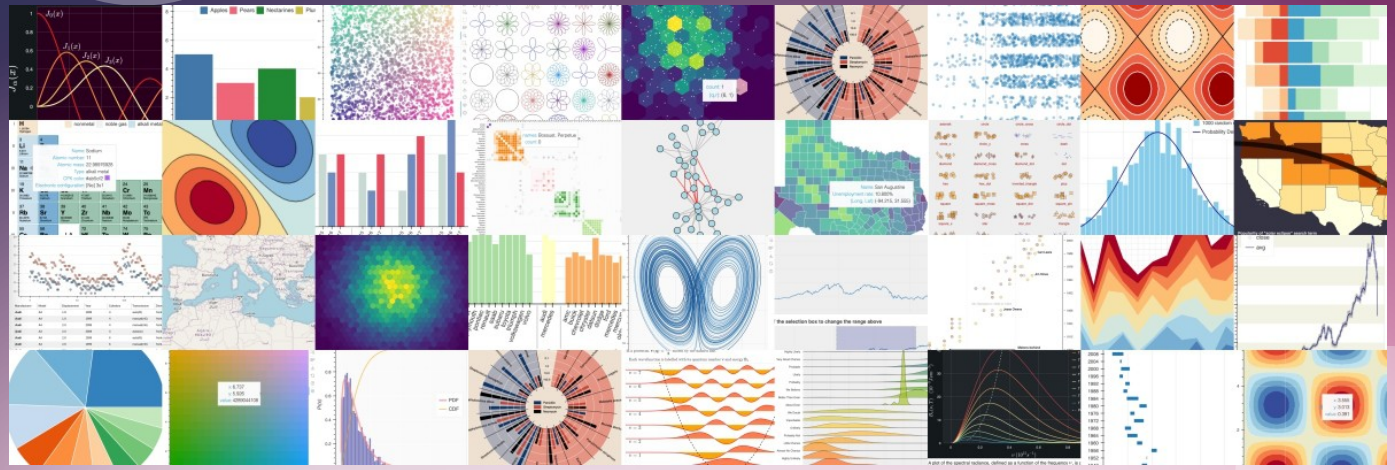
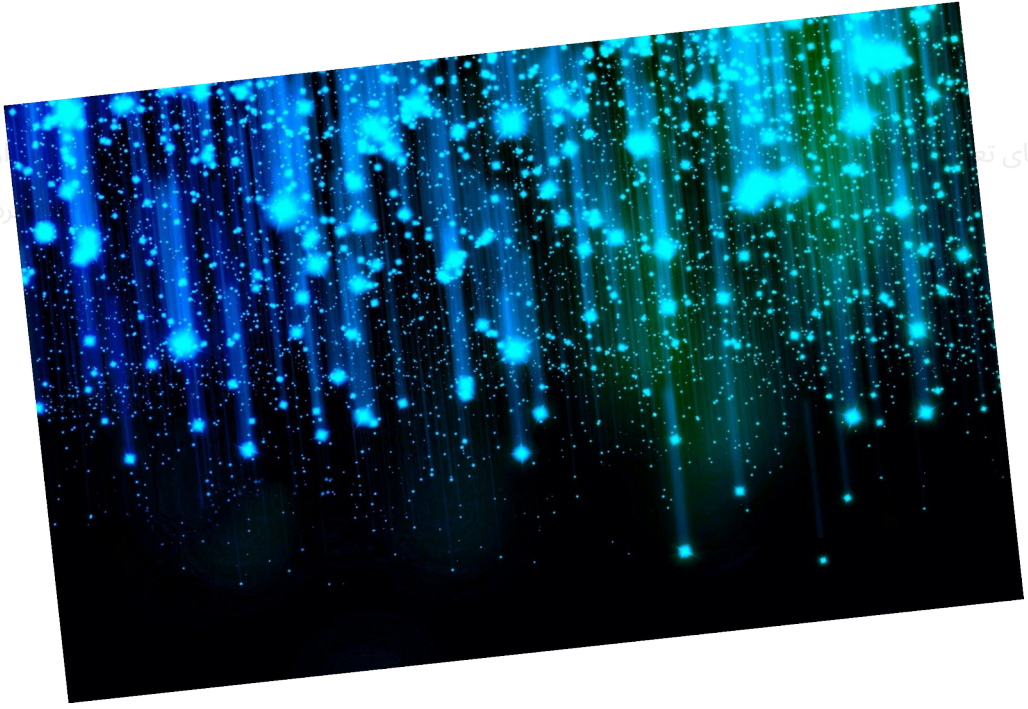
هدف اصلی این کتاب ارائه دانش در مورد استفاده از کتابخانه‌های پایتون برای مصورسازی داده‌ها است.

این فصل توابع مختلف موجود در ماژول Bokeh پایتون را معرفی می‌کند که برای ارائه داده‌ها در نمایش‌های گرافیکی استفاده می‌شوند.



در این فصل، موضوعات زیر را مورد بحث قرار خواهیم داد:



- **مقدمه**
- **وارد کردن Bokeh و مستندات آن**
- **درک نمودارها، اشکال و محورها**
- **پالت‌ها و رنگ‌ها**
- **ایجاد نمودارها**
- **ایجاد نمودارهای تعاملی**
- **ایجاد چندین نمودار**
- **ذخیره نمودار**

* وارد کردن Bokeh و مستندات آن

برای دسترسی به مستندات Bokeh، می‌توانید به وب‌سایت Bokeh به آدرس زیر مراجعه کنید:

https://docs.bokeh.org/en/latest/index.html

یا می‌توانید از تابع help در پایتون برای نمایش مستندات یک تابع یا ماژول خاص استفاده کنید. برای مثال، برای نمایش مستندات ماژول plotting در Bokeh، می‌توانید از دستور زیر استفاده کنید:

```
help("bokeh.plotting")
```

این دستور مستندات تابع figure را در کنسول نمایش می‌دهد، که شامل توضیحی از تمامی توابع، پارامترها و مقدار بازگشتی آن‌ها، به همراه مثال‌هایی از نحوه استفاده است.

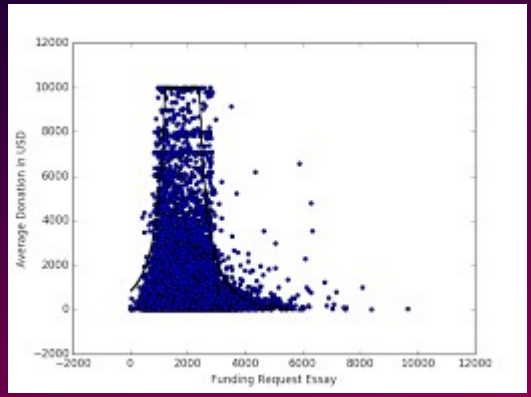
برای مبتدیان، مستندات موجود در وب‌سایت رسمی Bokeh را بررسی و به آن مراجعه کنید:

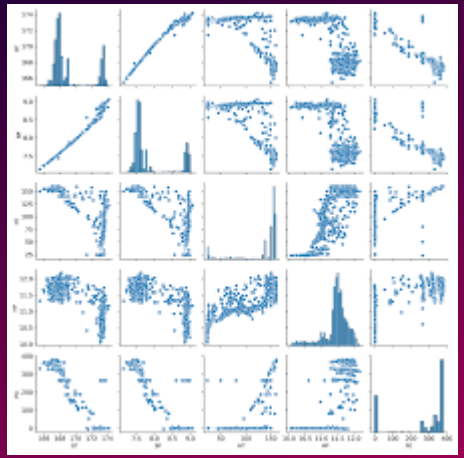
[https://docs.bokeh.org/en/latest/docs/first_steps.html](https://docs.bokeh.org/en/latest/docs/)

.first_steps.html

درک نمودارها، اشکال، محورها و پالت‌ها

Bokeh یک کتابخانه مصورسازی داده‌ها است که ابزارهایی برای ایجاد نمودارهای تعاملی، تجسم‌ها و برنامه‌ها در مرورگرهای وب ارائه می‌دهد. سه مؤلفه اصلی یک نمودار Bokeh عبارتند از نمودار (Figure)، اشکال (glyphs) و محورها (axes).

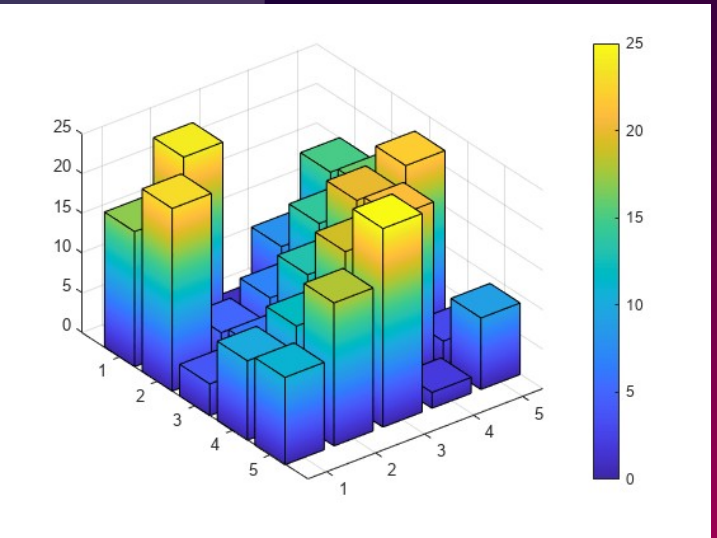


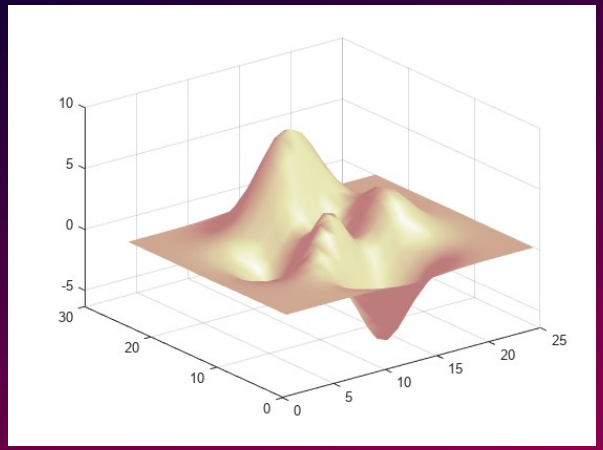


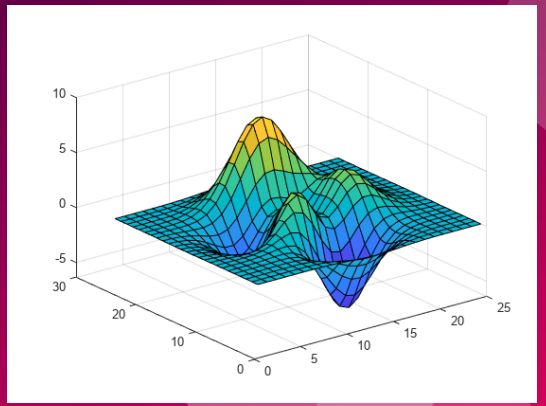
نمودار (Figure)

نمودار (Figure) شیء مرکزی در یک نمودار Bokeh است. این شیء یک بوم (canvas) فراهم می‌کند که روی آن اشکال (glyphs) کشیده می‌شوند و همچنین طرح و قالب کلی نمودار را تعیین می‌کند. شما می‌توانید با فراخوانی تابع figure در ماژول bokeh.plotting، یک شیء نمودار جدید ایجاد کنید. تابع figure تعدادی آرگومان می‌پذیرد که به شما اجازه می‌دهد ظاهر نمودار را سفارشی‌سازی کنید، از جمله اندازه، رنگ پس‌زمینه و خصوصیات فونت.

تابع figure() در Bokeh یک بوم یا نمودار جدید ایجاد می‌کند. این تابع تعدادی آرگومان می‌پذیرد که به شما اجازه می‌دهد ظاهر نمودار را سفارشی‌سازی کنید، مانند اندازه و رنگ پس‌زمینه نمودار، محدوده مقادیر برای محورهای x و y، و ظاهر برچسب‌های محور x و y.







در اینجا فهرستی از رایج‌ترین پارامترهای تابع figure() در Bokeh آمده است:

plot_width و plot_height: عرض و ارتفاع نمودار، به پیکسل. مقادیر پیش‌فرض به ترتیب 600 و 400 هستند.

title: عنوان نمودار. می‌توانید این پارامتر را به یک مقدار رشته‌ای تنظیم کنید.

x_axis_label و y_axis_label: برچسب‌های محورهای x و y به ترتیب.

x_range و y_range: محدوده مقادیری که روی محورهای x و y نمایش داده می‌شوند، به ترتیب. این مقادیر می‌توانند به صورت یک تپِل یا لیست مشخص شوند، با مقدار اول که حداقل مقدار و مقدار دوم که حداکثر مقدار را نشان می‌دهد. اگر مشخص نشود، محدوده به صورت خودکار بر اساس داده‌ها تعیین می‌شود.

x_axis_type و y_axis_type: نوع مقیاسی که برای محورهای x و y استفاده می‌شود، به ترتیب. مقدار پیش‌فرض "auto" است، اما شما می‌توانید از "log"، "linear" یا "datetime" نیز استفاده کنید.

background_fill_color و border_fill_color: رنگ‌هایی که برای پس‌زمینه و حاشیه نمودار استفاده می‌شوند، به ترتیب.

tools: لیستی از ابزارهایی که در نمودار گنجانده می‌شوند. مقدار پیش‌فرض "pan,box_zoom,wheel_zoom,reset,save" است، اما شما می‌توانید ابزارهای دیگری مانند "hover" یا "lasso_select" را نیز اضافه کنید.

toolbar_location: مکان نوار ابزار، که شامل ابزارهای نمودار است. مقدار پیش‌فرض "above" است، اما شما می‌توانید از "left"، "below" یا "right" نیز استفاده کنید.

حلقه (Annulus): یک شکل حلقه‌ای با شعاع داخلی و خارجی. اغلب برای نمایش یک نقطه داده با یک مقدار داخلی و خارجی مشخص استفاده می‌شود.

قوس (Arc): بخشی از یک دایره یا حلقه، که توسط زاویه شروع، زاویه پایان و شعاع مشخص می‌شود. برای ایجاد نمودارهای پای (Pie) یا تجسم‌های دایره‌ای دیگر مفید است.

دایره (Circle): یک شکل ساده دایره‌ای، که اغلب برای نمایش نقاط داده فردی استفاده می‌شود.

بیضی (Ellipse): یک شکل بیضی، که اغلب برای نمایش یک نقطه داده با یک مقدار x و y استفاده می‌شود.

تصویر (Image): یک شکل تصویری مستطیلی، که برای نمایش تصاویر در یک نمودار استفاده می‌شود.

خط (Line): یک شکل خط، که برای نمایش یک خط یا منحنی متصل‌کننده نقاط داده استفاده می‌شود.

چندخط (MultiLine): یک شکل چندخطی، که برای نمایش چندین خط یا منحنی متصل‌کننده مجموعه‌های مختلف نقاط داده استفاده می‌شود.

وصله (Patch): یک شکل چندضلعی پر شده، که برای نمایش نقاط داده با مقادیر x و y متعدد استفاده می‌شود.

چهارگوش (Quad): یک شکل مستطیلی، که برای نمایش نقاط داده با یک مقدار x و y ثابت، و یک عرض و ارتفاع استفاده می‌شود.

پرتو (Ray): یک شکل خطی که از یک نقطه داده شده شروع می‌شود و به طور نامحدود در یک جهت مشخص گسترش می‌یابد. برای ایجاد تجسم‌های جهت‌دار مفید است.

مستطیل (Rect): یک شکل مستطیلی با عرض و ارتفاع ثابت، که اغلب برای نمایش نقاط داده فردی استفاده می‌شود.

قطعه (Segment): یک شکل خطی که دو نقطه را متصل می‌کند، که اغلب برای ایجاد خطوط یا منحنی‌های قطعه‌قطعه‌ای استفاده می‌شود.

متن (Text): یک شکل متنی، که برای اضافه کردن متن به یک نمودار استفاده می‌شود.

مثلث (Triangle): یک شکل مثلثی، که برای نمایش نقاط داده با سه مقدار استفاده می‌شود.

بخش (Wedge): بخشی از یک شکل دایره‌ای یا حلقه‌ای، که توسط زاویه شروع، زاویه پایان و شعاع مشخص می‌شود. مشابه شکل قوس (Arc)، اما با شعاع داخلی ثابت و رنگ پر شده.

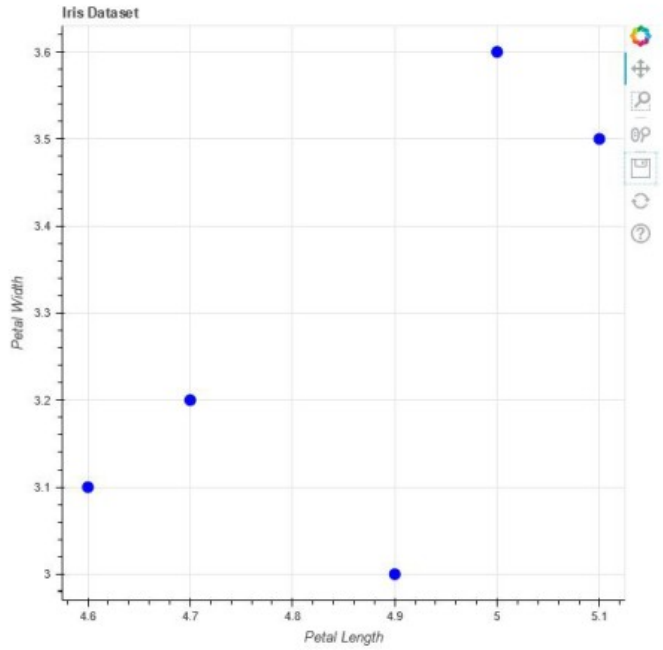
این اشکال را می‌توان با خصوصیات بصری مختلف مانند رنگ، اندازه، آلفا (شفافیت)، عرض خط، رنگ پر و بیشتر سفارشی‌سازی کرد تا تجسم‌های غنی و آموزنده‌ای ایجاد کرد.

در اینجا یک مثال از هر نوع شکل در Bokeh، با استفاده از مجموعه داده Iris آمده است:

```
import bokeh
from bokeh.plotting import figure, show
from bokeh.sampledata.iris import flowers
# Create a ColumnDataSource from the Iris dataset
source = bokeh.plotting.ColumnDataSource(flowers)
# Create a figure with a title and x/y axis labels
p = figure(title="Iris Dataset", x_axis_label="Petal Length", y_axis_label="Petal Width")
# Add glyphs for each type of glyph to the first five rows of the iris dataset
p.circle(x=flowers["sepal_length"][:5], y=flowers["sepal_width"][:5], size=10, color="blue")
# Show the plot
show(p)
p.square(x=flowers["sepal_length"][:5], y=flowers["sepal_width"][:5], size=10, color="red")
# Show the plot
show(p)
p.triangle(x=flowers["sepal_length"][:5], y=flowers["sepal_width"][:5], size=10,
color="green")
# Show the plot
show(p)
p.diamond(x=flowers["sepal_length"][:5], y=flowers["sepal_width"][:5], size=10,
color="purple")
# Show the plot
```

```
show(p)

p.cross(x=flowers["sepal_length"][:5], y=flowers["sepal_width"][:5], size=10,
color="orange")

# Show the plot
show(p)
```

Output of circle glyph:



این کد یک نمودار واحد با انواع مختلف نمادها ایجاد می‌کند و از ویژگی‌های بصری متفاوت برای هر کدام استفاده می‌کند. نمودارها در مرورگر وب باز می‌شوند. توجه داشته باشید که همه نمادها برای همه داده‌ها مناسب نیستند و انتخاب نماد درست اهمیت دارد.

محورها

محورها مقیاس‌ها و برچسب‌های محورهای x و y نمودار را فراهم می‌کنند. شما می‌توانید با استفاده از روش‌های مختلفی مانند xaxis و yaxis به نمودار محور اضافه کنید و ظاهر آن‌ها را با استفاده از ویژگی‌های مختلفی مانند اندازه فونت، برچسب‌های تیک و مکان‌های علامت تیک، سفارشی کنید. Bokeh همچنین ابزارهای اضافی برای تعامل با محورها، مانند زوم، جابجایی و انتخاب نقاط داده فراهم می‌کند. در اینجا چند نمونه از نحوه سفارشی‌سازی ظاهر محورها در Bokeh آورده شده است:

تغییر محدوده محور می‌توانید محدوده یک محور را با استفاده از ویژگی‌های x_range و y_range شیء Figure تنظیم کنید. برای مثال، برای تنظیم محدوده محور x از 0 تا 10:

```
p = figure(x_range=(0, 10), ...)
```

تنظیم علامت‌ها و برچسب‌های تیک می‌توانید علامت‌ها و برچسب‌های تیک روی یک محور را با استفاده از ویژگی‌هایی مانند major_tick_in، major_tick_out، minor_tick_in، minor_tick_out و ویژگی‌های مشابه کنترل کنید. برای مثال، برای تغییر اندازه فونت برچسب‌های محور x:

```
p.xaxis.major_label_text_font_size = "16pt"
```

تغییر رنگ و عرض خط محور

می‌توانید ظاهر خط محور را با استفاده از ویژگی‌های axis_line_color و axis_line_width سفارشی کنید. برای مثال، برای تغییر رنگ خط محور y به قرمز و افزایش عرض آن:

```
p.yaxis.axis_line_color = "red"
p.yaxis.axis_line_width = 3
```

قالب‌بندی برچسب‌های تیک

می‌توانید برچسب‌های تیک روی یک محور را با استفاده از ویژگی formatter قالب‌بندی کنید. به عنوان مثال، برای قالب‌بندی برچسب‌های تیک محور y به صورت درصد:

```
from bokeh.models import NumeralTickFormatter
p.yaxis.formatter = NumeralTickFormatter(format="0%")
```

این‌ها تنها چند نمونه از نحوه سفارشی‌سازی ظاهر محورها در Bokeh هستند. ویژگی‌ها و گزینه‌های بسیاری دیگری نیز موجود هستند، بنابراین توصیه می‌کنم برای اطلاعات بیشتر به مستندات Bokeh مراجعه کنید.

به طور خلاصه، Figure ساختار و چیدمان کلی نمودار را فراهم می‌کند، نمادها (glyphs) نمایش بصری داده‌ها را ارائه می‌دهند و محورها مقیاس‌ها و برچسب‌های محورهای x و y را تأمین می‌کنند. این اجزاء به شما اجازه می‌دهند تا در Bokeh نمودارهای تعاملی و غنی ایجاد کنید.

پالت‌ها و رنگ‌ها

Bokeh یک کتابخانه قدرتمند پایتون برای ایجاد نمودارهای تعاملی است. این کتابخانه روش‌های مختلفی برای کنترل پالت‌ها و رنگ‌ها در نمودارها ارائه می‌دهد. شما می‌توانید پالت‌ها و رنگ‌ها را در نمودارها با استفاده از پارامتر palette و ویژگی color کنترل کنید.

پالت‌ها

ماژول bokeh.palettes بخشی از کتابخانه پایتون Bokeh است که برای ایجاد تجسم‌های تعاملی در مرورگرهای وب استفاده می‌شود. این ماژول مجموعه‌ای از پالت‌های رنگی را شامل می‌شود که می‌توانند برای ایجاد نمودارها و چارت‌های جذاب بصری استفاده شوند.

یک پالت رنگ مجموعه‌ای از رنگ‌هاست که به گونه‌ای طراحی شده‌اند که به خوبی با هم کار کنند. Bokeh تعدادی پالت از پیش تعریف‌شده فراهم می‌کند که می‌توان به صورت مستقیم از آنها استفاده کرد یا به عنوان نقطه شروعی برای ایجاد پالت‌های سفارشی از آنها بهره برد.

در اینجا یک مرور کلی از برخی پالت‌های پرکاربرد در ماژول bokeh.palettes آورده شده است:

- Category10 : پالت ۱۰ رنگ که می‌تواند برای داده‌های دسته‌ای استفاده شود.
- Category20 : پالت ۲۰ رنگ که می‌تواند برای داده‌های دسته‌ای استفاده شود.
- Viridis : پالت رنگ‌هایی که به گونه‌ای طراحی شده‌اند که از لحاظ بصری یکنواخت باشند و برای داده‌های پیوسته به خوبی کار کنند.
- Inferno : پالت رنگ‌هایی که مشابه Viridis است اما دارای طرح رنگی متفاوتی است.
- Magma : پالت رنگ‌هایی که مشابه Inferno است اما دارای طرح رنگی متفاوتی است.
- Plasma : پالت رنگ‌هایی که مشابه Inferno و Magma است اما دارای طرح رنگی متفاوتی است.

توربو

پالتی از رنگ‌ها که مشابه پالت Viridis است، اما با کنتراست بالاتر و دید بهتر برای تفاوت‌های کوچک.

برای استفاده از یک پالت، می‌توانید تابع پالت را با تعداد رنگ‌هایی که می‌خواهید در پالت باشد، به عنوان آرگومان فراخوانی کنید. به عنوان مثال، برای ایجاد یک پالت با ۵ رنگ با استفاده از پالت Viridis، می‌توانید از کد زیر استفاده کنید:

```
from bokeh.palettes import Viridis
my_palette = Viridis[5]
```

این کد یک لیست از ۵ رنگ ایجاد می‌کند که می‌توانید در نمودار خود استفاده کنید. همچنین می‌توانید از توابع plasma()، magma()، inferno() و turbo() برای تولید پالت‌ها به صورت مستقیم استفاده کنید.

علاوه بر این پالت‌های از پیش تعریف شده، می‌توانید پالت‌های سفارشی نیز با ترکیب رنگ‌ها یا انجام اینترپولیشن بین رنگ‌ها ایجاد کنید. توابع linear_palette() و log_palette() می‌توانند برای تولید پالت‌های رنگی خطی و لگاریتمی به ترتیب استفاده شوند. کلاس Color می‌تواند برای تعریف رنگ‌های سفارشی به روش‌های مختلف، از جمله مشخص کردن مقادیر RGB یا HSL، استفاده شود.

در کل، ماژول bokeh.palettes مجموعه‌ای از ابزارهای مفید برای ایجاد پالت‌های رنگی فراهم می‌کند که می‌توانند در انواع نمودارها و تجسم‌ها استفاده شوند. چه با داده‌های دسته‌ای کار کنید و چه با داده‌های پیوسته، احتمالاً پالتی وجود دارد که برای نیازهای شما مناسب باشد.

رنگ‌ها

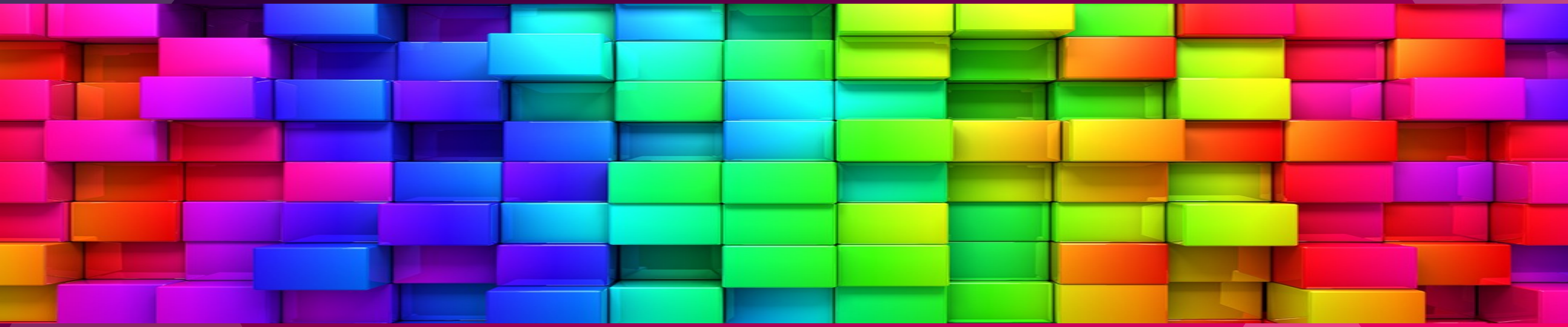
ماژول bokeh.colors بخشی از کتابخانه پایتون Bokeh است که ابزارهایی برای ایجاد تجسم‌های تعاملی در مرورگرهای وب فراهم می‌کند. این ماژول مجموعه‌ای از توابع و کلاس‌ها برای کار با رنگ‌ها در Bokeh ارائه می‌دهد.

یکی از اساسی‌ترین توابع در ماژول bokeh.colors تابع color() است که یک رشته نمایانگر رنگ را می‌گیرد و یک تاپل RGB از اعداد صحیح بازمی‌گرداند. به عنوان مثال، color("red") مقدار (255, 0, 0) را برمی‌گرداند.

تابع دیگر در ماژول bokeh.colors تابع RGB() است که سه عدد صحیح نمایانگر مقادیر قرمز، سبز و آبی یک رنگ را می‌گیرد و یک تاپل RGB بازمی‌گرداند. به عنوان مثال، RGB(255, 0, 0) مقدار (255, 0, 0) را برمی‌گرداند.

کلاس Color در ماژول bokeh.colors یک روش شیءگرا برای کار با رنگ‌ها فراهم می‌کند. شما می‌توانید با وارد کردن یک رشته رنگ، یک تاپل RGB یا یک رشته هگزادسیمال، یک شیء Color ایجاد کنید. به عنوان مثال، Color("red") یک شیء Color ایجاد می‌کند که رنگ قرمز را نشان می‌دهد. همچنین می‌توانید به اجزای قرمز، سبز و آبی یک شیء Color با استفاده از ویژگی‌های red، green و blue دسترسی پیدا کنید.

کلاس HSV در ماژول bokeh.colors یک روش برای کار با رنگ‌ها در فضای رنگی HSV (ته‌رنگ، اشباع، مقدار) فراهم می‌کند. شما می‌توانید با وارد کردن اجزای ته‌رنگ، اشباع و مقدار به عنوان اعداد اعشاری بین 0 و 1، یک شیء HSV ایجاد کنید. به عنوان مثال، HSV(0.0, 1.0, 1.0) یک شیء HSV ایجاد می‌کند که رنگ قرمز خالص را نشان می‌دهد.



نمودار پراکندگی

در اینجا یک مثال از چگونگی ایجاد یک نمودار پراکندگی با استفاده از Bokeh آمده است:

```
from bokeh.plotting import figure, output_file, show
from bokeh.models import ColumnDataSource

#Create sample data
X = [1, 2, 3, 4, 5]
Y = [6, 7, 2, 4, 5]

#Create a ColumnDataSource Object
source = ColumnDataSource(data = dict(x=X,y=Y))

# Create a new plot with a title and axis labels
p = figure(title ="Scatter Plot", x_axis_label='X - Axis', y_axis_label='Y - Axis')

# Add a circle glyph to the plot using the data from ColumnDataSource
p.circle('x','y', size=10, source=source)

#Save th eplot to an HTML file and display it
output_file("scatter.html")
show(p)
```

Output:

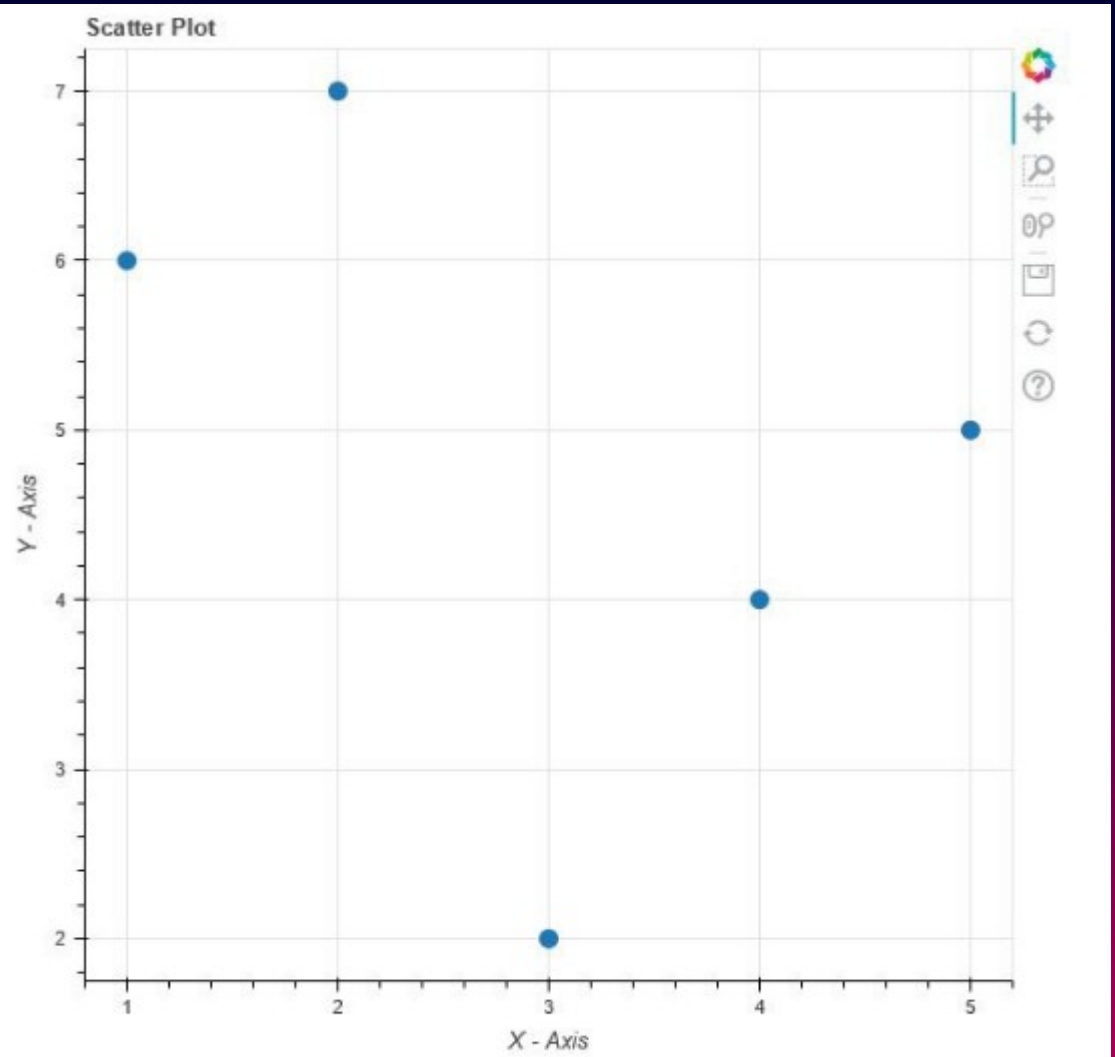


Figure 7.3: *Scatter Plot*

این کد یک نمودار پراکندگی با عنوان و برچسب‌های محوری ایجاد می‌کند و یک نماد دایره را به نمودار اضافه می‌کند با استفاده از داده‌های موجود در شیء ColumnDataSource. نمودار حاصل به یک فایل HTML ذخیره می‌شود و با استفاده از تابع show نمایش داده می‌شود.

در Bokeh، یک ColumnDataSource یک ساختار داده اساسی است که برای نگهداری داده‌هایی که در یک شکل (figure) نمایش داده خواهند شد استفاده می‌شود. اساساً یک شیء شبیه به دیکشنری است که نام‌های ستون رشته‌ای را به دنباله‌های مقادیر نقشه می‌کند. این دنباله‌ها می‌توانند لیست‌ها، آرایه‌ها یا سری‌های Pandas باشند.

استفاده از یک ColumnDataSource می‌تواند فرآیند ایجاد نمودارهای تعاملی را ساده‌تر کند، زیرا به Bokeh اجازه می‌دهد تا نمودار را در پاسخ به تعاملات کاربر مانند جابجایی، زوم یا هوورینگ (hovering) بر روی نمادها به‌روزرسانی کند. همچنین می‌توان از آن برای به اشتراک گذاشتن داده بین چندین نمودار در یک داشبورد استفاده کرد. به عنوان مثال، در زیر یک مثال از ایجاد یک ColumnDataSource در Bokeh آورده شده است:

اینجا یک مثال از ایجاد یک ColumnDataSource در Bokeh است:

```
from bokeh.plotting import figure, output_file, show
from bokeh.models import ColumnDataSource
import numpy as np

#Sample Data
x=np.linspace(0,10,100)
y=np.sin(x)

#Create a ColumnDataSource Object
source = ColumnDataSource(data = dict(x=X,y=Y))
```

در این مثال، ما برخی از داده‌ها را برای نمایش ایجاد می‌کنیم، در این حالت آرایه‌های x و y که شامل ۱۰۰ نقطه در طول یک موج سینوسی هستند. سپس یک شیء ColumnDataSource ایجاد می‌کنیم و یک دیکشنری با کلیدهای 'x' و 'y' و آرایه‌های x و y را به عنوان مقادیر به آن منتقل می‌کنیم.

نمودار خطی

یک مثال از چگونگی ایجاد یک نمودار خطی ساده با استفاده از Bokeh:

```
from bokeh.plotting import figure, output_file, show
#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 ="Line Plot", x_axis_label='X - Axis', y_axis_label='Y - Axis')

# Add a line glyph to the plot using the data from ColumnDataSource
p.line(X,Y, line_width=2)

#Save the plot to an HTML file and display it
output_file("line.html")
show(p)
```

Output:

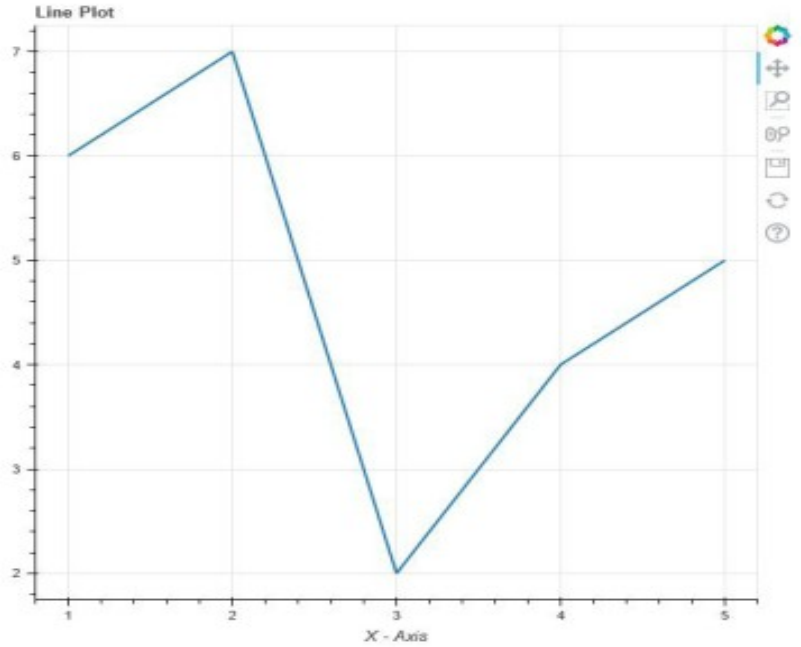


***Figure 7.4:** Line Plot*

این کد یک نمودار خطی با عنوان و برچسب‌های محوری ایجاد می‌کند و یک نماد خط را به نمودار با استفاده از داده‌های موجود در لیست‌های x و y اضافه می‌کند. آرگومان line_width عرض خط را تنظیم می‌کند. نمودار حاصل به یک فایل HTML ذخیره می‌شود و با استفاده از تابع show نمایش داده می‌شود.

می‌توانید با استفاده از آرگومان‌های اضافی به توابع figure و line، ظاهر نمودار را بیشتر سفارشی کنید. به عنوان مثال، می‌توانید با استفاده از آرگومان line_color رنگ خط را تنظیم کنید، یا نشانگرها را به خط اضافه کنید با استفاده از آرگومان‌های line_dash و line_dash_offset.

نمودار میله‌ای

یک مثال از چگونگی ایجاد یک نمودار میله‌ای ساده با استفاده از Bokeh:

```
from bokeh.plotting import figure, output_file, show

# Data
categories = ['Apple', 'Orange', 'Banana']
Values = [5, 3, 4]

# Create a new plot with a title and axis labels
p = figure(title ="Bar Plot", x_axis_label='Category', y_axis_label='Value',
        x_range=categories)

# Add a vbar glyph to the plot using the data from ColumnDataSource
p.vbar(categories,top=Values,bottom=0, width =0.6)

#Save th eplot to an HTML file and display it
output_file("bar.html")
show(p)
```

Output:

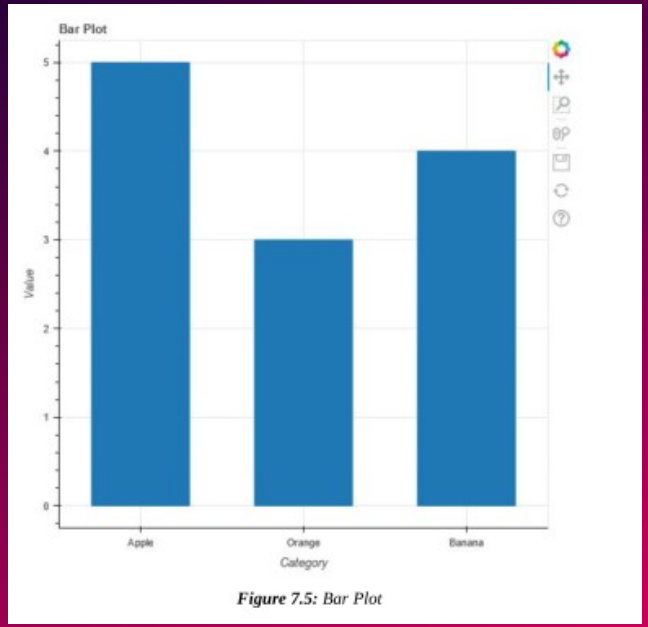


Figure 7.5: Bar Plot

```
from bokeh.plotting import figure, output_file, show
from bokeh.models import ColumnDataSource, LinearColorMapper
from bokeh.transform import transform
# Create some data
x = ['Mon', 'Tue', 'Wed', 'Thu', 'Fri']
y = ['Morning', 'Afternoon', 'Evening']
data = {'x': x*len(y), 'y': [i for i in y for _ in x],
        'values': [5, 3, 2, 7, 6, 4, 8, 1, 9, 2, 5, 7, 6, 2, 3]}
# Create a new plot with a title and axis labels
p = figure(title="Heatmap", x_range=x, y_range=y)
# Create a color mapper to map the values to colors
color_mapper = LinearColorMapper(palette="Viridis256",
                                 low=min(data['values']), high=max(data['values']))
# Add a Rect glyph to the plot using the data and color mapper
source = ColumnDataSource(data)
p.rect(x='x', y='y', width=1, height=1, source=source,
       line_color=None, fill_color=transform('values', color_mapper))
# Add a color bar to the plot
color_bar = bokeh.models.ColorBar(color_mapper=color_mapper, location=(0, 0))
p.add_layout(color_bar, 'right')
# Save the plot to an HTML file and display it
output_file("heatmap.html")
show(p)
```

این کد یک نمودار میله‌ای با عنوان و برچسب‌های محوری ایجاد می‌کند و یک نماد میله‌ای به نمودار با استفاده از داده‌های موجود در لیست‌های دسته‌بندی‌ها و مقادیر اضافه می‌شود. آرگومان x_range محدوده‌ی مقادیر را برای محور x تعیین می‌کند. آرگومان width عرض میله‌ها را تعیین می‌کند. نمودار حاصل به یک فایل HTML ذخیره می‌شود و با استفاده از تابع show نمایش داده می‌شود. می‌توانید با استفاده از آرگومان‌های اضافی به توابع figure و vbar، ظاهر نمودار را بیشتر سفارشی کنید. به عنوان مثال، می‌توانید با استفاده از آرگومان color رنگ میله‌ها را تنظیم کنید، یا یک افسانه به نمودار اضافه کنید با استفاده از آرگومان legend_label.

HeatMap

در Bokeh نوع چارت Heatmap به صورت پیش‌فرض موجود نیست، اما می‌توانید با استفاده از نماد Rect و ColorMapper یک Heatmap ایجاد کنید تا مقادیر را به رنگ‌ها نگاشت کنید. در زیر یک مثال از چگونگی ایجاد یک Heatmap ساده با استفاده از Bokeh آورده شده است:

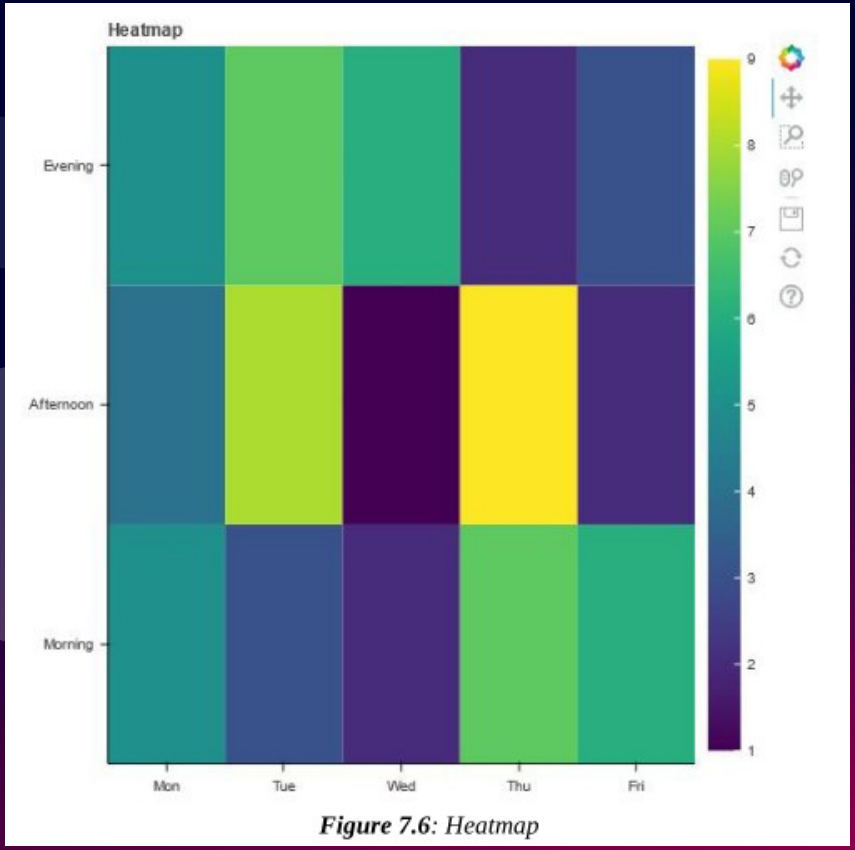


***Figure 7.6**: Heatmap*

این کد یک Heatmap با عنوان و برچسب‌های محوری ایجاد می‌کند و یک نماد Rect را به نمودار اضافه می‌کند با استفاده از داده‌های موجود در لیست‌های x، y و values. آرگومان‌های width و height اندازه هر مستطیل را تعیین می‌کنند. آرگومان line_color به None تنظیم شده است تا حاشیه‌ها را از دور مستطیل‌ها حذف کند. آرگومان fill_color رنگی را برای پر کردن مستطیل تعیین می‌کند. در این حالت، تابع transform() برای اعمال یک color_mapper به ستون مقادیر منبع داده‌ی منبع استفاده می‌شود، که نشان می‌دهد رنگ پر شدن ممکن است بر اساس نوعی نگاشت داده یا نرمال‌سازی باشد. نمودار حاصل به یک فایل HTML ذخیره می‌شود و با استفاده از تابع show نمایش داده می‌شود.

می‌توانید با استفاده از آرگومان‌های اضافی به توابع figure و rect، ظاهر نمودار را بیشتر سفارشی کنید. به عنوان مثال، می‌توانید از طریق آرگومان‌های width، height و location سازنده ColorBar اندازه و مکان نوار رنگ را تنظیم کنید.

Histogram

Bokeh تابع histogram را برای ایجاد هیستوگرام‌ها فراهم می‌کند. در زیر یک مثال از چگونگی ایجاد یک هیستوگرام ساده با استفاده از Bokeh آورده شده است:

```
from bokeh.plotting import figure, output_file, show
from bokeh.sampledata.iris import flowers
import numpy as np
 # Create a new plot with a title and axis labels
p = figure(title="Histogram", x_axis_label="Sepal Length", y_axis_label="Count")
 # Get the data
values, edges = np.histogram(flowers['sepal_length'], bins=20)
 # Add the histogram to the plot
p.quad(top=values, bottom=0, left=edges[:-1], right=edges[1:],
       fill_color="navy", line_color="white", alpha=0.5)
 # Save the plot to an HTML file and display it
output_file("histogram.html")
show(p)
```

Output:

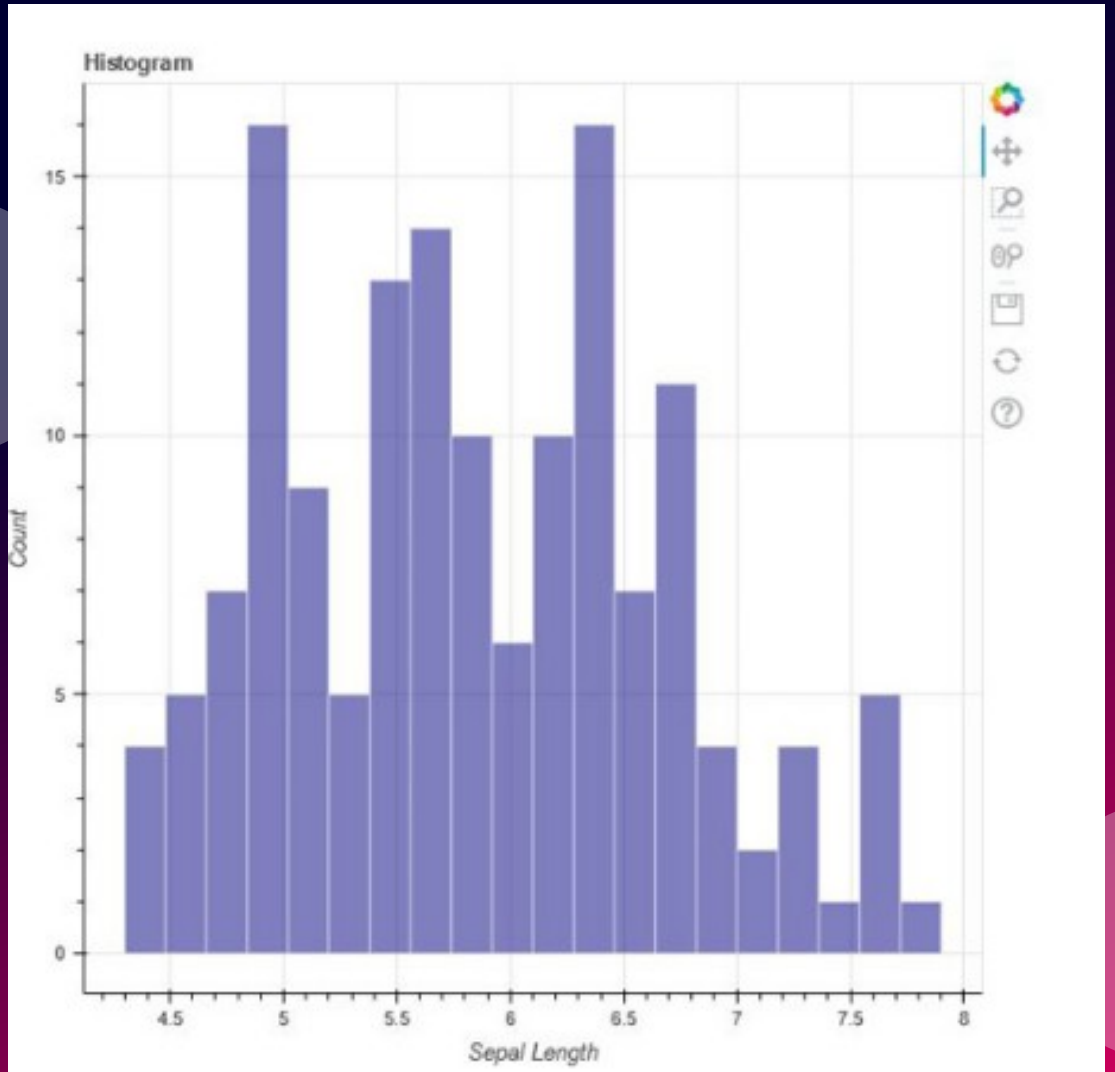


***Figure 7.7**: Histogram*

این کد یک هیستوگرام از طول گلبرگ‌ها در مجموعه داده Iris ایجاد می‌کند. آرایه‌های values و edges با استفاده از تابع numpy.histogram به دست می‌آیند که داده‌ها را به ۲۰ بازه با فواصل مساوی تقسیم می‌کند.

تابع quad برای ایجاد یک مستطیل برای هر بازه استفاده می‌شود، با بالای مستطیل تنظیم شده به تعداد نقاط داده در بازه. این خط کد یک نماد جدید با استفاده از p.quad() ایجاد می‌کند و آن را به شی p plot اضافه می‌کند. این تابع چندین آرگومان را می‌پذیرد:

top=values: این آرگومان بالای هر مستطیل را به مقدار متناظر در آرایه values تنظیم می‌کند که ارتفاع هر بازه در هیستوگرام را نشان می‌دهد.

bottom=0: این آرگومان پایین هر مستطیل را به 0 تنظیم می‌کند، که اطمینان حاصل می‌شود که مستطیل‌ها از محور x شروع می‌شوند.

left=edges[:-1]: این آرگومان لبه چپ هر مستطیل را به مقدار متناظر در آرایه edges، بدون در نظر گرفتن آخرین مقدار، تنظیم می‌کند. این اطمینان حاصل می‌شود که هر مستطیل از ابتدای هر بازه شروع می‌شود.

right=edges[1:]: این آرگومان لبه راست هر مستطیل را به مقدار متناظر در آرایه edges، بدون در نظر گرفتن اولین مقدار، تنظیم می‌کند. این اطمینان حاصل می‌شود که هر مستطیل تا انتهای هر بازه می‌رسد.

fill_color="navy": این آرگومان رنگ پر کردن هر مستطیل را به "navy" تنظیم می‌کند.

line_color="white": این آرگومان رنگ حاشیه مستطیل‌ها را به "white" تنظیم می‌کند.

alpha=0.5: این آرگومان شفافیت هر مستطیل را به 0.5 تنظیم می‌کند، باعث می‌شود تا جزئی از شفافیت داشته باشند. نمودار حاصل به یک فایل HTML ذخیره می‌شود و با استفاده از تابع show نمایش داده می‌شود.

می‌توانید با استفاده از آرگومان‌های اضافی به توابع figure و quad، ظاهر نمودار را بیشتر سفارشی کنید. به عنوان مثال، می‌توانید با استفاده از آرگومان‌های fill_color و line_width تابع quad، رنگ و عرض خطوط مستطیل‌ها را تنظیم کنید، یا از طریق آرگومان legend_label تابع quad یک افسانه اضافه کنید.

نمودار Patch

یک نماد Patch در Bokeh برای رسم چندضلعی‌های بسته دلخواه در یک نمودار استفاده می‌شود. این نماد Patch می‌تواند برای برجسته کردن مناطق خاصی از یک نمودار یا ایجاد اشکال سفارشی استفاده شود.

در زیر یک مثال از ایجاد یک نمودار Patch در Bokeh آورده شده است:

```
from bokeh.plotting import figure, show
from bokeh.models import ColumnDataSource
# create a sample polygon
x = [1, 2, 3, 4, 5]
y = [1, 3, 4, 2, 1]
polygon = [(xi, yi) for xi, yi in zip(x, y)]
# create a data source for the polygon
source = ColumnDataSource(data=dict(x=x, y=y))
# create a figure and add a patch glyph
p1 = figure(title="Patch plot")
p1.patch(x='x', y='y', fill_alpha=0.4, line_width=2, source=source)
```

```
show(p1)
```

Output:

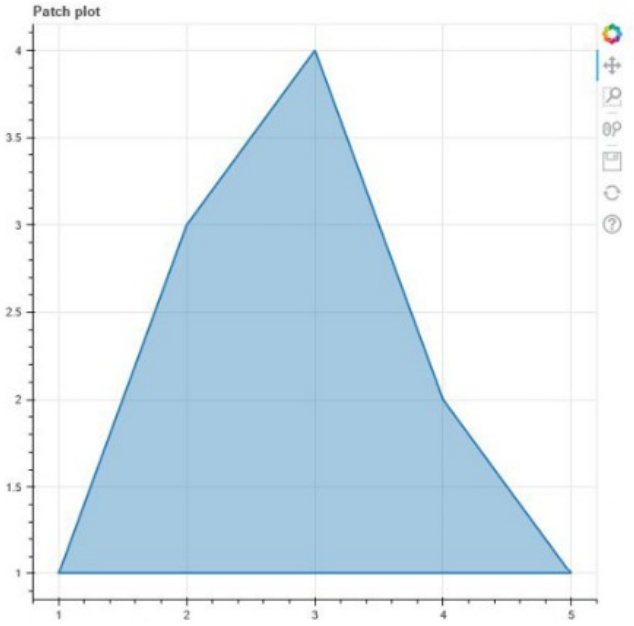


Figure 7.8: *Patch Plot*

در این مثال، ابتدا یک چندضلعی نمونه با پنج رأس تعریف می‌کنیم. سپس یک شیء ColumnDataSource را برای ذخیره داده‌های چندضلعی ایجاد می‌کنیم. ما چندضلعی را به منبع داده اضافه می‌کنیم با استفاده از یک دیکشنری که نام‌های ستون را به آرایه‌های داده نقشه می‌دهد.

سپس یک شیء figure ایجاد می‌کنیم و با استفاده از متد patch یک نماد Patch به آن اضافه می‌کنیم. نام‌های ستون منبع داده را به پارامترهای x و y متد patch منتقل می‌کنیم تا مختصات چندضلعی را مشخص کنیم. همچنین، پارامترهای fill_alpha و line_width را مشخص می‌کنیم تا ظاهر چندضلعی را کنترل کنیم.

در نهایت، تابع show را فراخوانی می‌کنیم تا نمودار را در رابط کاربری پیش‌فرض Bokeh نمایش دهیم. نمودار حاصل چندضلعی را که توسط مختصات x و y تعریف شده است نشان می‌دهد.

نمودارهای مساحتی

نمودارهای مساحتی یک نوع نمودار هستند که داده‌های کمی را با رسم مقادیر تجمعی یک متغیر در طول زمان یا متغیر مستقل دیگر نمایش می‌دهند. در Bokeh، نمودارهای مساحتی می‌توانند با استفاده از روش‌های varea یا harea کلاس Figure ایجاد شوند، بسته به اینکه آیا نمودار مساحتی عمودی یا افقی می‌خواهید.

در زیر یک مثال از چگونگی ایجاد یک نمودار مساحتی عمودی ابتدایی با استفاده از Bokeh آورده شده است:

```
from bokeh.plotting import figure, show
x = [1, 2, 3, 4, 5]
y1 = [2, 4, 3, 6, 4]
y2 = [1, 3, 2, 4, 3]

p = figure(title="Area Plot", x_axis_label="X-axis", y_axis_label="Y-axis")
p.varea(x=x, y1=y1, y2=y2, fill_color="blue")
Show(p)
```

Output:

در این مثال، از متد varea برای ایجاد نمودار ناحیه‌ای عمودی استفاده شده است که در آن x به عنوان متغیر مستقل و y1 و y2 به عنوان متغیرهای تجمعی به کار می‌روند. با استفاده از آرگومان fill_color، رنگ ناحیه پر شده بین دو خط تعیین می‌شود.

شما همچنین می‌توانید ظاهر نمودار ناحیه‌ای را با استفاده از خواص و گزینه‌های مختلف مانند line_alpha، line_color، line_width، و fill_alpha سفارشی‌سازی کنید.

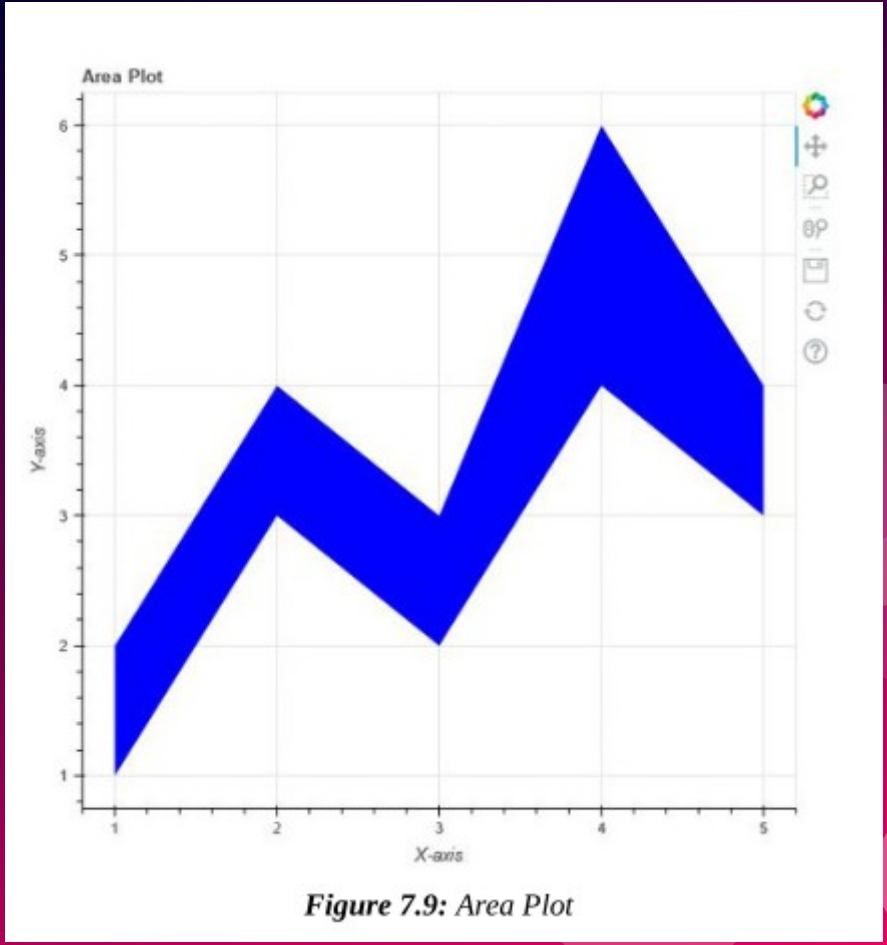


Figure 7.9: *Area Plot*

در این مثال، از متد varea برای ایجاد نمودار ناحیه‌ای عمودی استفاده شده است که در آن x به عنوان متغیر مستقل و y1 و y2 به عنوان متغیرهای تجمعی به کار می‌روند. با استفاده از آرگومان fill_color، رنگ ناحیه پر شده بین دو خط تعیین می‌شود.

شما همچنین می‌توانید ظاهر نمودار ناحیه‌ای را با استفاده از خواص و گزینه‌های مختلف مانند line_alpha، line_color، line_width، و fill_alpha سفارشی‌سازی کنید.

برای مثال، برای شفاف‌تر کردن الگوی ناحیه و افزایش شفافیت ناحیه پر شده:

```
# add the vertical area plot with a hatching pattern and color
p.varea(x=x, y1=y1, y2=y2, fill_color="blue", hatch_color="red", hatch_pattern="dot")
show(p)
```

در این مثال، ما یک الگوی هاشور سبز را به رنگ پرکننده آبی اضافه می‌کنیم با استفاده از پارامتر hatch_color که به "قرمز" تنظیم شده و پارامتر hatch_pattern که به "نقطه" تنظیم شده است.

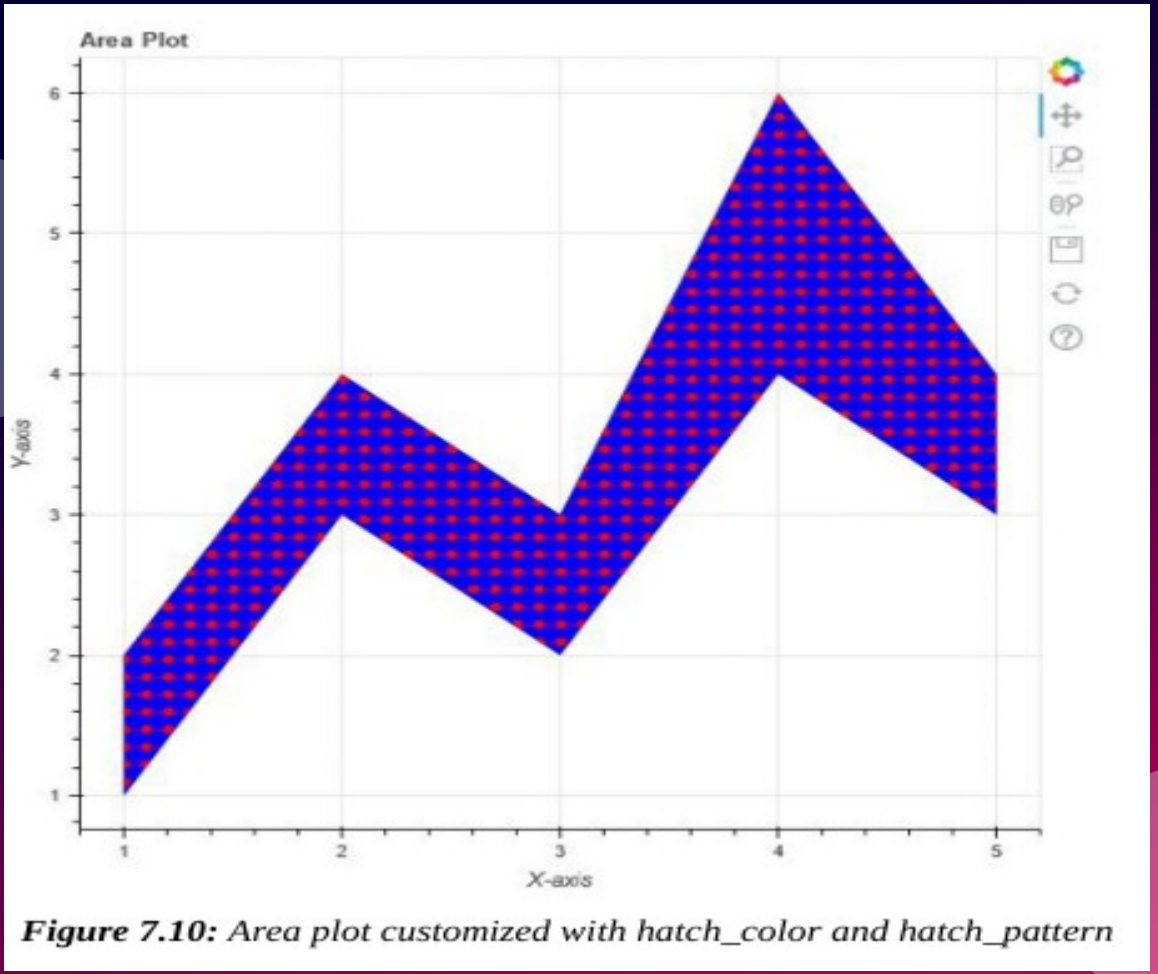


Figure 7.10: *Area plot customized with hatch_color and hatch_pattern*

همچنین می‌توانید چند نمودار ناحیه‌ای را روی هم قرار دهید تا یک نمودار ناحیه‌ای انباشته ایجاد کنید. برای مثال، برای انباشتن دو نمودار ناحیه‌ای عمودی:

```
source = ColumnDataSource(data=dict(x=x,y1=y1,y2=y2))
p.varea_stack(['y1', 'y2'], x='x',
              fill_color=["lightblue", "lightgreen"],
              source=source, legend_label=["Area 1", "Area 2"])
show(p)
```

در این مثال، از متد varea_stack برای انباشتن y1 و y2 روی یکدیگر استفاده شده است. با استفاده از آرگومان fill_color، رنگ هر ناحیه پر شده تعیین می‌شود و با آرگومان legend_label، برچسب هر نمودار ناحیه‌ای در افسانه تنظیم می‌گردد.

این‌ها تنها چند نمونه از نحوه ایجاد و سفارشی‌سازی نمودارهای ناحیه‌ای در Bokeh هستند. این کتابخانه گزینه‌ها و خواص بسیاری دیگر را برای ایجاد انواع مختلف نمودارهای ناحیه‌ای فراهم می‌کند.

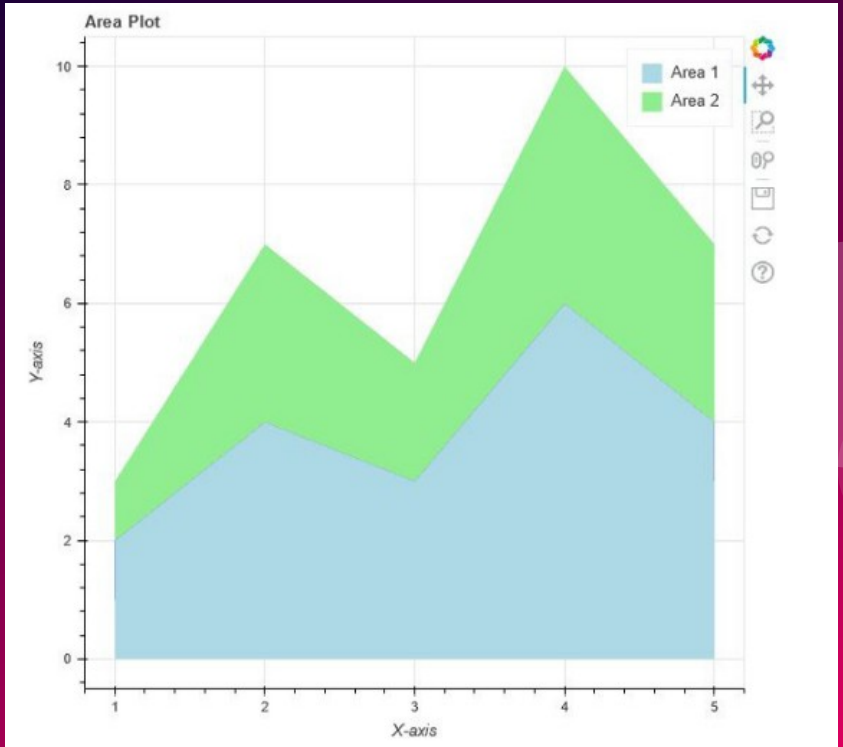


Figure 7.11: *Area Plot Customized with fill_color*

نمودار میله‌ای انباشته

نمودار میله‌ای انباشته نوعی نمودار است که توزیع یک متغیر دسته‌ای را با انباشتن مقادیر هر دسته روی یکدیگر نشان می‌دهد. در Bokeh، نمودارهای میله‌ای انباشته می‌توانند با استفاده از متدهای vbar_stack یا hbar_stack از کلاس Figure ایجاد شوند، بسته به این که شما یک نمودار میله‌ای عمودی یا افقی انباشته می‌خواهید.

در اینجا یک مثال از نحوه ایجاد یک نمودار میله‌ای عمودی انباشته ساده با استفاده از Bokeh آورده شده است:

```
from bokeh.plotting import figure, show
from bokeh.palettes import Spectral6
x = ["Books", "Magazine", "Posters"]
data = {'Product': x,
        '2020': [20, 30, 40],
        '2021': [10, 15, 20],
        '2022': [5,10,15]}
color=Spectral6[0:3]
# Convert the data to a ColumnDataSource
source = ColumnDataSource(data=data)
p = figure(x_range=x, plot_height=350, title="Product Sales by Year",
           toolbar_location=None, tools="")

p.vbar_stack(['2020','2021','2022'], x='Product', color=color,
             source=source)

show(p)
```

Output:

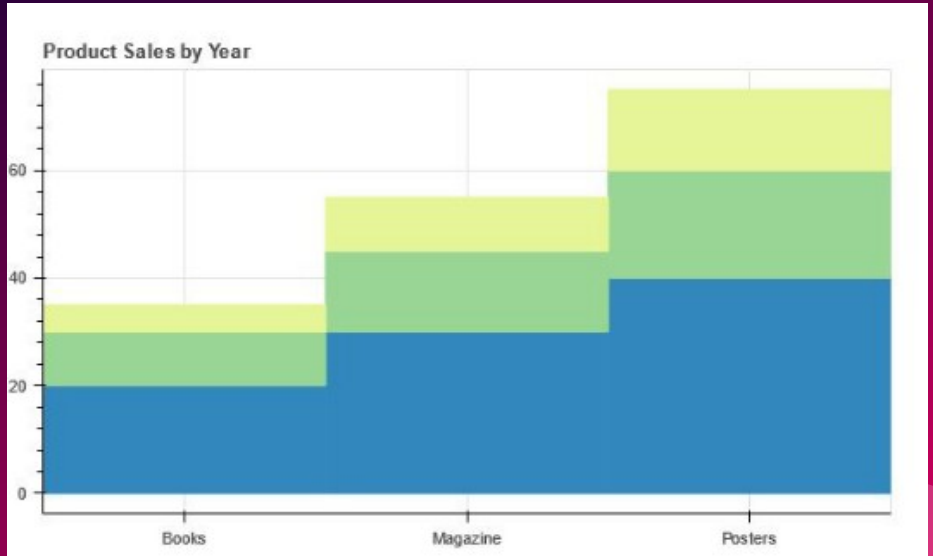


Figure 7.12: *Stacked Bar Plot*

در این مثال، نمودار میله‌های عمودی انباشته را نمایش می‌دهد تا فروش محصولات مختلف را در سال‌های ۲۰۲۰، ۲۰۲۱ و ۲۰۲۲ نشان دهد. محور x نمودار سه نوع محصول مختلف "کتاب‌ها"، "مجله‌ها" و "پوسترها" را نشان می‌دهد. محور y نمودار مجموع فروش هر نوع محصول را برای سال‌های ۲۰۲۰، ۲۰۲۱ و ۲۰۲۲ نمایش می‌دهد.

پالت Spectral6 از ماژول bokeh.palettes برای تعریف رنگ‌های میله‌ها استفاده شده است. پالت Spectral6 شامل شش رنگ است و در این کد از سه رنگ اول آن (با استفاده از Spectral6[0:3]) برای نمایش سه سال استفاده شده است.

تابع vbar_stack برای ایجاد میله‌های عمودی انباشته استفاده شده است. اولین پارامتر لیستی از کلیدهای دیکشنری داده‌ها است که به سال‌هایی که می‌خواهیم انباشته شوند اشاره می‌کند. پارامتر x به "Product" تنظیم شده تا نشان دهد که می‌خواهیم میله‌ها برای هر نوع محصول انباشته شوند. پارامتر color به لیست Spectral6[0:3] تنظیم شده تا رنگ‌های میله‌ها را تعریف کند. در نهایت، پارامتر source به شیء ColumnDataSource که قبلاً ایجاد کرده‌ایم تنظیم شده تا داده‌ها را برای نمودار فراهم کند. آرگومان legend_label برای تنظیم برچسب هر انباشت در افسانه استفاده می‌شود.

همچنین می‌توانید ظاهر نمودار میله‌ای انباشته را با استفاده از خواص و گزینه‌های مختلفی مانند line_alpha، line_color، line_width و fill_alpha سفارشی‌سازی کنید.

شما همچنین می‌توانید یک نمودار میله‌ای افقی انباشته با استفاده از متد hbar_stack ایجاد کنید، که به‌طور مشابه با متد vbar_stack کار می‌کند، اما با تغییر جهت نمودار و ابعاد میله‌ها.

این کتابخانه گزینه‌ها و خواص بسیاری دیگر را برای ایجاد انواع مختلف نمودارهای میله‌ای انباشته فراهم می‌کند.

ایجاد نمودارهای تعاملی

ایجاد یک نمودار تعاملی فرآیندی ساده است که شامل ایجاد یک شکل (figure) در Bokeh، تعریف داده‌های نمودار پراکندگی و افزودن ویژگی‌های تعاملی به نمودار می‌باشد.

در اینجا یک مثال از یک نمودار پراکندگی تعاملی در Bokeh آورده شده است:

```
from bokeh.plotting import figure, show, output_file
from bokeh.models import HoverTool
import pandas as pd

#Data
df = pd.DataFrame({
    'x':[1,2,3,4,5],
    'y':[4,7,1,6,3],
    'size':[10,20,30,40,50],
    'color':['red','green','blue','orange','purple']})
#create Bokeh figure and add scatter plot
p=figure(title='Interactive Scatter Plot',
        tools='box_select,lasso_select,reset')
p.scatter('x','y',size='size',color='color',alpha=0.5, source=df)
#Add hover tooltip
hover = HoverTool(tooltips=[('x', '@x'),('y','@y')])
p.add_tools(hover)
#Show plot in output file or in notebook
output_file('interactive_Scatter.html')
show(p)
```

Output:

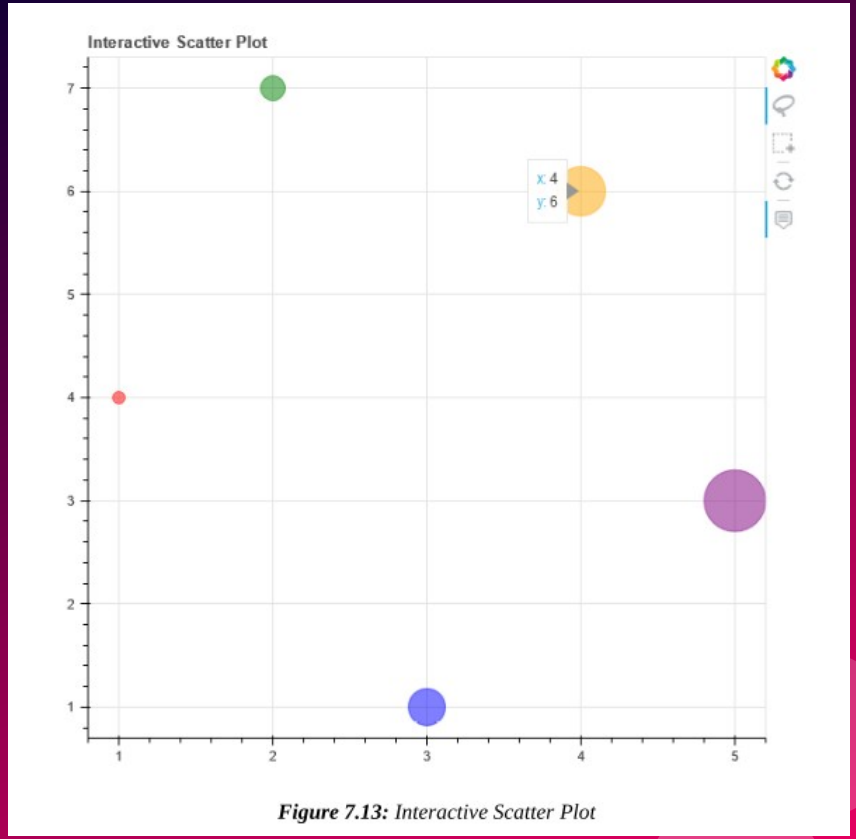


Figure 7.13: *Interactive Scatter Plot*

در این مثال، ابتدا یک دیتافریم نمونه با داده‌های تصادفی ایجاد می‌کنیم. سپس با استفاده از تابع figure یک شکل Bokeh ایجاد کرده و داده‌های نمودار پراکندگی را با استفاده از تابع scatter اضافه می‌کنیم. پارامترهای size و color برای تنظیم اندازه و رنگ هر نقطه پراکندگی بر اساس ستون‌های size و color در دیتافریم به کار می‌روند. پارامتر alpha برای تنظیم شفافیت هر نقطه پراکندگی استفاده می‌شود.
سپس، با استفاده از HoverTool از ماژول bokeh.models، یک ابزار نمایش اطلاعات (tooltip) به نمودار پراکندگی اضافه می‌کنیم. پارامتر tooltips برای تعریف اطلاعاتی که در tooltip نمایش داده می‌شود هنگامی که کاربر بر روی یک نقطه پراکندگی قرار می‌گیرد، استفاده می‌شود.
در نهایت، نمودار Bokeh را در فایل خروجی یا در نوت‌بوک با استفاده از توابع output_file و show نمایش می‌دهیم. پارامتر tools برای اضافه کردن ابزارهای تعاملی به نمودار استفاده می‌شود، مانند انتخاب جعبه‌ای (box select) و انتخاب طنابی (lasso select)، که به کاربران اجازه می‌دهند نقاط داده‌ای را در نمودار پراکندگی انتخاب و برجسته کنند.
این تنها یک مثال ساده از نحوه ایجاد یک نمودار پراکندگی تعاملی در Bokeh است. Bokeh ابزارها و گزینه‌های بسیاری دیگر برای ایجاد نمودارهای پراکندگی پویا و پاسخ‌گو فراهم می‌کند، از جمله قابلیت زوم و پیمایش، تعاملات پیوندی و بازگشت‌های جاوااسکریپت سفارشی.

ایجاد نمودارهای متعدد

Bokeh روش‌های مختلفی برای ایجاد نمودارهای متعدد و ترکیب آن‌ها در یک چیدمان واحد فراهم می‌کند. در اینجا چند روش برای ایجاد نمودارهای متعدد با استفاده از Bokeh آورده شده است:

1. استفاده از gridplot(): شما می‌توانید از تابع gridplot() در ماژول bokeh.layouts برای ایجاد یک شبکه از نمودارها استفاده کنید. تابع gridplot() یک لیست از لیست‌های نمودارها را می‌گیرد که هر لیست داخلی نمایانگر یک ردیف از نمودارها است. در اینجا یک مثال آورده شده است:

```
from bokeh.layouts import gridplot
from bokeh.plotting import figure, show

#create two figures
p1=figure(title="Plot 1",plot_width=250, plot_height=250)
p1.circle([1,2,3],[4,5,6])

p2=figure(title="Plot 2",plot_width=250, plot_height=250)
p2.square([1,2,3],[4,5,6])

#create grid og plots
grid=gridplot([[p1,p2]])

show(grid)
```

در این مثال، دو شکل ایجاد می‌کنیم، چند نماد به آن‌ها اضافه می‌کنیم، و سپس آن‌ها را به عملکرد gridplot() منتقل می‌کنیم تا یک شبکه از نمودارها ایجاد شود.

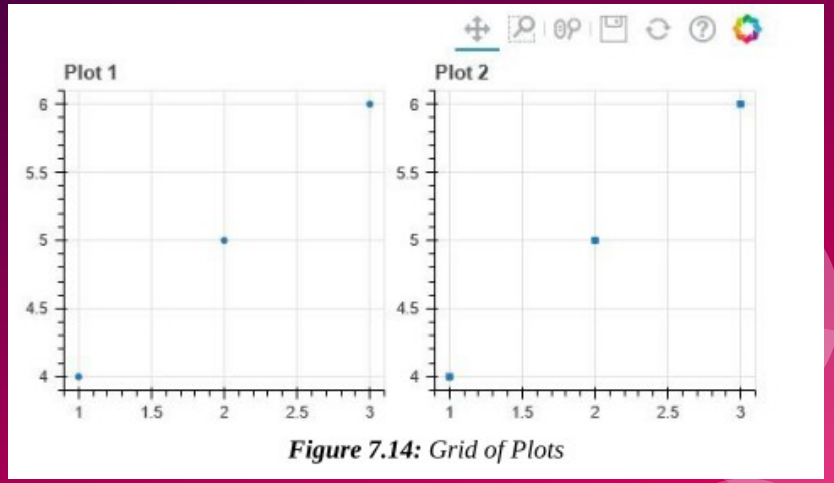


Figure 7.14: *Grid of Plots*

تابع gridplot() در بوکه برای ترتیب دادن چند شکل به صورت شبکه‌ای استفاده می‌شود. در زیر پارامترهایی که می‌توان به تابع gridplot() ارسال کرد، آمده است:

- children (لازم): یک لیست از لیست‌های شیء شکل برای ترتیب دادن در یک شبکه. هر ردیف در شبکه توسط یک لیست از اشیاء شکل نمایان می‌شود و ردیف‌ها به صورت عمودی در شبکه ترتیب داده می‌شوند.
- plot_width (اختیاری): عرض هر نمودار در شبکه، به پیکسل.
- plot_height (اختیاری): ارتفاع هر نمودار در شبکه، به پیکسل.
- toolbar_location (اختیاری): مکان نوار ابزار برای هر نمودار در شبکه. این می‌تواند "بالا"، "پایین"، "چپ" یا "راست" باشد.
- toolbar_options (اختیاری): یک دیکشنری از گزینه‌ها که به نوار ابزار ارسال می‌شود. گزینه‌های ممکن شامل "لوگو"، "کمک"، "ذخیره" و "جا" هستند. به عنوان مثال، toolbar_options={"لوگو": None، "کمک": None}، لوگو و دکمه‌های کمک را در نوار ابزار غیرفعال می‌کند.
- merge_tools (اختیاری): اگر True باشد، نوار ابزار برای هر نمودار در شبکه به یک نوار ابزار واحد ادغام خواهد شد. پیش‌فرض False است.
- sizing_mode (اختیاری): تعیین می‌کند که چگونه اندازه شبکه محاسبه خواهد شد. مقادیر ممکن شامل "ثابت"، "کشیدن_هردو"، "مقیاس_عرض" و "مقیاس_ارتفاع" می‌شود. پیش‌فرض "ثابت" است که اندازه شبکه را بر اساس پارامترهای plot_width و plot_height تعیین می‌کند.

```
from bokeh.plotting import figure, show
from bokeh.models import ColumnDataSource
# create a sample data
x = [1, 2, 3, 4, 5]
y = [1, 3, 4, 2, 1]
y1 = [2, 4, 3, 6, 4]
y2 = [1, 3, 2, 4, 3]
#create polygon with x and y
polygon = [(xi, yi) for xi, yi in zip(x, y)]
# create a First plot as patch glyph
source = ColumnDataSource(data=dict(x=x, y=y))
p1 = figure(title="Patch plot")
p1.patch(x='x', y='y', fill_alpha=0.4, line_width=2, source=source)
# create a Second plot as varea
p2 = figure(title="Area Plot", x_axis_label="X-axis", y_axis_label="Y-axis")
source = ColumnDataSource(data=dict(x=x,y1=y1,y2=y2))

  p2.varea_stack(['y1', 'y2'], x='x', fill_color=["lightblue", "lightgreen"],
          source=source, legend_label=["Area 1", "Area 2"])
  #Create Row of Plots
  from bokeh.layouts import row
  row=row(p1,p2)
  show(row)
```

2. استفاده از row() و column(): شما می‌توانید از توابع row() و column() در ماژول bokeh.layouts استفاده کنید تا ردیف‌ها یا ستون‌هایی از نمودارها ایجاد کنید. تابع 'row(*children, sizing_mode='stretch_width')' یک لیست از نمودارها یا اشیاء طرح را به عنوان آرگومان‌ها می‌پذیرد و آن‌ها را به صورت افقی در یک ردیف ترتیب می‌دهد. پارامتر sizing_mode کنترل می‌کند که چگونه طرح باید نسبت به ظرف والدین تغییر اندازه دهد. مقدار پیش‌فرض 'sizing_mode 'stretch_width است که به این معنی است که عرض طرح باید به منظور تطابق با عرض ظرف والدین کشیده شود.

تابع 'column(*children, sizing_mode='stretch_height')' یک لیست از نمودارها یا اشیاء طرح را به عنوان آرگومان‌ها می‌پذیرد و آن‌ها را به صورت عمودی در یک ستون ترتیب می‌دهد. پارامتر sizing_mode کنترل می‌کند که چگونه طرح باید نسبت به ظرف والدین تغییر اندازه دهد. مقدار پیش‌فرض 'sizing_mode 'stretch_height است که به این معنی است که ارتفاع طرح باید به منظور تطابق با ارتفاع ظرف والدین کشیده شود.

اینجا یک مثال است:

ایجاد ردیف‌ها/ستون‌ها برای نمایش نمودارهایی که قبلاً ایجاد شده‌اند - patch و varea_stack

Output:

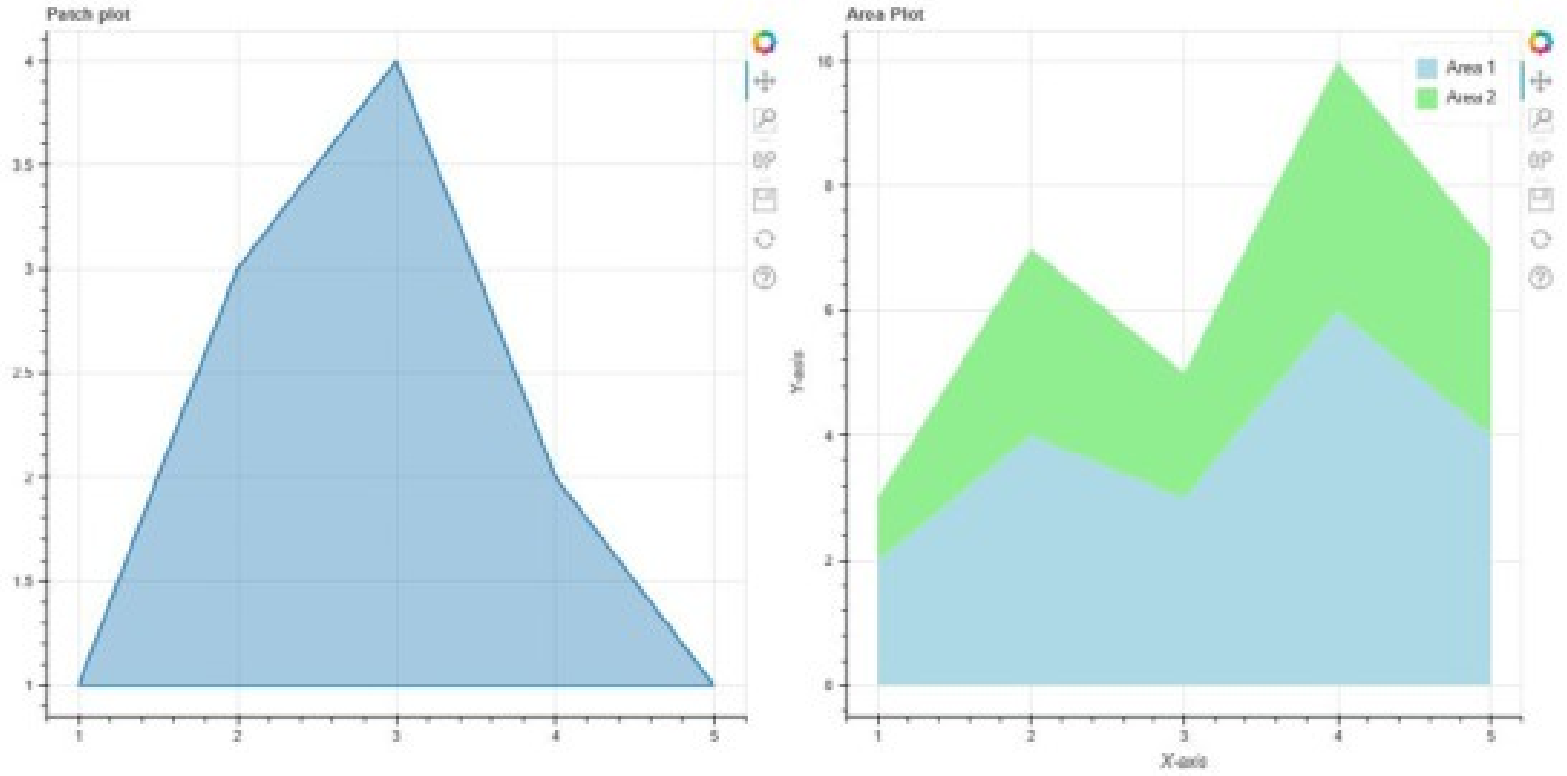


Figure 7.15: *Plotting in a row*

در این مثال، دو شکل ایجاد می‌کنیم، چند نماد به آن‌ها اضافه می‌کنیم، و سپس آن‌ها را به توابع row() و column() منتقل می‌کنیم تا یک ردیف یا ستون از نمودارها ایجاد شود.

```
# create a row of plots
row = row(p1, p2)
show(row)
# create a column of plots
column = column(p1, p2)
```

3. استفاده از tabs(): شما می‌توانید از کلاس Tabs() در ماژول bokeh.models برای ایجاد یک مجموعه از تب‌ها استفاده کنید، هرکدام حاوی یک نمودار متفاوت است. در بوکه، Tabs یک ظرفیت طرح است که یک یا چند پنل با تب‌ها را نمایش می‌دهد تا بین آن‌ها جابجا شود. سازنده Tabs یک لیست از اشیاء تب‌ها را به عنوان آرگومان‌ها می‌پذیرد.

Tab یک کلاس است که یک پنل تنها را در یک طرح Tabs نمایش می‌دهد. دو آرگومان را می‌پذیرد:

child: یک شیء طرح بوکه، مانند یک شکل، ستون، ردیف یا gridplot، که در پنل نمایش داده خواهد شد.

title: یک رشته که به عنوان عنوان تب پنل نمایش داده می‌شود.

سازنده Tabs دارای چندین پارامتر است که می‌توان از آن‌ها برای سفارشی‌سازی ظاهر و رفتار طرح تب استفاده کرد:

tabs: یک لیست از اشیاء تب برای نمایش در طرح تب. این پارامتر لازم است.

active: یک عدد صحیح که شاخص تب فعال را مشخص می‌کند هنگامی که طرح تب برای اولین بار نمایش داده می‌شود. مقدار پیش‌فرض 0 است که به این معنی است که اولین تب فعال خواهد بود.

sizing_mode: یک رشته که نحوه تغییر اندازه طرح تب نسبت به ظرف والدین را مشخص می‌کند. مقدار پیش‌فرض 'stretch_both' است که به این معنی است که طرح باید هم افقی و هم عمودی کشیده شود تا با اندازه ظرف والدین مطابقت داشته باشد.

width: یک عدد صحیح که عرض طرح تب را به پیکسل مشخص می‌کند. اگر مشخص نشده باشد، طرح عرض دسترسی‌پذیر ظرف والدین را پر می‌کند.

height: یک عدد صحیح که ارتفاع طرح تب را به پیکسل مشخص می‌کند. اگر مشخص نشده باشد، طرح ارتفاع دسترسی‌پذیر ظرف والدین را پر می‌کند.

tabs_location: یک رشته که موقعیت تب‌ها را نسبت به محتوای پنل مشخص می‌کند. مقادیر مجاز 'left'، 'below'، 'above' و 'right' هستند. مقدار پیش‌فرض 'above' است.

orientation: یک رشته که جهت تب‌ها را نسبت به محتوای پنل مشخص می‌کند. مقادیر مجاز 'horizontal' و 'vertical' هستند. مقدار پیش‌فرض 'horizontal' است.

css_classes: یک لیست از رشته‌ها که کلاس‌های CSS اضافی را برای افزودن به طرح تب مشخص می‌کند.

اینجا یک مثال است:

ایجاد تب‌ها برای نمایش نمودارهای قبلاً ایجاد شده - patch و varea_stack

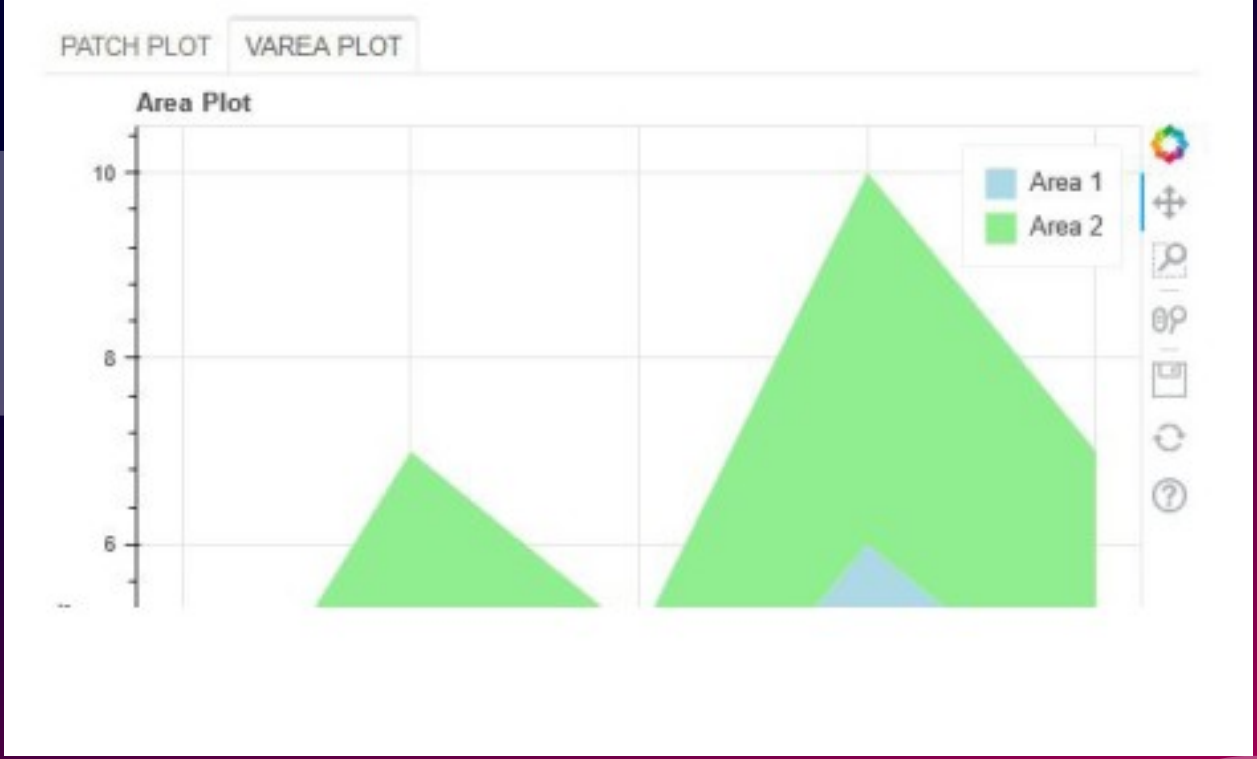
```
from bokeh.plotting import figure, show
from bokeh.models import ColumnDataSource

# create a sample data
x = [1, 2, 3, 4, 5]
y = [1, 3, 4, 2, 1]
y1 = [2, 4, 3, 6, 4]
y2 = [1, 3, 2, 4, 3]

#create polygon with x and y
polygon = [(xi, yi) for xi, yi in zip(x, y)]

# create a First plot as patch glyph
source = ColumnDataSource(data=dict(x=x, y=y))
p1 = figure(title="Patch plot")
p1.patch(x='x', y='y', fill_alpha=0.4, line_width=2, source=source)

# create a Second plot as varea
p2 = figure(title="Area Plot",
        x_axis_label="X-axis", y_axis_label="Y-axis")
source = ColumnDataSource(data=dict(x=x,y1=y1,y2=y2))
p2.varea_stack(['y1', 'y2'], x='x',
            fill_color=["lightblue", "lightgreen"],
            source=source, legend_label=["Area 1", "Area 2"])

#Create tabs
from bokeh.models import Panel, Tabs
```

```
tab1 = Panel(child=p1,title="PATCH PLOT")
tab2=Panel(child=p2,title="VAREA PLOT")
tabs=Tabs(tabs=[tab1,tab2])
output_file("Plots_in_tab.html", tabs)
show(tabs)
```

Output:

Figure 7.16: *Creating Tabs for plot display*

در این مثال، دو شکل ایجاد می‌کنیم، چند نماد به آن‌ها اضافه می‌کنیم، و سپس دو شیء پنل ایجاد می‌کنیم که هرکدام شامل یک نمودار و یک عنوان است.



سپس این پنل‌ها را به کلاس Tabs() منتقل می‌کنیم تا یک مجموعه از تب‌ها ایجاد شود، هرکدام حاوی یک نمودار متفاوت است.

ذخیره نمودار

می‌توانید یک نمودار بوکه را به یک فایل HTML با استفاده از تابع save() ذخیره کنید. تابع save() یک شیء bokeh.plotting.figure.Figure و یک نام فایل می‌پذیرد.

اینجا یک مثال است:

```
from bokeh.plotting import figure, output_file, save
# create a figure
p = figure()
# add some glyphs to the figure
p.circle([1, 2, 3], [4, 5, 6])
# specify the output file name
output_file("my_plot.html")
# save the plot
save(p)
```

در این مثال، یک شیء Figure ایجاد می‌کنیم، چند نماد به آن اضافه می‌کنیم، و نام فایل خروجی را با استفاده از تابع output_file() مشخص می‌کنیم. سپس تابع save() را فراخوانی می‌کنیم تا نمودار را در فایل مشخص‌شده ذخیره کنیم.

به طور پیش‌فرض، تابع save() یک فایل HTML ایجاد می‌کند که شامل تمام فایل‌های JavaScript و CSS لازم است. همچنین می‌توانید یک فایل HTML مستقل را که نیازی به اتصال اینترنت یا سرور Bokeh ندارد با ارسال mode="inline" به تابع save() ذخیره کنید:

```
save(p، filename="my_plot.html"، title="My Plot"، mode="inline")
```

این عمل، یک فایل HTML مستقل با عنوان داده‌شده که شامل کلیه کدهای JavaScript و CSS لازم است، ایجاد می‌کند.

در زیر پارامترهایی که می‌توان به تابع save() ارسال کرد را مشاهده می‌کنید:

fig (لازم): شیء Figure برای ذخیره‌سازی.

filename (اختیاری): نام فایل خروجی. اگر این ارائه نشود، یک نام پیش‌فرض استفاده می‌شود.

title (اختیاری): عنوان سند HTML.

resources (اختیاری): کنترل می‌کند چگونه منابع BokehJS (برای مثال JavaScript، CSS) در سند HTML شامل می‌شوند. پیش‌فرض "inline" است، که تمام منابع را در فایل HTML جاسازی می‌کند. گزینه‌های دیگر شامل "cdn" (که منابع را از یک شبکه توزیع محتوا بارگیری می‌کند) و "relative" (که پیوندهای نسبی به منابع ایجاد می‌کند) می‌شود.

mode (اختیاری): تعیین می‌کند چگونه فایل خروجی در سند HTML جاسازی شود. پیش‌فرض "cdn" است، که کد JavaScript BokehJS را از یک شبکه توزیع محتوا شامل می‌کند. گزینه‌های دیگر شامل "inline" (که کد BokehJS را در فایل HTML شامل می‌کند) و "absolute" (که یک URL مطلق برای کد BokehJS ایجاد می‌کند) می‌شود.

root_dir (اختیاری): اگر منابع با استفاده از حالت "relative" در حال درج هستند، این پارامتر دایرکتوری ریشه را برای استفاده از URLهای نسبی مشخص می‌کند.

template (اختیاری): قالب Jinja2 برای استفاده در تولید سند HTML. اگر ارائه نشود، قالب پیش‌فرض Bokeh استفاده می‌شود.

در زیر یک مثال از نحوه استفاده از برخی از این پارامترها آمده است:

```
from bokeh.plotting import figure, save
p = figure(title="My Plot")
# Add some glyphs to the figure...
save(p, filename="my_plot.html", title="My Plot", resources="cdn")
```

در این مثال، یک شیء Figure ایجاد می‌کنیم و چند نماد به آن اضافه می‌کنیم. تابع save() با پارامترهای filename و title فراخوانی شده و نام فایل خروجی و عنوان سند مشخص می‌شوند. برای پارامتر "cdn"، "resources" ذکر شده است که منابع BokehJS را از یک شبکه توزیع محتوا بارگیری خواهد کرد.

برای ذخیره یک نمودار Bokeh به فرمت PNG یا SVG، به ترتیب از توابع export_png() و export_svg() استفاده کنید. در زیر یک قطعه کد مثال آورده شده است که نحوه استفاده از توابع export_png() و export_svg() در Bokeh برای ذخیره یک نمودار در یک فایل PNG یا SVG را نشان می‌دهد:

```
from bokeh.plotting import figure, output_file, show
from bokeh.io import export_png, export_svg
# Define the plot
```

```
p = figure(title='Example Plot', x_axis_label='X Axis', y_axis_label='Y Axis')
p.line(x=[1, 2, 3, 4, 5], y=[2, 4, 6, 8, 10], line_width=2)
# Save the plot to PNG format
export_png(p, filename='example_plot.png')
# Save the plot to SVG format
export_svg(p, filename='example_plot.svg')
```

نتیجه‌گیری

در این فصل، آموختیم که Bokeh یک کتابخانه قدرتمند و انعطاف‌پذیر برای بصری‌سازی داده است که ابزارهای متنوعی برای ایجاد بصری‌سازی‌های تعاملی و مبتنی بر وب در پایتون فراهم می‌کند. این کتابخانه یک رابط سطح بالا برای ایجاد انواع شائع نمودارها مانند نمودار پراکندگی، نمودار خطی و نمودار میله‌ای فراهم می‌کند، همچنین بصری‌سازی‌های پیشرفته‌تری مانند نقشه‌های حرارتی، نمودارهای موازی هماهنگی و نقشه‌های کرولت را نیز پشتیبانی می‌کند. Bokeh همچنین انواع منابع داده را از جمله DataFrameهای Pandas، آرایه‌های NumPy و داده‌های JSON پشتیبانی می‌کند.

قدرت اصلی Bokeh در تعاملی بودن آن است که به کاربران امکان می‌دهد بصری‌سازی‌های پویا و واکنش‌پذیری ایجاد کنند که به راحتی در مرورگرهای وب قابل به اشتراک‌گذاری و بررسی هستند. این ویژگی آن را به یک انتخاب محبوب برای برنامه‌هایی مانند داشبوردها، ابزارهای کاوش داده و بصری‌سازی‌های علمی و مهندسی می‌کند.

Bokeh یک انتخاب عالی برای ایجاد بصری‌سازی‌های تعاملی با کیفیت بالا در پایتون است و مستندات گسترده و جامعت آن، یادگیری و استفاده از آن را آسان می‌کند. به یاد داشته باشید که با اشکال و رنگ‌های مختلف آزمایش کنید و از ترکیب‌های خلاقانه با نمودارهای خود استفاده کنید. در فصل بعد، شما یاد خواهید گرفت که EDA را انجام دهید و توانمندی خود را در استخراج اطلاعات معنی‌دار از مجموعه داده‌های خود تقویت کنید. با استفاده از بصری‌سازی‌ها و روش‌های آماری مختلف، روابط را کشف کرده، نقاط ناپذیری را تشخیص دهید، توزیع‌ها را کاوش کنید و در نهایت نتیجه‌های ارزشمندی که می‌تواند به تصمیم‌گیری‌های آگاهانه کمک کند، استخراج کنید. EDA به عنوان پایه‌ای برای کاوش داده‌های بیشتر و ساخت مدل‌ها عمل می‌کند، این مهارت برای هر داده‌شناس یا تحلیل‌گر داده اساسی است.

نکاتی که به خاطر بسپارید:

یک نمودار به عنوان پنجره یا ظرف کلی که شامل بصری‌سازی، شامل هرگونه محورها، افسانه‌ها و نمادها است.

نمادها عناصر بصری هستند که برای نمایش نقاط داده استفاده می‌شوند، مانند نقاط، خطوط، میله‌ها یا متن.

محورها عناصر بصری هستند که مقیاس و جهت بصری‌سازی را تعیین می‌کنند، شامل محور x، محور y و هر محور فرعی دیگری است.

محور x معمولاً برای نمایش متغیر مستقل یا زمان استفاده می‌شود، در حالی که محور y متغیر وابسته را نمایش می‌دهد.

محورها قابلیت سفارشی‌سازی را دارند که شامل خطوط تیک، برچسب‌ها، خطوط شبکه و سایر عناصر بصری است که به تفسیر داده‌ها کمک می‌کنند.

نمادها قابلیت سفارشی‌سازی را دارند که شامل رنگ، اندازه، شکل و سایر عناصر بصری است که به تمییز بین نقاط داده مختلف کمک می‌کنند.

فهم رابطه بین نمادها و محورها برای ایجاد بصری‌سازی‌های موثر که پیام مورد نظر را انتقال می‌دهند، حیاتی است.

انواع مختلفی از بصری‌سازی‌ها، مانند نمودارهای پراکندگی، نمودارهای خطی و نمودارهای میله‌ای، ترکیب‌های مختلفی از نمادها و محورها را برای انتقال موثر داده نیاز دارند.

Bokeh امکان تعامل بالایی را فراهم می‌کند، با ویژگی‌هایی مانند ابزارهای راهنمای هوور، بزرگنمایی، کشو، و انتخاب.

سفارشی‌سازی ظاهر یک نمودار شامل تنظیم ویژگی‌هایی مانند رنگ پس‌زمینه، اندازه فونت و برچسب‌های محور است.

Bokeh از طیف گسترده‌ای از فرمت‌های خروجی پشتیبانی می‌کند، از جمله PNG، HTML و SVG.

Bokeh می‌تواند با کتابخانه‌های دیگر پایتون، مانند Pandas، NumPy و Scikit-learn، برای تجزیه و تحلیل داده و یادگیری ماشین یکپارچه شود. برای ایجاد چندین نمودار در Bokeh، از تابع gridplot() برای ترتیب نمودارهای فردی در یک چیدمان شبکه استفاده کنید.

برای اتصال چندین نمودار با هم، از توابع Range1d ،()link و ColumnDataSource() برای همگام‌سازی دامنه‌ها و منابع داده استفاده کنید.

برای ذخیره یک نمودار Bokeh در فرمت HTML، از تابع output_file() استفاده کنید و نام فایل و مکان فایل خروجی را مشخص کنید.

برای ذخیره یک نمودار Bokeh در فرمت PNG یا SVG، به ترتیب از توابع export_png() یا export_svg() استفاده کنید.

در هنگام ذخیره یک نمودار Bokeh، مهم است که اندازه و وضوح فایل خروجی، و همچنین هرگونه سبک یا فرمت‌بندی اضافی را که ممکن است لازم باشد، در نظر گرفته شود.

در جستجوی داده‌های پنهان، ایجاد الگوهای جدید، و روشنایی دریایی اطلاعات، به‌عنوان یک دانشمند علوم داده، هر روز به یافتن راه‌حل‌های جذاب و مهم‌ترین مسائل جهانی گام می‌گذارید.