codecademy / PRO

INTENSIVE

Unit 1: Data Visualization with Matplotlib

Function	Description
`plt.plot(x_values, y_values)`	Creates a line. `x_values` and `y_values` are lists of numbers (i.e., `[1, 2, 3, 4`]). Also accepts the following keyword arguments: - `marker` - a symbol that will appear at each (x, y) point. Options include '*' for a start, 'o' for a circle, or 's' for a square. - `linestyle` - whether the line is solid ('-') or dashed ('--' or ':') or no line at all ('') - `linewidth` - a number representing the thickness of the line; default is 1 - `color` - the color of the line (can be a HEX code or any html color name)
`plt.show()`	Displays any previous plot commands
`plt.close('all')`	Closes all previous figures
`plt.figure(figsize=(width, length))`	Creates a new figure with a specified length and width. `width` and `length` are both numbers in inches.
`plt.title('My Chart Title')`	A title for a chart.
`plt.xlabel('My X-Label')`	A label for the x-axis
`plt.ylabel('My Y-Label')`	A label for the y-axis
`plt.subplot(num_rows, num_cols, subplot_index)`	Creates a subplot for a grid with `num_rows` rows and `num_cols` columns. The new subplot is at a position defined by `subplot_index`. For instance, `plt.subplot(2, 3, 4)` would create a grid with 2 rows and 3 columns and would create a plot in the second row and the first column (4 "steps" if moving left to right and top to bottom).

Function	Description
`ax = plt.subplot()`	Creates and axes object (`ax`) that can be used for adjusting the position and labeling of x- and y-axis tick marks.
`plt.legend([‘label1’, ‘label2’])`	Creates a legend using the labels given.
`plt.legend()`	Creates a legend using any `label` keywords that were given in `plt.plot` commands.
`ax.set_xticks([0, 1, 2, 3, 4])`	Creates tick marks at positions `[0, 1, 2, 3, 4]` on the x-axis. Requires that you created an axes object by using `ax = plt.subplot()`.
`ax.set_yticks([0, 1, 2, 3, 4])`	Creates tick marks at positions `[0, 1, 2, 3, 4]` on the y-axis. Requires that you created an axes object by using `ax = plt.subplot()`.
`ax.set_xticklabels([‘label1’, ‘label2’, ‘label3’])`	Modifies the first three labels of the x-axis ticks marks to be `‘label1’, ‘label2’, ‘label3`’ Requires that you created an axes object by using `ax = plt.subplot()`. You’ll probably want to start by specify the positions of your x-ticks using `ax.set_xticks`.
`ax.set_yticklabels([‘label1’, ‘label2’, ‘label3’])`	Modifies the first three labels of the y-axis ticks marks to be `‘label1’, ‘label2’, ‘label3’` Requires that you created an axes object by using `ax = plt.subplot()`. You’ll probably want to start by specify the positions of your y-ticks using `ax.set_yticks`.

Function	Description
`plt.bar(x_values, heights)`	Creates a bar chart with bars at each value of `x_values` using the heights given in `heights`. If you’re only creating one bar chart, `x_values` can be equal to `range(len(heights))`.

`# n: Number of the series`
`# t: Total number of series to plot`
`# d: Number of data points`
`# w: width of each bart (default is 0.8)`
`[t*x + w*n for x in range(d)]`
	Creates x-values for side-by-side bar charts.

`plt.bar(x_values, heights, bottom =other_heights)`	Creates a second bar chart stacked on top of another bar chart whose heights are `other_heights`.
`plt.bar(x_values, heights, yerr=errors, capsize=capsize)`	Creates a bar chart with error bars where each bar’s error height is given by `errors` and the width of each error bar is given by the integer `capsize`.
`plt.hist(dataset, bins=num_bins, range=(min, max))`	Creates a histogram of the dataset `dataset`. The default number of bins is 10, but if the `bins` keyword is given, then the number of bins is `num_bins`. The default `range` is all values of dataset, but if the keyword range is given, only values between `min` and `max` will be used.
`plt.hist(dataset, alpha=0.5)`	Creates a semi-transparent histogram
`plt.hist(dataset, histtype=’step’)`	Creates a histogram with a dark outline and no internal fill color.
`plt.hist(dataset, normed=True)`	Creates a normalized histogram where the sum of the area under the curve is 1. This is useful when graphing two histograms with different sized datasets.

Function	Description
`plt.pie(pie_chart_data)`	Creates a pie chart where each slice of the pie is given by the value of `pie_chart_data` divided by `sum(pie_chart_data)`
`plt.axis(‘equal’)`	Makes a pie chart round and not oblong
`plt.pie(pie_chart_data, labels=[‘label1’, ‘label2’, ‘label3’])`	Creates labels for each slice of the pie chart
`plt.pie(pie_chart_data, autopct=‘%d%%’)`	Labels for each slice of the pie chart with its percentage (i.e., ‘23%’)
`plt.pie(pie_chart_data, autopct=‘%0.2f%%’)`	Labels for each slice of the pie chart with its percentage (i.e., ‘23.05%’)
`plt.fill_between(x_values, lower_y, upper_y)`	Creates a shaded area between the upper and lower bounds given. Usually used to show error around a line plot.