Visualization with seaborn

Bruno Gonçalves

www.data4sci.com/newsletter

https://github.com/DataForScience/Seaborn

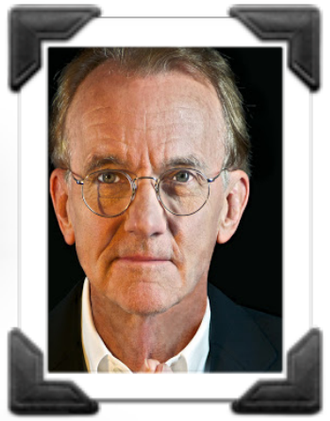
Who Am I?

- Name: Bruno Gonçalves
- Work: Senior Data Scientist
- Background: Physics (PhD) and Computer Science (MS)
- Experience: 10+ years using large scale datasets to analyze individual human behavior.
- Website: www.data4sci.com
- Twitter: @data4sci
- Email: bgoncalves@data4sci.com

Why Visualization?

"Information Visualization is a form of knowledge compression"
D. McCandless

"There is no such thing as information overload. There is only bad design."
E. Tufte



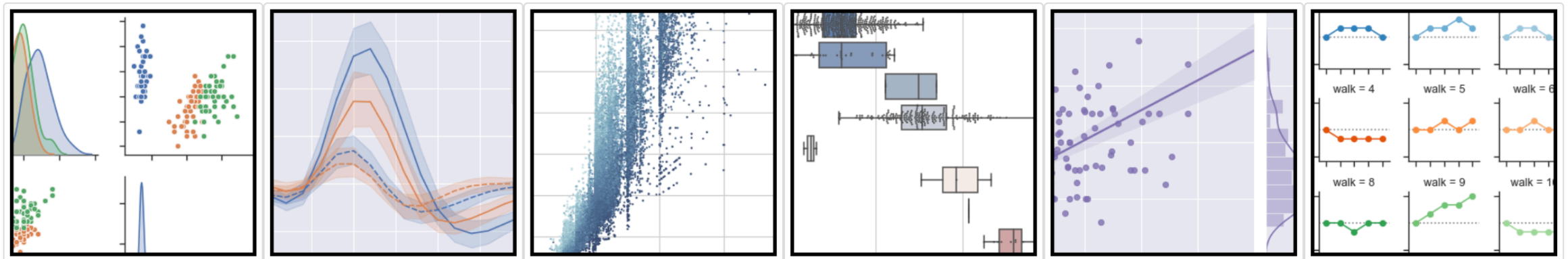
"Never leave a number all by itself. Never believe that one number on its own can be meaningful. If you are offered one number, always ask for at least one more. Something to compare it with."
H. Rosling

"The greatest value of a picture is when it forces us to notice what we never expected to see."
J. Tukey

Why seaborn?

seaborn.pydata.org

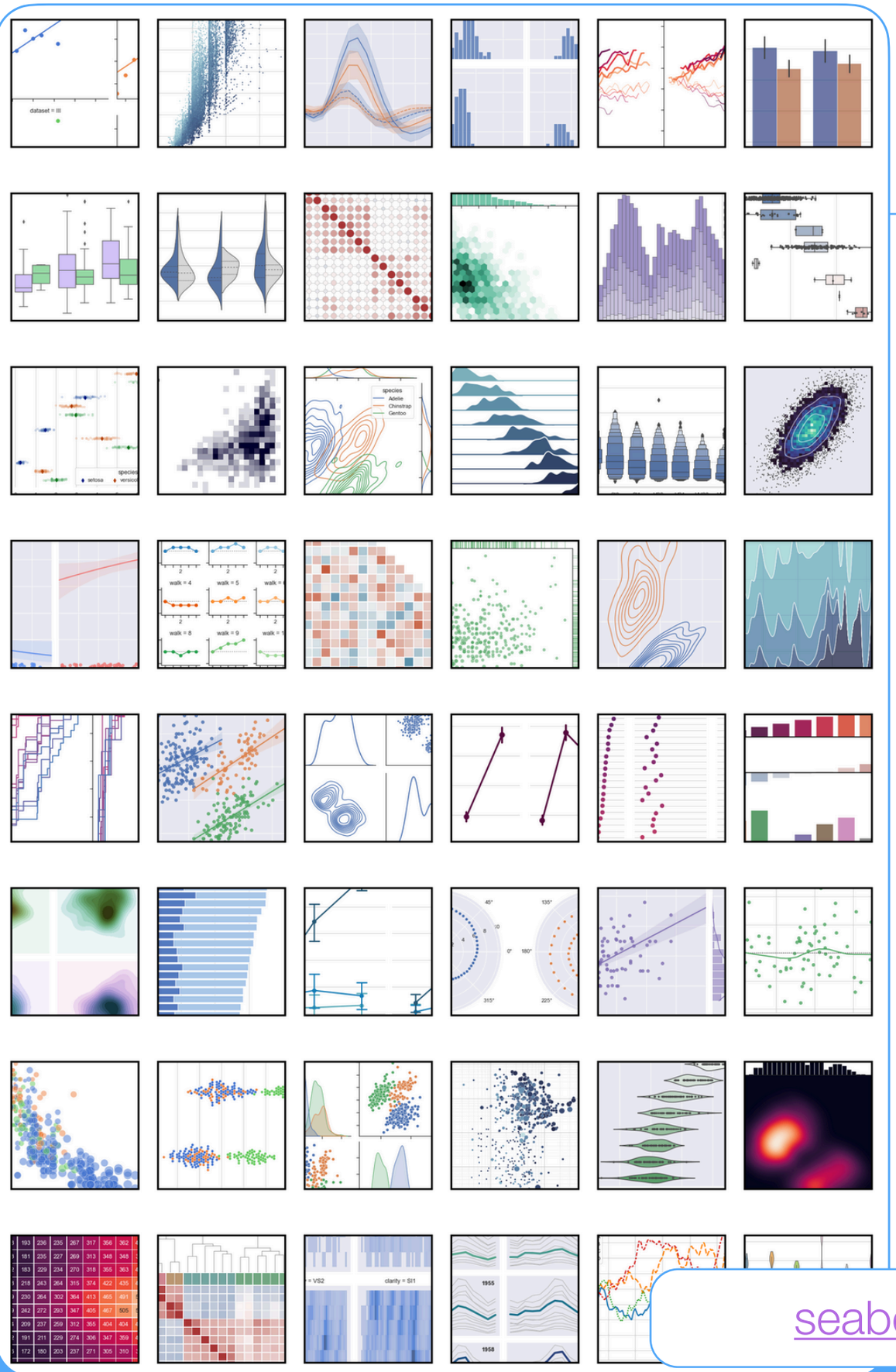
seaborn: statistical data visualization



"Seaborn is a library for making statistical graphics in Python. It is built on top of matplotlib and closely integrated with pandas data structures.

Seaborn aims to make visualization a central part of exploring and understanding data. Its dataset-oriented plotting functions operate on dataframes and arrays containing whole datasets and internally perform the necessary semantic mapping and statistical aggregation to produce informative plots."

seaborn



Extensive gallery that provides the source code for all the figures shown

seaborn.pydata.org

@bgoncalves

seaborn

seaborn.pydata.org

- The first step is to import the seaborn module:

```
import seaborn as sns
```

- The convention is to import seaborn as sns (sns is a geeky reference to Sam Norman Seaborn, a fictional character on the television show The West Wing)
- The sns module contains all the functions we will use as direct members.

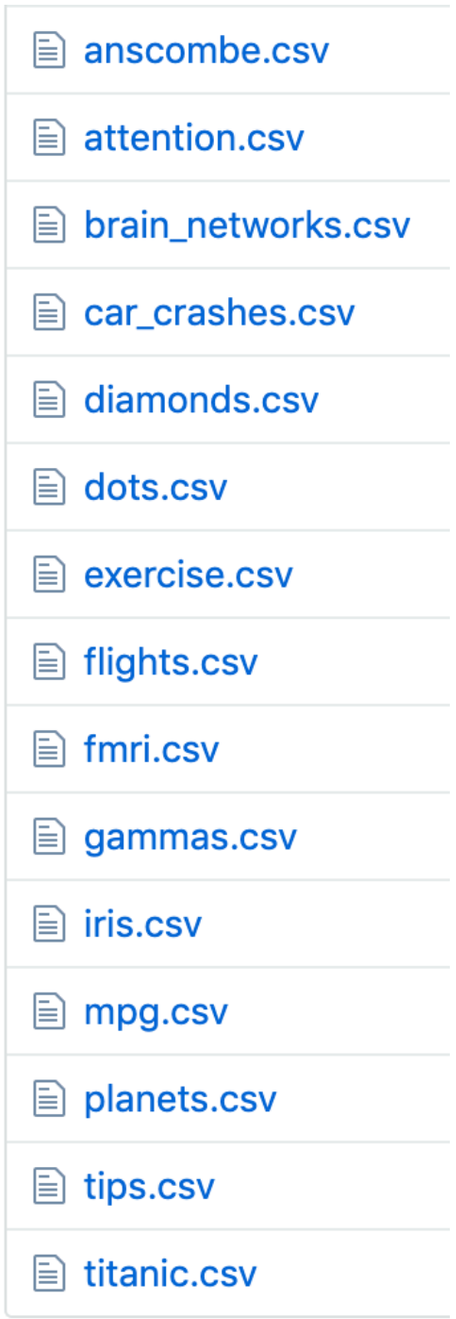
Datasets

seaborn.pydata.org

- The first step is to import the seaborn module:

```
import seaborn as sns
```

- The convention is to import seaborn as sns (sns is a geeky reference to Sam Norman Seaborn, a fictional character on the television show The West Wing)
- The sns module contains all the functions we will use as direct members.
- For ease of learning and demonstration, seaborn makes it easy to access standard datasets.
- The datasets can be access easily by calling the load_dataset function with the file name (sans extension) as a parameter.



Datasets

seaborn.pydata.org

- The first step is to import the seaborn module:

```
import seaborn as sns
```

- The convention is to import seaborn as sns (sns is a geeky reference to Sam Norman Seaborn, a fictional character on the television show The West Wing)
- The sns module contains all the functions we will use as direct members.
- For ease of learning and demonstration, seaborn makes it easy to access standard datasets.
- The datasets can be access easily by calling the load_dataset function with the file name (sans extension) as a parameter.
- .load_dataset(name) - where name is “iris”, “mpg”, etc…
- datasets are returned as "tidy" pandas dataframes

	total_bill	tip	sex	smoker	day	time	size
0	16.99	1.01	Female	No	Sun	Dinner	2
1	10.34	1.66	Male	No	Sun	Dinner	3
2	21.01	3.50	Male	No	Sun	Dinner	3
3	23.68	3.31	Male	No	Sun	Dinner	2
4	24.59	3.61	Female	No	Sun	Dinner	4

- anscombe.csv
- attention.csv
- brain_networks.csv
- car_crashes.csv
- diamonds.csv
- dots.csv
- exercise.csv
- flights.csv
- fmri.csv
- gammas.csv
- iris.csv
- mpg.csv
- planets.csv
- tips.csv
- titanic.csv

Types of plots

- seaborn can handle a large number of different kinds of plots

- sns.scatterplot()
- sns.lineplot()

Relationship plots

- sns.stripplot()
- sns.swarmplot()
- sns.boxplot()
- sns.violinplot()
- sns.boxenplot()
- sns.pointplot()
- sns.barplot()
- sns.countplot()

Categorical plots

Types of plots

Axes level

- sns.scatterplot()
- sns.lineplot()

- sns.stripplot()
- sns.swarmplot()
- sns.boxplot()
- sns.violinplot()
- sns.boxenplot()
- sns.pointplot()
- sns.barplot()
- sns.countplot()

Figure level

Relationship plots

sns.relplot()

- kind="scatter"
- kind="line"

Figure level functions are equivalent to the axes level ones, (with the right kind setting) but more adapted for ease of exploration

Categorical plots

sns.catplot()

- kind="strip"
- kind="swarm"
- kind="box"
- kind="violin"
- kind="boxen"
- kind="point"
- kind="bar"
- kind="count"

Figure vs Axes level

github.com/DataForScience/Seaborn

Figure level

```
1  fg = sns.relplot(x="total_bill", y="tip", data=tips, kind="scatter")
```

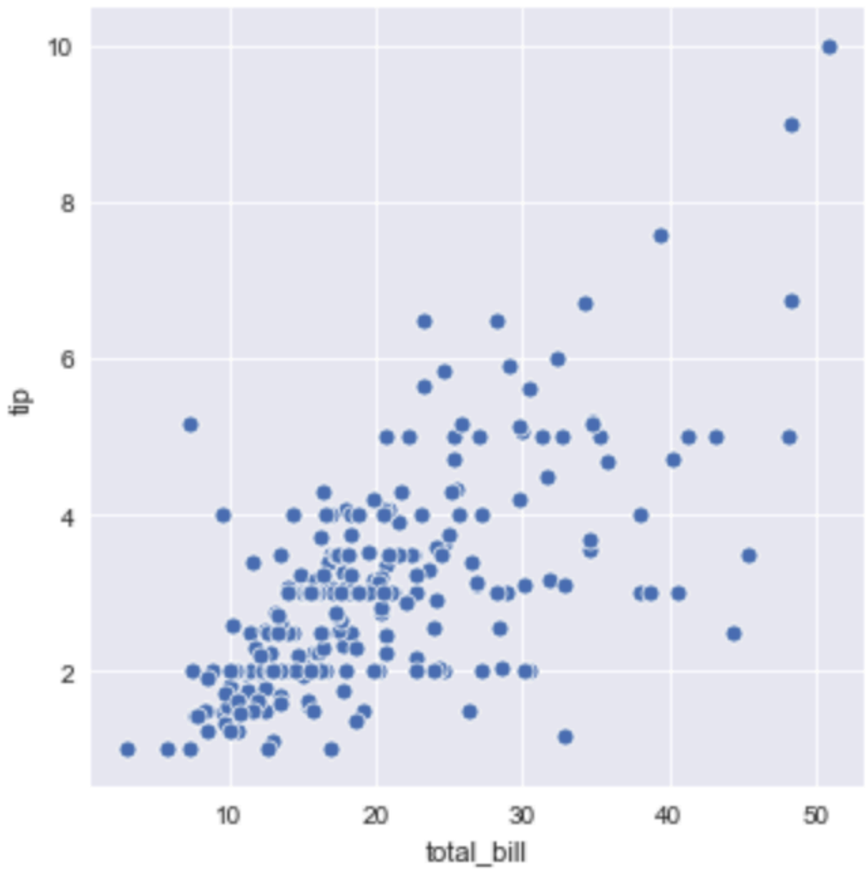
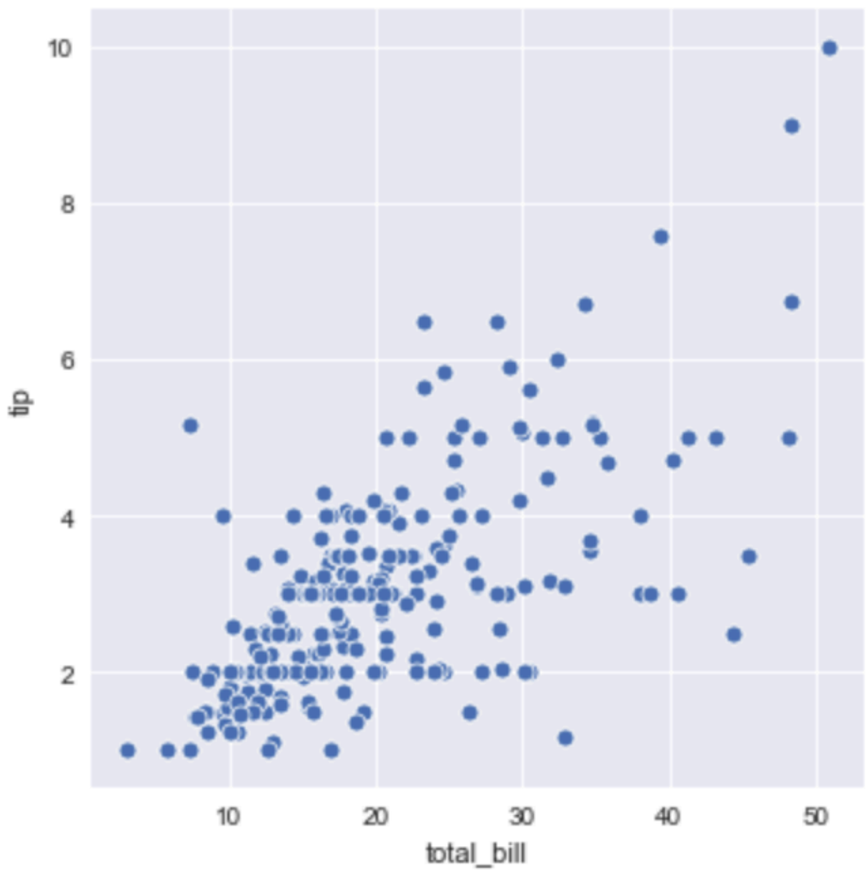


Figure vs Axes level

github.com/DataForScience/Seaborn

Figure level

```
1  fg = sns.relplot(x="total_bill", y="tip", data=tips, kind="scatter")
```



Axes level

```
1  ax = sns.scatterplot(x="total_bill", y="tip", data=tips)
```

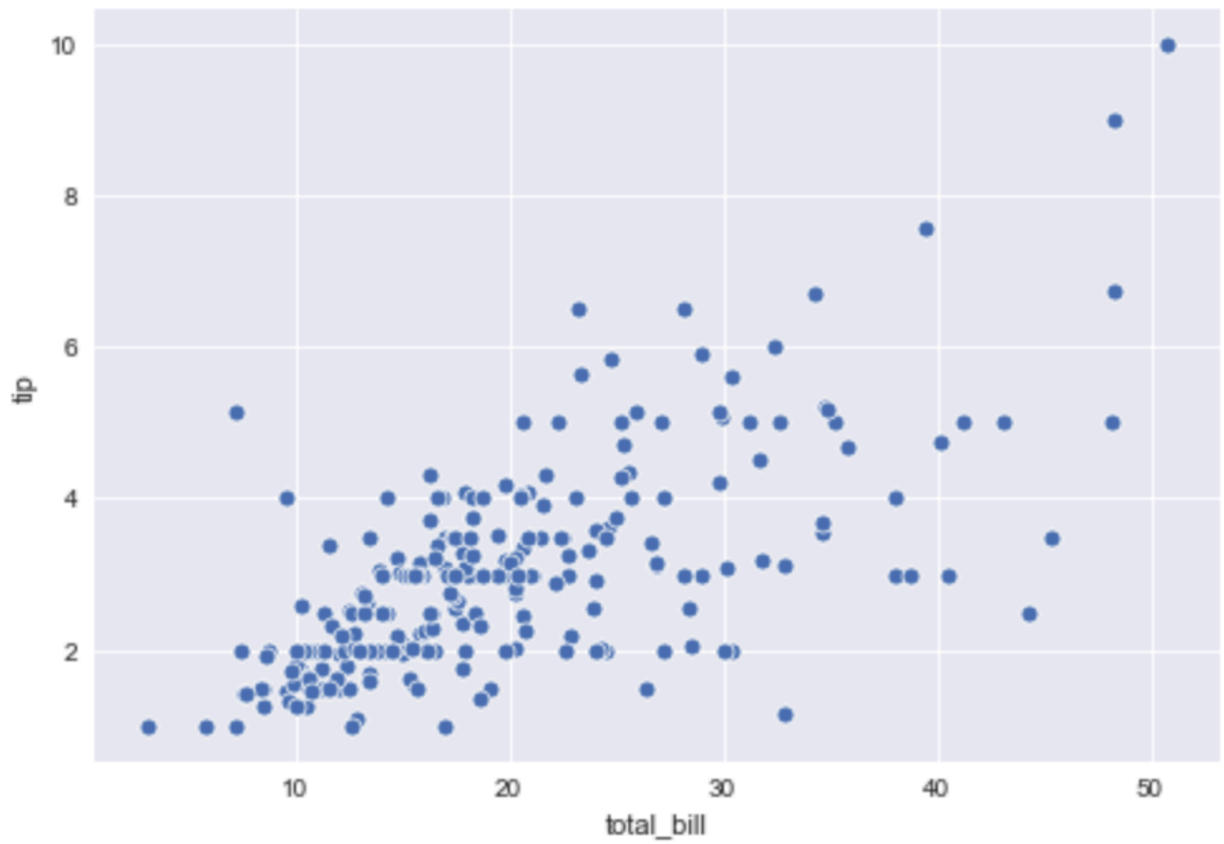
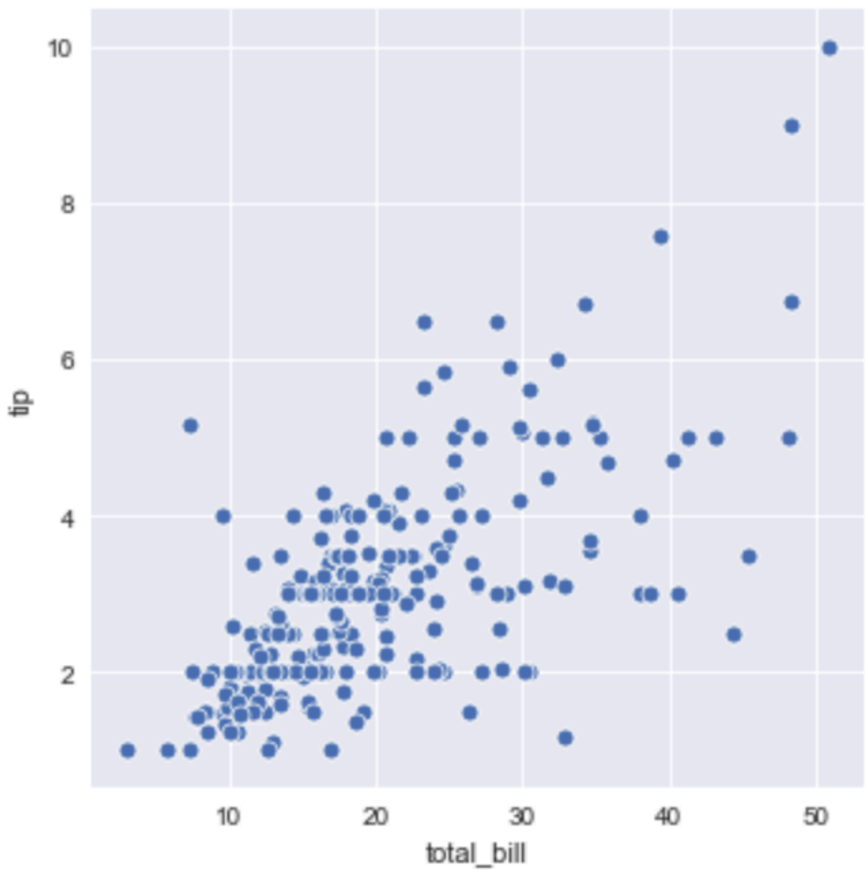


Figure vs Axes level

github.com/DataForScience/Seaborn

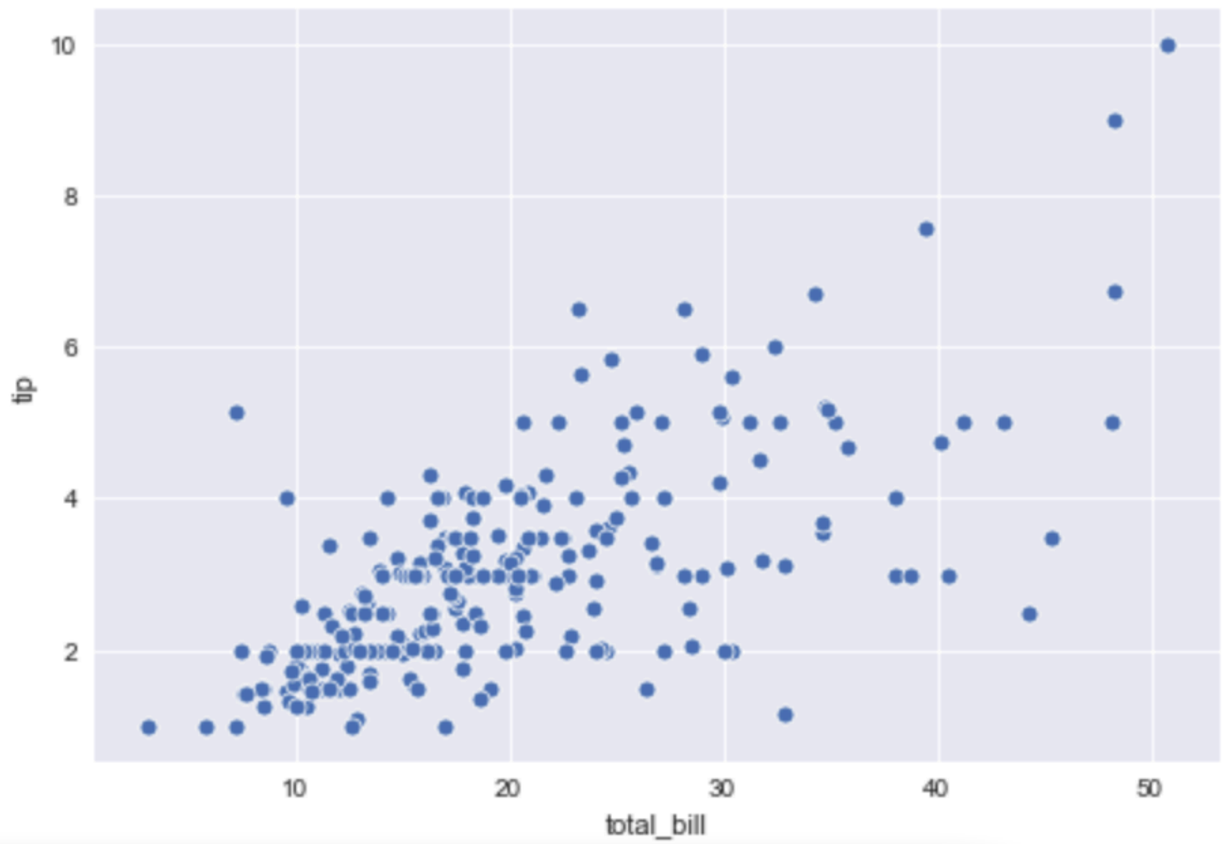
Figure level

```
1  fg = sns.relplot(x="total_bill", y="tip", data=tips, kind="scatter")
```



Axes level

```
1  ax = sns.scatterplot(x="total_bill", y="tip", data=tips)
```



The two approaches produce slightly different results.
For convenience, we will focus mostly on Figure level functions

Basic function structure

- The basic plotting functions are designed to work directly with pandas data frames
- Every plotting function takes a data parameter that is used to pass the correct DataFrame to the function
- There is no limitation on the number of columns that the DataFrame can contain.
- The specific columns to be plotted are passed by setting other parameters to the respective column names. In particular:
  - x - column to plot along the x-axis
  - y - column to plot along the y axis
- Plot properties such as hue, size, style, etc can also be set using column names
- In the case of figure level functions, the kind parameter determines what specific type of plot is produced.

Figure level functions

- sns.relplot(x, y, hue, size, style, data, kind)
  - x, y - column names to plot in each axes
  - hue - column name specifying how to color each data element
  - size - column name to use to determine the size of each element
  - style - column name to use to determine the style of each element
  - data - dataframe containing the data to plot
  - kind - string specifying the type of plot to produce
- sns.catplot(x, y, hue, order, data, orient, kind)
  - orient - specifies the orientation of the plot ('v' or 'h' for vertical or horizontal)
  - order - specifies the order in which the categorical variable (x or y) is plotted
  - *_order - family of parameters that determiner the order in which the respective categorical value (hue_order, style_order, size_order, etc…) is plotted

Figure level functions

- As we saw, axes level and figure level functions are equivalent. Every plot you can generate with an axes level function can also be generated by a figure level function
- The converse is, however, not true.
- One of the main advantages of figure level functions is their ability to easily generate subplots by just passing a couple of extra parameters:
    - row, col - specifies the column names to be used to split the dataset and plot along rows and columns
    - col_wrap - width at which to wrap the column. Incompatible with a row setting.
    - row_order, col_order - work similarly to the other *_order parameters
- The return types of **axes level** functions and **figure level** function are also different:
    - Axes level functions return a matplotlib Axis object
    - Figure level functions return a seaborn FacetGrid object
- Axes level functions can be easily added to a pre-existing matplotlib plot by passing an matplotlib Axis object to the ax parameter.

Figure level functions

github.com/DataForScience/Seaborn

```
sns.relplot(x="total_bill", y="tip", data=tips, kind="scatter",
            hue="sex", size="size",
            col='smoker', row='day', style='time',
            row_order=["Thur", "Fri", "Sat", 'Sun']
            )
```

Figure level functions

github.com/DataForScience/Seaborn

```
sns.relplot(x="total_bill", y="tip", data=tips, kind="scatter",
            hue="sex", size="size",
            col='smoker', row='day', style='time',
            row_order=["Thur", "Fri", "Sat", 'Sun']
            )
```

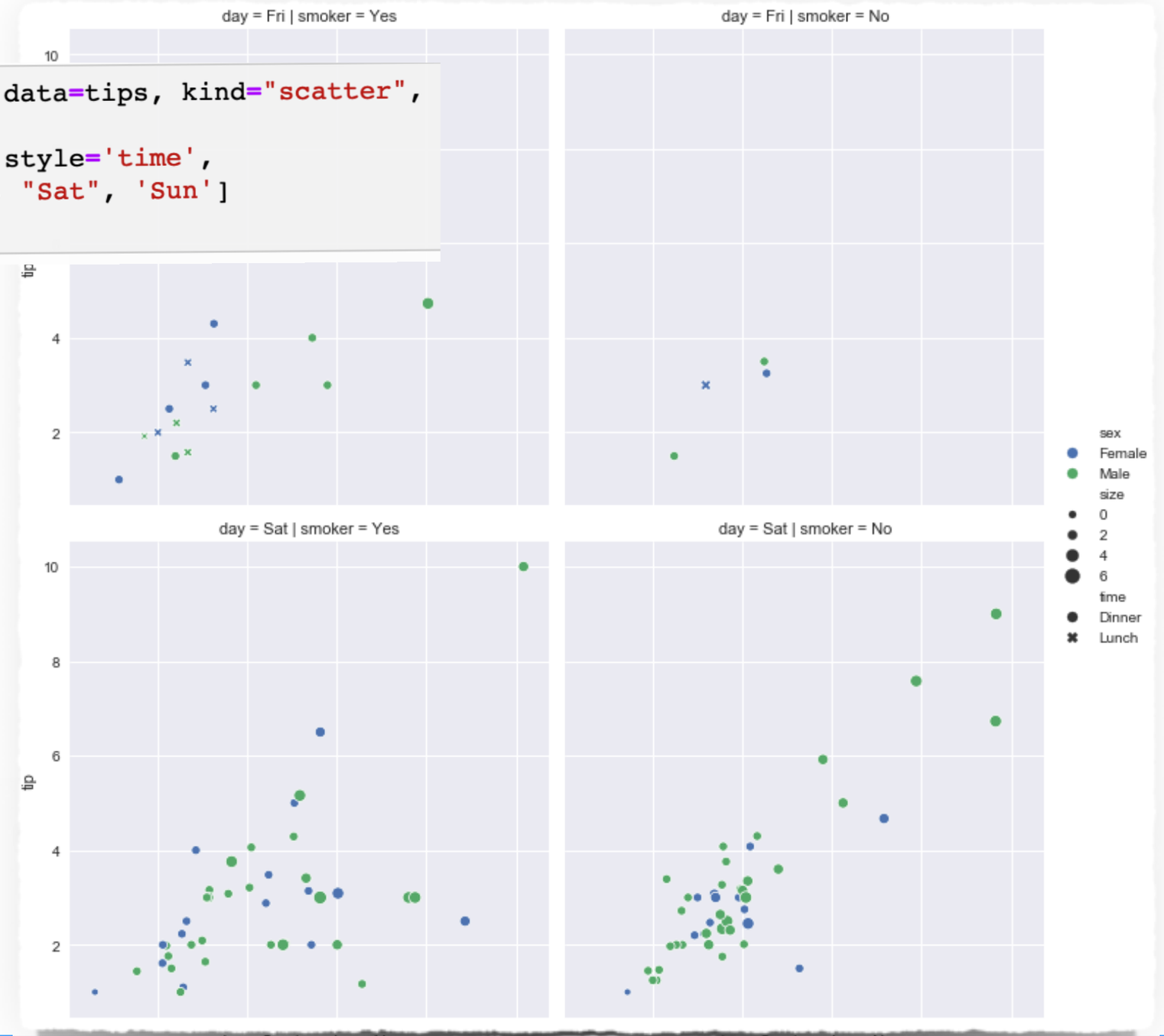


Figure level functions

github.com/DataForScience/Seaborn

```
sns.relplot(x="total_bill", y="tip", data=tips, kind="scatter",
            hue="sex", size="size",
            col='smoker', row='day', style='time',
            row_order=["Thur", "Fri", "Sat", 'Sun']
            )
```

FacetGrid

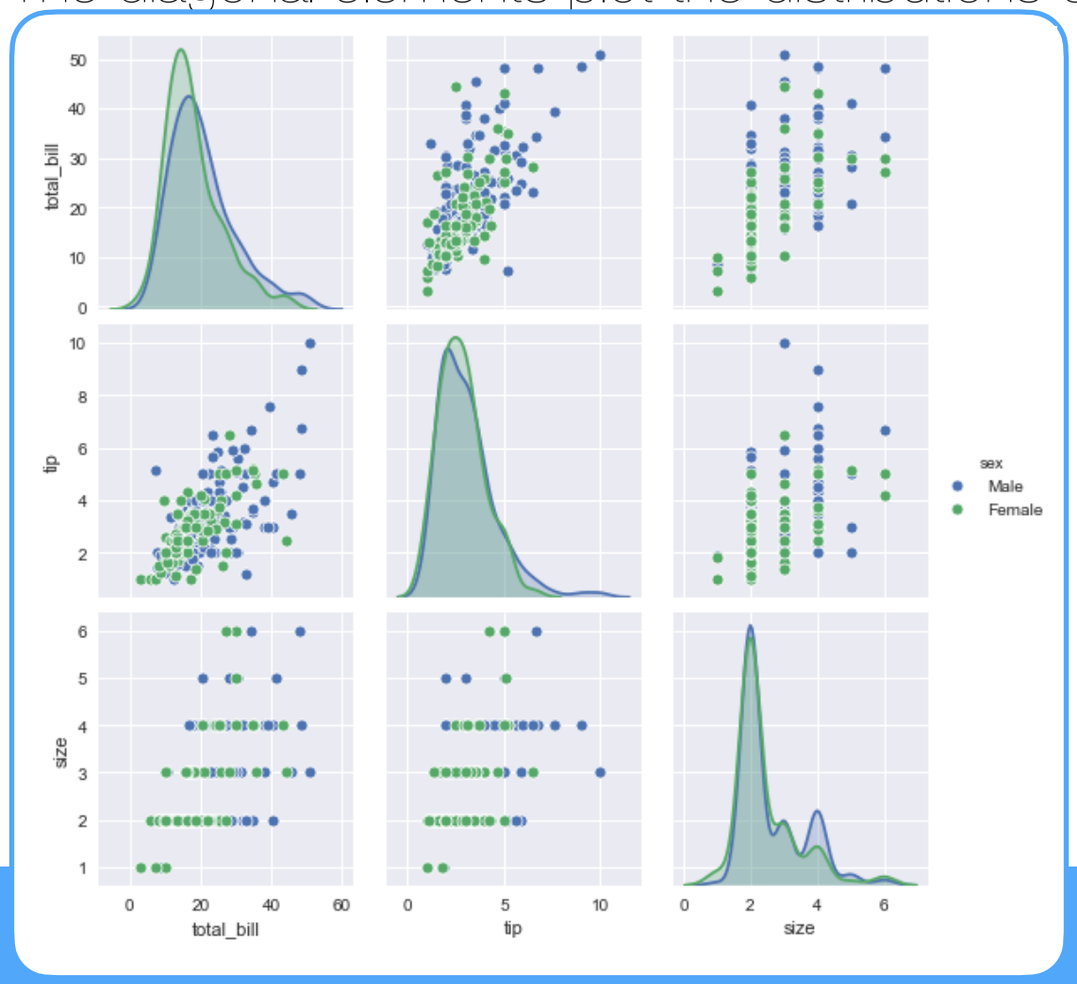
https://seaborn.pydata.org/generated/seaborn.FacetGrid.html

- FacetGrid is the object that seaborn uses in the background to generate the subplots.
- You can easily use FacetGrid with any matplotlib "compatible" plotting function.
- The process is divided into two parts:
    - First, you instantiate a FacetGrid object with the correct data frame and basic plotting parameters (these are the same parameters you would use with any other seaborn function).
    - Second, you use the FacetGrid.map method to initialize the FacetGrid object with the function that will generate the plot along with any extra parameters you require.
- Any function that modifies the currently active Axis object can be used (both matplotlib functions and custom functions)
- Any extra parameters passed to map() will be passed directly to the plotting function. This function should accept a **kwargs argument
- FacetGrid will pass the sliced data as Series to the plotting function

FacetGrid

https://seaborn.pydata.org/generated/seaborn.FacetGrid.html

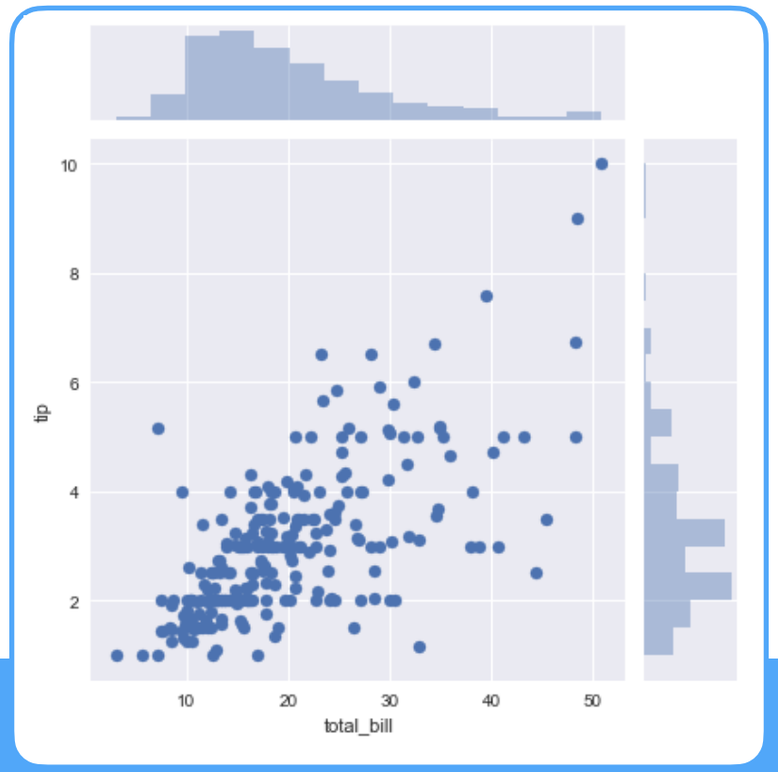
- After plotting the FacetGrid will contain a field axes containing an array with references to all the matplotlib Axes objects it generated.
- We can further customize the plot by directly using matplotlib functionality.
- FacetGrid is a general concept with several specialized variations within seaborn
  - PairPlot - specialized FacetGrid that automatically plots every pairwise relationship of numerical features. The diagonal elements plot the distributions of the respective feature.



FacetGrid

https://seaborn.pydata.org/generated/seaborn.FacetGrid.html

- After plotting the FacetGrid will contain a field axes containing an array with references to all the matplotlib Axes objects it generated.
- We can further customize the plot by directly using matplotlib functionality.
- FacetGrid is a general concept with several specialized variations within seaborn
  - PairPlot - specialized FacetGrid that automatically plots every pairwise relationship of numerical features. The diagonal elements plot the distributions of the respective feature.
  - JointPlot - specialized FacetGrid that uses marginal right and top plots to visualize the distributions of the X and Y features.



Distribution plots

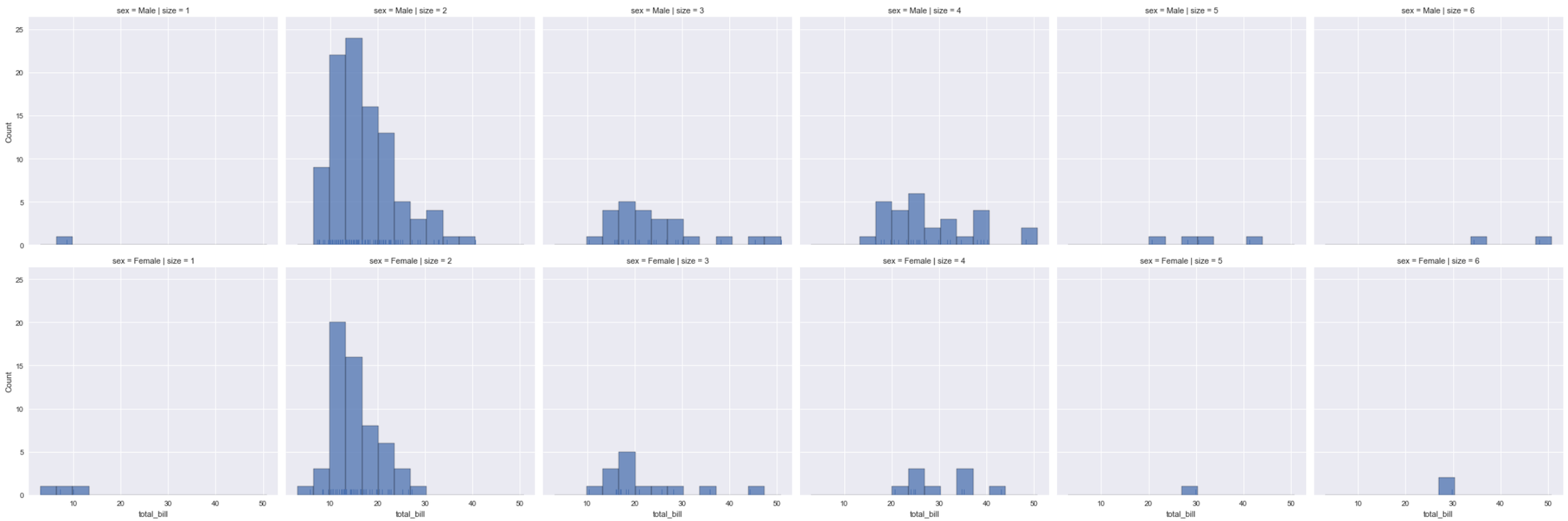
https://seaborn.pydata.org/generated/seaborn.displot.html

- We have covered the two main kinds of Figure Level plotting functions, sns.relplot() for Relationship Plots and sns.catplot() for Categorical Plots.

- The third class of function is sns.displot() for Distribution Plots and it allows us to generate a FacetGrid plot of the distributions underlying our data.

- Supports univariate and bivariate distribution of data

- Allows slicing using hue, row, col, etc.



Distribution plots

https://seaborn.pydata.org/generated/seaborn.displot.html

- sns.displot(x, y, hue, data, kind)
  - x, y - column names to plot in each axes. Specify only x for univariate distribution and both x and y for bivariate distributions
  - hue - column name specifying how to color each data element
  - data - dataframe containing the data to plot
  - row, col - Column names to define the rows and columns of the FacetGrid
  - kind - string specifying the type of plot to produce
  - rug - Boolean flag determining whether or not to add a rug plot highlighting the locations of the data points
  - kind - The kind of plot to generate:
    - “hist” - Plot 1D or 2D histogram
    - “kde” - Plot histogram using Kernel Density Estimation
    - “ecdf” - Plot empirical Cumulative Distribution Function

Types of plots

Axes level

- sns.histplot()
- sns.kdeplot()
- sns.ecdfplot()

Figure level

Distribution plots

sns.displot()

- kind="hist"
- kind="kde"
- kind="ecdf"

Code - Seaborn

https://github.com/DataForScience/Seaborn

Thank you!

www.data4sci.com/newsletter

- Hope you enjoyed this talk. Looking forward to your questions and thoughts.
- Don't forget to download the Jupyter notebook from GitHub: github.com/DataForScience/Seaborn
- Please feel free to reach out on Social Media!
- Website: www.data4sci.com
- Twitter: @data4sci
- Email: bgoncalves@data4sci.com

END