assignment=26

July 30, 2023

1 Q1. How can you create a Bokeh plot using Python code?

Displaying Bokeh figure in Jupyter notebook is very similar to the above. The only change you need to make is to import output_notebook instead of output_file from bokeh.plotting module.

Call to output_notebook() function sets Jupyter notebook's output cell as the destination for show() function as shown below –

[1]:
```
from bokeh.sampledata.autompg import autompg_clean as autompg_df
automp_df.head()
```

[1]:
```
    mpg  cyl  displ   hp  weight  accel  yr         origin  \
0  18.0    8  307.0  130    3504   12.0  70  North America
1  15.0    8  350.0  165    3693   11.5  70  North America
2  18.0    8  318.0  150    3436   11.0  70  North America
3  16.0    8  304.0  150    3433   12.0  70  North America
4  17.0    8  302.0  140    3449   10.5  70  North America

                        name        mfr
0  chevrolet chevelle malibu  chevrolet
1          buick skylark 320      buick
2         plymouth satellite   plymouth
3              amc rebel sst        amc
4                ford torino       ford
```

[3]:
```
from bokeh.io import show
from bokeh.plotting import figure
import numpy as np

fig = figure(plot_width=600, plot_height=300, title="Sample Line Plot")

fig.line(x=range(10), y=np.random.randint(1,50, 10))

show(fig)
```

[12]:
```
from bokeh.plotting import figure, show
```

[13]:
```
# prepare some data
x = [1, 2, 3, 4, 5]
y = [6, 7, 2, 4, 5]
```

[14]:
```
# create a new plot with a title and axis labels
p = figure(title="Simple line example", x_axis_label='x', y_axis_label='y')
```

[15]:
```
# add a line renderer with legend and line thickness to the plot
p.line(x, y, legend_label="Temp.", line_width=2)
```

[15]:
```
GlyphRenderer(id='1637', …)
```

[16]:
```
# show the results
show(p)
```

[18]:
```
from bokeh.plotting import figure, show

# prepare some data
x = [1, 2, 3, 4, 5]
y = [6, 7, 2, 4, 5]

# create a new plot with a title and axis labels
p = figure(title="Simple line example", x_axis_label="x", y_axis_label="y")

# add a line renderer with legend and line thickness
p.line(x, y, legend_label="Temp.", line_width=2)

# show the results
show(p)
```

[19]:
```
# prepare some data
x = [1, 2, 3, 4, 5]
y1 = [6, 7, 2, 4, 5]
y2 = [2, 3, 4, 5, 6]
y3 = [4, 5, 5, 7, 2]
```

[20]:
```
# create a new plot with a title and axis labels
p = figure(title="Multiple line example", x_axis_label='x', y_axis_label='y')
```

[21]:
```
# add multiple renderers
p.line(x, y1, legend_label="Temp.", color="blue", line_width=2)
p.line(x, y2, legend_label="Rate", color="red", line_width=2)
p.line(x, y3, legend_label="Objects", color="green", line_width=2)
```

[21]:
```
GlyphRenderer(id='2040', …)
```

[22]:
```python
from bokeh.plotting import figure, show

# prepare some data
x = [1, 2, 3, 4, 5]
y1 = [6, 7, 2, 4, 5]
y2 = [2, 3, 4, 5, 6]
y3 = [4, 5, 5, 7, 2]

# create a new plot with a title and axis labels
p = figure(title="Multiple line example", x_axis_label="x", y_axis_label="y")

# add multiple renderers
p.line(x, y1, legend_label="Temp.", color="blue", line_width=2)
p.line(x, y2, legend_label="Rate", color="red", line_width=2)
p.line(x, y3, legend_label="Objects", color="green", line_width=2)

# show the results
show(p)
```

2 Q2. What are glyphs in Bokeh, and how can you add them to a Bokeh plot? Explain with an example.

Any plot is usually made up of one or many geometrical shapes such as line, circle, rectangle, etc. These shapes have visual information about the corresponding set of data. In Bokeh terminology, these geometrical shapes are called gylphs. Bokeh plots constructed using bokeh.plotting interface use a default set of tools and styles. However, it is possible to customize the styles using available plotting tools.

`Types of Plots:`

Different types of plots created using glyphs are as given below —

`Line plot:`

This type of plot is useful for visualizing the movements of points along the x-and y-axes in the form of a line. It is used to perform time series analytics.

`Bar plot:`

This is typically useful for indicating the count of each category of a particular column or field in your dataset.

`Patch plot:`

This plot indicates a region of points in a particular shade of color. This type of plot is used to distinguish different groups within the same dataset.

`scare plot:`

This type of plot is used to visualize relationship between two variables and to indicate the strength of correlation between them.

Different glyph plots are formed by calling appropriate method of Figure class. The Figure object is obtained by following constructor –

[25]:
```
from bokeh.plotting import figure, show
fig = figure()
fig.line(x,y)
show(fig)
```

[26]:
```
from bokeh.plotting import figure, output_file, show
x = [1,2,3,4,5]
y = [2,4,6,8,10]
output_file('line.html')
fig = figure(title = 'Line Plot example', x_axis_label = 'x', y_axis_label =␣
  ↪'y')
fig.line(x,y)
show(fig)
```

[27]:
```
from bokeh.plotting import figure, output_file, show
fig = figure(plot_width = 400, plot_height = 200)
fig.hbar(y = [2,4,6], height = 1, left = 0, right = [1,2,3], color = "Cyan")
output_file('bar.html')
show(fig)
```

[30]:
```
from bokeh.plotting import figure, output_file, show
p = figure(plot_width = 300, plot_height = 300)
p.patch(x = [1, 3,2,4], y = [2,3,5,7], color = "green")
output_file('patch.html')
show(p)
```

patches() This method is used to draw multiple polygonal patches. It needs following arguments –

1 xs The x-coordinates for all the patches, given as a "list of lists". 2 ys The y-coordinates for all the patches, given as a "list of lists". As an example of patches() method, run the following code –

[1]:
```
from bokeh.plotting import figure, output_file, show
xs = [[5,3,4], [2,4,3], [2,3,5,4]]
ys = [[6,4,2], [3,6,7], [2,4,7,8]]
fig = figure()
fig.patches(xs, ys, fill_color = ['red', 'blue', 'black'], line_color = 'white')
output_file('patch_plot.html')
show(fig)
```

varea() Output of the varea() method is a vertical directed area that has one x coordinate array, and two y coordinate arrays, y1 and y2, which will be filled between.

1 x The x-coordinates for the points of the area. 2 y1 The y-coordinates for the points of one side of the area. 3 y2 The y-coordinates for the points of the other side of the area.

```
[3]: from bokeh.plotting import figure, output_file, show
     fig = figure()
     x = [1, 2, 3, 4, 5]
     y1 = [2, 6, 4, 3, 5]
     y2 = [1, 4, 2, 2, 3]
     fig.varea(x = x,y1 = y1,y2 = y2)
     output_file('jupyuter notebook')
     show(fig)
```

```
[2]: from bokeh.plotting import figure, output_file, show
     fig = figure()
     fig.scatter([1, 4, 3, 2, 5], [6, 5, 2, 4, 7], marker = "circle", size = 20,␣
      ↪fill_color = "grey")
     output_file('scatter.html')
     show(fig)
```

```
[3]: from bokeh.plotting import figure, output_file, show
     fig = figure()
     y = [1, 2, 3, 4, 5]
     x1 = [2, 6, 4, 3, 5]
     x2 = [1, 4, 2, 2, 3]
     fig.harea(x1 = x1,x2 = x2,y = y)
     output_file('area.html')
     show(fig)
```

3 Q3. How can you customize the appearance of a Bokeh plot, including the axes, title, and legend?

With Bokeh's themes, you can quickly change the appearance of your plot. Themes are a set of pre-defined design parameters such as colors, fonts, or line styles.

Bokeh comes with five built-in themes: caliber, dark_minimal, light_minimal, night_sky, and contrast. Additionally, you can define your own custom themes.

To use one of the built-in themes, assign the name of the theme you want to use to the theme property of your document:

```
[5]: from bokeh.io import curdoc
     from bokeh.plotting import figure, show

     # prepare some data
     x = [1, 2, 3, 4, 5]
     y = [4, 5, 5, 7, 2]

     # apply theme to current document
     curdoc().theme = "dark_minimal"
```

```
# create a plot
p = figure(sizing_mode="stretch_width", max_width=500, height=250)

# add a renderer
p.line(x, y)

# show the results
show(p)
```

[6]:
```
from bokeh.plotting import figure, show

# prepare some data
x = [1, 2, 3, 4, 5]
y = [4, 5, 5, 7, 2]

# create a new plot with a specific size
p = figure(
    title="Plot sizing example",
    width=350,
    height=250,
    x_axis_label="x",
    y_axis_label="y",
)

# add circle renderer
circle = p.circle(x, y, fill_color="red", size=15)

# show the results
show(p)
```

[8]:
```
from bokeh.plotting import figure, show

# prepare some data
x = [1, 2, 3, 4, 5]
y1 = [4, 5, 5, 7, 2]
y2 = [2, 3, 4, 5, 6]

# create a new plot
p = figure(title="Legend example")

# add circle renderer with legend_label arguments
line = p.line(x, y1, legend_label="Temp.", line_color="blue", line_width=2)
circle = p.circle(
    x,
    y2,
    legend_label="Objects",
    fill_color="red",
    fill_alpha=0.5,
    line_color="blue",
    size=80,
)

# display legend in top left corner (default is top right corner)
p.legend.location = "top_left"

# add a title to your legend
p.legend.title = "Obervations"

# change appearance of legend text
p.legend.label_text_font = "times"
p.legend.label_text_font_style = "italic"
p.legend.label_text_color = "navy"

# change border and background of legend
p.legend.border_line_width = 3
p.legend.border_line_color = "navy"
p.legend.border_line_alpha = 0.8
p.legend.background_fill_color = "navy"
p.legend.background_fill_alpha = 0.2

# show the results
show(p)
```

[9]:
```
from bokeh.plotting import figure, show

# prepare some data
x = [1, 2, 3, 4, 5]
y = [6, 7, 2, 4, 5]

# create new plot
p = figure(title="Headline example")

# add line renderer with a legend
p.line(x, y, legend_label="Temp.", line_width=2)

# change headline location to the left
p.title_location = "left"

# change headline text
p.title.text = "Changing headline text example"

# style the headline
p.title.text_font_size = "25px"
p.title.align = "right"
```

```
p.title.background_fill_color = "darkgrey"
p.title.text_color = "white"

# show the results
show(p)
```

[10]:
```
from bokeh.models import BoxAnnotation
```

[11]:
```
low_box = BoxAnnotation(top=20, fill_alpha=0.2, fill_color="#F0E442")
mid_box = BoxAnnotation(bottom=20, top=80, fill_alpha=0.2, fill_color="#009E73")
high_box = BoxAnnotation(bottom=80, fill_alpha=0.2, fill_color="#F0E442")
```

[12]:
```
p.add_layout(low_box)
p.add_layout(mid_box)
p.add_layout(high_box)
```

[13]:
```
import random

from bokeh.models import BoxAnnotation
from bokeh.plotting import figure, show

# generate some data (1-50 for x, random values for y)
x = list(range(0, 51))
y = random.sample(range(0, 100), 51)

# create new plot
p = figure(title="Box annotation example")

# add line renderer
line = p.line(x, y, line_color="#000000", line_width=2)

# add box annotations
low_box = BoxAnnotation(top=20, fill_alpha=0.2, fill_color="#F0E442")
mid_box = BoxAnnotation(bottom=20, top=80, fill_alpha=0.2, fill_color="#009E73")
high_box = BoxAnnotation(bottom=80, fill_alpha=0.2, fill_color="#F0E442")

# add boxes to existing figure
p.add_layout(low_box)
p.add_layout(mid_box)
p.add_layout(high_box)

# show the results
show(p)
```

4 Q4. What is a Bokeh server, and how can you use it to create interactive plots that can be updated in real time?

Bokeh server makes it easy to create interactive web applications that connect front-end UI events to running Python code.

Bokeh creates high-level Python models, such as plots, ranges, axes, and glyphs, and then converts these objects to JSON to pass them to its client library, BokehJS. For more information on the latter, see Contributing to BokehJS.

This flexible and decoupled design offers some advantages. For instance, it is easy to have other languages, such as R or Scala, drive Bokeh plots and visualizations in the browser.

However, keeping these models in sync between the Python environment and the browser would provide further powerful capabilities:

respond to UI and tool events in the browser with computations or queries using the full power of Python

automatically push server-side updates to the UI elements such as widgets or plots in the browser

use periodic, timeout, and asynchronous callbacks to drive streaming updates

This is where the Bokeh server comes into play:

The primary purpose of the Bokeh server is to synchronize data between the underlying Python environment and the BokehJS library running in the browser.

[14]:
```
# myapp.py

from random import random

from bokeh.layouts import column
from bokeh.models import Button
from bokeh.palettes import RdYlBu3
from bokeh.plotting import figure, curdoc

# create a plot and style its properties
p = figure(x_range=(0, 100), y_range=(0, 100), toolbar_location=None)
p.border_fill_color = 'black'
p.background_fill_color = 'black'
p.outline_line_color = None
p.grid.grid_line_color = None

# add a text renderer to the plot (no data yet)
r = p.text(x=[], y=[], text=[], text_color=[], text_font_size="26px",
           text_baseline="middle", text_align="center")

i = 0

ds = r.data_source
```

```
# create a callback that adds a number in a random location
def callback():
    global i

    # BEST PRACTICE --- update .data in one step with a new dict
    new_data = dict()
    new_data['x'] = ds.data['x'] + [random()*70 + 15]
    new_data['y'] = ds.data['y'] + [random()*70 + 15]
    new_data['text_color'] = ds.data['text_color'] + [RdYlBu3[i%3]]
    new_data['text'] = ds.data['text'] + [str(i)]
    ds.data = new_data

    i = i + 1

# add a button widget and configure with the call back
button = Button(label="Press Me")
button.on_click(callback)

# put the button and plot in a layout and add to the document
curdoc().add_root(column(button, p))
```

Standalone Bokeh server You can have the Bokeh server running on a network for users to interact with your app directly. This can be a simple solution for local network deployment, provided the capabilities of the hardware running the server match your app requirements and the expected number of users.

However, if you have authentication, scaling, or uptime requirements, you'll have to consider more sophisticated deployment configurations.

SSH tunnels To run a standalone instance of the Bokeh server on a host with restricted access, use SSH to "tunnel" to the server.

In the simplest scenario, the user accesses the Bokeh server from another location, such as a laptop with no intermediary machines.

Run the server as usual on the remote host.

bokeh serve

You can configure the Bokeh server to terminate SSL connections and serve secure HTTPS and WSS sessions directly. To do so, you'll have to supply the –ssl-certfile argument with the value of the path to a single PEM file containing a certificate as well as any number of CA certificates needed to establish the certificate's authenticity.

The Bokeh server is scalable by design. If you need more capacity, you can simply run additional servers. In this case, you'll generally want to run all the Bokeh server instances behind a load balancer so that new connections are distributed among individual servers.

../../_images/bokeh_serve_scale.svg The Bokeh server is horizontally scalable. To add more capacity, you can run more servers behind a load balancer.

Nginx can help with load balancing. This section describes some of the basics of one possible configuration, but please also refer to the Nginx load balancer documentation. For instance, there are different strategies available for choosing what server to connect to next.

First, you need to add an upstream stanza to the Nginx configuration. This typically goes above the server stanza and looks something like the following:

Nginx can help with load balancing. This section describes some of the basics of one possible configuration, but please also refer to the Nginx load balancer documentation. For instance, there are different strategies available for choosing what server to connect to next.

hen a Bokeh server receives an HTTP request, it immediately returns a script that initiates a WebSocket connection. All subsequent communication happens over the WebSocket.

To reduce the risk of cross-site misuse, the Bokeh server will only initiate WebSocket connections from the origins that are explicitly allowed. Requests with Origin headers that are not on the allowed list will generate HTTP 403 error responses.

5 Q5. How can you embed a Bokeh plot into a web page or dashboard using Flask or Django?

several options:

file_html: generate the html output, it is not useful for rest.

server_document: It requires a Bokeh server.

components: It returns a js script and a div to include.

autoload_static: It returns a js function and a div to include.

In the django layout, I used:

[27]:
```python
# scatter.py

from bokeh.plotting import figure
from bokeh.models import Range1d
from bokeh.embed import components

# create some data
x1 = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
y1 = [0, 8, 2, 4, 6, 9, 5, 6, 25, 28, 4, 7]
x2 = [2, 5, 7, 15, 18, 19, 25, 28, 9, 10, 4]
y2 = [2, 4, 6, 9, 15, 18, 0, 8, 2, 25, 28]
x3 = [0, 1, 0, 8, 2, 4, 6, 9, 7, 8, 9]
y3 = [0, 8, 4, 6, 9, 15, 18, 19, 19, 25, 28]

# select the tools you want
TOOLS="pan,wheel_zoom,box_zoom,reset,save"

# the red and blue graphs share this data range
```

```python
xr1 = Range1d(start=0, end=30)
yr1 = Range1d(start=0, end=30)

# only the green graph uses this data range
xr2 = Range1d(start=0, end=30)
yr2 = Range1d(start=0, end=30)

# build the figures
p1 = figure(x_range=xr1, y_range=yr1, tools=TOOLS, width=300, height=300)
p1.scatter(x1, y1, size=12, color="red", alpha=0.5)

p2 = figure(x_range=xr1, y_range=yr1, tools=TOOLS, width=300, height=300)
p2.scatter(x2, y2, size=12, color="blue", alpha=0.5)

p3 = figure(x_range=xr2, y_range=yr2, tools=TOOLS, width=300, height=300)
p3.scatter(x3, y3, size=12, color="green", alpha=0.5)

# plots can be a single Bokeh model, a list/tuple, or even a dictionary
plots = {'Red': p1, 'Blue': p2, 'Green': p3}

script, div = components(plots)
print(script)
print(div)
```

```
    <script type="text/javascript">
        (function() {
  const fn = function() {
    Bokeh.safely(function() {
      (function(root) {
        function embed_document(root) {
        const docs_json = '{"f067809a-7a01-4911-91a2-
```

```
5f94709ff551":{"defs":[],"roots":{"references":[{"attributes":{"data":{"x":[0,1,
2,3,4,5,6,7,8,9,10],"y":[0,8,2,4,6,9,5,6,25,28,4,7]},"selected":{"id":"2880"},"s
election_policy":{"id":"2879"}},"id":"2799","type":"ColumnDataSource"},{"attribu
tes":{"coordinates":null,"group":null},"id":"2926","type":"Title"},{"attributes"
:{},"id":"2777","type":"LinearScale"},{"attributes":{},"id":"2878","type":"AllLa
bels"},{"attributes":{"bottom_units":"screen","coordinates":null,"fill_alpha":0.
5,"fill_color":"lightgrey","group":null,"left_units":"screen","level":"overlay",
"line_alpha":1.0,"line_color":"black","line_dash":[4,4],"line_width":2,"right_un
its":"screen","syncable":false,"top_units":"screen"},"id":"2825","type":"BoxAnno
tation"},{"attributes":{},"id":"2846","type":"BasicTicker"},{"attributes":{},"id
":"2788","type":"WheelZoomTool"},{"attributes":{"fill_alpha":{"value":0.2},"fill
_color":{"value":"red"},"hatch_alpha":{"value":0.2},"hatch_color":{"value":"red"
},"line_alpha":{"value":0.2},"line_color":{"value":"red"},"size":{"value":12},"x
":{"field":"x"},"y":{"field":"y"}},"id":"2802","type":"Scatter"},{"attributes":{
},"id":"2775","type":"LinearScale"},{"attributes":{},"id":"2821","type":"WheelZo
omTool"},{"attributes":{},"id":"2900","type":"AllLabels"},{"attributes":{"fill_a
```

lpha":{"value":0.1},"fill_color":{"value":"red"},"hatch_alpha":{"value":0.1},"ha
tch_color":{"value":"red"},"line_alpha":{"value":0.1},"line_color":{"value":"red
"},"size":{"value":12},"x":{"field":"x"},"y":{"field":"y"}},"id":"2801","type":"
Scatter"},{"attributes":{"below":[{"id":"2845"}],"center":[{"id":"2848"},{"id":"
2852"}],"height":300,"left":[{"id":"2849"}],"renderers":[{"id":"2869"}],"title":
{"id":"2926"},"toolbar":{"id":"2859"},"width":300,"x_range":{"id":"2770"},"x_sca
le":{"id":"2841"},"y_range":{"id":"2771"},"y_scale":{"id":"2843"}},"id":"2838","
subtype":"Figure","type":"Plot"},{"attributes":{},"id":"2791","type":"SaveTool"}
,{"attributes":{"coordinates":null,"formatter":{"id":"2874"},"group":null,"major
_label_policy":{"id":"2875"},"ticker":{"id":"2784"}},"id":"2783","type":"LinearA
xis"},{"attributes":{"axis":{"id":"2779"},"coordinates":null,"group":null,"ticke
r":null},"id":"2782","type":"Grid"},{"attributes":{"coordinates":null,"formatter
":{"id":"2932"},"group":null,"major_label_policy":{"id":"2933"},"ticker":{"id":"
2846"}},"id":"2845","type":"LinearAxis"},{"attributes":{"tools":[{"id":"2820"},{
"id":"2821"},{"id":"2822"},{"id":"2823"},{"id":"2824"}]},"id":"2826","type":"Too
lbar"},{"attributes":{"axis":{"id":"2812"},"coordinates":null,"group":null,"tick
er":null},"id":"2815","type":"Grid"},{"attributes":{"below":[{"id":"2812"}],"cen
ter":[{"id":"2815"},{"id":"2819"}],"height":300,"left":[{"id":"2816"}],"renderer
s":[{"id":"2836"}],"title":{"id":"2893"},"toolbar":{"id":"2826"},"width":300,"x_
range":{"id":"2768"},"x_scale":{"id":"2808"},"y_range":{"id":"2769"},"y_scale":{
"id":"2810"}},"id":"2805","subtype":"Figure","type":"Plot"},{"attributes":{},"id
":"2874","type":"BasicTickFormatter"},{"attributes":{},"id":"2841","type":"Linea
rScale"},{"attributes":{"coordinates":null,"data_source":{"id":"2799"},"glyph":{
"id":"2800"},"group":null,"hover_glyph":null,"muted_glyph":{"id":"2802"},"nonsel
ection_glyph":{"id":"2801"},"view":{"id":"2804"}},"id":"2803","type":"GlyphRende
rer"},{"attributes":{"bottom_units":"screen","coordinates":null,"fill_alpha":0.5
,"fill_color":"lightgrey","group":null,"left_units":"screen","level":"overlay","
line_alpha":1.0,"line_color":"black","line_dash":[4,4],"line_width":2,"right_uni
ts":"screen","syncable":false,"top_units":"screen"},"id":"2792","type":"BoxAnnot
ation"},{"attributes":{"axis":{"id":"2783"},"coordinates":null,"dimension":1,"gr
oup":null,"ticker":null},"id":"2786","type":"Grid"},{"attributes":{},"id":"2824"
,"type":"SaveTool"},{"attributes":{},"id":"2902","type":"Selection"},{"attribute
s":{},"id":"2823","type":"ResetTool"},{"attributes":{"axis":{"id":"2845"},"coord
inates":null,"group":null,"ticker":null},"id":"2848","type":"Grid"},{"attributes
":{},"id":"2929","type":"BasicTickFormatter"},{"attributes":{"source":{"id":"283
2"}},"id":"2837","type":"CDSView"},{"attributes":{},"id":"2780","type":"BasicTic
ker"},{"attributes":{"fill_alpha":{"value":0.2},"fill_color":{"value":"blue"},"h
atch_alpha":{"value":0.2},"hatch_color":{"value":"blue"},"line_alpha":{"value":0
.2},"line_color":{"value":"blue"},"size":{"value":12},"x":{"field":"x"},"y":{"fi
eld":"y"}},"id":"2835","type":"Scatter"},{"attributes":{},"id":"2930","type":"Al
lLabels"},{"attributes":{},"id":"2820","type":"PanTool"},{"attributes":{},"id":"
2932","type":"BasicTickFormatter"},{"attributes":{},"id":"2879","type":"UnionRen
derers"},{"attributes":{},"id":"2843","type":"LinearScale"},{"attributes":{},"id
":"2933","type":"AllLabels"},{"attributes":{"coordinates":null,"data_source":{"i
d":"2865"},"glyph":{"id":"2866"},"group":null,"hover_glyph":null,"muted_glyph":{
"id":"2868"},"nonselection_glyph":{"id":"2867"},"view":{"id":"2870"}},"id":"2869
","type":"GlyphRenderer"},{"attributes":{},"id":"2784","type":"BasicTicker"},{"a
ttributes":{"data":{"x":[0,1,0,8,2,4,6,9,7,8,9],"y":[0,8,4,6,9,15,18,19,19,25,28

]},"selected":{"id":"2935"},"selection_policy":{"id":"2934"}},"id":"2865","type"
:"ColumnDataSource"},{"attributes":{},"id":"2899","type":"BasicTickFormatter"},{
"attributes":{},"id":"2787","type":"PanTool"},{"attributes":{},"id":"2877","type
":"BasicTickFormatter"},{"attributes":{},"id":"2934","type":"UnionRenderers"},{"
attributes":{"source":{"id":"2799"}},"id":"2804","type":"CDSView"},{"attributes"
:{"coordinates":null,"data_source":{"id":"2832"},"glyph":{"id":"2833"},"group":n
ull,"hover_glyph":null,"muted_glyph":{"id":"2835"},"nonselection_glyph":{"id":"2
834"},"view":{"id":"2837"}},"id":"2836","type":"GlyphRenderer"},{"attributes":{}
,"id":"2810","type":"LinearScale"},{"attributes":{},"id":"2853","type":"PanTool"
},{"attributes":{"fill_alpha":{"value":0.5},"fill_color":{"value":"green"},"hatc
h_alpha":{"value":0.5},"hatch_color":{"value":"green"},"line_alpha":{"value":0.5
},"line_color":{"value":"green"},"size":{"value":12},"x":{"field":"x"},"y":{"fie
ld":"y"}},"id":"2866","type":"Scatter"},{"attributes":{"source":{"id":"2865"}},"
id":"2870","type":"CDSView"},{"attributes":{},"id":"2897","type":"AllLabels"},{"
attributes":{},"id":"2935","type":"Selection"},{"attributes":{"overlay":{"id":"2
792"}},"id":"2789","type":"BoxZoomTool"},{"attributes":{"fill_alpha":{"value":0.
1},"fill_color":{"value":"green"},"hatch_alpha":{"value":0.1},"hatch_color":{"va
lue":"green"},"line_alpha":{"value":0.1},"line_color":{"value":"green"},"size":{
"value":12},"x":{"field":"x"},"y":{"field":"y"}},"id":"2867","type":"Scatter"},{
"attributes":{"data":{"x":[2,5,7,15,18,19,25,28,9,10,4],"y":[2,4,6,9,15,18,0,8,2
,25,28]},"selected":{"id":"2902"},"selection_policy":{"id":"2901"}},"id":"2832",
"type":"ColumnDataSource"},{"attributes":{},"id":"2808","type":"LinearScale"},{"
attributes":{},"id":"2854","type":"WheelZoomTool"},{"attributes":{},"id":"2856",
"type":"ResetTool"},{"attributes":{"tools":[{"id":"2787"},{"id":"2788"},{"id":"2
789"},{"id":"2790"},{"id":"2791"}]},"id":"2793","type":"Toolbar"},{"attributes":
{"bottom_units":"screen","coordinates":null,"fill_alpha":0.5,"fill_color":"light
grey","group":null,"left_units":"screen","level":"overlay","line_alpha":1.0,"lin
e_color":"black","line_dash":[4,4],"line_width":2,"right_units":"screen","syncab
le":false,"top_units":"screen"},"id":"2858","type":"BoxAnnotation"},{"attributes
":{},"id":"2790","type":"ResetTool"},{"attributes":{"overlay":{"id":"2825"}},"id
":"2822","type":"BoxZoomTool"},{"attributes":{"overlay":{"id":"2858"}},"id":"285
5","type":"BoxZoomTool"},{"attributes":{},"id":"2875","type":"AllLabels"},{"attr
ibutes":{},"id":"2817","type":"BasicTicker"},{"attributes":{"axis":{"id":"2816"}
,"coordinates":null,"dimension":1,"group":null,"ticker":null},"id":"2819","type"
:"Grid"},{"attributes":{},"id":"2857","type":"SaveTool"},{"attributes":{},"id":"
2813","type":"BasicTicker"},{"attributes":{"coordinates":null,"formatter":{"id":
"2929"},"group":null,"major_label_policy":{"id":"2930"},"ticker":{"id":"2850"}},
"id":"2849","type":"LinearAxis"},{"attributes":{"fill_alpha":{"value":0.2},"fill
_color":{"value":"green"},"hatch_alpha":{"value":0.2},"hatch_color":{"value":"gr
een"},"line_alpha":{"value":0.2},"line_color":{"value":"green"},"size":{"value":
12},"x":{"field":"x"},"y":{"field":"y"}},"id":"2868","type":"Scatter"},{"attribu
tes":{"coordinates":null,"group":null},"id":"2871","type":"Title"},{"attributes"
:{"coordinates":null,"formatter":{"id":"2899"},"group":null,"major_label_policy"
:{"id":"2900"},"ticker":{"id":"2813"}},"id":"2812","type":"LinearAxis"},{"attrib
utes":{"fill_alpha":{"value":0.1},"fill_color":{"value":"blue"},"hatch_alpha":{"
value":0.1},"hatch_color":{"value":"blue"},"line_alpha":{"value":0.1},"line_colo
r":{"value":"blue"},"size":{"value":12},"x":{"field":"x"},"y":{"field":"y"}},"id
":"2834","type":"Scatter"},{"attributes":{"coordinates":null,"formatter":{"id":"

```
2877"},"group":null,"major_label_policy":{"id":"2878"},"ticker":{"id":"2780"}},"
id":"2779","type":"LinearAxis"},{"attributes":{"fill_alpha":{"value":0.5},"fill_
color":{"value":"red"},"hatch_alpha":{"value":0.5},"hatch_color":{"value":"red"}
,"line_alpha":{"value":0.5},"line_color":{"value":"red"},"size":{"value":12},"x"
:{"field":"x"},"y":{"field":"y"}},"id":"2800","type":"Scatter"},{"attributes":{"
coordinates":null,"group":null},"id":"2893","type":"Title"},{"attributes":{"belo
w":[{"id":"2779"}],"center":[{"id":"2782"},{"id":"2786"}],"height":300,"left":[{
"id":"2783"}],"renderers":[{"id":"2803"}],"title":{"id":"2871"},"toolbar":{"id":
"2793"},"width":300,"x_range":{"id":"2768"},"x_scale":{"id":"2775"},"y_range":{"
id":"2769"},"y_scale":{"id":"2777"}},"id":"2772","subtype":"Figure","type":"Plot
"},{"attributes":{"end":30},"id":"2770","type":"Range1d"},{"attributes":{"end":3
0},"id":"2768","type":"Range1d"},{"attributes":{"axis":{"id":"2849"},"coordinate
s":null,"dimension":1,"group":null,"ticker":null},"id":"2852","type":"Grid"},{"a
ttributes":{},"id":"2850","type":"BasicTicker"},{"attributes":{"end":30},"id":"2
771","type":"Range1d"},{"attributes":{"end":30},"id":"2769","type":"Range1d"},{"
attributes":{},"id":"2880","type":"Selection"},{"attributes":{"tools":[{"id":"28
53"},{"id":"2854"},{"id":"2855"},{"id":"2856"},{"id":"2857"}]},"id":"2859","type
":"Toolbar"},{"attributes":{"coordinates":null,"formatter":{"id":"2896"},"group"
:null,"major_label_policy":{"id":"2897"},"ticker":{"id":"2817"}},"id":"2816","ty
pe":"LinearAxis"},{"attributes":{"fill_alpha":{"value":0.5},"fill_color":{"value
":"blue"},"hatch_alpha":{"value":0.5},"hatch_color":{"value":"blue"},"line_alpha
":{"value":0.5},"line_color":{"value":"blue"},"size":{"value":12},"x":{"field":"
x"},"y":{"field":"y"}},"id":"2833","type":"Scatter"},{"attributes":{},"id":"2901
","type":"UnionRenderers"},{"attributes":{},"id":"2896","type":"BasicTickFormatt
er"}],"root_ids":["2772","2805","2838"]},"title":"Bokeh
Application","version":"2.4.3"}}';
        const render_items = [{"docid":"f067809a-7a01-4911-91a2-
5f94709ff551","root_ids":["2772","2805","2838"],"roots":{"2772":"218e91b4-4e07-
461a-8dd2-0e2ade944905","2805":"aff1625a-0f37-40ca-b840-
26960eb77976","2838":"c80afb37-32ff-4372-93d2-26beb654b340"}}];
        root.Bokeh.embed.embed_items(docs_json, render_items);
        }
        if (root.Bokeh !== undefined) {
          embed_document(root);
        } else {
          let attempts = 0;
          const timer = setInterval(function(root) {
            if (root.Bokeh !== undefined) {
              clearInterval(timer);
              embed_document(root);
            } else {
              attempts++;
              if (attempts > 100) {
                clearInterval(timer);
                console.log("Bokeh: ERROR: Unable to run BokehJS code because
BokehJS library is missing");
              }
            }
```

```
     }, 10, root)
    }
   })(window);
  });
 };
 if (document.readyState != "loading") fn();
 else document.addEventListener("DOMContentLoaded", fn);
})();
  </script>
```

```
{'Red': '<div class="bk-root" id="218e91b4-4e07-461a-8dd2-0e2ade944905" data-
root-id="2772"></div>', 'Blue': '<div class="bk-root"
id="aff1625a-0f37-40ca-b840-26960eb77976" data-root-id="2805"></div>', 'Green':
'<div class="bk-root" id="c80afb37-32ff-4372-93d2-26beb654b340" data-root-
id="2838"></div>'}
```

```
BokehUserWarning: ColumnDataSource's columns must be of the same length. Current
lengths: ('x', 11), ('y', 12)
```

You can also embed standalone documents with the autoload_static() function. This function provides a

tag that replaces itself with a Bokeh plot. This script also checks for BokehJS and loads it if necessary. This function lets you embed a plot with nothing but this

tag.

This function takes a Bokeh model, such as a plot, that you want to display, a Resources object, and a path to load a script from. Then autoload_static() returns a self-contained

tag and a block of JavaScript code. The JavaScript code saves to the path you provide and the

loads and runs it to display your plot on a web page.

Here is how you might use autoload_static() with a simple plot:

In addition to a single Bokeh model, such as a plot, the components() function can also accept a list or tuple of models or a dictionary of keys and models. Each returns a tuple with one script and a corresponding data structure for the target

elements.

You can also embed standalone documents with the autoload_static() function. This function provides a

tag that replaces itself with a Bokeh plot. This script also checks for BokehJS and loads it if necessary. This function lets you embed a plot with nothing but this

tag.

This function takes a Bokeh model, such as a plot, that you want to display, a Resources object, and a path to load a script from. Then autoload_static() returns a self-contained

tag and a block of JavaScript code. The JavaScript code saves to the path you provide and the

loads and runs it to display your plot on a web page.

[36]:
```
from bokeh.resources import CDN
from bokeh.plotting import figure
from bokeh.embed import autoload_static

plot = figure()
plot.circle([1,2], [3,4])

js, tag = autoload_static(plot, CDN, "some/path")
```

If an application is running on a Bokeh server that makes it available at some URL, you will typically want to embed the entire application in a web page. This way, the page will create a new session and display it to the user every time it loads.

You can achieve this with the server_document() function. This function accepts the URL to a Bokeh server application and returns a script that embeds a new session from that server every time the script executes.

Here is an example of the server_document() function in use:

Bokeh also provides a standard Jinja template that helps you quickly and flexibly embed different document roots by extending the "base" template. This is especially useful when you need to embed individual components of a Bokeh app in a non-Bokeh layout, such as Bootstrap.

[ ]:
```

```