CHEMICAL ENGINEERING
UNIVERSITY of WASHINGTON

Data Science Methods

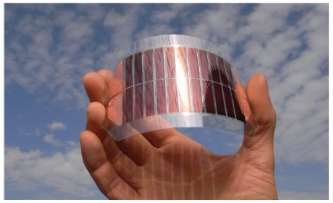
CHEM E 545

Prof. Shachi Mittal
shmittal@uw.edu ; BNS 257

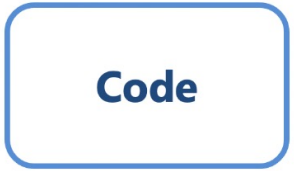
Lecture 4: Data Visualization

Goal: Better prepare you for the emerging opportunities in research and industry

Source: Harvard Clean Energy Project – DOI 10.1021/jz200866s



Data



Useful information

Data Description

- **2.3mil photovoltaic materials**
- Each has a **chemical identity**
- Experimental **measurements**:
  - Open circuit voltage
  - Energy gap
  - Etc.
- Synthesizing them is **expensive**

Interesting tasks

- Can we **predict** solar cell capability without having to synthesize a new material?
- How do the measurements taken **correlate** to each other
- Are there types of materials we should **explore**?

Data Visualization

- Scatter plots: used to observe relationship between variables and uses dots to represent the relationship between them.
- The **scatter()** method in the matplotlib library is used to draw a scatter plot. It takes in the following parameters:
- **x_axis_data-** An array containing x-axis data
- **y_axis_data-** An array containing y-axis data
- **s-** marker size (can be scalar or array of size equal to size of x or y)
- **c-** color of sequence of colors for markers
- marker- marker style
- **cmap-** cmap name
- **linewidths-** width of marker border
- **edgecolor-** marker border color
- **alpha-** blending value, between 0 (transparent) and 1 (opaque)

Data Visualization

- Subplots: pyplot.subplots creates a figure and a grid of subplots with a single call, while providing reasonable control over how the individual plots are created.

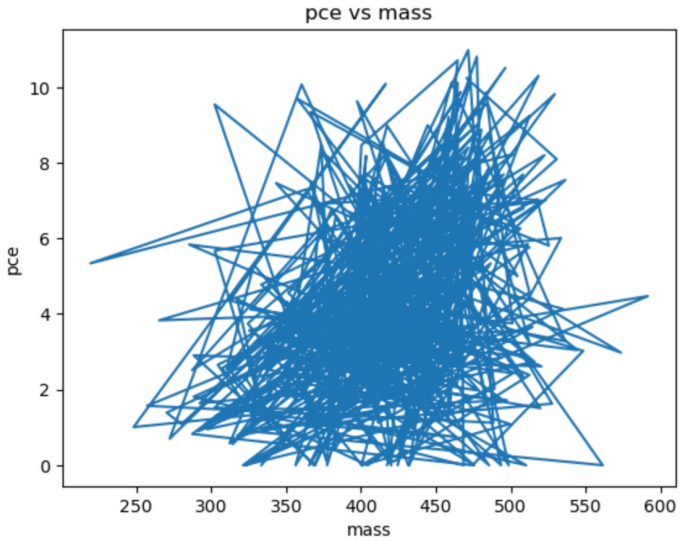
- subplots() without arguments returns a Figure and a single Axes.

- The first two optional arguments of pyplot.subplots define the number of rows and columns of the subplot grid.

Data Visualization

- Line Plot : **matplotlib.pyplot.plot**

Plot y versus x as lines and/or markers.



Parameters:

x, y : ***array-like or scalar***

The horizontal / vertical coordinates of the data points. *x* values are optional and default to `range(len(y))`.

Commonly, these parameters are 1D arrays.

They can also be scalars, or two-dimensional (in that case, the columns represent separate data sets).

These arguments cannot be passed as keywords.

fmt : ***str, optional***

A format string, e.g. 'ro' for red circles. See the *Notes* section for a full description of the format strings.

Format strings are just an abbreviation for quickly setting basic line properties. All of these and more can also be controlled by keyword arguments.

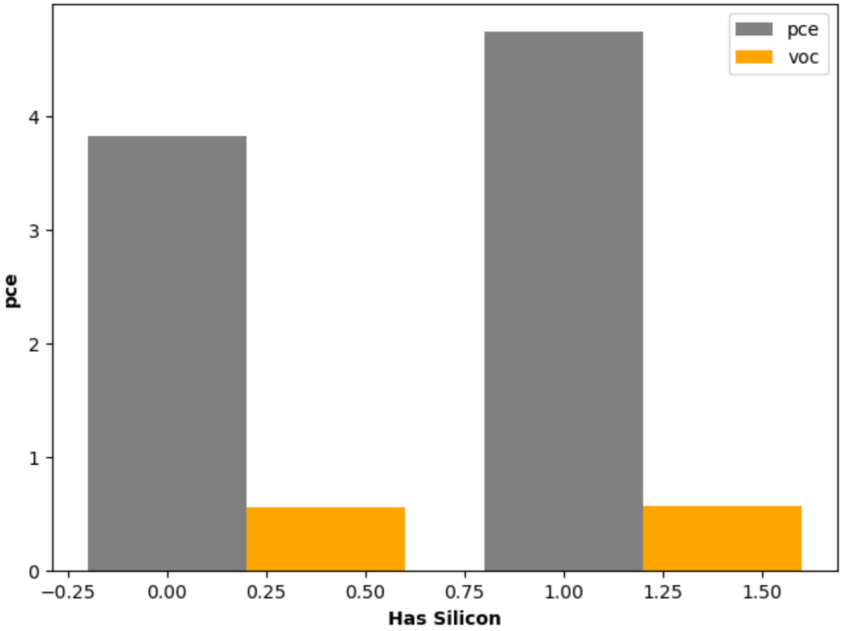
This argument cannot be passed as keyword.

data : ***indexable object, optional***

An object with labelled data. If given, provide the label names to plot in *x* and *y*.

Data Visualization

- Bar Plot : **matplotlib.pyplot.bar**



Parameters:

x : ***float or array-like***

The x coordinates of the bars. See also *align* for the alignment of the bars to the coordinates.

height : ***float or array-like***

The height(s) of the bars.
Note that if *bottom* has units (e.g. datetime), *height* should be in units that are a difference from the value of *bottom* (e.g. timedelta).

width : ***float or array-like, default: 0.8***

The width(s) of the bars.
Note that if *x* has units (e.g. datetime), then *width* should be in units that are a difference (e.g. timedelta) around the *x* values.

bottom : ***float or array-like, default: 0***

The y coordinate(s) of the bottom side(s) of the bars.
Note that if *bottom* has units, then the y-axis will get a Locator and Formatter appropriate for the units (e.g. dates, or categorical).

align : ***{'center', 'edge'}, default: 'center'***

Alignment of the bars to the *x* coordinates:

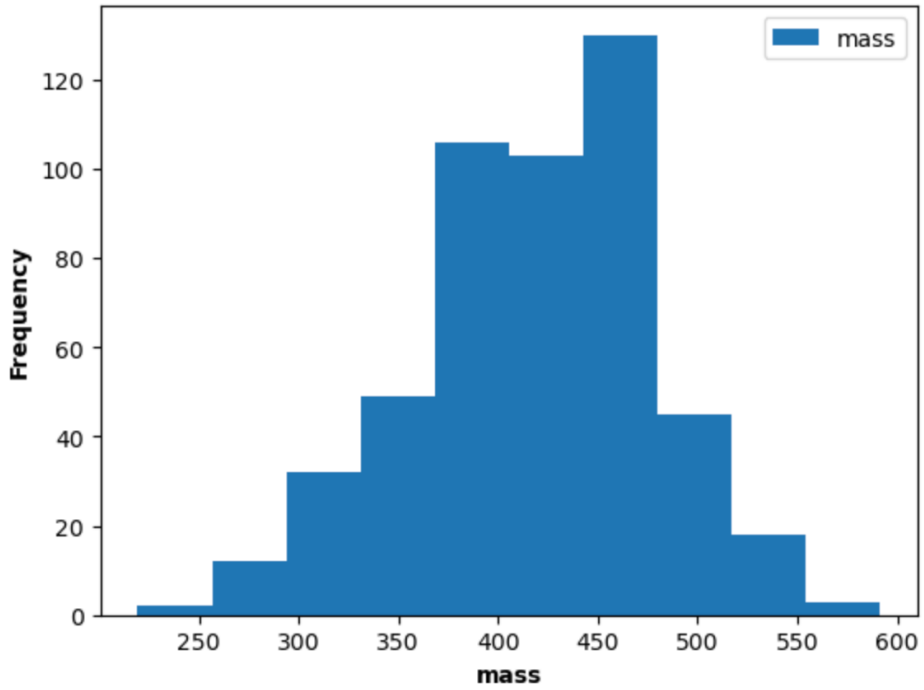
- 'center': Center the base on the *x* positions.
- 'edge': Align the left edges of the bars with the *x* positions.

To align the bars on the right edge pass a negative *width* and `align='edge'`.

Data Visualization

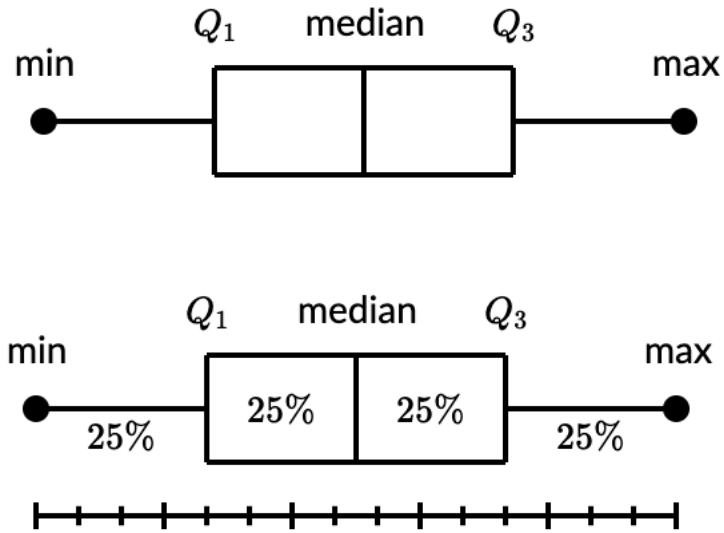
- Histogram : **matplotlib.pyplot.hist**

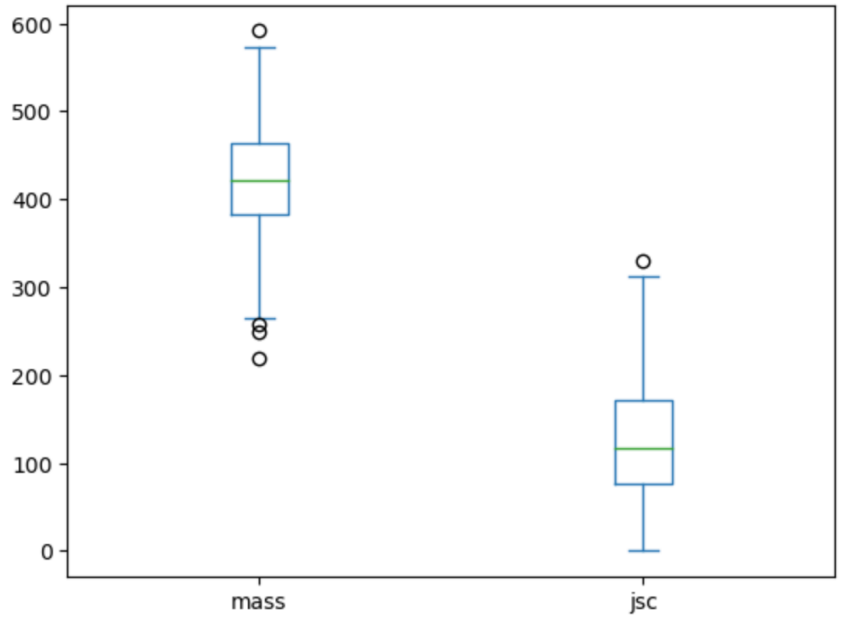
- This method uses numpy.histogram to bin the data in *x* and count the number of values in each bin



Data Visualization

- Box and Whisker Plot : matplotlib.pyplot.boxplot

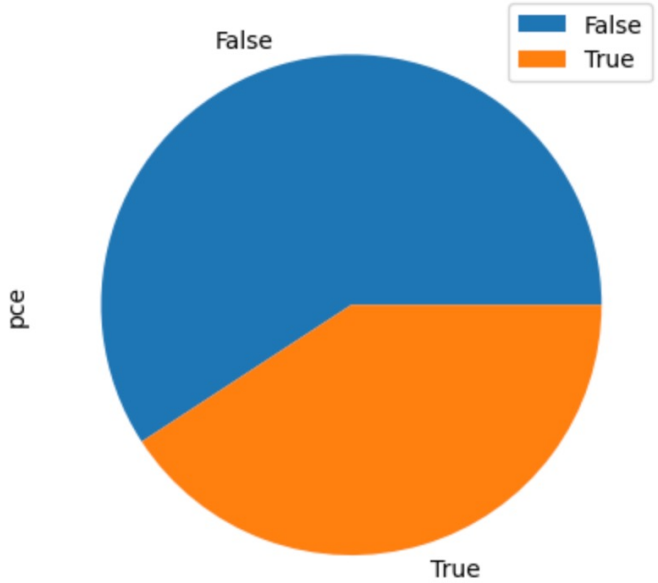




Data Visualization

- Pie Chart: **matplotlib.pyplot.pie**

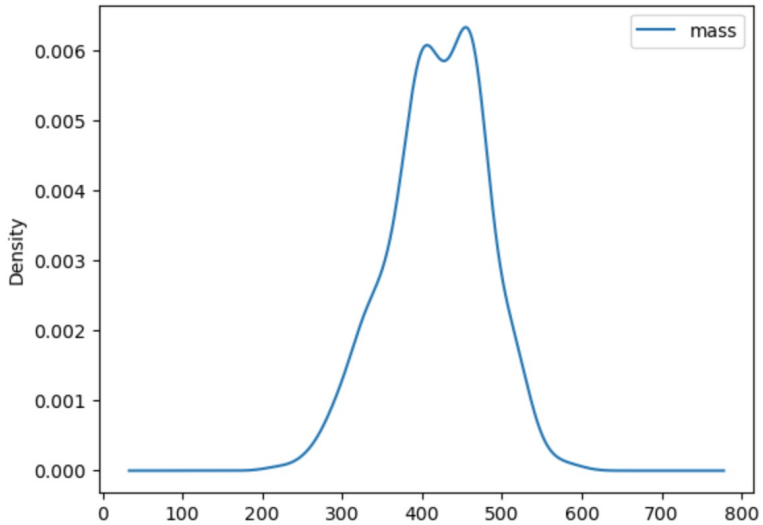
Make a pie chart of array *x*. The fractional area of each wedge is given by x/sum(x). The wedges are plotted counterclockwise, by default starting from the x-axis.



Data Visualization

Density Estimate

- A kernel density estimate (KDE) plot is a method for visualizing the distribution of observations in a dataset, analogous to a histogram. KDE represents the data using a continuous probability density curve in one or more dimensions.
- But it has the potential to introduce distortions if the underlying distribution is bounded or not smooth.

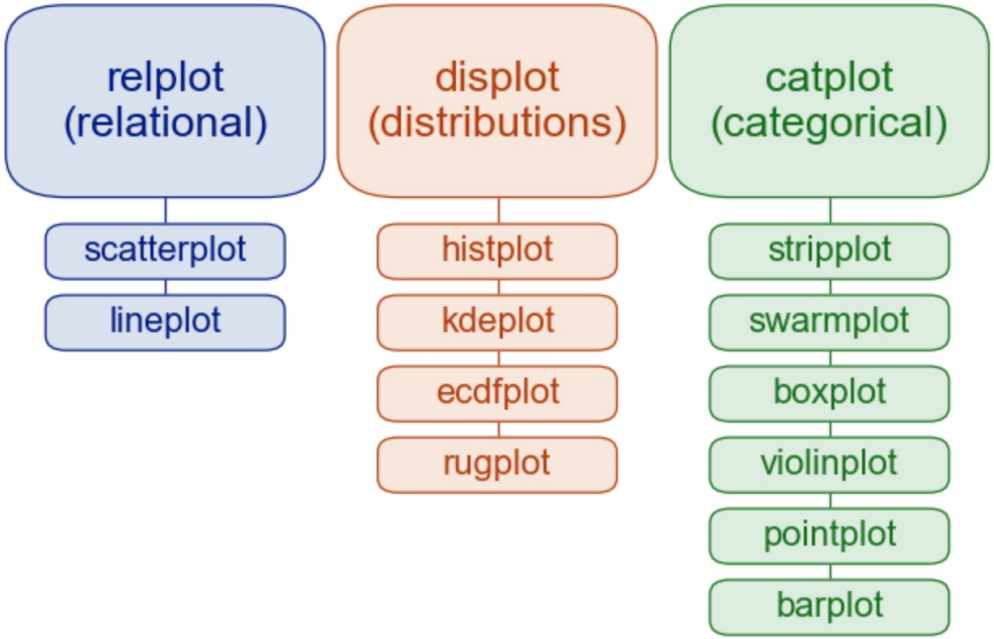


Lets move to JupyterHub

Class Poll

Data Visualization

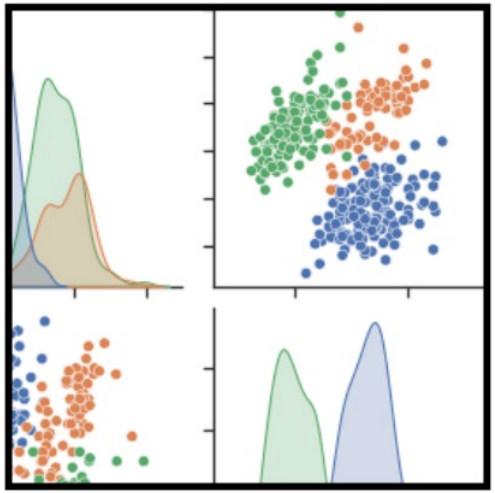
- Seaborn

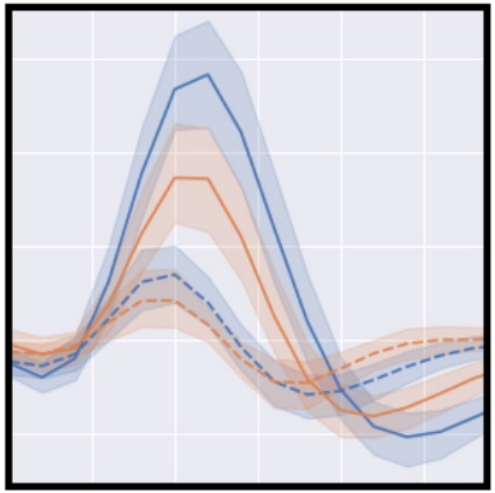


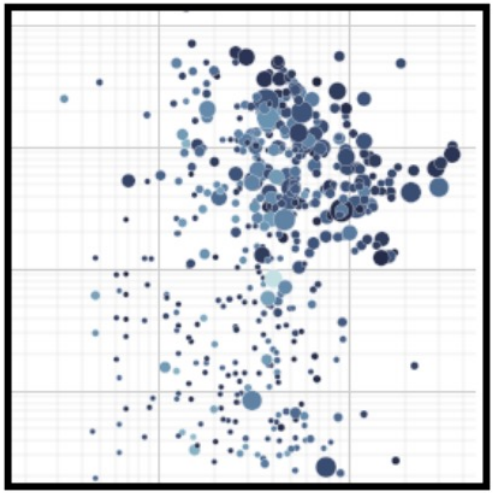
Data Visualization

- Seaborn

Seaborn is a Python data visualization library based on matplotlib. It provides a high-level interface for drawing attractive and informative statistical graphics.







Lets move to JupyterHub