15.3.2 Build a Bar Plot in ggplot2

For his first plots, Jeremy has decided to visualize some information about fuel economy. His practice dataset will be from the U.S. Environmental Protection Agency (EPA) and dated 1999 through 2008. He knows Colleen already has this data on lock, so he'll be able to check his work with her—and it will help them both brainstorm for their team's first big presentation to the CEO.

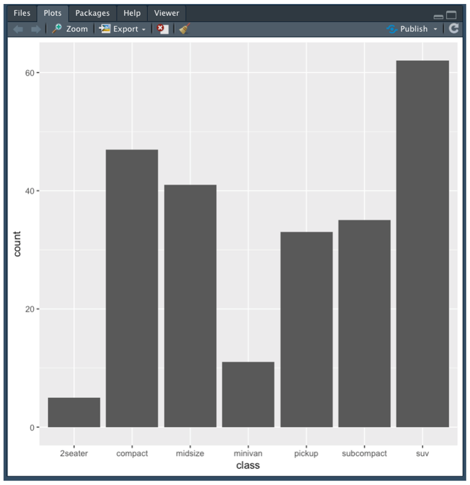
Now that we are familiar with setting up the `ggplot()`function, let's build our first plot using the mpg (miles per gallon) dataset. First, we'll take a moment to familiarize ourselves with the mpg dataset. In the R console, type the following statement:

```
> head(mpg)
```

The mpg dataset contains fuel economy data from the EPA for vehicles manufactured between 1999 and 2008. The mpg dataset is built into R and is used throughout R documentation due to its availability, diversity of variables, and overall cleanliness of data. For our purposes, we'll use the mpg data to demonstrate how to implement each of our ggplot visualizations.

The first plots we'll generate using ggplot2 will be bar plots. Bar plots are used to visualize categorical data. They can be used to represent the frequency of each categorical value in a list of categorical data. For example, if we want to create a bar plot that represents the distribution of vehicle classes from the mpg dataset, we would use the following statements in R:

```
> plt <- ggplot(mpg,aes(x=class)) #import dataset into ggplot2
> plt + geom_bar() #plot a bar plot
```



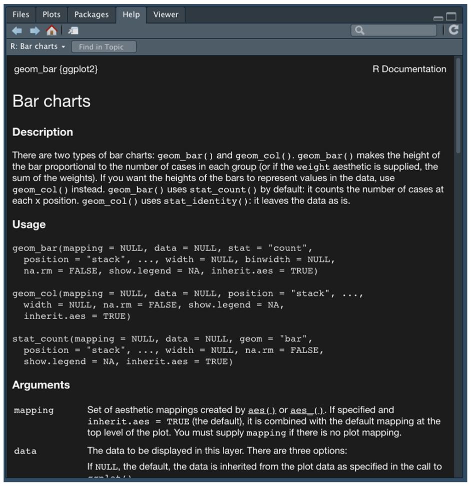
NOTE

> When you generate a plot in RStudio, the multi-tool pane will switch over to the Plot pane.

In this example, we're only trying to visualize univariate (single variable) data. Therefore, we only need to assign our `x` argument within the `aes()` function. After creating our `ggplot` object, we then generate a bar plot using `geom_bar()`.

Type the following code into the R console to look at the `geom_bar()` documentation in the Help pane:

```
> ?geom_bar()
```



geom_bar {ggplot2} R Documentation

Bar charts

Description

There are two types of bar charts: `geom_bar()` and `geom_col()`. `geom_bar()` makes the height of the bar proportional to the number of cases in each group (or if the `weight` aesthetic is supplied, the sum of the weights). If you want the heights of the bars to represent values in the data, use `geom_col()` instead. `geom_bar()` uses `stat_count()` by default: it counts the number of cases at each x position. `geom_col()` uses `stat_identity()`: it leaves the data as is.

Usage

```
geom_bar(mapping = NULL, data = NULL, stat = "count",
  position = "stack", ..., width = NULL, binwidth = NULL,
  na.rm = FALSE, show.legend = NA, inherit.aes = TRUE)

geom_col(mapping = NULL, data = NULL, position = "stack", ...,
  width = NULL, na.rm = FALSE, show.legend = NA,
  inherit.aes = TRUE)

stat_count(mapping = NULL, data = NULL, geom = "bar",
  position = "stack", ..., width = NULL, na.rm = FALSE,
  show.legend = NA, inherit.aes = TRUE)
```

Arguments

`mapping` Set of aesthetic mappings created by `aes()` or `aes_()`. If specified and `inherit.aes = TRUE` (the default), it is combined with the default mapping at the top level of the plot. You must supply `mapping` if there is no plot mapping.

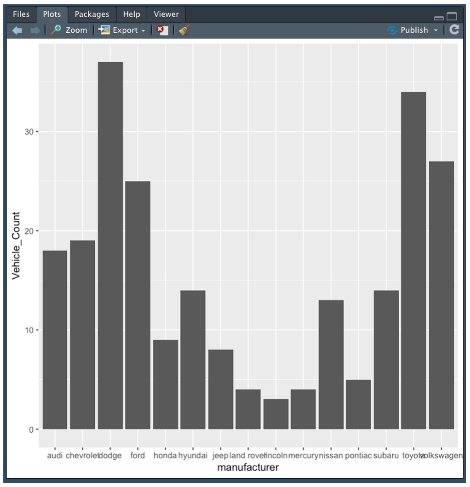
`data` The data to be displayed in this layer. There are three options:

If `NULL`, the default, the data is inherited from the plot data as specified in the call to

Unlike most of our previous R functions that we have explored, the `geom` functions from ggplot2 are very large. However, in most cases, we can leave all of the arguments alone and use the `geom ()` function by itself.

Another use for bar plots is to compare and contrast categorical results. For example, if we want to compare the number of vehicles from each manufacturer in the dataset, we can use dplyr's `summarize()` function to summarize the data, and ggplot2's `geom_col()` to visualize the results:

```
> mpg_summary <- mpg %>% group_by(manufacturer) %>% summarize(Vehicle_Count=
> plt <- ggplot(mpg_summary,aes(x=manufacturer,y=Vehicle_Count)) #import dat
> plt + geom_col() #plot a bar plot
```



As we practiced previously, creating a summary table for the manufacturer vehicles was done using dplyr's `group_by()` and `summarize()` functions. Our new summary table was then used as the input data for our `ggplot()` function.

In our first example, we only needed to assign one variable to our list of classes. In contrast, our second example required two variables—one for our categorical factors (assigned to x), and another for our calculated results (assigned to y). Once we generated our `ggplot` object, we then used an alternative method for creating a bar plot, `geom_col()`.

Functionally, both `geom_bar()` and `geom_col()`create bar plots; however, the two methods assume different inputs. `geom_bar()` expects one variable and generates frequency data, and `geom_col()`expects two variables where we provide the size of each category's bar.

NOTE

> Many of ggplot2 visualizations have alternative methods that accommodate different use cases. Feel free to look at the **ggplot2 documentation** (https://ggplot2.tidyverse.org/reference/index.html) if you have a specific use case in mind.

In its current state, our bar plot could be sufficient for personal use when drawing quick conclusions about the data. For instance, we can see from our bar plot that Dodge had the highest number of vehicles in the dataset and Lincoln had the fewest. However, our current bar plot would not be appropriate to use for an analytical report or for publishing. The two biggest issues with the current plot are:

- Our axis titles are not consistent and could be better formatted.
- Our x-axis labels are overlapping and run off the page.

We'll fix this by adding formatting functions.