SESSION 1

What is Data Visualization?

Data visualization is a field in data_analysis that deals with visual representation of data. It graphically plots data and is an effective way to communicate inferences from data.

Using data_visualization, we can get a visual summary of our data. With pictures, maps and graphs, the human mind has an easier time processing and understanding any given data. Data visualization plays a significant role in the representation of both small and large data sets, but it is especially useful when we have large data sets, in which it is impossible to see all of our data, let alone process and understand it manually.

Data visualization is part art and part science. The challenge is to get the art right without getting the science wrong and vice versa. Data visualization first has to accurately convey the data. It must not mislead or distort. If one number is twice as large as another is, but in the visualization they look to be about the same, then the visualization is wrong. At the same time, a data visualization should be aesthetically pleasing. Good visual presentations tend to enhance the message of the visualization. If a figure contains jarring colors, imbalanced visual elements, or other features that distract, then the viewer will find it harder to inspect the figure and interpret it correctly.

.

Ugly, bad, and wrong figures

We frequently show different versions of the same figures, some as examples of how to make a good visualization and some as examples of how not to. To provide a simple visual guideline of which examples should be emulated and which should be avoided, I am clearly labeling problematic figures as “ugly”, “bad”, or “wrong” (Figure 1.1):

ugly—A figure that has aesthetic problems but otherwise is clear and informative.

bad—A figure that has problems related to perception; it may be unclear, confusing, overly complicated, or deceiving.

wrong—A figure that has problems related to mathematics; it is objectively incorrect.

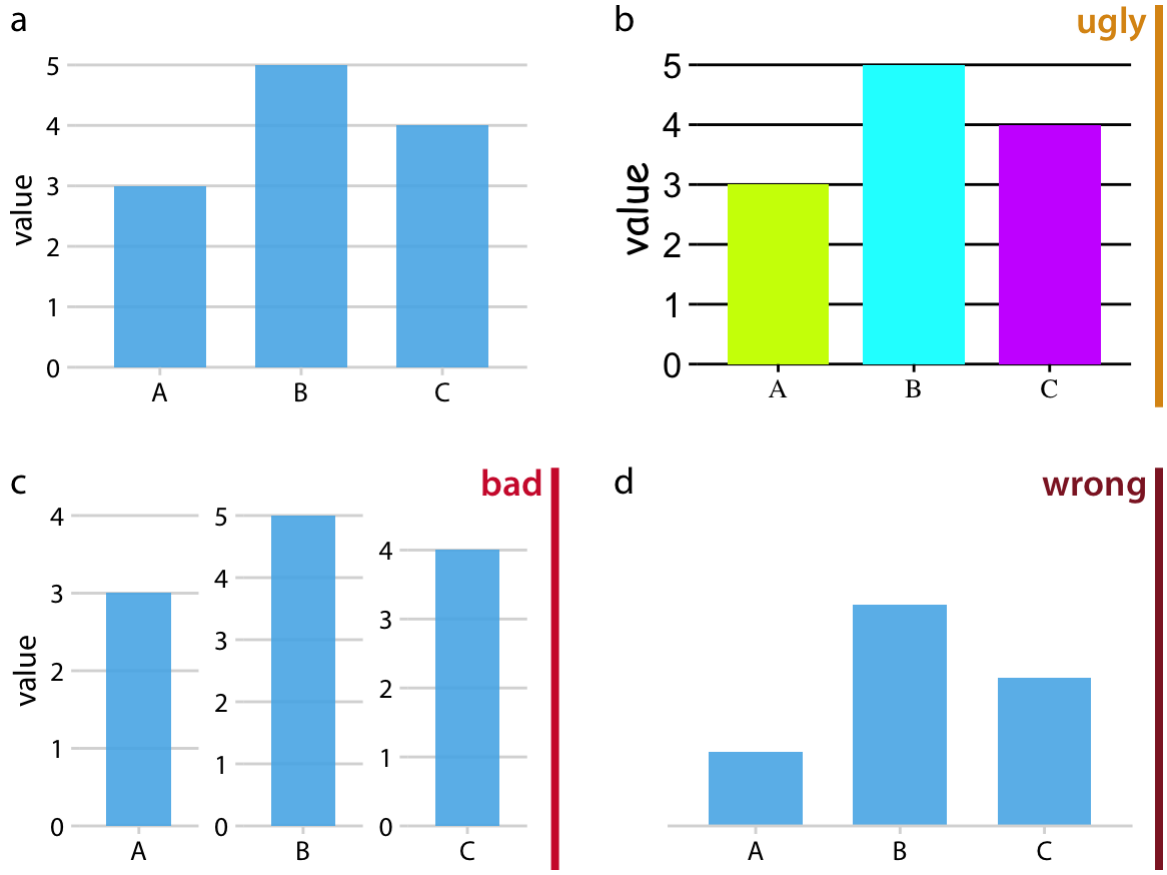

Figure 1.1: Examples of ugly, bad, and wrong figures. (a) A bar plot showing three values (A = 3, B = 5, and C = 4). This is a reasonable visualization with no major flaws. (b) An ugly version of part (a). While the plot is technically correct, it is not aesthetically pleasing. The colors are too bright and not useful. The background grid is too prominent. The text is displayed using three different fonts in three different sizes. (c) A bad version of part (a). Each bar is shown with its own *y*-axis scale. Because the scales don't align, this makes the figure misleading. One can easily get the impression that the three values are closer together than they actually are. (d) A wrong version of part (a). Without an explicit *y* axis scale, the numbers represented by the bars cannot be ascertained. The bars appear to be of lengths 1, 3, and 2, even though the values displayed are meant to be 3, 5, and 4.

We are not explicitly labeling good figures. Any figure that isn't clearly labeled as flawed should be assumed to be at least acceptable. It is a figure that is informative, looks appealing, and could be printed as is. Note that among the good figures, there will still be differences in quality, and some good figures will be better than others.

We are generally providing my rationale for specific ratings, but some are a matter of taste. In general, the "ugly" rating is more subjective than the "bad" or "wrong" rating.

Moreover, the boundary between “ugly” and “bad” is somewhat fluid. Sometimes poor design choices can interfere with human perception to the point where a “bad” rating is more appropriate than an “ugly” rating. In any case, I encourage you to develop your own eye and to critically evaluate my choices.

In today’s world, a lot of data is being generated on a daily basis. And sometimes to analyze this data for certain trends, patterns may become difficult if the data is in its raw format. To overcome this data visualization comes into play. Data visualization provides a good, organized pictorial representation of the data, which makes it easier to understand, observe, analyze. In this tutorial, we will discuss how to visualize data using Python.

Data visualization is the discipline of trying to understand data by placing it in a visual context so that patterns, trends, and correlations that might not otherwise be detected can be exposed. Python offers multiple great graphing libraries packed with lots of different features. Whether you want to create interactive or highly customized plots, Python has an excellent library for you.

Useful packages for visualizations in python

Matplotlib

Matplotlib is a visualization library in Python for 2D plots of arrays. Matplotlib is written in Python and makes use of the NumPy library. It can be used in Python and IPython shells, Jupyter notebook, and web application servers. Matplotlib comes with a wide variety of plots like line, bar, scatter, histogram, etc. which can help us, deep-dive, into understanding trends, patterns, correlations. It was introduced by John Hunter in 2002.

Matplotlib is the most basic library for visualizing data graphically. It includes many of the graphs that we can think of. Just because it is basic does not mean that it is not powerful, many of the other data visualization libraries we are going to talk about are based on it. Matplotlib’s charts are made up of two main components, the axes (the lines that delimit thearea of the chart) and the figure (where we draw the axes, titles and things that come out of the area of the axes).

If you are working with Python from the terminal or a script, after defining the graph with the functions we have written above use plt.show(). If you're working from jupyter notebook, add %matplotlib inline to the beginning of the file and run it before making the chart.

We can make multiple graphics in one figure. This goes very well for comparing charts or for sharing data from several types of charts easily with a single image.

Seaborn

Seaborn is a dataset-oriented library for making statistical representations in Python. It is developed atop matplotlib and to create different visualizations. It is integrated with pandas data structures. The library internally performs the required mapping and aggregation to create informative visuals It is recommended to use a Jupyter/IPython interface in matplotlib mode.

Seaborn is a library based on Matplotlib. What it gives us are nicer graphics and functions to make complex types of graphics with just one line of code. We import the library and initialize the style of the graphics with sns.set(), without this command the graphics would still have the same style as Matplotlib.

Bokeh

Bokeh is an interactive visualization library for modern web browsers. It is suitable for large or streaming data assets and can be used to develop interactive plots and dashboards. There is a wide array of intuitive graphs in the library which can be leveraged to develop solutions. It works closely with PyData tools. The library is well-suited for creating customized visuals according to required use-cases. The visuals can also be made interactive to serve a what-if scenario model. All the codes are open source and available on GitHub.

Bokeh is a library that allows you to generate interactive graphics. We can export them to an HTML document that we can share with anyone who has a web browser. It is a very useful

library when we are interested in looking for things in the graphics and we want to be able to zoom in and move around the graphic. Or when we want to share them and give the possibility to explore the data to another person. We start by importing the library and defining the file in which we will save the graph.

Altair

Altair is a declarative statistical visualization library for Python. Altair's API is user-friendly and consistent and built atop Vega-Lite JSON specification. Declarative library indicates that while creating any visuals, we need to define the links between the data columns to the channels (x-axis, y-axis, size, color). With the help of Altair, it is possible to create informative visuals with minimal code. Altair holds a declarative grammar of both visualization and interaction.

Plotly

plotly.py is an interactive, open-source, high-level, declarative, and browser-based visualization library for Python. It holds an array of useful visualization, which includes scientific charts, 3D graphs, statistical charts, financial charts among others. Plotly graphs can be viewed in Jupyter notebooks, standalone HTML files, or hosted online. Plotly library provides options for interaction and editing. The robust API works perfectly in both local and web browser mode.

ggplot

ggplot is a Python implementation of the grammar of graphics. The Grammar of Graphics refers to the mapping of data to aesthetic attributes (colour, shape, size) and geometric objects (points, lines, bars). The basic building blocks according to the grammar of graphics are data, geom (geometric objects), stats (statistical transformations), scale, coordinate system, and facet.Using ggplot in Python allows you to develop informative visualizations incrementally, understanding the nuances of the data first, and then tuning the components to improve the visual representations.