Data Visualization with R (GGPlot2) vs Python (MatPlotLib)

Basics

```
library(ggplot2)

g <- ggplot (data = data, aes(x, y, ...))

g +  <GEOM_FUNCTION> (mapping = aes( <MAPPINGS> ),
     stat = <STAT> , position = <POSITION> ) +
     <COORDINATE_FUNCTION> +
     <FACET_FUNCTION> +
     <SCALE_FUNCTION> +
     <LABELS_FUNCTION> +
     <THEME_FUNCTION>

import matplotlib as mpl
import matplotlib.pyplot as plt

fig, ax = plt.subplots()

ax.<PLOT_FUNCTION>
ax.<TICK_FUNCTION>
ax.<LABEL_FUNCTION>
ax.<THEME_FUNCTION>

plt.show()
```

Basic Plots <GEOM_FUNCTION> <PLOT_FUNCTION>

Scatter - Explore the relationship between two continuous variables

```
g + geom_point(x, y, alpha, color, fill, shape, size, stroke, ...)
ax.plot(kind='scatter', data=..., x, y, colormap, ...)
```

Bar - Explore the relationship of multivariate categorical variables

```
g + geom_bar(x, alpha, color, fill, linetype, size, weight, ...)
ax.plot(kind='bar', data=..., x, y, colormap, ...)
```

Line - Track changes over short and long periods of time

```
g + geom_line(x, y, alpha, color, group, linetype, size, ...)
ax.plot(kind='line', data=..., x, y, colormap, ...)
```

Contour - Give a sense of the density of the data at a glance

```
g + geom_contour(x, y, z, alpha, color, group, linetype, size, weight, ...)
ax.contour(data=..., [x, y,] z, [levels], colors, alpha, ...)
```

Step - Show changes that occur at irregular intervals

```
g + geom_step(direction = "...", x, y, alpha, color, group, linetype, size, ...)
ax.step(data=..., x, y, where='pre', ...)
```

Box - Represent a distribution of a one-dimensional continuous variable

```
g + geom_boxplot(x, y, lower, middle, upper, ymax, ymin, alpha, color, fill,, ...)
ax.plot(kind='box', data=..., x, y, colormap, ...)
```

Histogram - View data's empirical distribution within a set of intervals

```
g + geom_histogram(binwidth, x, y, alpha, color, fill, linetype, size, weight, ...)
ax.plot(kind='hist', data=..., x, y, colormap, ...)
```

Violin Plot - Visualize one discrete variable and one continuous variable

```
g + geom_violin(scale, x, y, alpha, color, fill, group, linetype, size, weight, ...)
ax.violinplot(data=..., [x, y, ...], positions, vert, widths, ...)
```

Hex Heatmaps - Compare different parameters while viewing their relative distributions

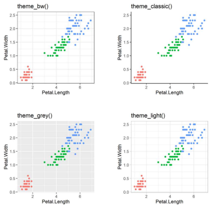
```
g + geom_hex(x, y, alpha, color, fill, size, ...)
ax.plot(kind='hexbin', data=..., x, y, colormap, ...)
```

Error Bar - Help see margins of error and standard deviations at a glance

```
g + geom_errorbar(x, ymax, ymin, alpha, color, group, linetype, size, width, ...)
ax.errorbar(data=..., x, y, yerr, xerr, ecolor, elinewidth, ...)
```

Styles and Themes <THEME_FUNCTION> <THEME_FUNCTION>

```
r + theme_theme()
plt.style.use(...)
```

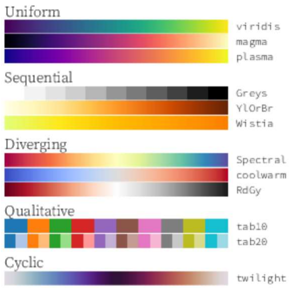


https://statisticsglobe.com/ggplot2-themes-r

Color Palettes

Use with aesthetics

```
color(...) or fill(...)
plt.get_cmap(...)
```



Labels <LABEL_FUNCTION> <LABEL_FUNCTION>

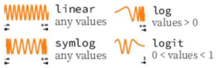
```
g + labs(x = "Add label to the x-axis",
         y = "Add label to the y-axis",
         title = "Add a title above plot",
         subtitle = "Add a subtitle below title",
         caption = "Add a caption below plot")
ax.set_xlabel("Add label to the x-axis")
ax.set_ylabel("Add label to the y-axis")
ax.set_title("Add a title above plot")
ax.suptitle("Add a subtitle below title")
ax.text(x, y, "Add a caption below plot")
```

X and Y Location Scales

Use with x or y aesthetics:

```
scale_[xy]_log10()
scale_x_reverse()
scale_x_sqrt()

ax.set_[xy]scale(scale,...)
```



10 Golden Rules for Data Visualization

1. Start with a Goal - Create a strong foundation to ensure the visualization achieves a well-defined goal

2. Know Your Data - Understand the data you are working with to use the best data to support the overall goal

3. Put Your Audience First - Customize the visualization to the stakeholder to guarantee they can focus on the message

4. Be Media Sensitive - Beware of the limitations and the responsiveness of your visualization platform to ensure it reaches the proper audience

5. Choose the Right Chart - Know the strengths and weaknesses of each graph and chart to ensure the use of the optimal visualization

6. Chart Smart - Do not force the data to fit into a certain graph; Be sure to not misrepresent, distort, or mislead the stakeholders

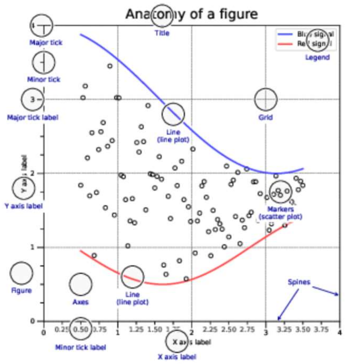
7. Use Labels Wisely - Provide a title and axis labels to give the stakeholder context; Allow the visualization to represent the data rather than captions

8. Design to the Point - Ensure all aspects of your visual is meaningful; Eliminate the any features that create distractions for the stakeholder

9. Let the Data Speak - Use visual cues to create a narration that will draw the stakeholder to important conclusions

10. Feedback is a Good Thing - Finetune your visualizations with feedback from the stakeholders

Anatomy of a Figure



Resources:

R Source -
Data Visualization PDF from https://rstudio.cloud/learn/cheat-sheets

Python Source -
https://www.proquest.com/docview/1906055447?parentSessionId=lQnHWjBk3oGBkYK1NzAbXw9S3drWM9F88fkPrTgYItA%3D

Golden Rules -
https://www.proquest.com/docview/1906055447?parentSessionId=lQnHWjBk3oGBkYK1NzAbXw9S3drWM9F88fkPrTgYItA%3D

Created By: Kylie Brothers