Generic data visualization tools for Python

By Vincent Roger

11/23/2023

About me

PhD in computer science.

Kiviak instrument's data scientist and machine learning guy.

LinkedIn

https://www.linkedin.com/in/vroger11/

Website

https://website.vincent-roger.fr/

Why data visualization with Python?

- Extract and prepare data is simplified with tools such as Pandas, Dask and others.
- It is a goto language for data scientists, machine learning and MLOPS people.
- So why learn a new language just for data visualization?

What types of tools

Specific (to create graphs)

- NetworkX
- Graph-tool
- PyGraphviz → Graphviz
- Pyvis

Generic

Focus of this presentation!

Plan

- Non web-oriented
- Web-oriented

Data we will use

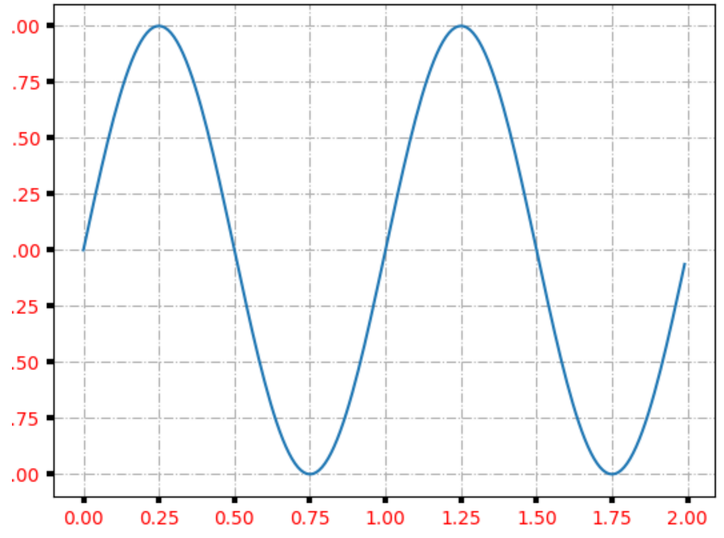
```python
import pandas as pd

df_iris = pd.read_csv('https://raw.githubusercontent.com/mwaskom/seaborn-data/master/iris.csv')

df_iris
```

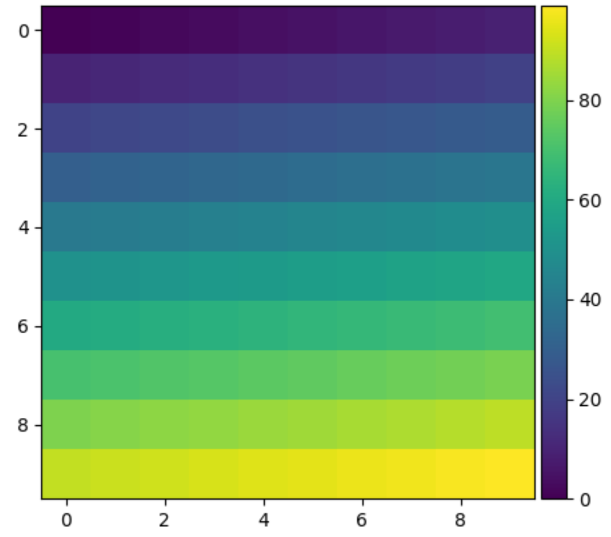
```
     sepal_length  sepal_width  petal_length  petal_width    species
0             5.1          3.5           1.4          0.2     setosa
1             4.9          3.0           1.4          0.2     setosa
2             4.7          3.2           1.3          0.2     setosa
3             4.6          3.1           1.5          0.2     setosa
4             5.0          3.6           1.4          0.2     setosa
..            ...          ...           ...          ...        ...
145           6.7          3.0           5.2          2.3  virginica
146           6.3          2.5           5.0          1.9  virginica
147           6.5          3.0           5.2          2.0  virginica
148           6.2          3.4           5.4          2.3  virginica
149           5.9          3.0           5.1          1.8  virginica

[150 rows x 5 columns]
```

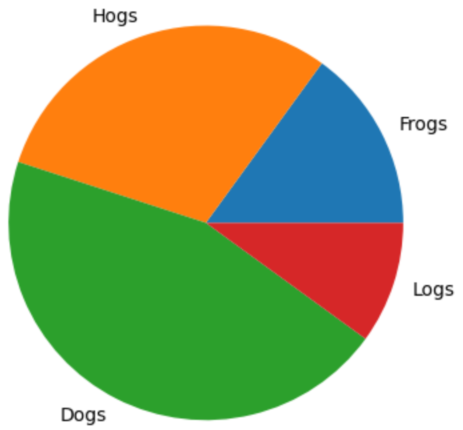
Non web-oriented

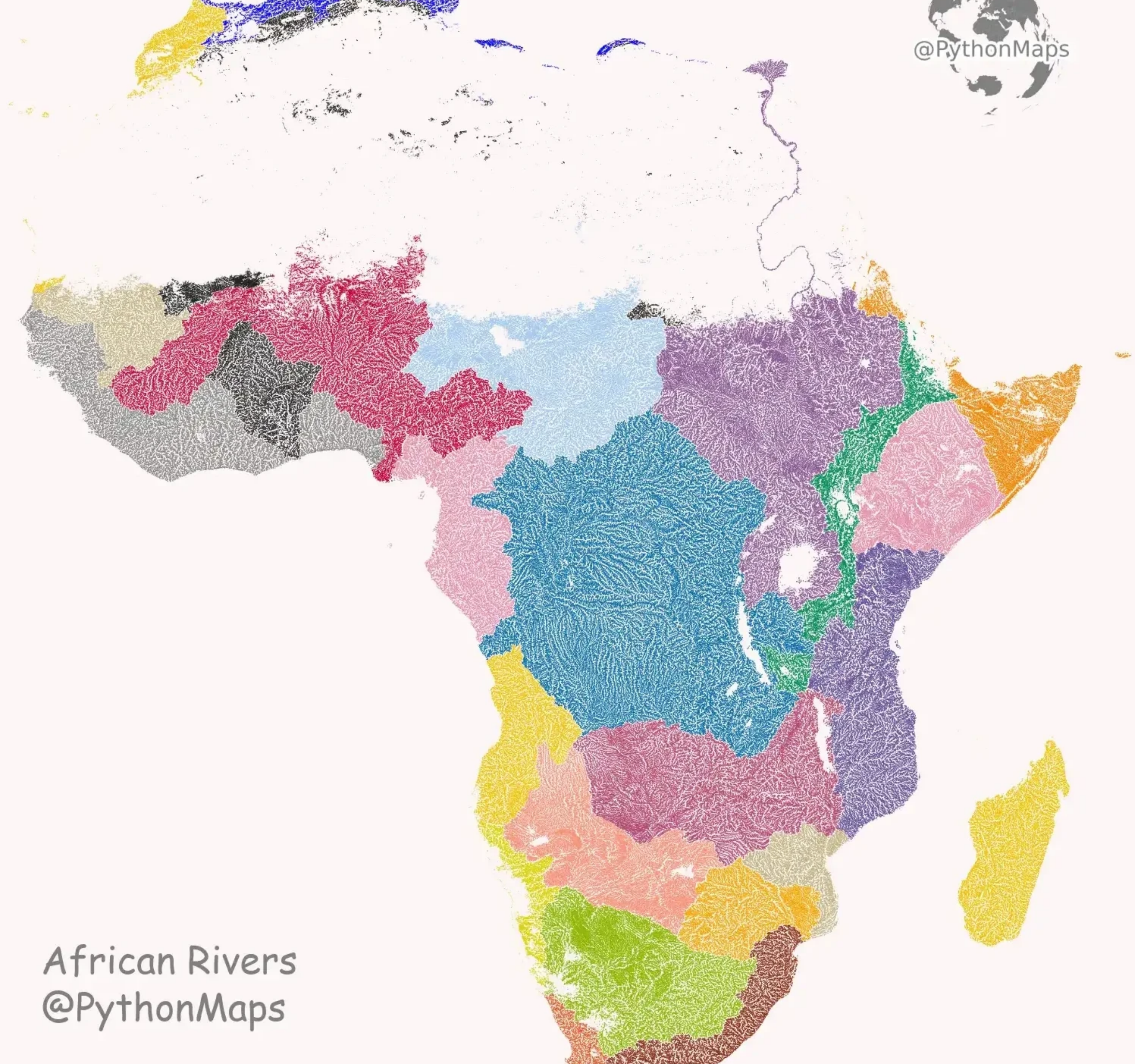
Matplotlib

- Based on Qt.
- Standard library.
- Default for multiple tools like Pandas.









Matplotlib

- Powerful if mastered!

Matplotlib - scatter plot of iris sepal

```python
import matplotlib.pyplot as plt

# Create a scatter plot with legends based on species
plt.figure(figsize=(8, 6))

# Iterate over each species and plot scatter points
for species in df_iris["species"].unique():
    species_data = df_iris[df_iris['species'] == species]
    plt.scatter(species_data['sepal_length'], species_data['sepal_width'], label=species)

# Add labels and title
plt.xlabel('Sepal Length')
plt.ylabel('Sepal Width')
plt.title('Scatter Plot of Iris Dataset')

# Add legend
plt.legend()

# Show the plot
plt.show()
```

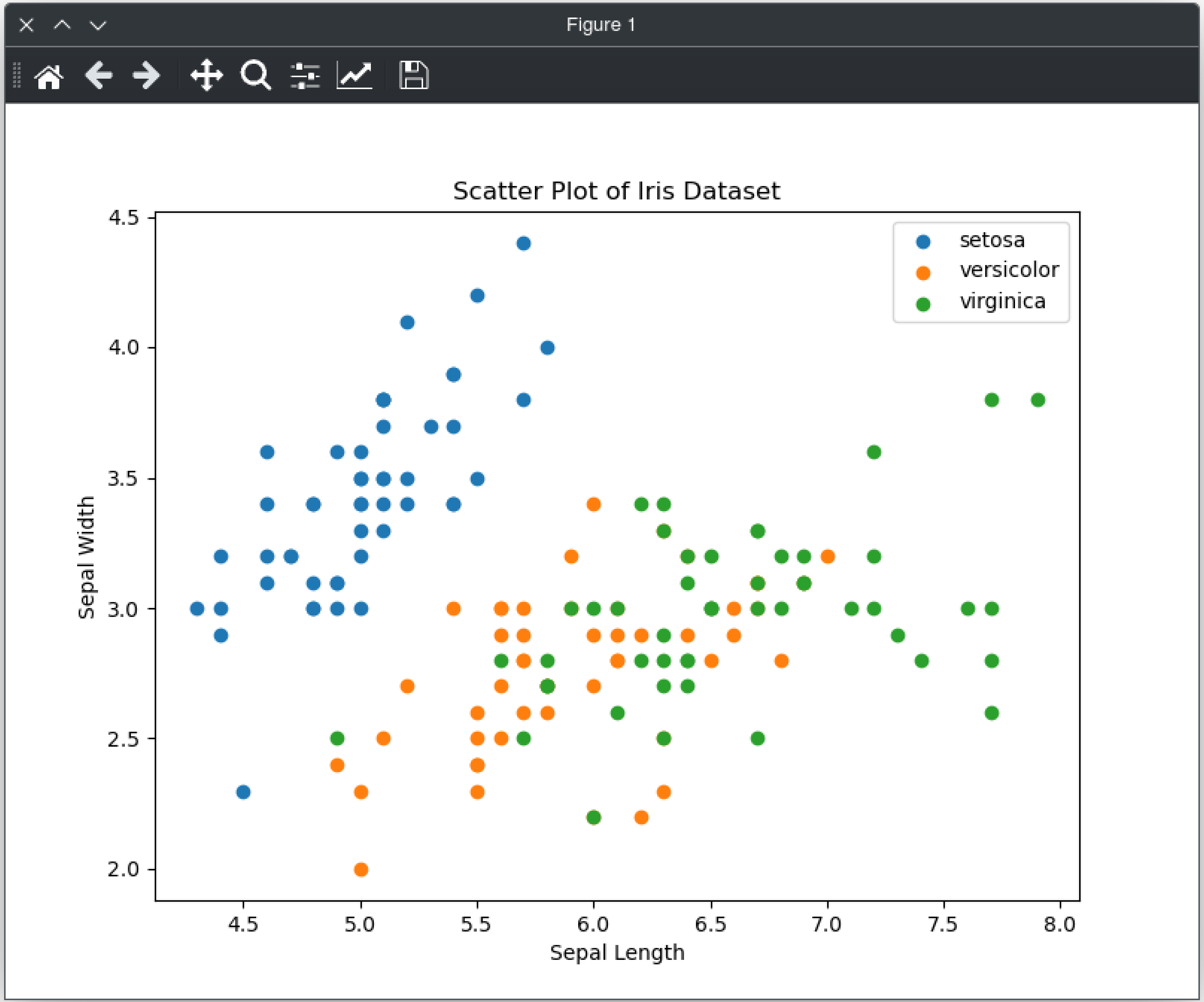


Figure 1
Scatter Plot of Iris Dataset
setosa
versicolor
virginica
Sepal Width
Sepal Length
4.5
4.0
3.5
3.0
2.5
2.0
4.5
5.0
5.5
6.0
6.5
7.0
7.5
8.0

Matplotlib

Advantages:

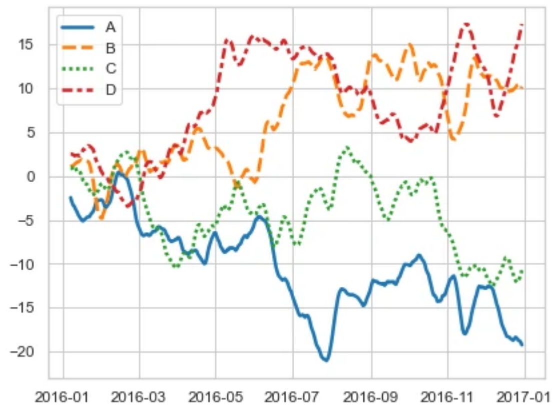
- The most commonly used.
- Work locally.
- Huge community.

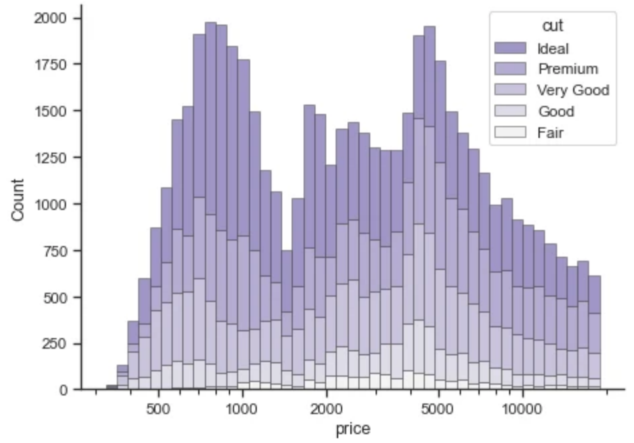
Disadvantages:

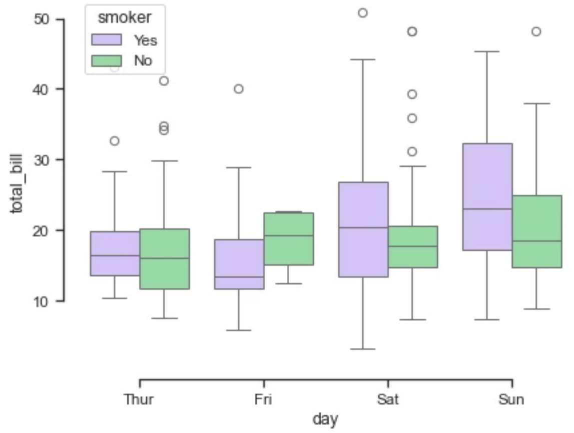
- Verbose.
- Not well adapted for dynamic plots (especially with many data points).
- Can be difficult to do some specific plots.

Seaborn

- Based on Matplotlib.
- Designed for statistical plots.







Seaborn - scatter plot of iris sepal

```python
import matplotlib.pyplot as plt
import seaborn as sns

# Create a scatter plot
sns.scatterplot(data=df_iris, x="sepal_length", y="sepal_width", hue="species")

# Add labels and title
plt.xlabel('Sepal length')
plt.ylabel('Sepal width')
plt.title('Scatter Plot of Sepal')

# Display the plot
plt.show()
```

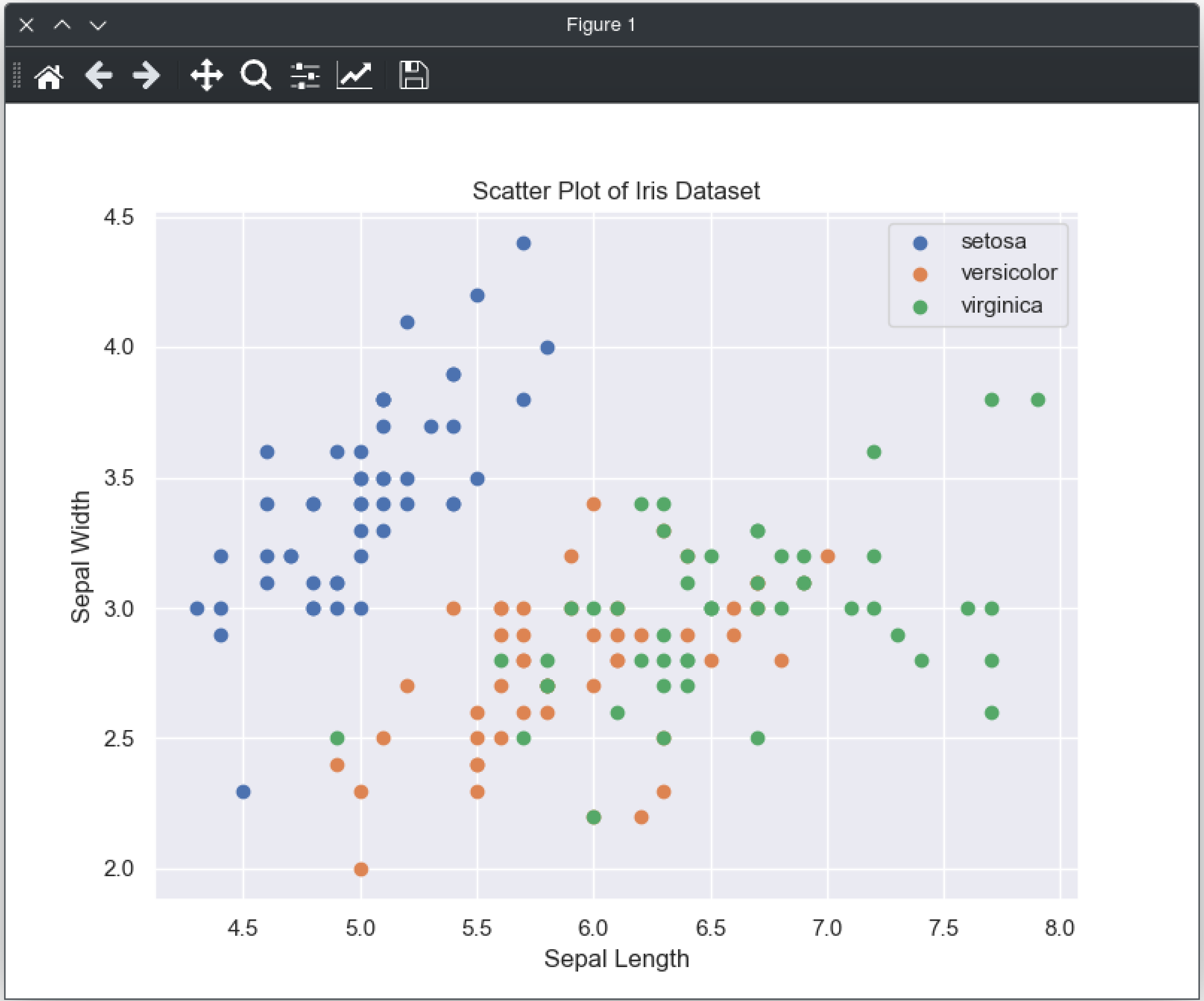
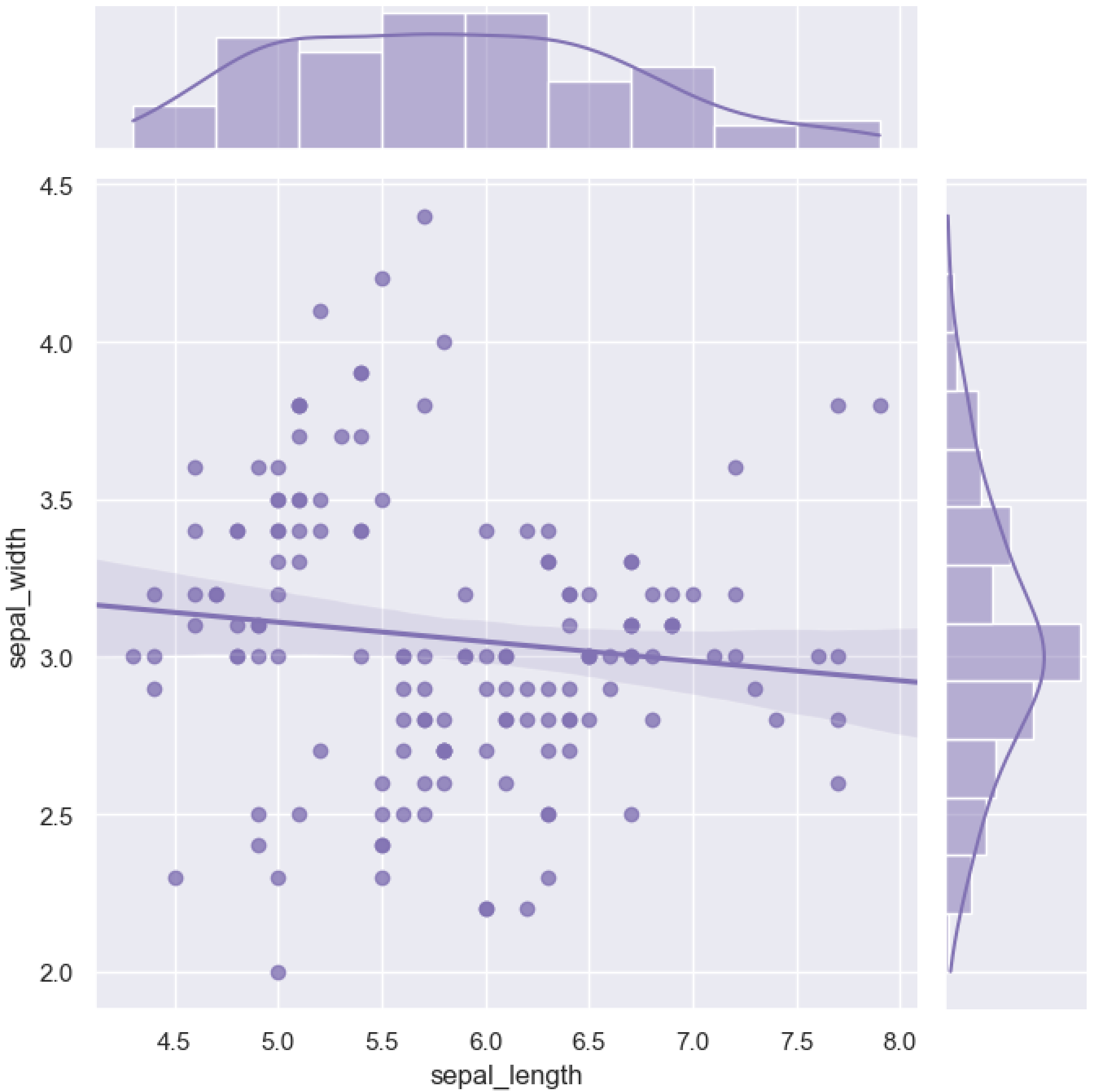


Figure 1
Scatter Plot of Iris Dataset
● setosa
● versicolor
● virginica
Sepal Width
Sepal Length
4.5
4.0
3.5
3.0
2.5
2.0
4.5
5.0
5.5
6.0
6.5
7.0
7.5
8.0

Seaborn - quickly do stats

```python
import matplotlib.pyplot as plt
import seaborn as sns

# Compute jointplot
sns.set_theme(style="darkgrid")
g = sns.jointplot(data=df_iris, x="sepal_length", y="sepal_width",
                  kind="reg", truncate=False,
                  color="m", height=7)

# Display the plot
plt.show()
```


4.5
4.0
3.5
3.0
2.5
2.0
sepal_width
4.5
5.0
5.5
6.0
6.5
7.0
7.5
8.0
sepal_length

Seaborn

Advantages:

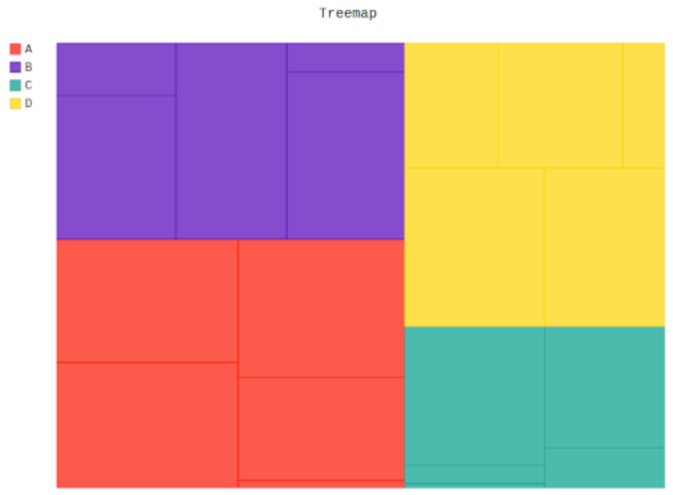
- Provides themes.
- Provide high-level functions for statistical analysis to ease some plots.

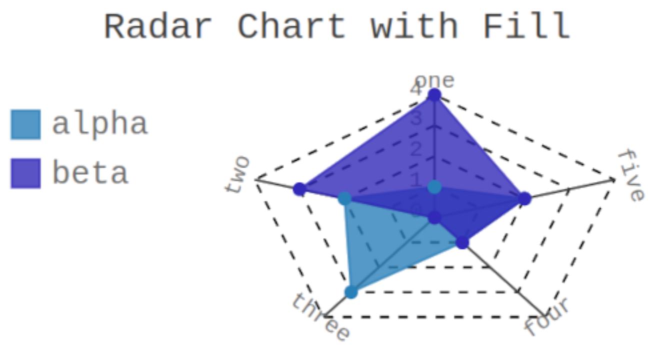
Disadvantages:

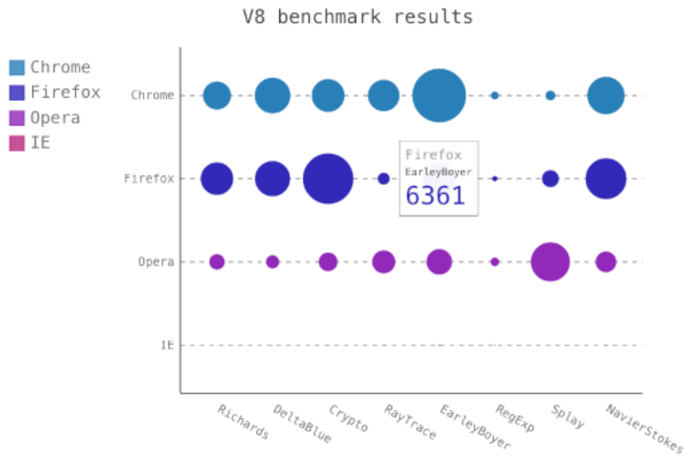
- Still a little bit verbose when modifying some plots.
- Not designed for dynamic plots.
- Not able to generate all kinds of plots.

Pygal

- Produce dynamic SVG.
- Written in Python.

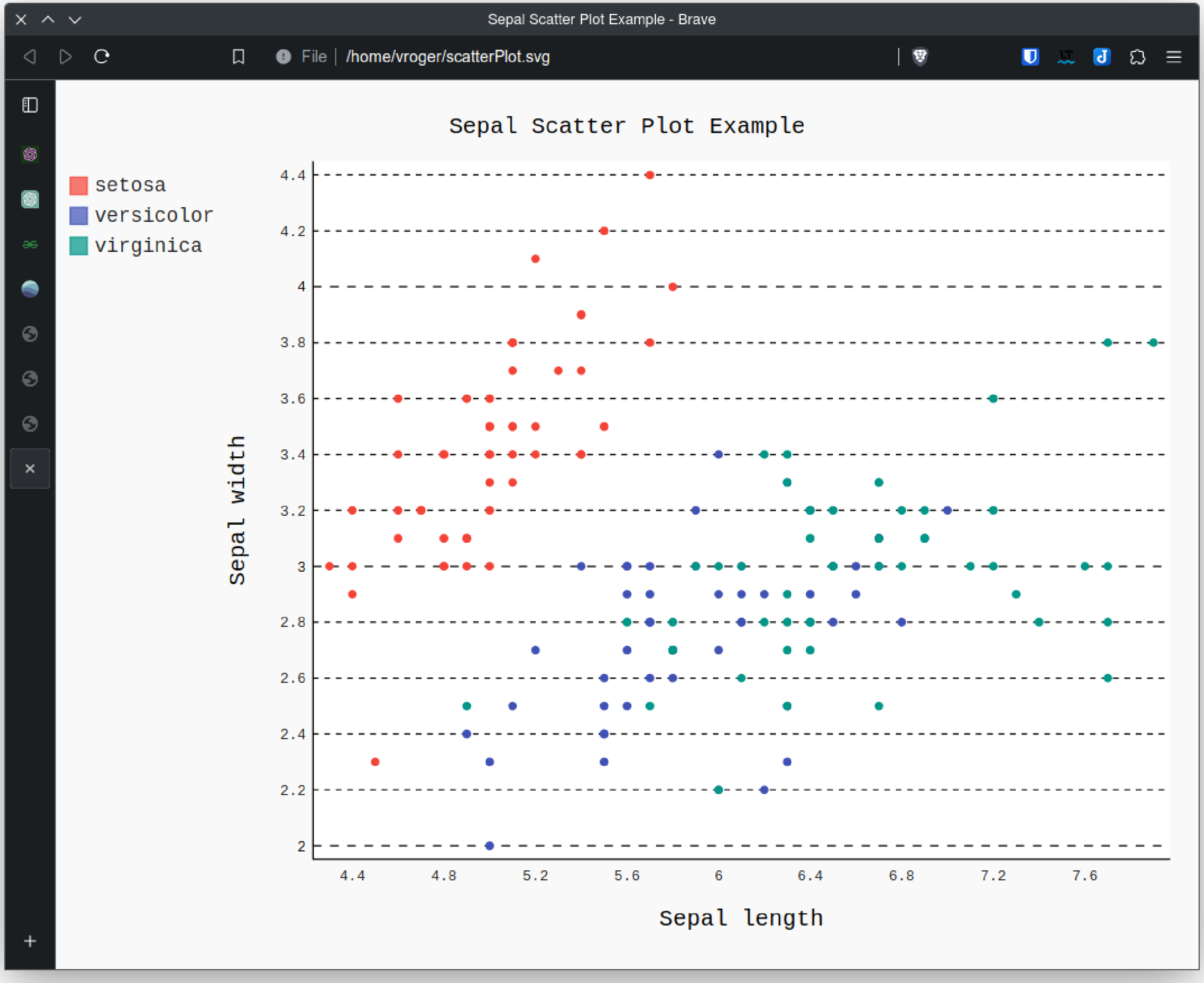






Pygal - scatter plot of iris sepal

```python
import pygal

scatter_chart = pygal.XY(stroke=False)
for specie in df_iris["species"].unique():
    df_s = df_iris[df_iris["species"] == specie]
    scatter_chart.add(f'{specie}', list(zip(df_s["sepal_length"], df_s["sepal_width"])))

scatter_chart.title = 'Sepal Scatter Plot Example'
scatter_chart.x_title = 'Sepal length'
scatter_chart.y_title = 'Sepal width'
scatter_chart.render_to_file('scatterPlot.svg')
```


Sepal Scatter Plot Example - Brave
File | /home/vroger/scatterPlot.svg
Sepal Scatter Plot Example
setosa
versicolor
virginica
Sepal width
4.4
4.2
4
3.8
3.6
3.4
3.2
3
2.8
2.6
2.4
2.2
2
4.4
4.8
5.2
5.6
6
6.4
6.8
7.2
7.6
Sepal length

Pygal

Advantages

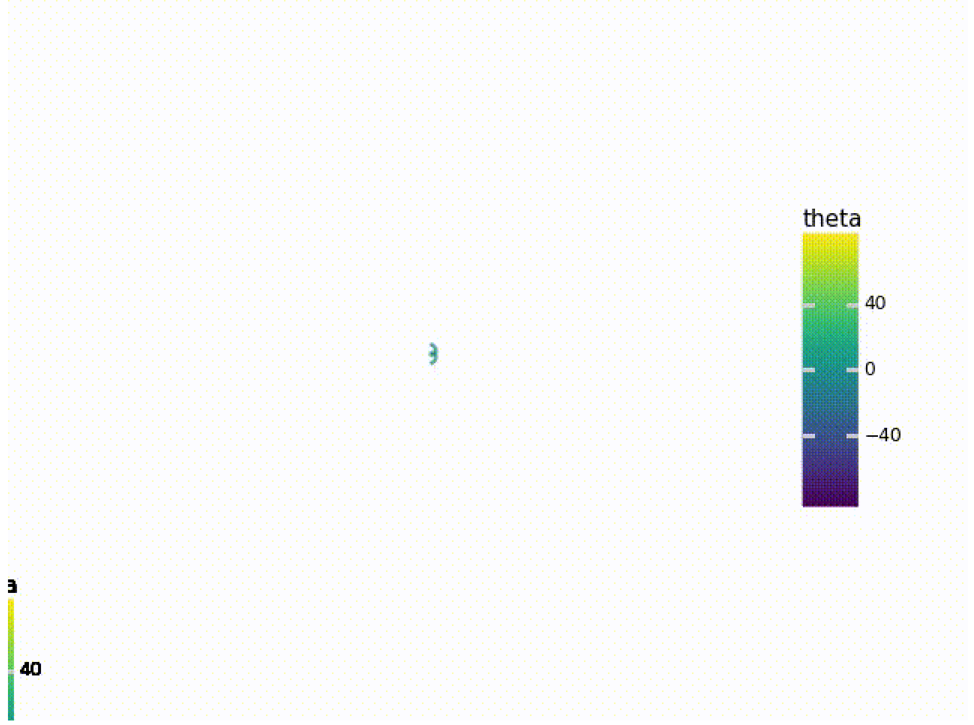
- Even if the output is SVG files, it is interactive by default!
- Great variety of charts.

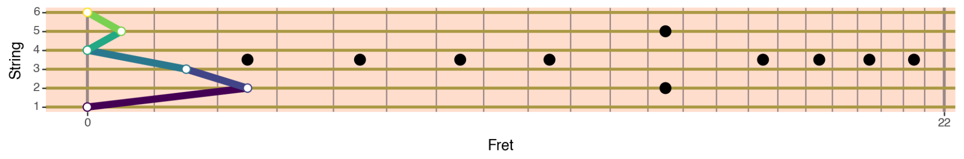
Disadvantages

- SVG compatible with web browsers, but not with software such as Inkscape.
- Chart is not clear with the defaults (especially the grid).

Plotnine

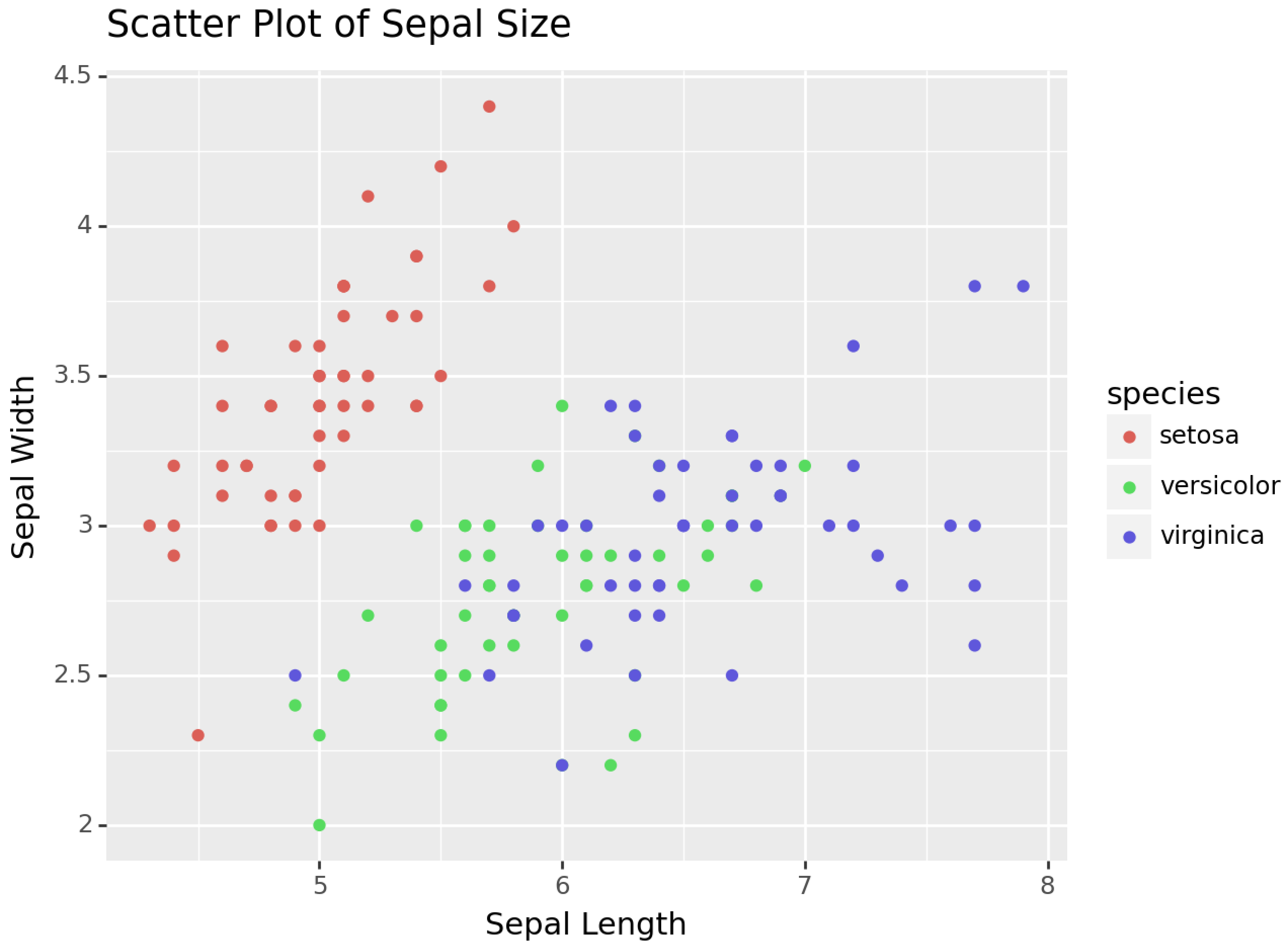
- Implementation of ggplot2's grammar in Python.





Plotnine - scatter plot of iris sepal

```python
from plotnine import ggplot, aes, geom_point, labs

# Create a scatter plot of sepal size
scatter_plot = (
    ggplot(df_iris, aes(x='sepal_length', y='sepal_width', color='species')) +
    geom_point() +
    labs(title='Iris Dataset Scatter Plot', x='Sepal Length', y='Sepal Width')
)

# Display the plot
print(scatter_plot)
```



Scatter Plot of Sepal Size
Sepal Width
4.5
4
3.5
3
2.5
2
5
6
7
8
Sepal Length
species
setosa
versicolor
virginica

Plotnine

Avantages

- As powerful as ggplot2.
- Tool adapted for R users using Python.
- Huge variety of plots possible.

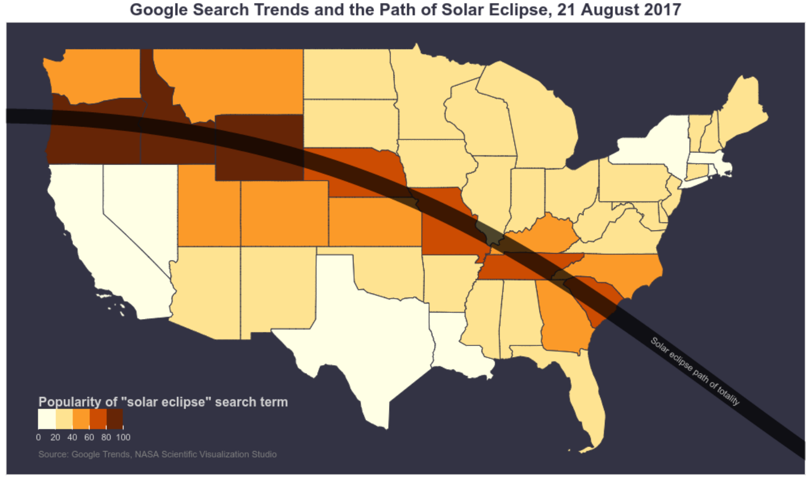
Disadvantages

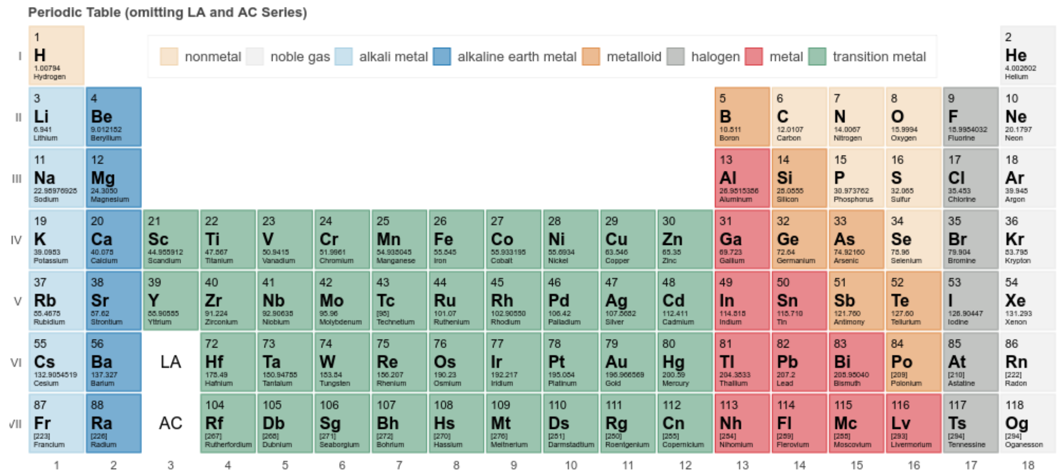
- Syntax is not Pythonic.
- Not for plotting large datasets.
- Plots limited in interactivity.

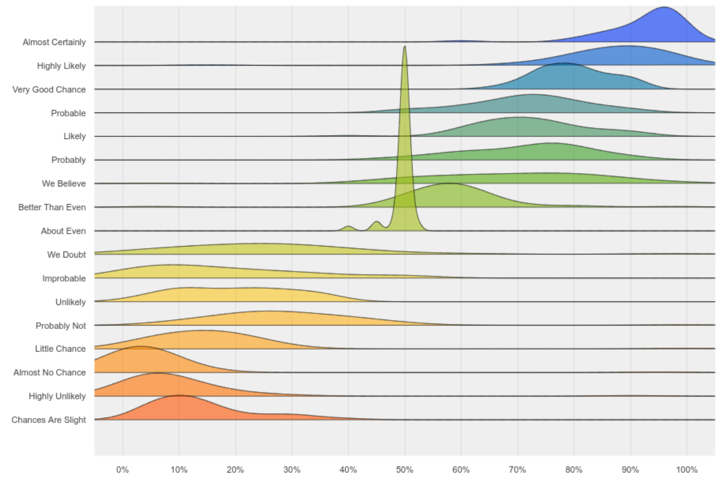
Web oriented

Bokeh

- Usable in Jupyter notebooks and IPython (output HTML by default).
- Capacity to create dashboards.
- Usage of WebGL is possible.

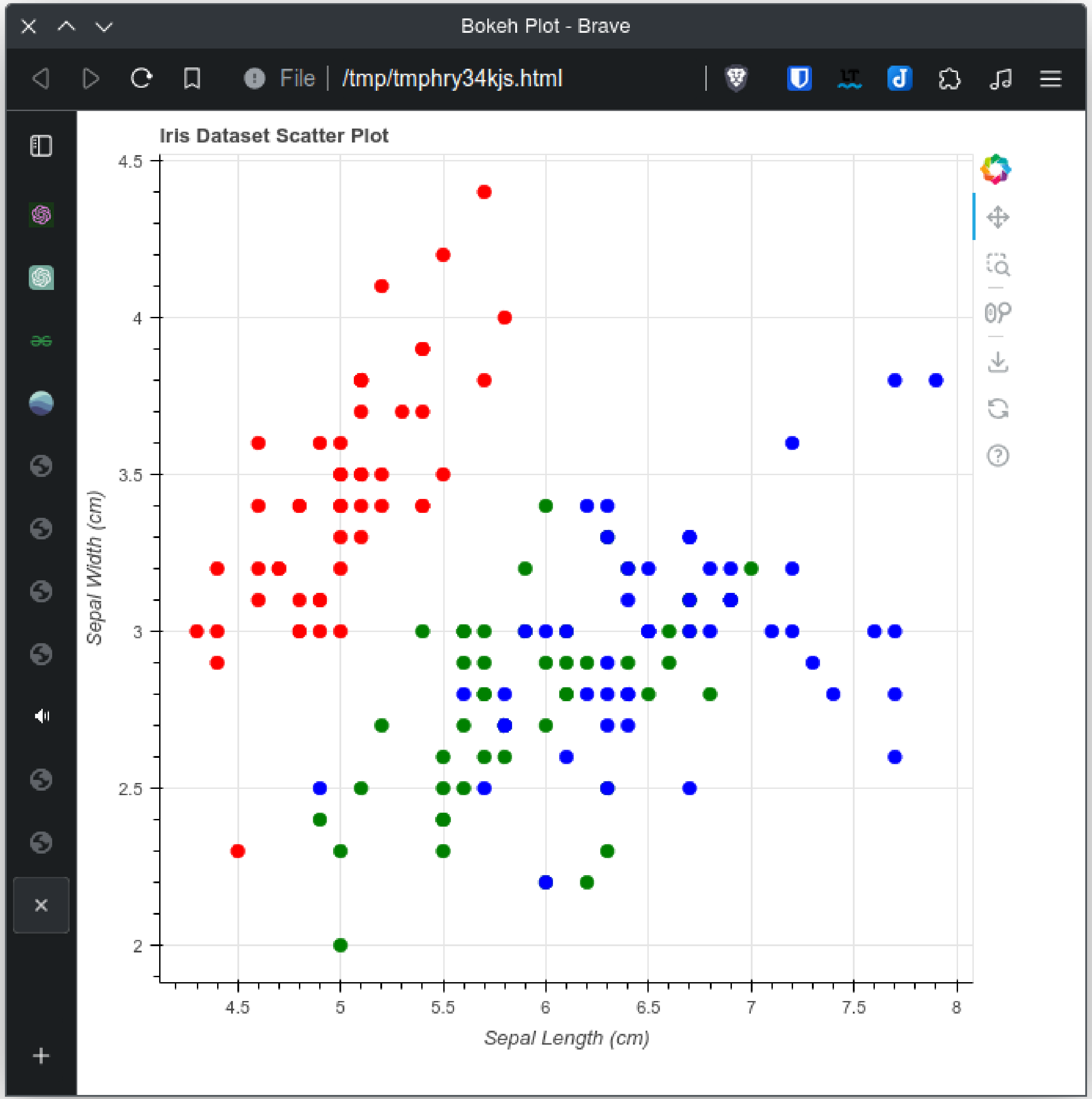

Google Search Trends and the Path of Solar Eclipse, 21 August 2017
Popularity of "solar eclipse" search term
0 20 40 60 80 100
Solar eclipse path of totality
Source: Google Trends, NASA Scientific Visualization Studio





Bokeh - scatter plot of iris sepal

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

# Create a ColumnDataSource
source = ColumnDataSource(df_iris)

# Create a scatter plot with titles
p = figure(title='Iris Dataset Scatter Plot', x_axis_label='Sepal Length (cm)', y_axis_label='Sepal Width (cm)')

p.circle(x='sepal_length', y='sepal_width', source=source,
               size=8,
             color=factor_cmap(
                        'species',
                         palette=['red', 'green', 'blue'],
                         factors=df_iris["species"].unique()
              )
)

# Show the plot
show(p)
```



Bokeh Plot - Brave
File
/tmp/tmphry34kjs.html
Iris Dataset Scatter Plot
Sepal Width (cm)
Sepal Length (cm)
4.5
4
3.5
3
2.5
2
4.5
5
5.5
6
6.5
7
7.5
8

Bokeh

Advantages

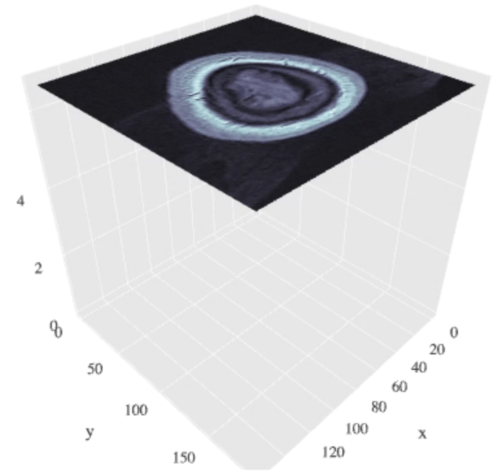
- Low level and "high" level API.
- Can handle a million points easily.
- Interactivity (widgets and dashboards).
- Capacity to do many different plots.

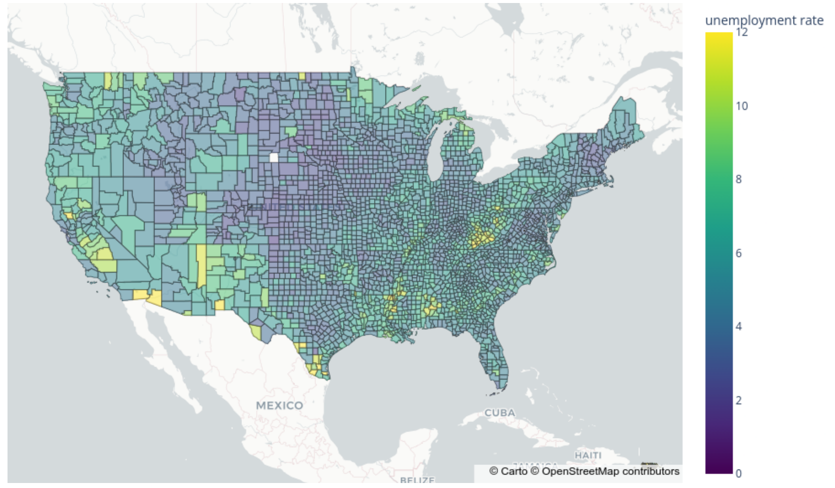
Disadvantages

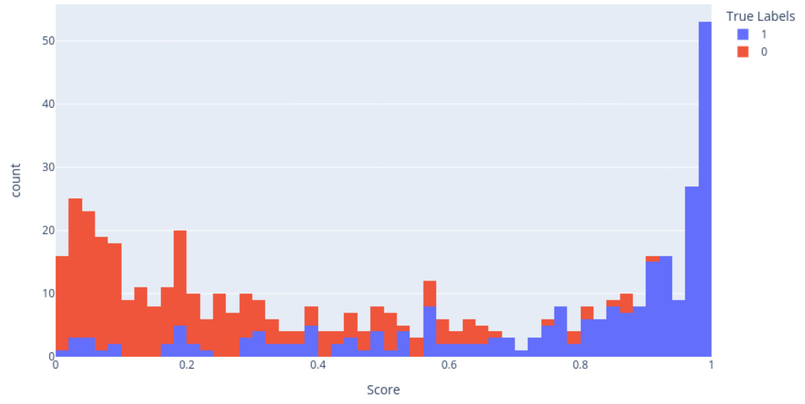
- Is harder to learn compared to other libraries.
- Less complete for 3D plots.
- If bokeh servers are down, your plots may not show properly (or at all).

Plotly

- Designed for interactive visualizations.
- Usage of WebGL is possible.
- Usable with Python, R and JavaScript.

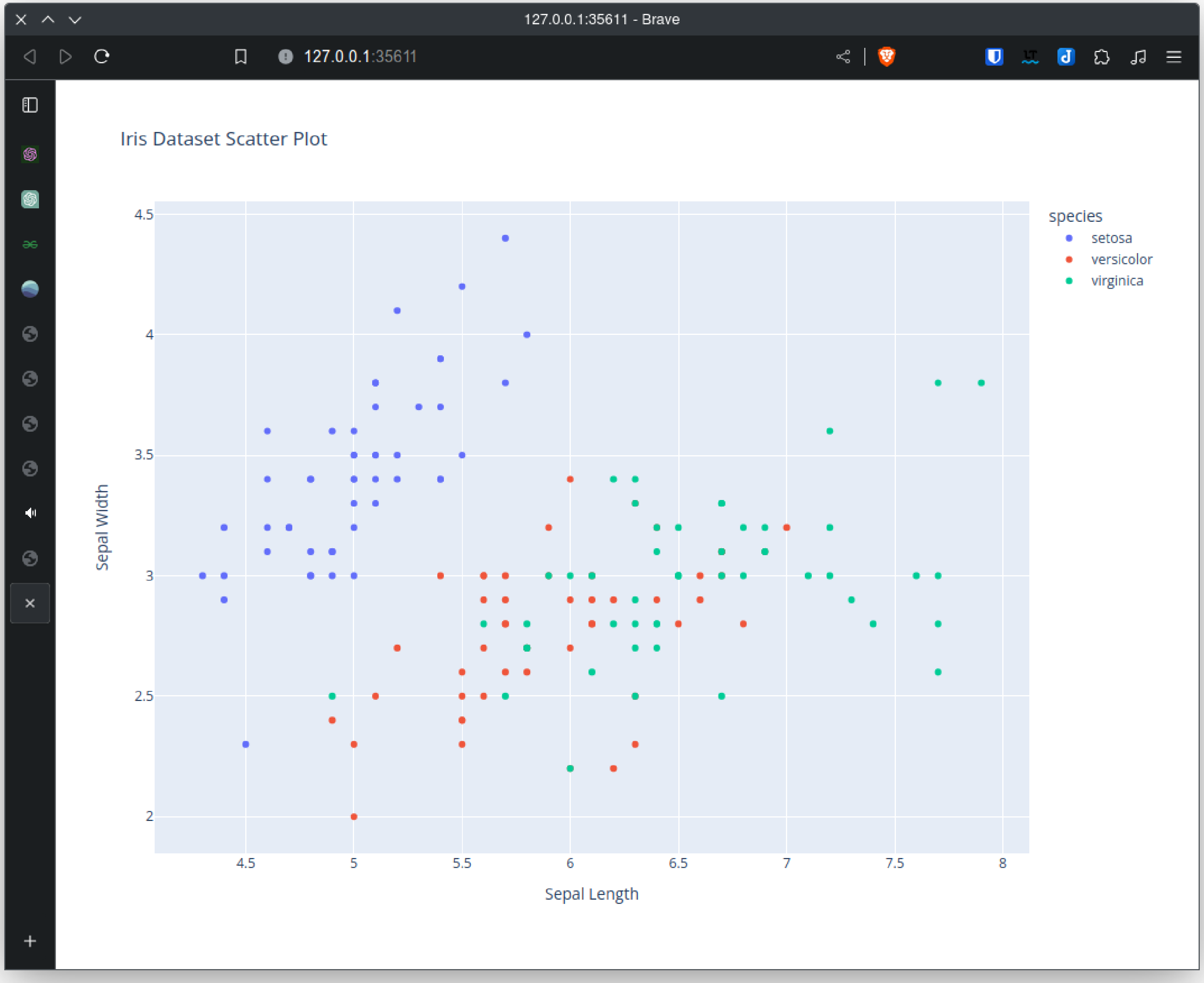






Plotly - scatter plot of iris sepal

```
import plotly.express as px

fig = px.scatter(df_iris, x='sepal_length', y='sepal_width', color='species',
                 title='Iris Dataset Scatter Plot', labels={'sepal_length': 'Sepal Length', 'sepal_width': 'Sepal Width'})

# Show the plot
fig.show()
```


127.0.0.1:35611 - Brave
127.0.0.1:35611
Iris Dataset Scatter Plot
species
● setosa
● versicolor
● virginica
Sepal Width
Sepal Length
4.5
4
3.5
3
2.5
2
4.5
5
5.5
6
6.5
7
7.5
8

Plotly

Advantages

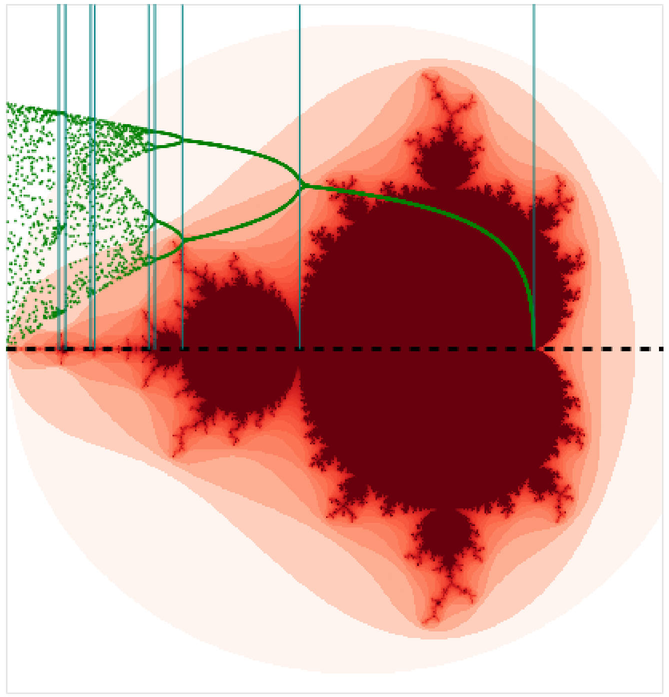
- The documentation is crystal clear for simple plots.
- Dashboard possible with Dash (well integrated).
- Auto resize plot by default.

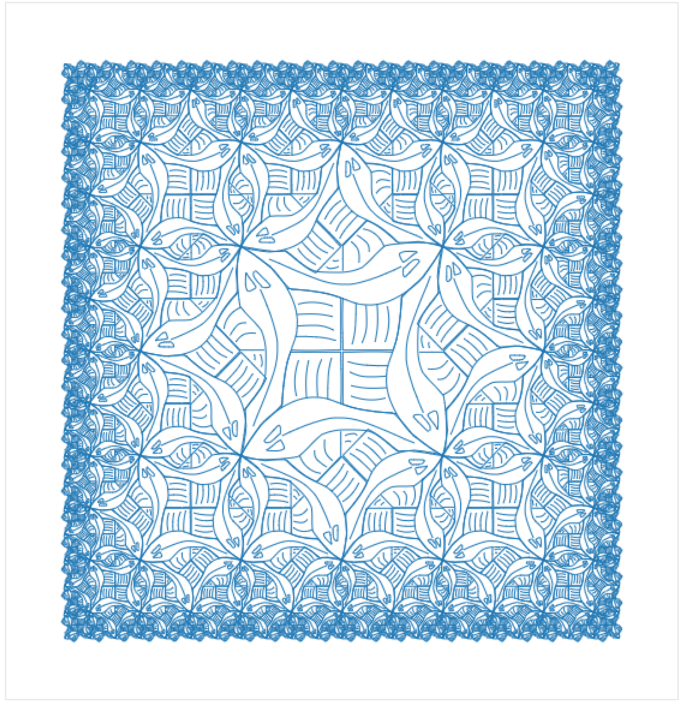
Disadvantages

- If Plotly servers are down, your plots may not show properly (or at all).
- Hard to use images as ticks.

Holoviews

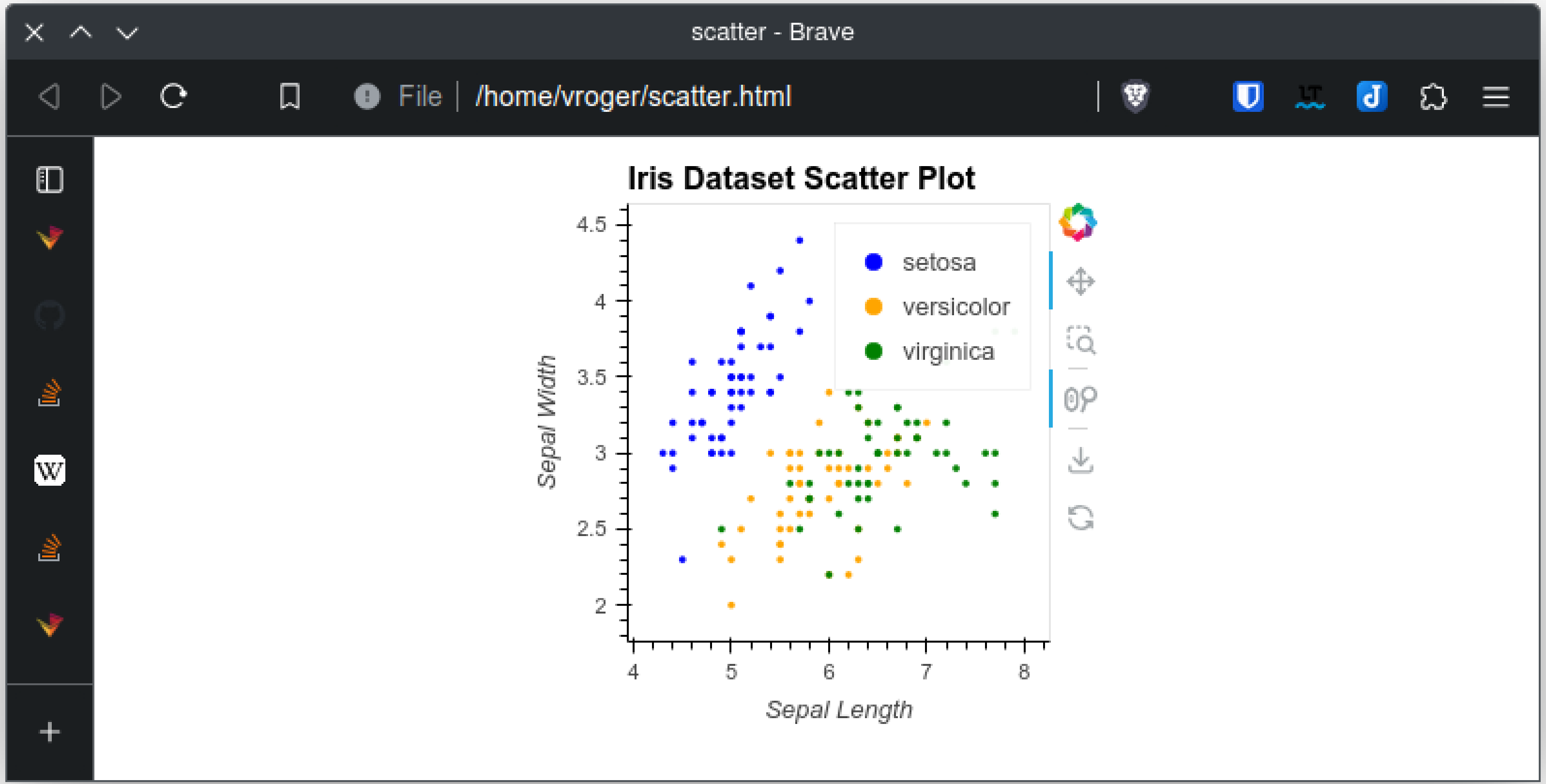
A high-level library for Bokeh, Matplotlib and Plotly.





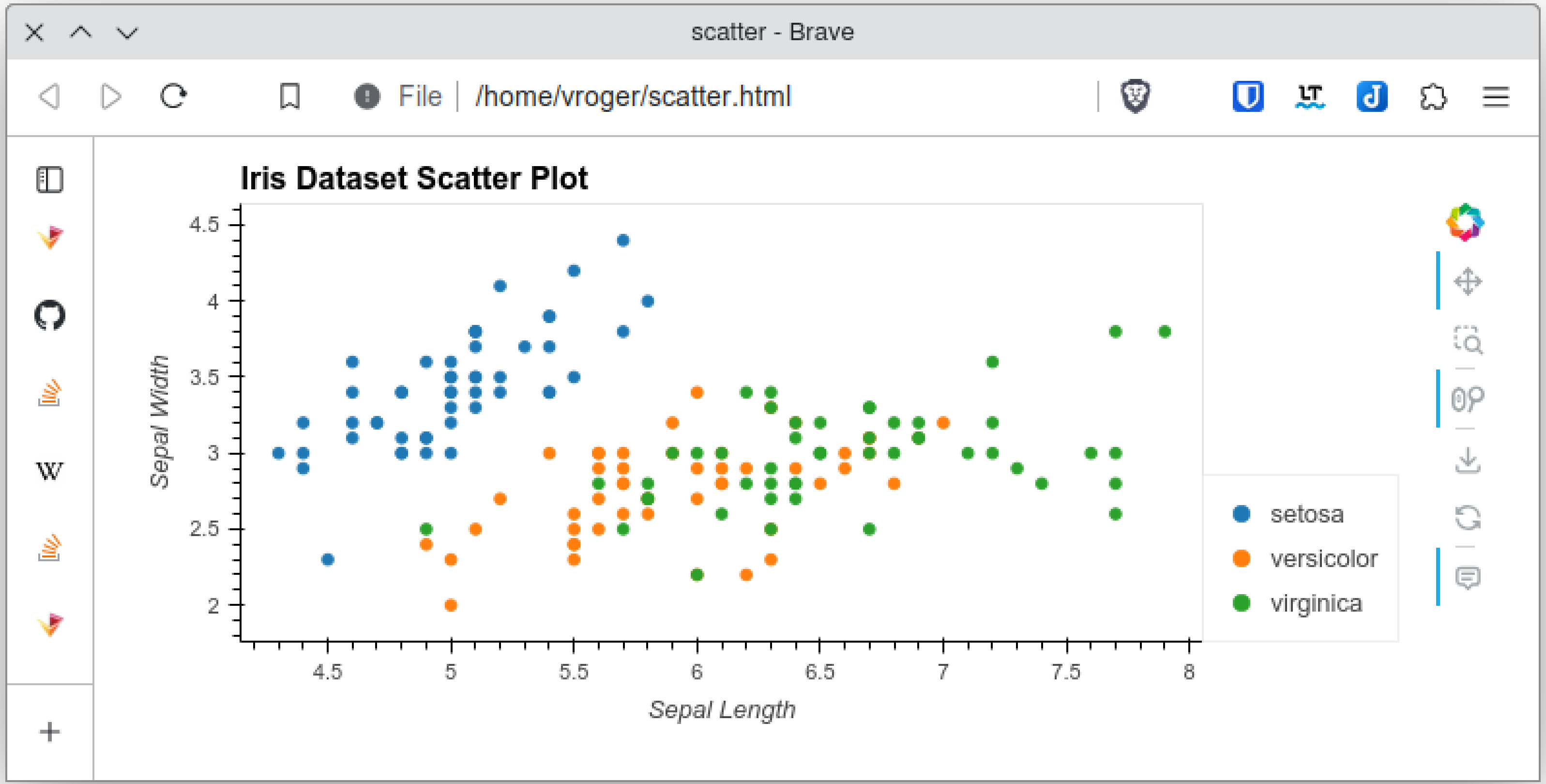
Holoviews - scatter plot of iris sepal

```python
import holoviews as hv

scatter = hv.Scatter(
    data=df_iris, kdims=['sepal_length'], vdims=['sepal_width', 'species'],
).opts(color='species', cmap=['blue', 'orange', 'green'])

scatter = scatter.opts(
    title="Iris Dataset Scatter Plot",
    xlabel='Sepal Length',
    ylabel='Sepal Width',
)

hv.save(scatter, 'scatter.html', backend='bokeh')
```

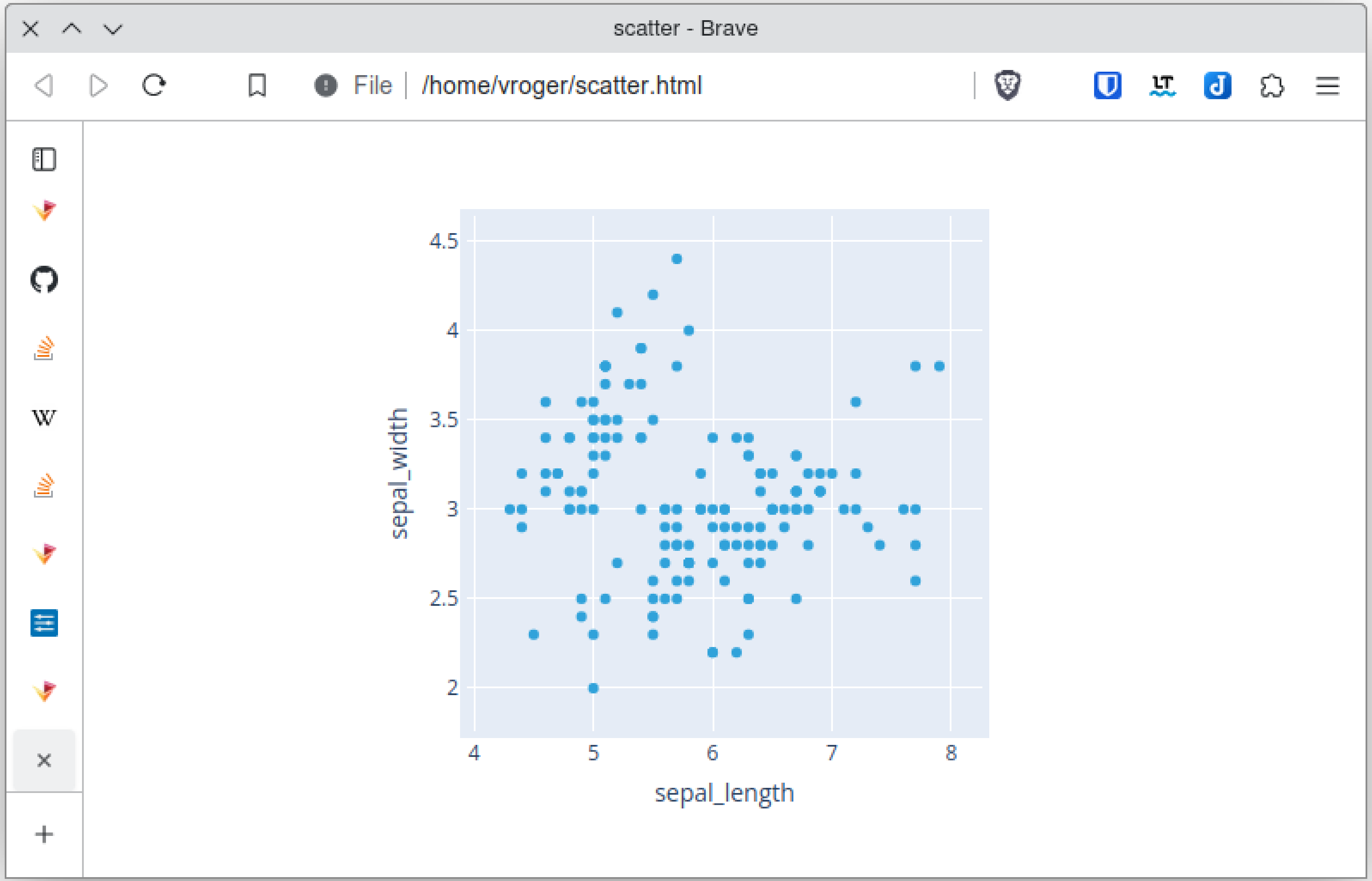

scatter - Brave
File
/home/vroger/scatter.html
Iris Dataset Scatter Plot
setosa
versicolor
virginica
Sepal Width
Sepal Length
4.5
4
3.5
3
2.5
2
4
5
6
7
8

Holoviews - scatter plot of iris sepal using pandas API

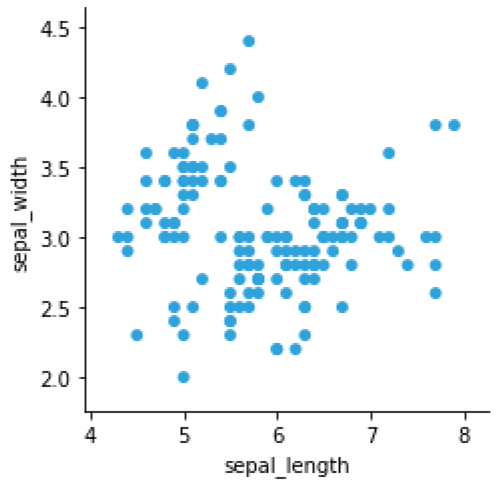
```python
import hvplot
import hvplot.pandas
import holoviews as hv

scatter = df_iris.hvplot(kind='scatter', x='sepal_length', y='sepal_width', color='species')

scatter = scatter.opts(
    title="Iris Dataset Scatter Plot",
    xlabel='Sepal Length',
    ylabel='Sepal Width',
)

hv.save(scatter, 'scatter.html', backend='bokeh')
```


scatter - Brave
File
/home/vroger/scatter.html
Iris Dataset Scatter Plot
Sepal Width
Sepal Length
4.5
4
3.5
3
2.5
2
4.5
5
5.5
6
6.5
7
7.5
8
setosa
versicolor
virginica

```python
hv.save(scatter, 'scatter.html', backend='plotly')
```


scatter - Brave
File
/home/vroger/scatter.html
sepal_width
sepal_length
4.5
4
3.5
3
2.5
2
4
5
6
7
8

```
hv.save(scatter, 'scatter.png', backend='matplotlib')
```


4.5
4.0
3.5
3.0
2.5
2.0
sepal_width
4
5
6
7
8
sepal_length

NOT THAT GENERIC!
imgflip.com

Holoviews

Advantages

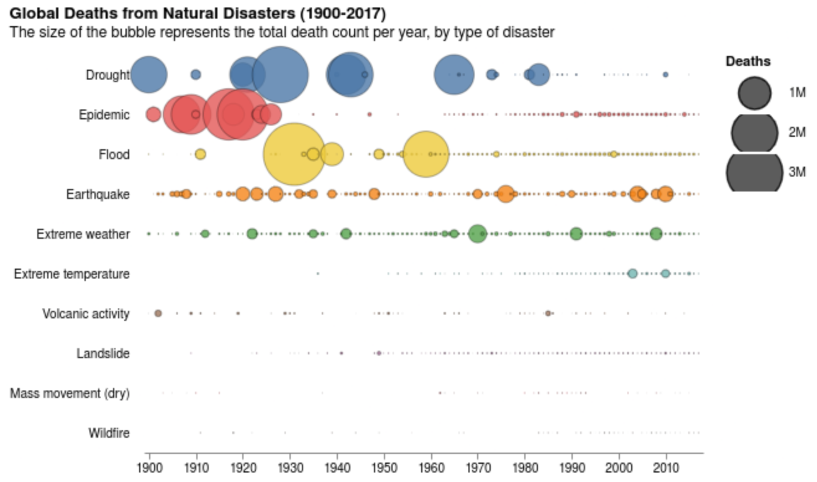
- Works well with Bokeh.
- Hight level API.

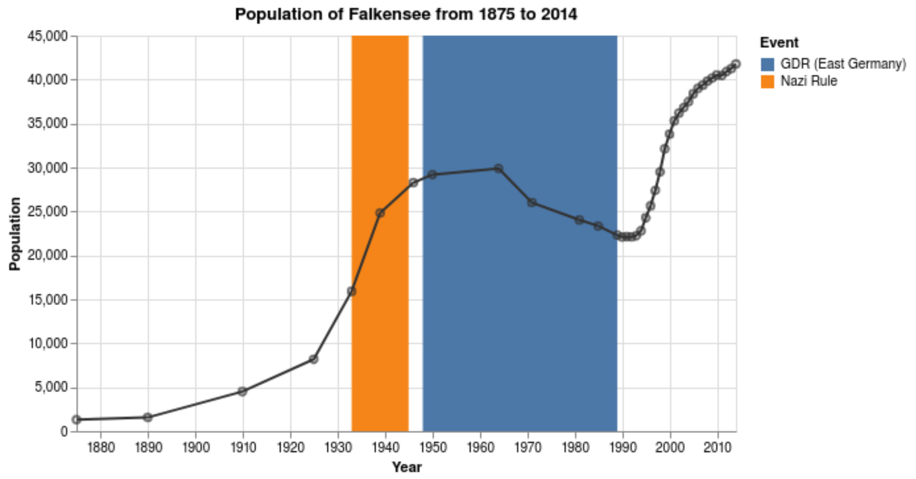
Disadavantages

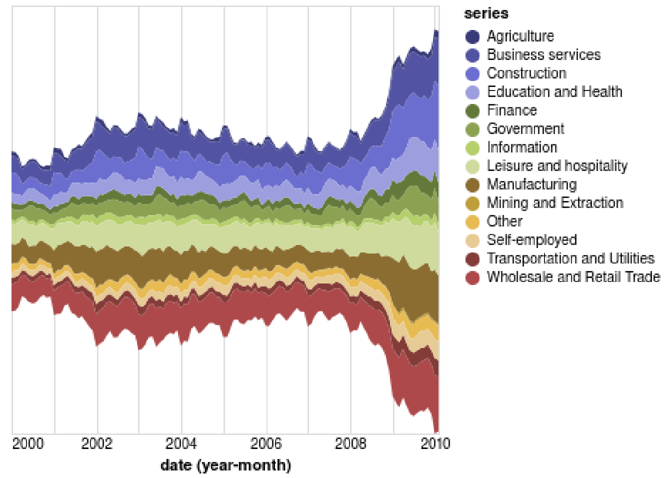
- Not generic (at least for now).

Altair

- Designed for notebooks.
- Use Vega engine (which produces JavaScript code).

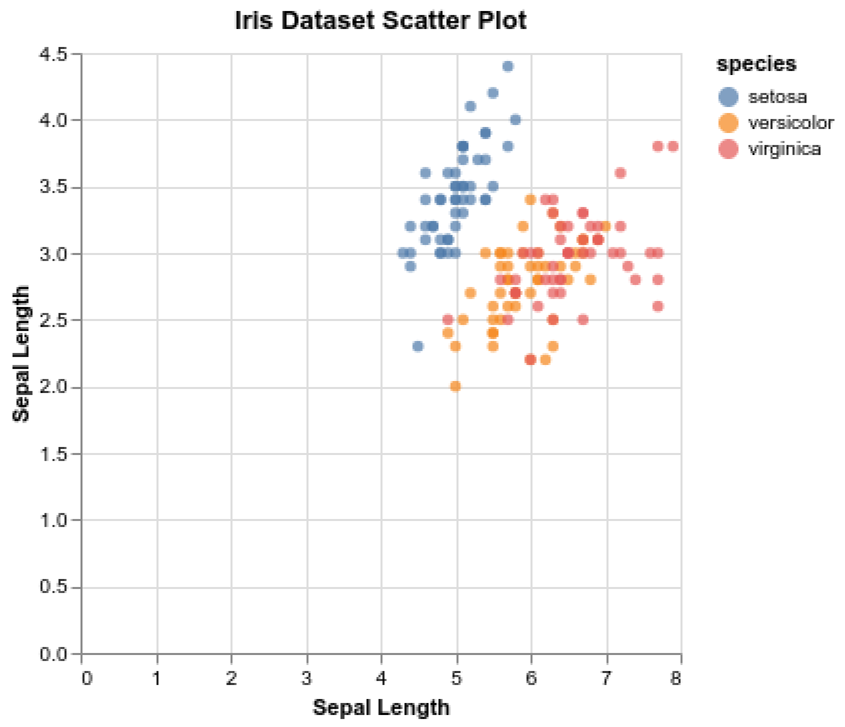






Altair - scatter plot of iris sepal

```python
import altair as alt

# Create the scatter plot
scatter_plot = alt.Chart(df_iris).mark_circle().encode(
    x=alt.X('sepal_length', title='Sepal Length'),
    y=alt.Y('sepal_width', title='Sepal Length'),
    color='species'
).properties(
    title='Iris Dataset Scatter Plot'
)

# Show the plot
scatter_plot
```


Iris Dataset Scatter Plot
Sepal Length
Sepal Length
species
setosa
versicolor
virginica
4.5
4.0
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
0
1
2
3
4
5
6
7
8

Altair

Advantages

- Simple use of dataframes.
- Better performance than Plotly in a VS Code notebook.

Disadvantages

- Defaults are not that great for statistical plots.
- No viewer for the last versions (5.x).
- Limited chart customization.

Conclusion

My choice

Plotly

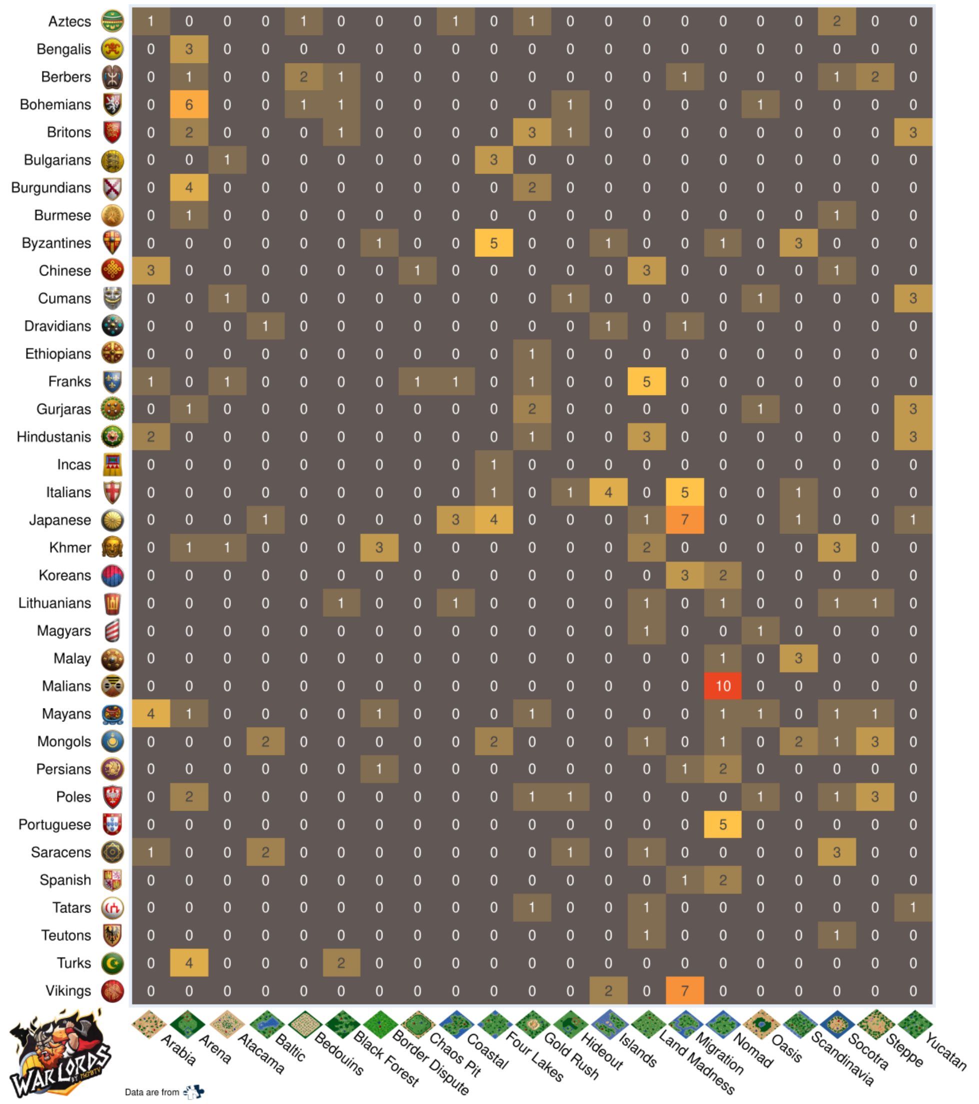
Pros

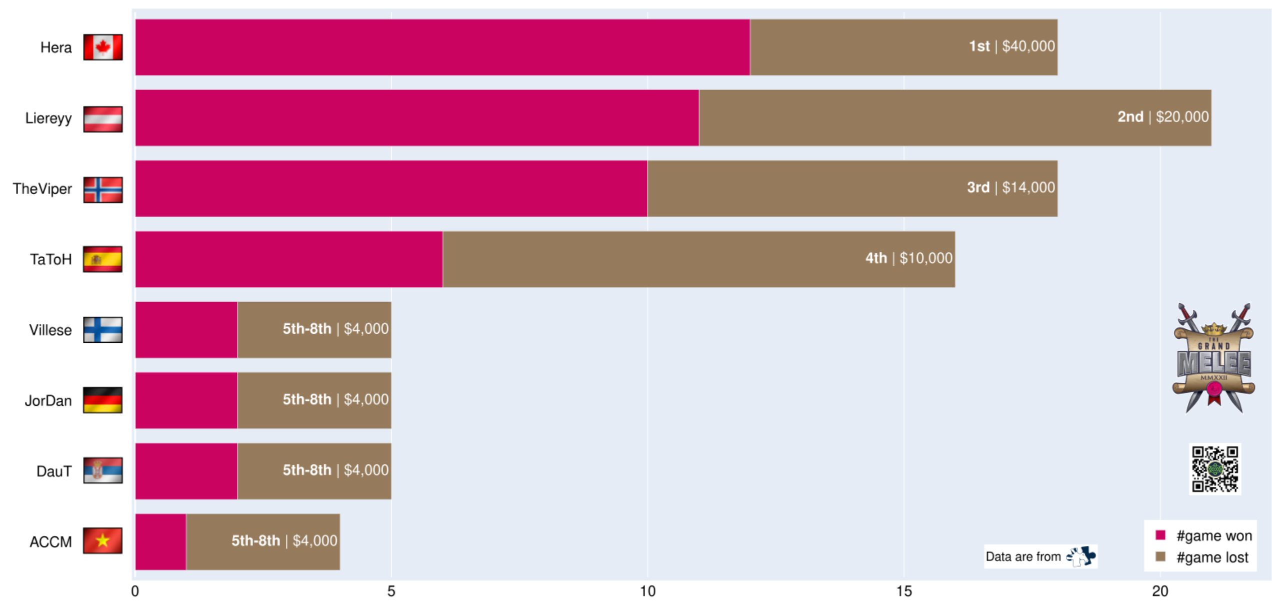
- Great defaults that reduce my codes.
- Great documentation.
- Supported by MLFlow.

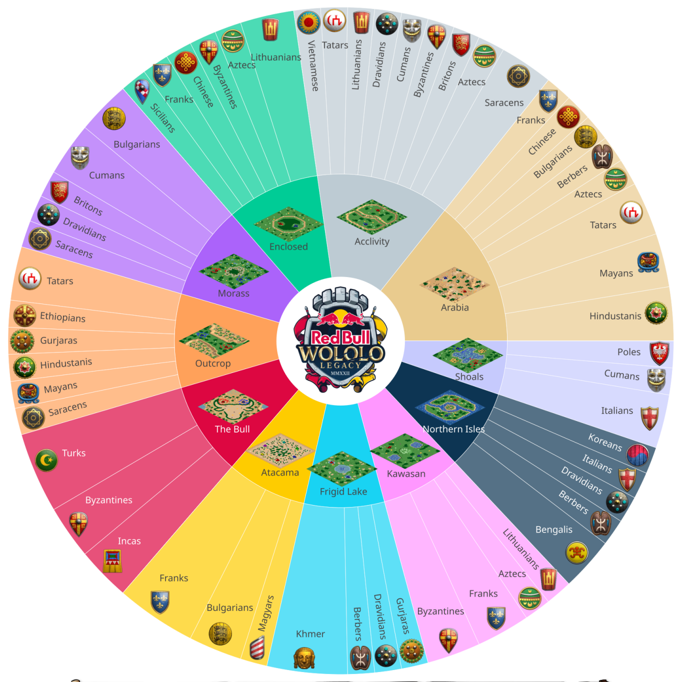
Cons

- A lot of plotting in VSCode notebook's causes slowdowns.
- Having images as ticks is a mess to implement (if you know a simple way, don't hesitate).

Some work of mine

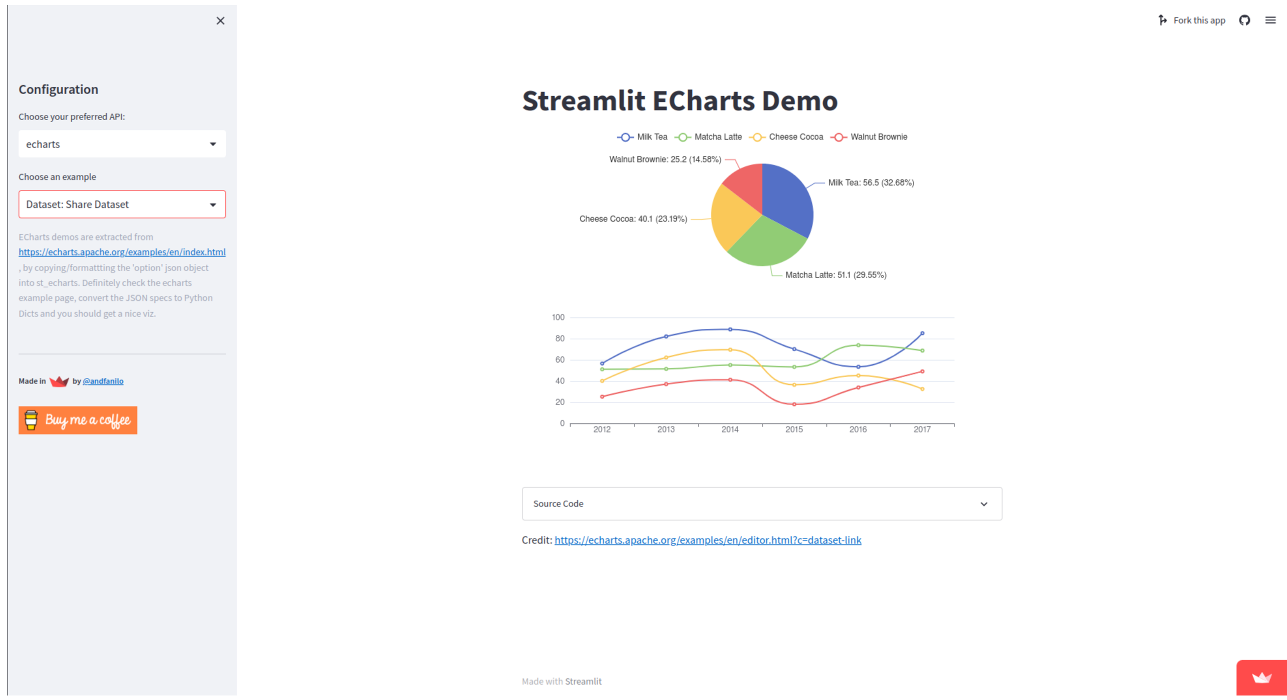






RedBull Wololo Legacy - Winning Civilizations on the Main Event

Bonus dashboard/webapp with Streamlit



Configuration

Choose your preferred API: echarts

Choose an example: Dataset: Share Dataset

ECharts demos are extracted from https://echarts.apache.org/examples/en/index.html , by copying/formattting the 'option' json object into st_echarts. Definitely check the echarts example page, convert the JSON specs to Python Dicts and you should get a nice viz.

Made in by @andfanilo

Buy me a coffee

Fork this app

Streamlit ECharts Demo

Source Code

Credit: https://echarts.apache.org/examples/en/editor.html?c=dataset-link

Made with Streamlit

Thank you for listening!