Stat 133 Post: ggplot2 Comprehensive Overview

Ronak Modi

10/31/2017

Introduction / Motivation:

Data Visualization is one of the most important topics in the universe of Data Computing. In R, an important standard library to visualize your data is ggplot2. I'm not sure about you, but as a student in this class, ggplot2 was pretty confusing, and for me personally, it became evident I wasn't as good at ggplot2 as I thought I was when I encountered that midterm question. The different arguments and keywords aren't intuitive for me, and I thought to myself that it would be great if someone could show me how to use ggplot2, but through the lens of a fellow student. Well, here I am! Let's get this started!

Pre-Requisites:

First off, you have to install ggplot2. What many may not know is that ggplot2 is only one of the many tools available in a package called tidyverse, which is also standard to R. In this post, I will only focus on ggplot2, but tidyverse has a bunch of other cool features that you should definitely check out if you are interested.

Creating a basic ggPlot:

One cool thing about ggplot2 is that it already has some data frames inside of it to play with. One of them is called mpg. This has information about cars miles per gallon. Check it out below!

```
mpg
```

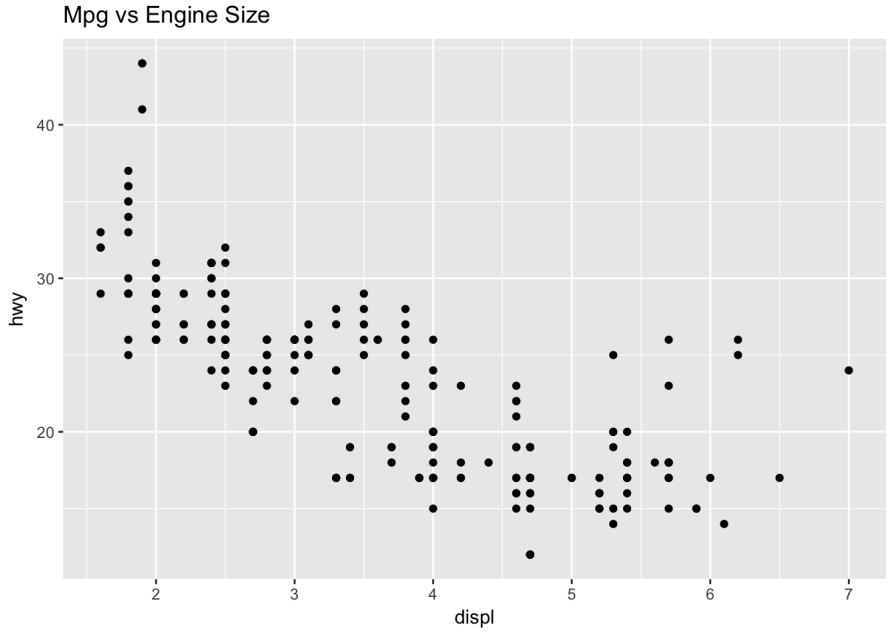
```
## # A tibble: 234 x 11
##    manufacturer      model displ  year   cyl      trans   drv   cty   hwy
##           <chr>      <chr> <dbl> <int> <int>      <chr> <chr> <int> <int>
##  1         audi         a4   1.8  1999     4   auto(l5)     f    18    29
##  2         audi         a4   1.8  1999     4 manual(m5)     f    21    29
##  3         audi         a4   2.0  2008     4 manual(m6)     f    20    31
##  4         audi         a4   2.0  2008     4   auto(av)     f    21    30
##  5         audi         a4   2.8  1999     6   auto(l5)     f    16    26
##  6         audi         a4   2.8  1999     6 manual(m5)     f    18    26
##  7         audi         a4   3.1  2008     6   auto(av)     f    18    27
##  8         audi a4 quattro   1.8  1999     4 manual(m5)     4    18    26
##  9         audi a4 quattro   1.8  1999     4   auto(l5)     4    16    25
## 10         audi a4 quattro   2.0  2008     4 manual(m6)     4    20    28
## # ... with 224 more rows, and 2 more variables: fl <chr>, class <chr>
```

Let's create a basic ggplot using the mpg data frame!

The below command will create a basic dotplot with engine size (displ) as the x and mpg (hwy) as the y. Let me run through the code as well to make it simple for you. ggplot creates a coordinate system which I can add "layers" to. The first argument of ggplot is the data, which I set to our data frame mpg. I add a layer to the plot using geom_point, which takes in a mapping argument letting me set the x and y axes to what I want them to be.

ggtitle lets me make a title for my graph!

```
ggplot(data = mpg) + geom_point(mapping = aes(x=displ , y=hwy)) + ggtitle("Mpg vs Engine Size")
```

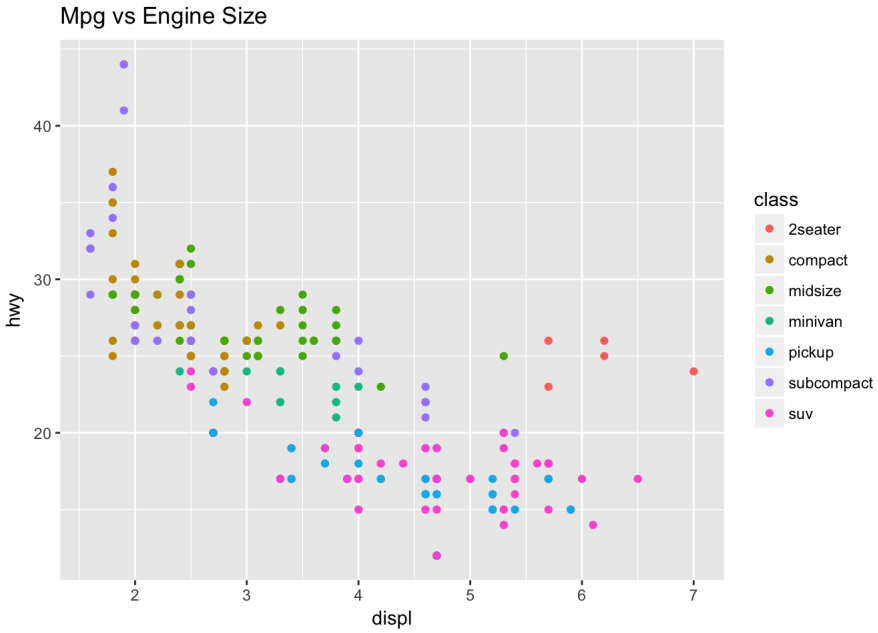


Aesthetic Mappings:

The last plot we made was pretty basic, so let's jump into some stuff that are a little more fun. In our dataset, we have a variable class, which classifies cars into groups (SUVs, Hybrids, Minivans, etc). We can take into account the class variable by mapping it to something special called an aesthetic. Aesthetics are intuitive, because they have to do with what you would expect: size, shape, and color of your plot.

Below, I'm going to change the color of the points based on which class of car the point is.

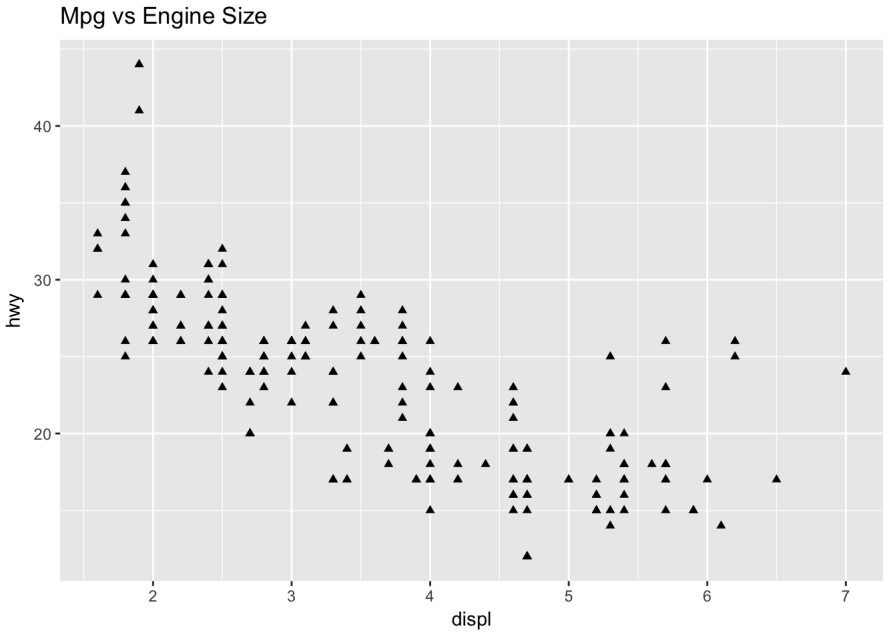
```
#you can notice that this is the exact same as the below plot, all I am changing is the color to be different base
d on the class of car, using the keyword 'color ='
ggplot(data = mpg) + geom_point(mapping = aes(x=displ , y=hwy , color=class)) + ggtitle("Mpg vs Engine Size")
```



In addition to color, you can change the size, shape, and even transparency of your points using a keyword different than color=. For size, use 'size=' , for shape use 'shape=' and for transparency use 'alpha='. However, be careful because ggplot will only use six different shapes, so if you have more variables than that as your 'shape=' value, it will probably not come out as you expect.

Also, you don't have to make a variable aesthetic for all your points. You can use any of the keywords I showed above outside the aes() as I'll show you below with the 'shape=' keyword.

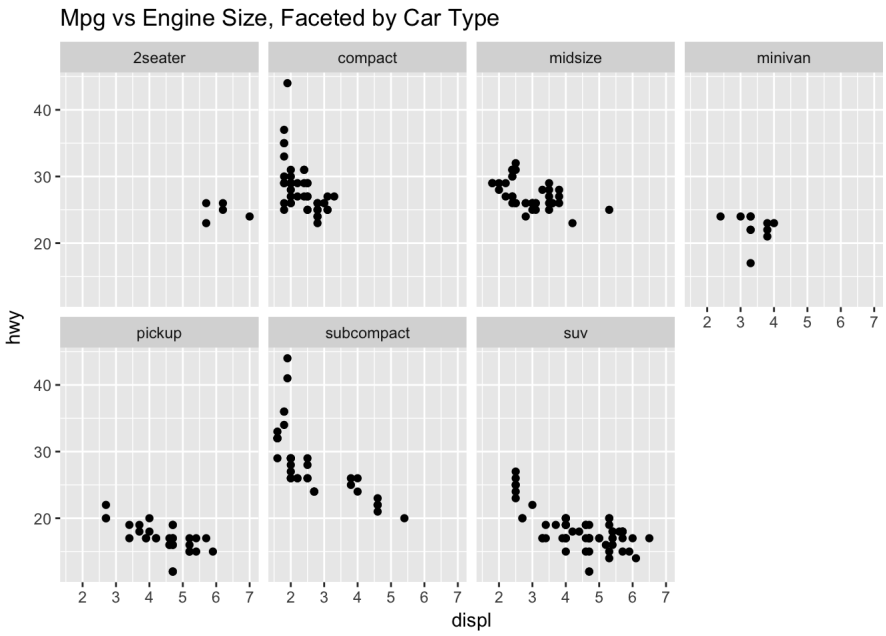
```
#shape takes a integer corresponding to a certain shape. the mapping of shape integers is provided in the ggplot2
documentation.
#17 is a triangle
ggplot(data = mpg) + geom_point(mapping = aes(x=displ , y=hwy), shape = 17) + ggtitle("Mpg vs Engine Size")
```



Facets:

The first question a lot of people (including me) wondered about Facets is what exactly are they? Facets are a way to split your data into multiple subplots based on a variable. You do this by adding a + facet_wrap() call to the end of your plot. You pass in a 'formula' object and the number of rows you would like to facet your plot into. This does seem confusing in writing, however let me show you an example!

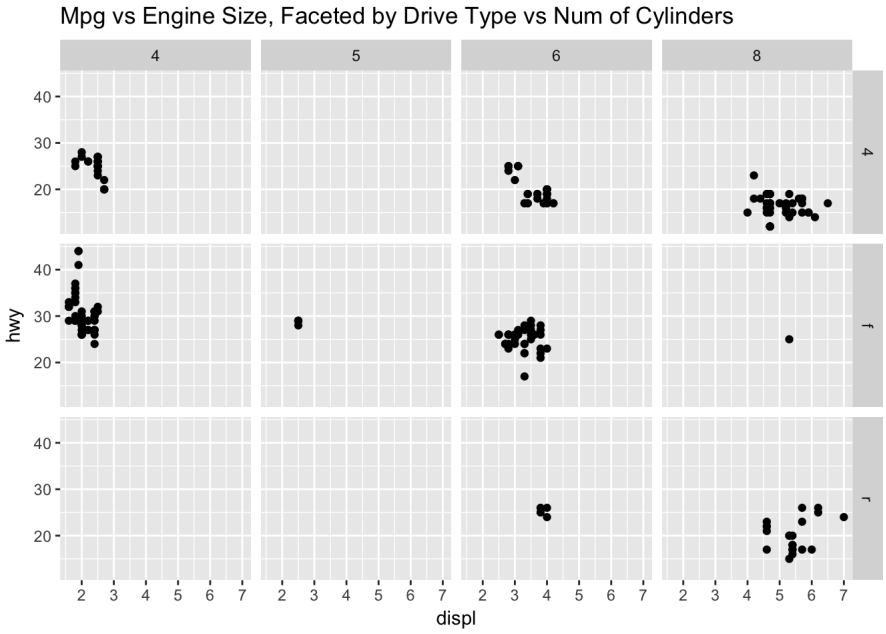
```
#in this case we are faceting by class, so each miniplot is of only a certain class of car. Also, nrow isn't doing
anything too complex, it is just for display purposes to make the graph look neater.
ggplot(data = mpg) +
  geom_point(mapping = aes(x = displ, y = hwy)) +
  facet_wrap(~ class, nrow = 2) +
  ggtitle("Mpg vs Engine Size, Faceted by Car Type")
```



It is cool to facet your plots by one variable, but what if you want to see a facet of two variables? First of all let's talk about what it means to facet a plot by two variables, because if you can visualize it in your head, it is a lot more intuitive to think about its uses and use it for your own data visualization projects whenever applicable. When we faceted our data by one variable, we split our plot into subplots based off of 1 variable, so when our facet variable took a different value, we had a different subplot. Now in a two variable case, everytime we have a different combination of (x , y) for our two variables, we will have a different subplot.

The way to use two variables is to add the command '+ facet_grid().' I will show a simple example below, faceting on the variables 'drv' and 'cyl', which are the drive type (4 wheel , rear , or front) and number of cylinders.

```
ggplot(data = mpg) +
  geom_point(mapping = aes(x = displ, y = hwy)) +
  facet_grid(drv ~ cyl) +
  ggtitle("Mpg vs Engine Size, Faceted by Drive Type vs Num of Cylinders")
```



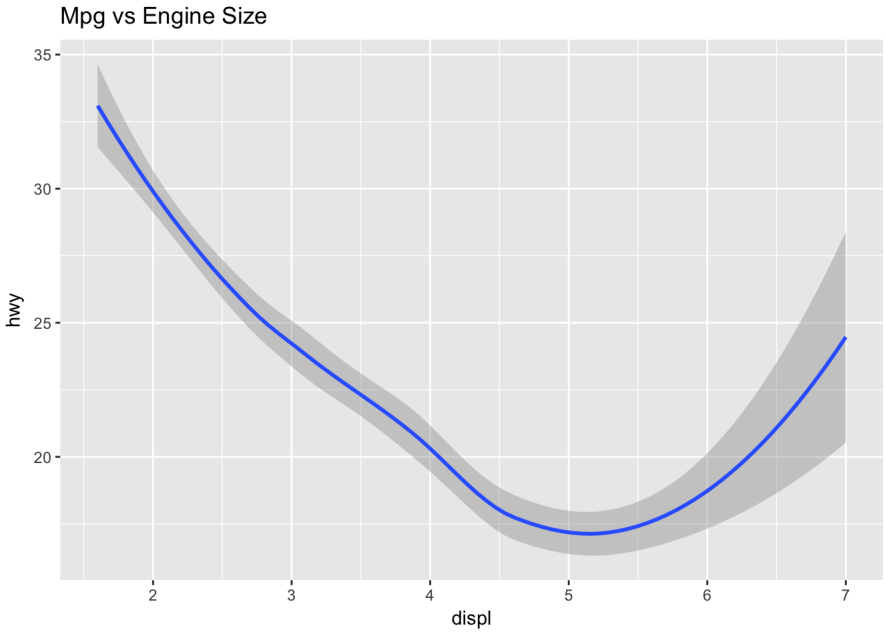
Geometric Objects:

This whole time, we have been using points to represent our data. Points are great, and oftentimes very informative, but if all we had in our data visualization toolbox were points, we would miss out on many important data science concepts, like regression for example. Luckily, ggplot2 has this covered as well! You can choose to represent your data with different geometric objects. 'geom_point' is just one of the ways you can represent your data. You can choose to use 'geom_smooth' (which creates a smooth line through your points), 'geom_boxplot' (boxplot) , and many other geometric objects that you can easily find in the ggplot2 documentation!

Below, I'll show an example with geom_smooth:

```
ggplot(data = mpg) +
  geom_smooth(mapping = aes(x = displ, y = hwy)) +
  ggtitle("Mpg vs Engine Size")
```

```
## `geom_smooth()` using method = 'loess'
```

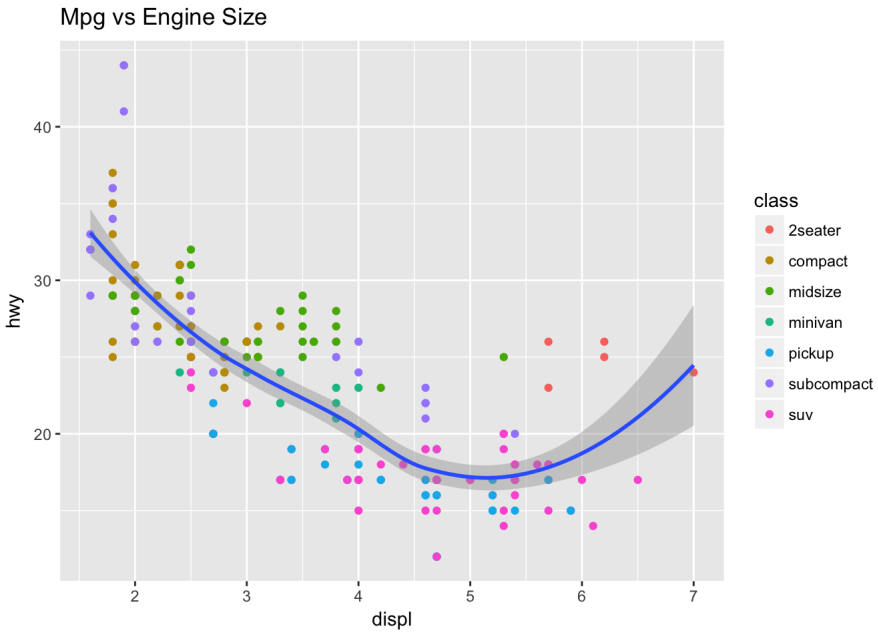


Another cool application of the geometric objects in ggplot2 is the ability to overlay multiple objects in something called layers. Basically, every '+' that you use adds a layer on to ggplot2. In the above examples, we have only been using facet layers and a title layer. However, now that we can use different geometric objects, we can use multiple 'geom' layers. There is an interesting twist to this though, if we put the usual 'aes' tag in the ggplot() call, then all layers will have that particular aesthetic applied to it.

I will illustrate this below:

```
#I have done the x and y aesthetic in the ggplot, and it applies to both the point and smooth layer.
ggplot(data = mpg, mapping = aes(x = displ, y = hwy)) +
  geom_point(mapping = aes(color = class)) +
  geom_smooth() +
  ggtitle("Mpg vs Engine Size")
```

```
## `geom_smooth()` using method = 'loess'
```



Overall Takeaway:

When I first started working with ggplot2 in this class, I was very discouraged by the help I received from the documentation provided by the instructor. It seemed like whoever created these documentations didn't take into account that what they were creating was meant to be used by people who had very little experience with data visualization. That is why the goal of my post was to create a really simple sort of documentation that anyone could understand. My post does not even begin to encompass the variety of things you can do with ggplot2, but my hope is that if you understand the fundamentals that I have put forth here, you can become great at ggplot2, just as I have!

Thanks for reading!

-Ronak

References List:

http://r4ds.had.co.nz/data-visualisation.html

http://ggplot2.tidyverse.org/reference/

http://www.sthda.com/english/wiki/ggplot2-title-main-axis-and-legend-titles

http://vita.had.co.nz/papers/layered-grammar.pdf

https://www.youtube.com/watch?v=WxSUsTDcMTg

http://www.dummies.com/education/math/statistics/plotting-t-ggplot2/

https://en.wikipedia.org/wiki/Ggplot2