post02-andrea-tu: Making ggplot2 interactive with ggiraph

Andrea Tu

December 3, 2017

In my last post, I talked about how to use the package `ggplot2`. `ggplot2` is a really useful data visualization packages that allows people to create beautiful and practical visualizations of data. While it has many of the same capabilities as does baseR, such as creating histograms, bar graphs, and scatterplots, it is still has some extra cool features that allows for easy reading and interpreting.

In this post, I will be introducing the package `ggiraph`. This package allows users to make ggplot2 visualizations interactive. I thought this was a nice new package that would bring together some of the things we've been learning, especially as we went into how to make ggplot2 and also how to use Shiny, which is an interactive web app. While Shiny has many capabilities, learning how to use it can be a little challenging, as it is a bit less intuitive than other packages and things we've used. Now with `ggiraph`, users can see interact on the ggplot2 graph, where it is much simpler and can also provide some interaction for the user. Let's get right into it!

Installing and loading ggplot2 and ggiraph packages

First, let's start off with loading the `ggplot2` and `ggiraph` packages. (We'll also need `dplyr`.)If you haven't used these packages before, you will probably need to install it, as I will do below:

```
install.packages(c('ggplot2', 'ggiraph'))
```

```
library(ggplot2)
library(ggiraph)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

Great! Now that we have `ggiraph` installed and loaded, let's move onto actually using the package with some data.

Downloading the data

Before we get started working with data, we need to get the data. In this post, I will be using a dataset I found on Kaggle that has Nutrition Facts for McDonalds Menu. You can go to the link yourself and download the dataset – don't worry, it's not that big of a file. We'll need to then load this data into a table so we can use it in our code. I'm also only going to choose a few categories of food to look at.

```
mcdonalds <- read.csv('../data/menu.csv', stringsAsFactors = FALSE)
mcdonalds <- filter(mcdonalds, Category %in% c('Breakfast', 'Chicken & Fish', 'Beef & Pork'))

# checking out the data
names(mcdonalds)
```

```
##  [1] "Category"                      "Item"
##  [3] "Serving.Size"                  "Calories"
##  [5] "Calories.from.Fat"             "Total.Fat"
##  [7] "Total.Fat....Daily.Value."     "Saturated.Fat"
##  [9] "Saturated.Fat....Daily.Value." "Trans.Fat"
## [11] "Cholesterol"                   "Cholesterol....Daily.Value."
## [13] "Sodium"                        "Sodium....Daily.Value."
## [15] "Carbohydrates"                 "Carbohydrates....Daily.Value."
## [17] "Dietary.Fiber"                 "Dietary.Fiber....Daily.Value."
## [19] "Sugars"                        "Protein"
## [21] "Vitamin.A....Daily.Value."     "Vitamin.C....Daily.Value."
## [23] "Calcium....Daily.Value."       "Iron....Daily.Value."
```

```
head(mcdonalds, 10)
```

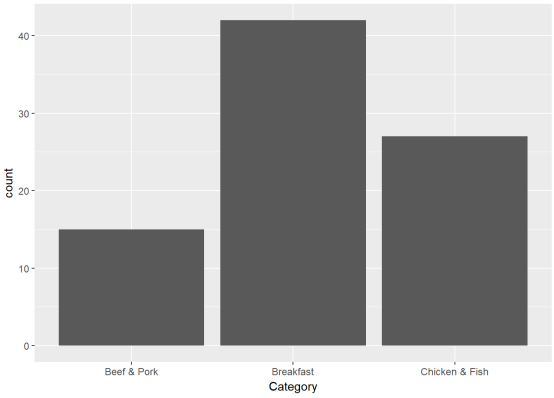
```
##     Category                                                 Item
## 1  Breakfast                                       Egg McMuffin
## 2  Breakfast                                  Egg White Delight
## 3  Breakfast                                    Sausage McMuffin
## 4  Breakfast                           Sausage McMuffin with Egg
## 5  Breakfast                     Sausage McMuffin with Egg Whites
## 6  Breakfast                                  Steak & Egg McMuffin
## 7  Breakfast          Bacon, Egg & Cheese Biscuit (Regular Biscuit)
## 8  Breakfast            Bacon, Egg & Cheese Biscuit (Large Biscuit)
## 9  Breakfast Bacon, Egg & Cheese Biscuit with Egg Whites (Regular Biscuit)
## 10 Breakfast   Bacon, Egg & Cheese Biscuit with Egg Whites (Large Biscuit)
##      Serving.Size Calories Calories.from.Fat Total.Fat
## 1  4.8 oz (136 g)      300              120        13
## 2  4.8 oz (135 g)      250               70         8
## 3  3.9 oz (111 g)      370              200        23
## 4  5.7 oz (161 g)      450              250        28
## 5  5.7 oz (161 g)      400              210        23
## 6  6.5 oz (185 g)      430              210        23
## 7  5.3 oz (150 g)      460              230        26
## 8  5.8 oz (164 g)      520              270        30
## 9  5.4 oz (153 g)      410              180        20
## 10 5.9 oz (167 g)      470              220        25
##    Total.Fat....Daily.Value. Saturated.Fat Saturated.Fat....Daily.Value.
## 1                         20             5                            25
## 2                         12             3                            15
## 3                         35             8                            42
## 4                         43            10                            52
## 5                         35             8                            42
## 6                         36             9                            46
## 7                         40            13                            65
## 8                         47            14                            68
## 9                         32            11                            56
## 10                        38            12                            59
##    Trans.Fat Cholesterol Cholesterol....Daily.Value. Sodium
## 1          0         260                         87    750
## 2          0          25                          8    770
## 3          0          45                         15    780
## 4          0         285                         95    860
## 5          0          50                         16    880
## 6          1         300                        100    960
## 7          0         250                         83   1300
## 8          0         250                         83   1410
## 9          0          35                         11   1300
## 10         0          35                         11   1420
##    Sodium....Daily.Value. Carbohydrates Carbohydrates....Daily.Value.
## 1                      31            31                            10
## 2                      32            30                            10
## 3                      33            29                            10
## 4                      36            30                            10
## 5                      37            30                            10
## 6                      40            31                            10
## 7                      54            38                            13
## 8                      59            43                            14
## 9                      54            36                            12
## 10                     59            42                            14
##    Dietary.Fiber Dietary.Fiber....Daily.Value. Sugars Protein
## 1              4                           17      3      17
## 2              4                           17      3      18
## 3              4                           17      2      14
## 4              4                           17      2      21
## 5              4                           17      2      21
## 6              4                           18      3      26
## 7              2                            7      3      19
## 8              3                           12      4      19
## 9              2                            7      3      20
## 10             3                           12      4      20
##    Vitamin.A....Daily.Value. Vitamin.C....Daily.Value.
## 1                         10                         0
## 2                          6                         0
## 3                          8                         0
## 4                         15                         0
## 5                          6                         0
## 6                         15                         2
## 7                         10                         8
## 8                         15                         8
## 9                          2                         8
## 10                         6                         8
##    Calcium....Daily.Value. Iron....Daily.Value.
## 1                       25                   15
## 2                       25                    8
## 3                       25                   10
## 4                       30                   15
## 5                       25                   10
## 6                       30                   20
## 7                       15                   15
## 8                       20                   20
## 9                       15                   10
## 10                      15                   15
```

The data containing nutrition facts for the McDonalds menu is now accessible through our mcdonalds table! Now, let's really get into the data visualization part, using `ggplot2` and `ggiraph`.

Using ggiraph with ggplot2

In using `ggplot2`, we saw that in addition to simply making ggplot, we needed to add another geom function to specify what kind of plot or visualization we wanted to display. Some examples of these are geom_bar, geom_histogram, geom_point, and geom_boxplot. Here is an example of the typical bar chart that can be made using `ggplot2`:

```
ggplot(data = mcdonalds) + geom_bar(aes(x = Category)) + theme(axis.text.x = element_text(size = 9))
```



The bar chart above shows the counts of number of items in each category on the McDonalds menu (at least, the ones on this csv). You can see that we used geom_bar to make this bar chart. In `ggiraph`, many of the functions are actually very similar. The thing that is different is that each function has an additional "_interactive" after the function name. For instance, some examples of the new `geom` functions are:

- geom_bar_interactive
- geom_point_interactive
- geom_text_interactive

… and more! Basically, for any of the original geom functions, you can simply add the little bit at the end to make the plot interactive. In the following examples, I will be using geom_point_interactive, but all the other functions work in similar ways, so feel free to explore on your own, as we also explore three of the aesthetic features of `ggiraph`.

tooltip

`tooltip` is an aesthetic that allows a certain column from the dataset to be displayed when the particular element is moused over. This could be useful when we just have a lot of points on two axes, and we want to know the object that actually has these two characteristics. In the following example, I will make a plot

```
sample <- ggplot(data = mcdonalds, aes(x = Total.Fat, y = Calories, color = Sodium))
example <- sample +
  geom_point_interactive(aes(tooltip = Item)) +
  xlim(0, 70) +
  xlab('Total Fat (in grams)') +
  ylim(0, 1250) +
  ggtitle('McDonalds Items by Total Fat and Calories')
ggiraph(code = {print(example)})
```

```
## Warning: Removed 1 rows containing missing values (geom_interactive_point).
```

Here we have a plot of the food items on total fat and calories, with the color based on the amount of sodium is in the product. We can see how useful the `tooltip` aesthetic is – now we can simply mouse over a point and see which item is the one that has the most calories and highest total fat. We can see that a Hash Brown has about 9 grams of total fat and 100-ish calories! Of course, there are more useful ways to use this function, but I wanted to look at McDonalds nutrition facts haha. :)

data_id

`data_id` allows a column of the dataset that contains an id to be associated with elements. Similar to `tooltip`, `data_id` takes in the name of a column, on which we can have the hover effect. The hover effect allows there to be different features on the plot when the user's mouse hovers over a point (or bar, or whatever other type of geom function).

The hover effect modifies the SVG graphic elements when a mouse is over the element. These effects can be configured using the `hover_css` argument, which is taken relative to SVG elements. Some common SVG elements include fill, stroke, and r (circle radius). Below, I will use basically the same plot and data as above, but also use `data_id` and `hover_css` to show how the hover effect can be implemented.

```
example2 <- sample +
  geom_point_interactive(aes(tooltip = Item, data_id = Item)) +
  xlim(0, 70) +
  xlab('Total Fat (in grams)') +
  ylim(0, 1250) +
  ggtitle('McDonalds Items by Total Fat and Calories')
ggiraph(code = {print(example2)}, hover_css = "fill:red;stroke:green")
```

```
## Warning: Removed 1 rows containing missing values (geom_interactive_point).
```

You can see that now when we hover over an element, the point becomes red with a green outline (in the holiday spirit!). This helps to show the user that they are indeed hovering over the element that they intended to. It might be more clear if there were less points and if the graph were more zoomed in, but in this example we're just looking at this big chunk of data.

zoom_max

Here's another argument that you can include once you've identified a `data_id`! This one is called `zoom_max`, which limits the amount that a user can zoom into the graph. This is useful because sometimes when you zoom in too much to a photo or something, it gets very pixelated and not very helpful anyways (I'm sure most of you have experienced this before).

```
example3 <- sample +
  geom_point_interactive(aes(tooltip = Item, data_id = Item)) +
  xlim(0, 70) +
  xlab('Total Fat (in grams)') +
  ylim(0, 1250) +
  ggtitle('McDonalds Items by Total Fat and Calories')
ggiraph(code = {print(example3)}, zoom_max = 10)
```

```
## Warning: Removed 1 rows containing missing values (geom_interactive_point).
```

Final example – putting everything together!

Now that we've learned a bit about how to use `ggiraph`, let's put all these aspects together to create plot.

```
example_final <- sample +
  geom_point_interactive(aes(tooltip = Item, data_id = Item)) +
  xlim(0, 70) +
  xlab('Total Fat (in grams)') +
  ylim(0, 1250) +
  ggtitle('McDonalds Items by Total Fat and Calories')
ggiraph(code = {print(example_final)}, hover_css = "fill:red;stroke:green", zoom_max = 10)
```

```
## Warning: Removed 1 rows containing missing values (geom_interactive_point).
```

In this plot, we can hover over points, where it will give us the identity of the item and change the fill and stroke of the point we are hovering over. We are now also able to zoom in to this graph for a more clear and precise view of all the data.

I hope that through this post, you were able to learn about some of the features of `ggiraph` and understand why it can be a useful package as an aid to `ggplot2`. If you guys are interested in learning more, there's so many more resources online, and feel free to check out some of my references as well!

References

Nutrition Facts for McDonalds Menu

Changing font size of axes in ggplot2

Data Visualization wtih ggplot2

Set scale limits

Quick Intro to ggiraph

Custom Animation Graphics

Examples of ggiraph

SVG Attributes