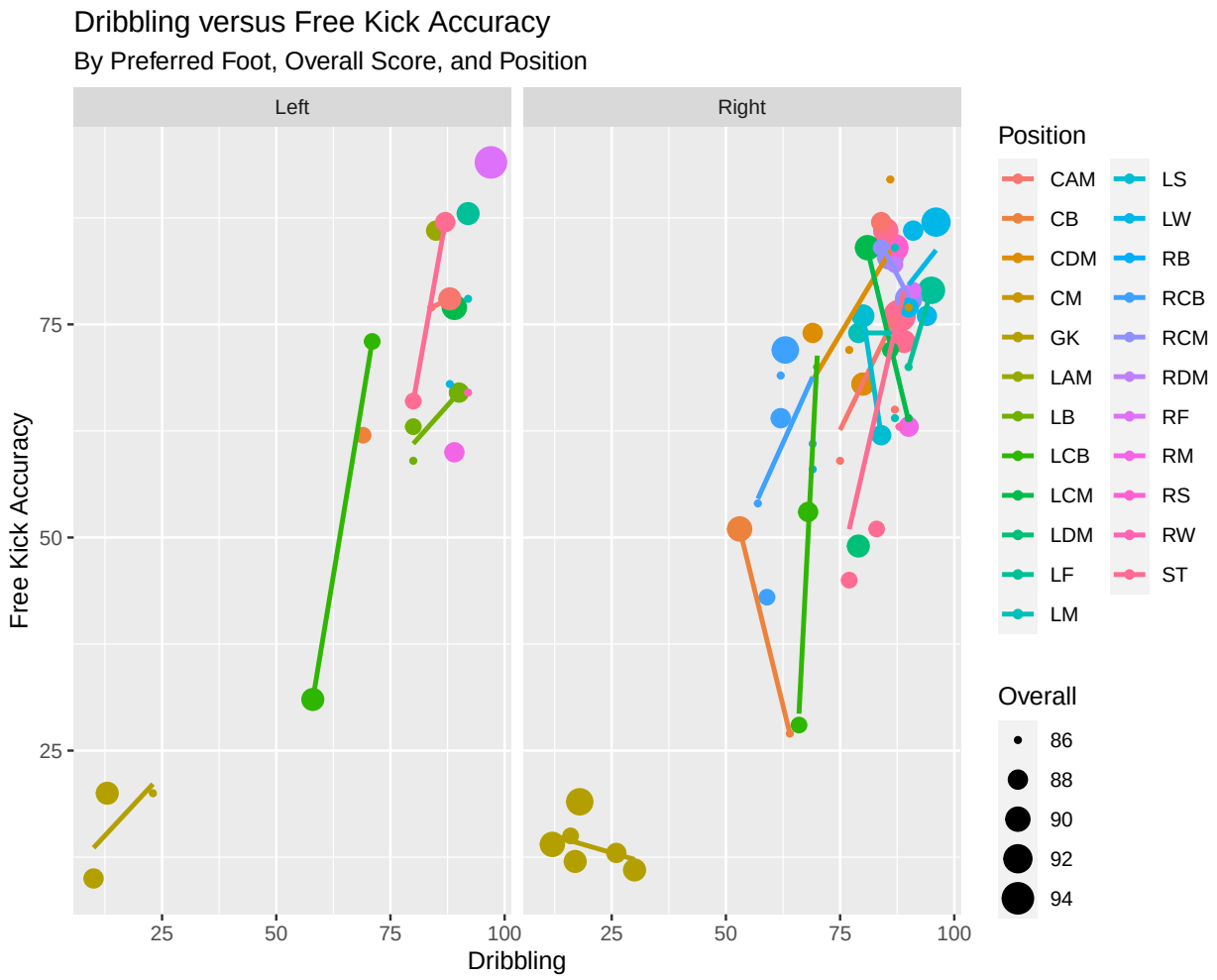
```
ggplot(data = Fifasmall, aes(x = Dribbling, y = FKAccuracy, color = Position)) +
  geom_point() +
  facet_wrap(~ PreferredFoot) +
  labs(title = "Dribbling versus Free Kick Accuracy",
       subtitle = "By Preferred Foot and Position",
       y = "Free Kick Accuracy")
```



And we could play with point size still, and maybe we want those regression lines back.

```
ggplot(data = Fifasmall, aes(x = Dribbling, y = FKAccuracy, color = Position)) +
  geom_point(aes(size = Overall)) +
  geom_smooth(method = "lm", se = FALSE)+
  facet_wrap(~ PreferredFoot) +
  labs(title = "Dribbling versus Free Kick Accuracy",
       subtitle = "By Preferred Foot, Overall Score, and Position",
       y = "Free Kick Accuracy")
```

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



This is not a very useful plot (Personally, I can't distinguish all those colors and some of those points are tiny), but you get the idea. You can layer on a lot here, and setting aesthetics will allow you to incorporate a lot of information into your plots. The plot also has to be re-sized so the legend stays on the page in the .pdf.

Your text has more examples that demonstrate (among other things):

- mosaic plots (two categorical variables)
- adding arrows and text features to plots (geom_curve, geom_text)
- geom_line for lines
- geom_linerange
- tribbles - small data frames built for quick use (often to add context)
- a lot of the data wrangling commands in the next few chapters (filter, group_by, summarize, mutate, %in%, unnest, pivot_wider)

Don't worry if the data wrangling commands are unclear. That's what we'll be tackling in the coming weeks.

We examined scale changes and color a little bit, but there's a lot more you can do there as well. You can also do more with editing legends, including their positions, or simply removing them.