IMD0105 - Special Issues in Information Technology VI

Interactive Data Visualization with Bokeh II

Natal-RN
March 2017

Previously on last class (...)

Part I

- Introduction about Bokeh
- Glyphs
- Lines
- Plot (numpy, pandas)
- Column data source

Part II

- Layout
- Rows of plot
- Columns of plot
- Nested plots
- Grid plots
- Tabbed plots
- Links selections
- Legends

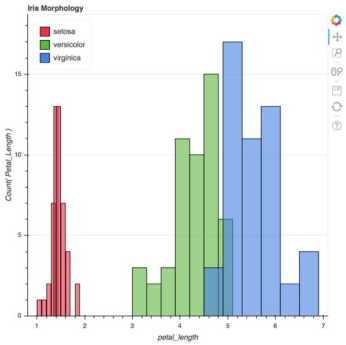
Agenda

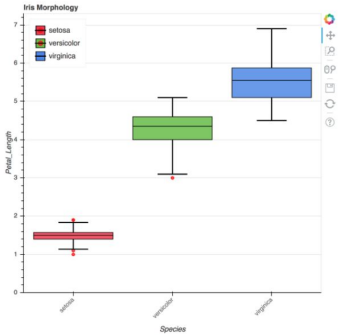
- Part III
  - High level charts
  - Histogram, Box, Scatter
- Part IV
  - Building interactive apps
  - Widgets (button, slider, select, checkbox, radiogroup, so on)

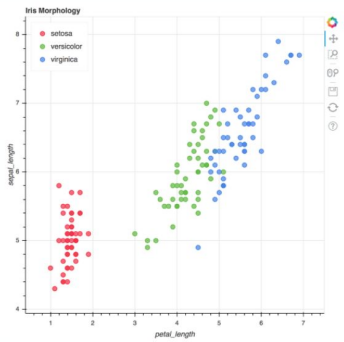
Part III

High Level Chart

In addition to versatile data-driven glyphs, Bokeh comes with a variety of high-level statistical chart types built in, so that you can get quick exploratory charts with very little code.

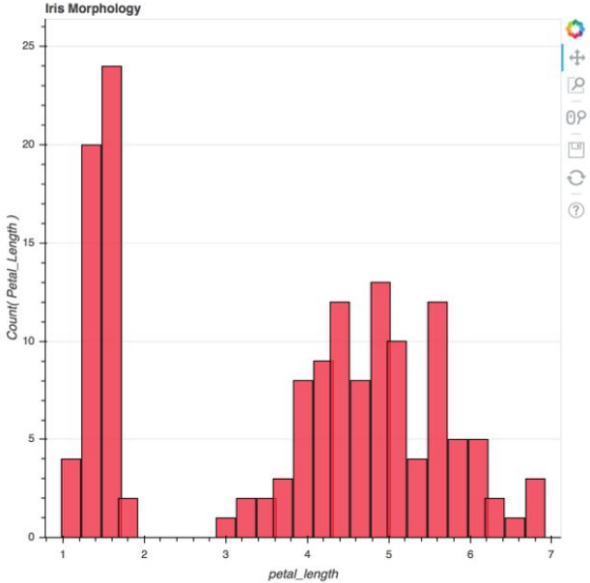






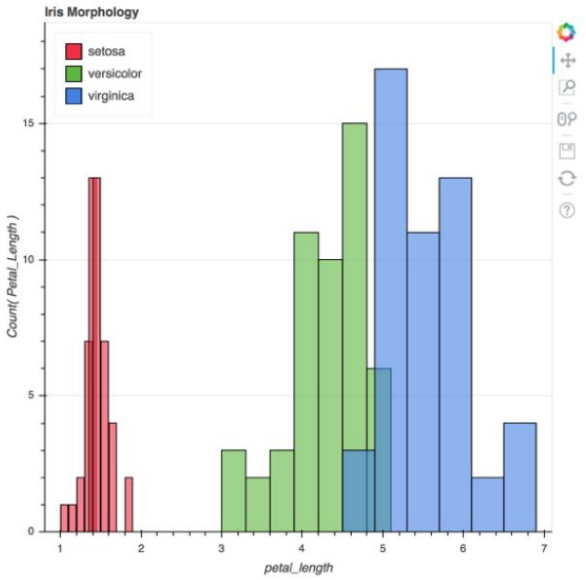
Histogram

```
In [1]: from bokeh.charts import Histogram

In [2]: from bokeh.sampledata.iris import flowers as df

In [3]: p = Histogram(df, 'petal_length', bins=25,
   ...:                 title='Iris Morphology')

In [4]: output_file('histogram.html')

In [5]: show(p)
```



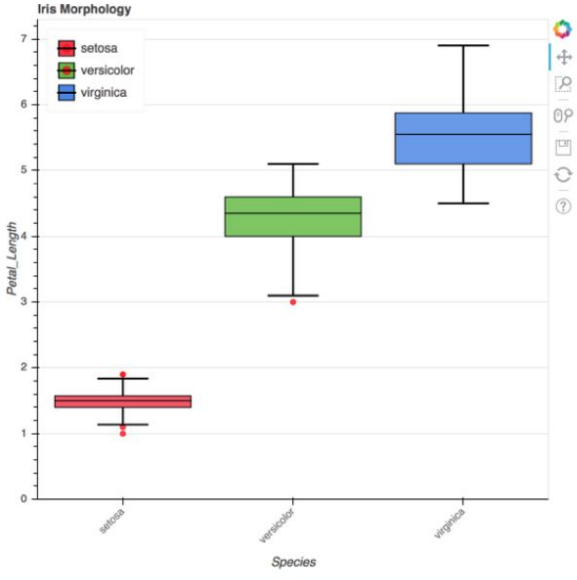
Multiple Histograms

```
In [1]: from bokeh.charts import Histogram

In [2]: from bokeh.sampledata.iris import flowers as df

In [3]: p = Histogram(df,'petal_length',
   ...:                  color='species',
   ...:                  title='Iris Morphology')

In [4]: output_file('histogram.html')

In [5]: show(p)
```



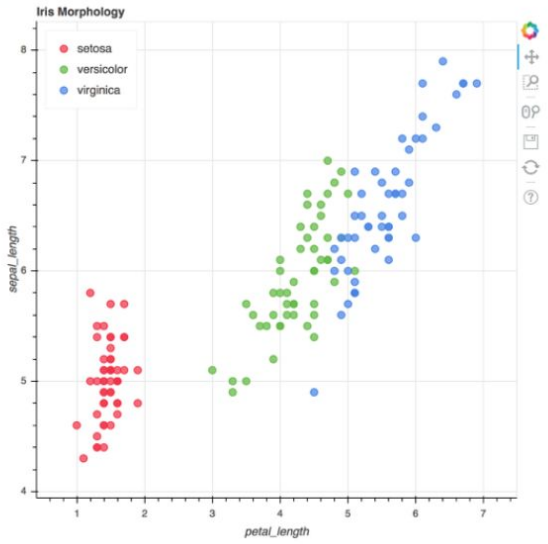
Box Plot

```
In [1]: from bokeh.charts import BoxPlot

In [2]: from bokeh.sampledata.iris import flowers as df

In [3]: p = BoxPlot(df, values='petal_length',
   ...:             label='species',color='species',
   ...:             title='Iris Morphology')

In [4]: output_file('boxplot.html')

In [5]: show(p)
```



Scatter Plot

```
In [1]: from bokeh.charts import Scatter

In [2]: from bokeh.sampledata.iris import flowers as df

In [3]: p = Scatter(df, x='petal_length',
   ...:             y='sepal_length',color='species',
   ...:             title='Iris Morphology')

In [4]: output_file('scatter.html')

In [5]: show(p)
```

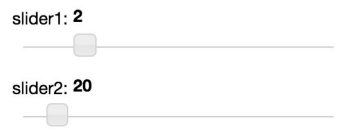


Part IV

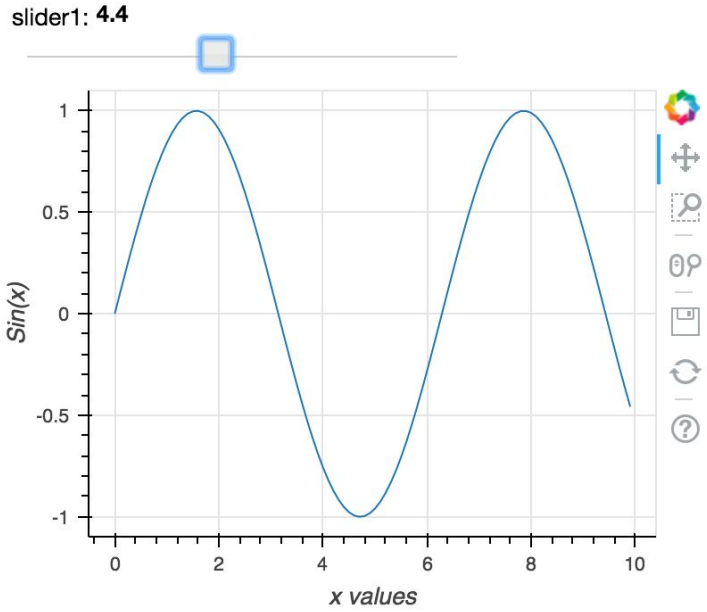
Building interactive apps with Bokeh

Connecting Sliders to Plot

```
# Perform the necessary imports
from bokeh.io import output_notebook, show
from bokeh.layouts import widgetbox
from bokeh.models import Slider

# Create first slider: slider1
slider1 = Slider(title='slider1', start=0, end=10, step=0.1, value=2)

# Create second slider: slider2
slider2 = Slider(title='slider2', start=10, end=100, step=1, value=20)

# Add slider1 and slider2 to a widgetbox
layout = widgetbox(slider1,slider2)

# Call the output_notebook()
output_notebook()
show(layout)
```



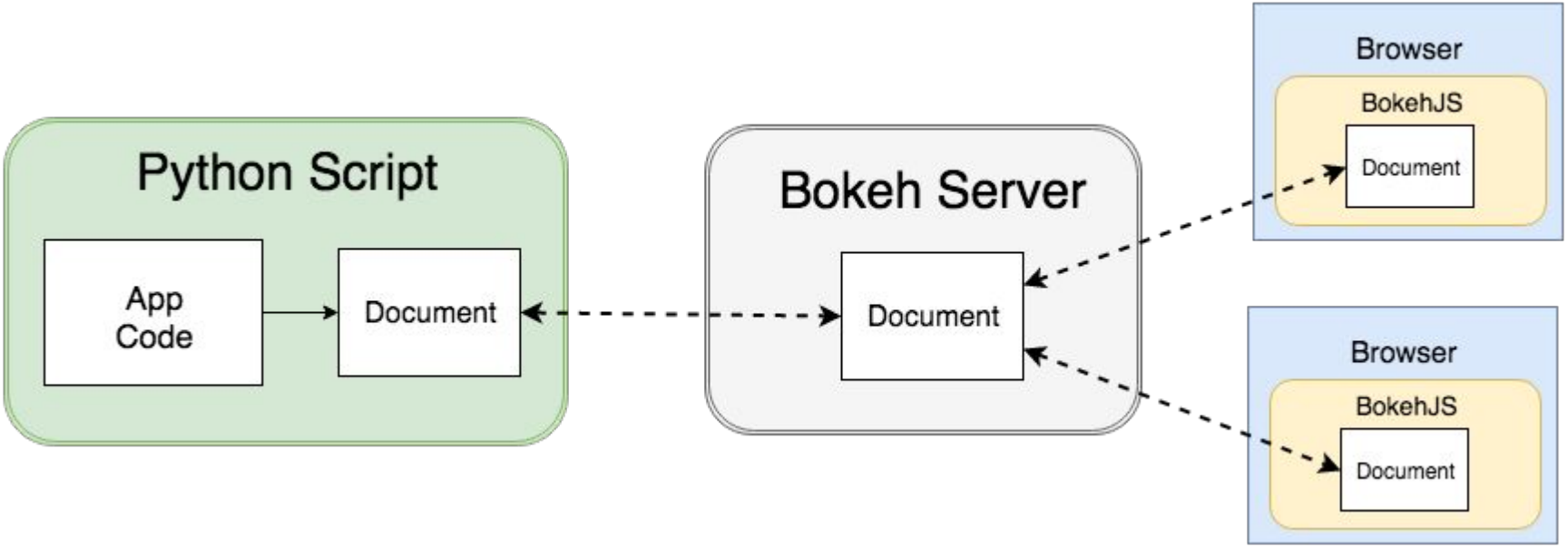
How to combine Bokeh models into layouts




```
# Create a figure
p = figure(x_axis_label='x values', y_axis_label='Sin(x)',
          width=400, height=300)

# Add a line to the plot
p.line('x', 'y', source=source)

# Create first slider: slider1
slider = Slider(title='slider1', start=0, end=10, step=0.1, value=2)

# Create a column layout: layout
layout = column(widgetbox(slider),p)

# Call the output_notebook()
output_notebook()
show(layout)
```

Interactive???

Running Bokeh Applications



Running Bokeh Applications

- Run single module apps at the shell or Windows command prompt:

```
bokeh serve --show myapp.py
```

- “Directory” style apps run similarly:

```
bokeh serve --show myappdir/
```

Writing a Python file from Jupyter

```
%%writefile mybokeh.py
#write/save cell contents into mybokeh.py (use -a to append).

# Import the ColumnDataSource, figure class from bokeh.plotting
from bokeh.plotting import ColumnDataSource, figure

# Import the widgets
from bokeh.layouts import widgetbox, column
from bokeh.models import Slider

# Import numpy
import numpy as np

# Import current document
from bokeh.io import curdoc
```

Learn about callbacks

```
%%writefile -a mybokeh.py
#write/save cell contents into mybokeh.py (use -a to append).


# Define a callback function: callback
def callback(attr, old, new):

    # Read the current value of the slider: scale
    scale = slider.value

    # Compute the updated y using np.cos(scale/x): new_y
    new_y = np.sin(scale/x)

    # Update source with the new data values
    source.data = {'x': x, 'y': new_y}
```

Installing callbacks

```
%%writefile -a mybokeh.py

# Attach the callback to the 'value' property of slider
slider.on_change('value',callback)

# Create layout and add to current document
layout = column(widgetbox(slider), p)
curdoc().add_root(layout)
```

Running mybokeh.py

```
# Open a terminal and execute
# bokeh serve mybokeh.py

# Open the browser and typed
# http://localhost:5006/mybokeh
```

Proof of Concept (PoC)

Embedding a Bokeh server in a Notebook

https://github.com/bokeh/bokeh/blob/0.12.4/examples/howto/server_embed/notebook_embed.ipynb