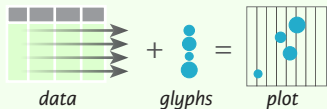# Python For Data Science *Cheat Sheet*
Bokeh

Plotting With Bokeh

The Python interactive visualization library **Bokeh** enables high-performance visual presentation of large datasets in modern web browsers.

Bokeh's mid-level general purpose `bokeh.plotting` interface is centered around two main components: data and glyphs.

data + glyphs = plot

The basic steps to creating plots with the `bokeh.plotting` interface are:

1. Prepare some data:
Python lists, NumPy arrays, Pandas DataFrames and other sequences of values
2. Create a new plot
3. Add renderers for your data, with visual customizations
4. Specify where to generate the output
5. Show or save the results

```
>>> from bokeh.plotting import figure
>>> from bokeh.io import output_file, show
>>> x = [1, 2, 3, 4, 5]                    Step 1
>>> y = [6, 7, 2, 4, 5]
>>> p = figure(title="simple line example",  Step 2
            x_axis_label='x',
            y_axis_label='y')
>>> p.line(x, y, legend="Temp.", line_width=2)  Step 3
>>> output_file("lines.html")              Step 4
>>> show(p)                                Step 5
```

1 Data
Also see Lists, NumPy & Pandas

Under the hood, your data is converted to Column Data Sources. You can also do this manually:

```
>>> import numpy as np
>>> import pandas as pd
>>> df = pd.DataFrame(np.array([[33.9,4,65, 'US'],
                                [32.4,4,66, 'Asia'],
                                [21.4,4,109, 'Europe']]),
                    columns=['mpg','cyl', 'hp', 'origin'],
                    index=['Toyota', 'Fiat', 'Volvo'])
```
```
>>> from bokeh.models import ColumnDataSource
>>> cds_df = ColumnDataSource(df)
```

2 Plotting

```
>>> from bokeh.plotting import figure
>>> p1 = figure(plot_width=300, tools='pan,box_zoom')
>>> p2 = figure(plot_width=300, plot_height=300,
            x_range=(0, 8), y_range=(0, 8))
>>> p3 = figure()
```

3 Renderers & Visual Customizations

Glyphs

Scatter Markers
```
>>> p1.circle(np.array([1,2,3]), np.array([3,2,1]),
            fill_color='white')
>>> p2.square(np.array([1.5,3.5,5.5]), [1,4,3],
            color='blue', size=1)
```
Line Glyphs
```
>>> p1.line([1,2,3,4], [3,4,5,6], line_width=2)
>>> p2.multi_line(pd.DataFrame([[1,2,3],[5,6,7]]),
                pd.DataFrame([[3,4,5],[3,2,1]]),
                color="blue")
```

Rows & Columns Layout

Rows
```
>>> from bokeh.layouts import row
>>> layout = row(p1,p2,p3)
```
Columns
```
>>> from bokeh.layouts import columns
>>> layout = column(p1,p2,p3)
```
Nesting Rows & Columns
```
>>>layout = row(column(p1,p2), p3)
```

Grid Layout
```
>>> from bokeh.layouts import gridplot
>>> row1 = [p1,p2]
>>> row2 = [p3]
>>> layout = gridplot([[p1,p2],[p3]])
```

Tabbed Layout
```
>>> from bokeh.models.widgets import Panel, Tabs
>>> tab1 = Panel(child=p1, title="tab1")
>>> tab2 = Panel(child=p2, title="tab2")
>>> layout = Tabs(tabs=[tab1, tab2])
```

Linked Plots
Also see Data

Linked Axes
```
>>> p2.x_range = p1.x_range
>>> p2.y_range = p1.y_range
```
Linked Brushing
```
>>> p4 = figure(plot_width = 100, tools='box_select,lasso_select')
>>> p4.circle('mpg', 'cyl', source=cds_df)
>>> p5 = figure(plot_width = 200, tools='box_select,lasso_select')
>>> p5.circle('mpg', 'hp', source=cds_df)
>>> layout = row(p4,p5)
```

Customized Glyphs
Also see Data

Selection and Non-Selection Glyphs
```
>>> p = figure(tools='box_select')
>>> p.circle('mpg', 'cyl', source=cds_df,
            selection_color='red',
            nonselection_alpha=0.1)
```
Hover Glyphs
```
>>> hover = HoverTool(tooltips=None, mode='vline')
>>> p3.add_tools(hover)
```
Colormapping
```
>>> color_mapper = CategoricalColorMapper(
            factors=['US', 'Asia', 'Europe'],
            palette=['blue', 'red', 'green'])
>>> p3.circle('mpg', 'cyl', source=cds_df,
            color=dict(field='origin',
            transform=color_mapper),
            legend='Origin'))
```

Legends

Legend Location
Inside Plot Area
```
>>> p.legend.location = 'bottom_left'
```
Outside Plot Area
```
>>> r1 = p2.asterisk(np.array([1,2,3]), np.array([3,2,1])
>>> r2 = p2.line([1,2,3,4], [3,4,5,6])
>>> legend = Legend(items=[("One" , [p1, r1]),("Two" , [r2])], location=(0, -30))
>>> p.add_layout(legend, 'right')
```

Legend Orientation
```
>>> p.legend.orientation = "horizontal"
>>> p.legend.orientation = "vertical"
```

Legend Background & Border
```
>>> p.legend.border_line_color = "navy"
>>> p.legend.background_fill_color = "white"
```

4 Output

Output to HTML File
```
>>> from bokeh.io import output_file, show
>>> output_file('my_bar_chart.html', mode='cdn')
```

Notebook Output
```
>>> from bokeh.io import output_notebook, show
>>> output_notebook()
```

Embedding

Standalone HTML
```
>>> from bokeh.embed import file_html
>>> html = file_html(p, CDN, "my_plot")
```
Components
```
>>> from bokeh.embed import components
>>> script, div = components(p)
```

Statistical Charts With Bokeh
Also see Data

Bokeh's high-level `bokeh.charts` interface is ideal for quickly creating statistical charts

Bar Chart
```
>>> from bokeh.charts import Bar
>>> p = Bar(df, stacked=True, palette=['red','blue'])
```

Box Plot
```
>>> from bokeh.charts import BoxPlot
>>> p = BoxPlot(df, values='vals', label='cyl',
            legend='bottom_right')
```

Histogram
```
>>> from bokeh.charts import Histogram
>>> p = Histogram(df, title='Histogram')
```

Scatter Plot
```
>>> from bokeh.charts import Scatter
>>> p = Scatter(df, x='mpg', y='hp', marker='square',
            xlabel='Miles Per Gallon',
            ylabel='Horsepower')
```

5 Show or Save Your Plots
```
>>> show(p1)            >>> save(p1)
>>> show(layout)        >>> save(layout)
```