Matplotlib Review

Nick Sun

2/4/2020

Part 0

I have used numpy in the past so for this notebook, the only new things that I really learned are about broadcasting. In my previous R background, I had only ever used a handful of functions to create data and hardly every made use of matrices that did not have matching dimensions. I have never heard the term broadcasting being used in this context, so this will be confusing for me since I have used Julia in a previous course and in that language, broadcasting refers to applying a function to a vector of data.

Part 1

I am vaguely familiar with matplotlib and have used it sparingly before, so I knew about the general idea behind graphical backends (Julia has a similar framework). One thing that I will definitely forget, especially coming from R, is to use `plt.show()` to actually display plots. As I worked through the notebook, I tried to make sure I picked up the differences between matplotlib and ggplot. For example, setting axis titles and breaks in base R is done within the `plot()` function but for matplotlib, we have to call methods on the figure object. Creating subplots in R is usually done using `par(mfrow = c(2,2))`, but in matplotlib, we have to use the subplots method.

It will take a while for me to get used to the syntax of matplotlib, and really it will just take a lot of practice. I can already picture myself trying to plot something in Python using base R or ggplot syntax. Overall though, the plotting functionality of matplotlib doesn't seem too complicated and the syntax reminds me a lot of Julia.

Here is how I created the plot at the end. Not entirely sure if this is the most Pythonic way to do it, since I ended up using a loop:

```
fig, axes = plt.subplots(nrows=3, ncols=1)
y = [y1, y2, y3]

axes[0].set(title=names[0])
axes[1].set(title=names[1])
axes[2].set(title=names[2])

# To iterate over all items in a multidimensional numpy array, use the `flat` attribute
i = 0
for ax in axes.flat:
    # Remove all xticks and yticks...
    ax.set(xticks=[], yticks=[])
    ax.plot(x, y[i], color = 'black')
    i = i+1

plt.show()
```

Part 2

For this notebook, I also really enjoyed doing the exercises since it helps me actually get a feel for matplotlib. I didn't get the exact look of the plots to replicated, but I did get most of the geometries:

```
fig, ax = plt.subplots()
ax.bar(x = x_pos + (bar_width/2),
      height = y_avg,
      yerr = y_err,
      color = barcolor,
      width = bar_width,
      edgecolor='black')
ax.fill_between(x = x_pred,
                y1 = y_max_pred,
                y2 = y_min_pred,
                color = fillcolor)
ax.plot(x_raw,
       y_raw,
       color = linecolor)
plt.show()
```

Overall, I think this exercise was useful in helping me distinguish between ggplot and matplotlib syntaxes. In R, these different geometries would be called as layers on a ggplot object and in matplotlib, the syntax is more reminiscent of functions which manipulate the same plot object.

Part 3

I will definitely keep this notebook as a reference for further work with matplotlib, I have a similar markdown document for ggplot and I find it's super useful. One thing that I think is interesting is that linetype and character are defined in R using numeric codes whereas matplotlib allows you to specify using actual ASCII like "–" what the appearance of certain geometries should be. Julia has a very similar syntax, so it's cool to see it appear here in Python.

I did not know that matplotlib had some built-in support for LaTeX, so I'll certainly keep that in mind for the future! I also did not know that the defaults for matplotlib were held in an editable file. That is a feature I wish R had! As far as I know, you have to manually specify in each of your files which custom palette you are going to use, there is no way to tell ggplot that you'll always want to use a new color scheme by default. I'm a big fan of playing around with config files, so I might be giving this a go later.

Part 4

Again, this notebook will be a handy reference since I always forget how to do relatively simple things like adjusting the tickmarks on my plots.