data-visualization-demo

November 12, 2025

1 Data Visualization Techniques Demo

This notebook demonstrates a range of data visualization techniques using Matplotlib, Seaborn, and Plotly. Datasets used: `iris`, `tips`, and `flights` from seaborn.

1.1 Setup: installs and imports

Run the next cell to import libraries. If Plotly is not installed, the cell will install it programmatically.

```
libraries loaded
```

1.2 Load example datasets

We'll load `iris`, `tips`, and `flights` from seaborn and inspect them briefly.

```
iris: (150, 5)

   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
```

```
tips: (244, 7)

   total_bill   tip     sex smoker  day    time  size
0       16.99  1.01  Female     No  Sun  Dinner     2
1       10.34  1.66    Male     No  Sun  Dinner     3
2       21.01  3.50    Male     No  Sun  Dinner     3
3       23.68  3.31    Male     No  Sun  Dinner     2
4       24.59  3.61  Female     No  Sun  Dinner     4
```

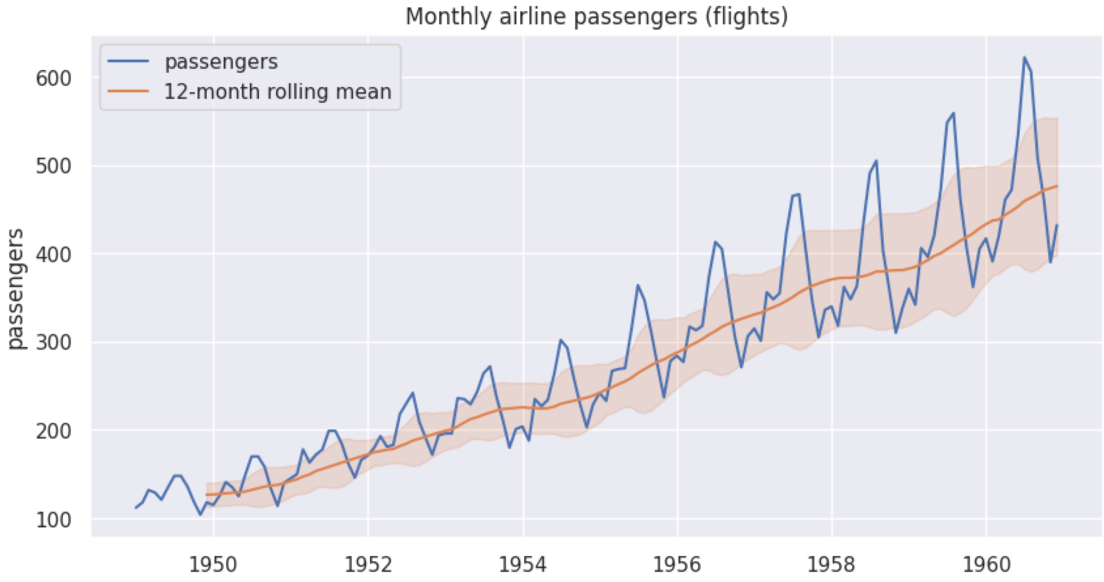
```
flights: (144, 3)

   year month  passengers
0  1949   Jan         112
1  1949   Feb         118
2  1949   Mar         132
3  1949   Apr         129
4  1949   May         121
```

1.3 Line plots & time series (Matplotlib / pandas)

Aggregate the `flights` dataset into a time series and demonstrate line plotting with rolling mean and shaded confidence band.

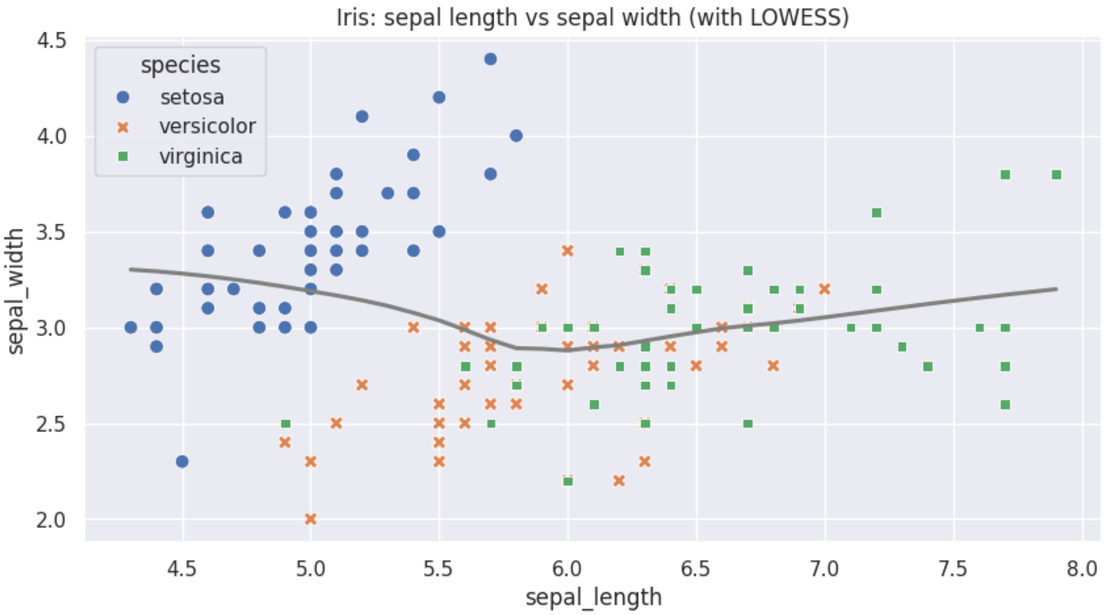
```
<matplotlib.legend.Legend at 0x7fd5a43e27b0>
```



1.4 Scatter plots and regression overlays

Scatter plot for `iris` (sepal length vs sepal width) with Seaborn regression overlay.

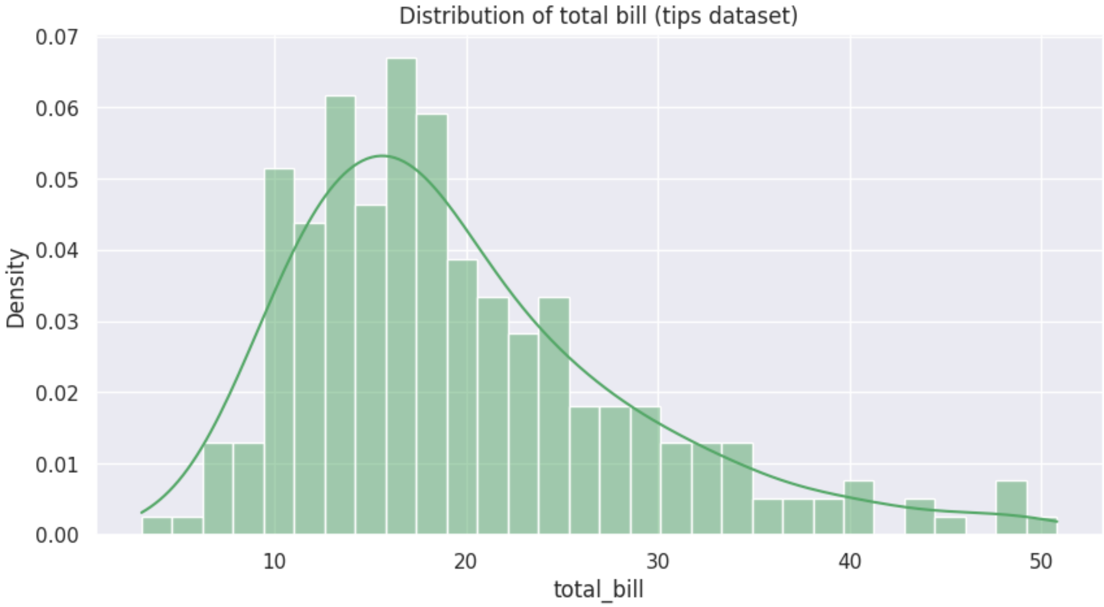
```
<matplotlib.legend.Legend at 0x7fd5a4d1af60>
```



1.5 Histograms and KDE

Histogram and KDE of `tips.total_bill` comparing density and bins.

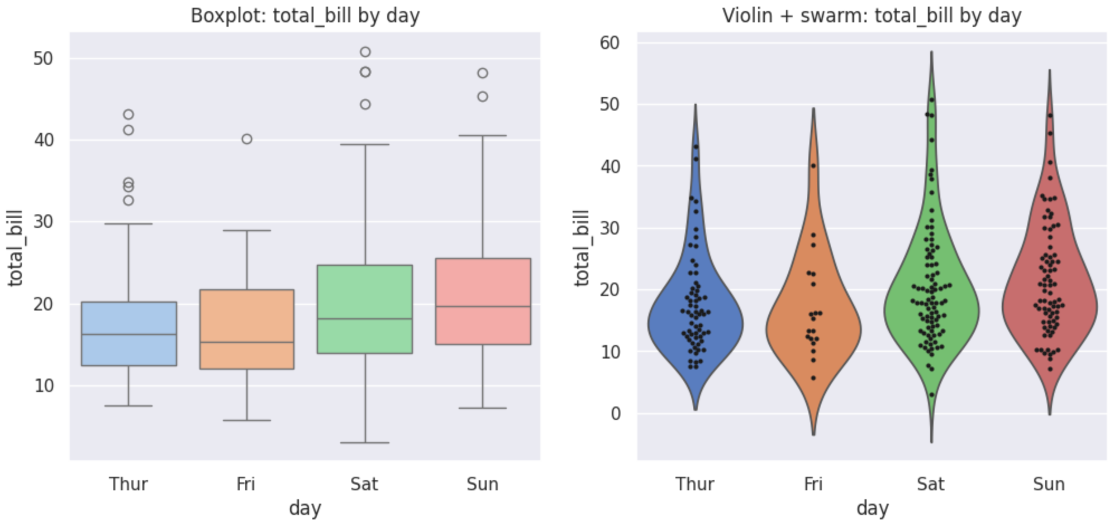
```
Text(0.5, 1.0, 'Distribution of total bill (tips dataset)')
```



1.6 Boxplots and violin plots

Boxplot of total bill by day and violinplot with swarm overlay.

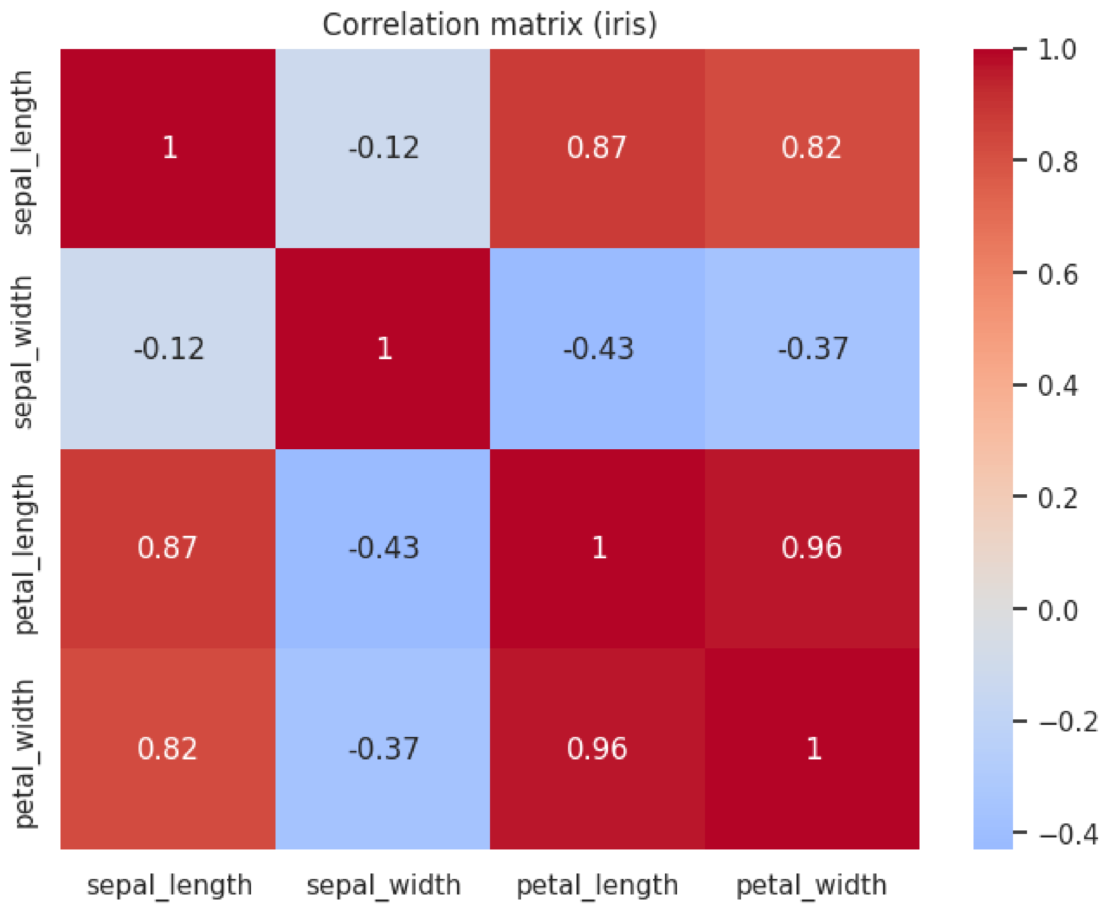
```
Text(0.5, 1.0, 'Violin + swarm: total_bill by day')
```



1.7 Heatmap and correlation matrix

Show correlations for the `iris` dataset.

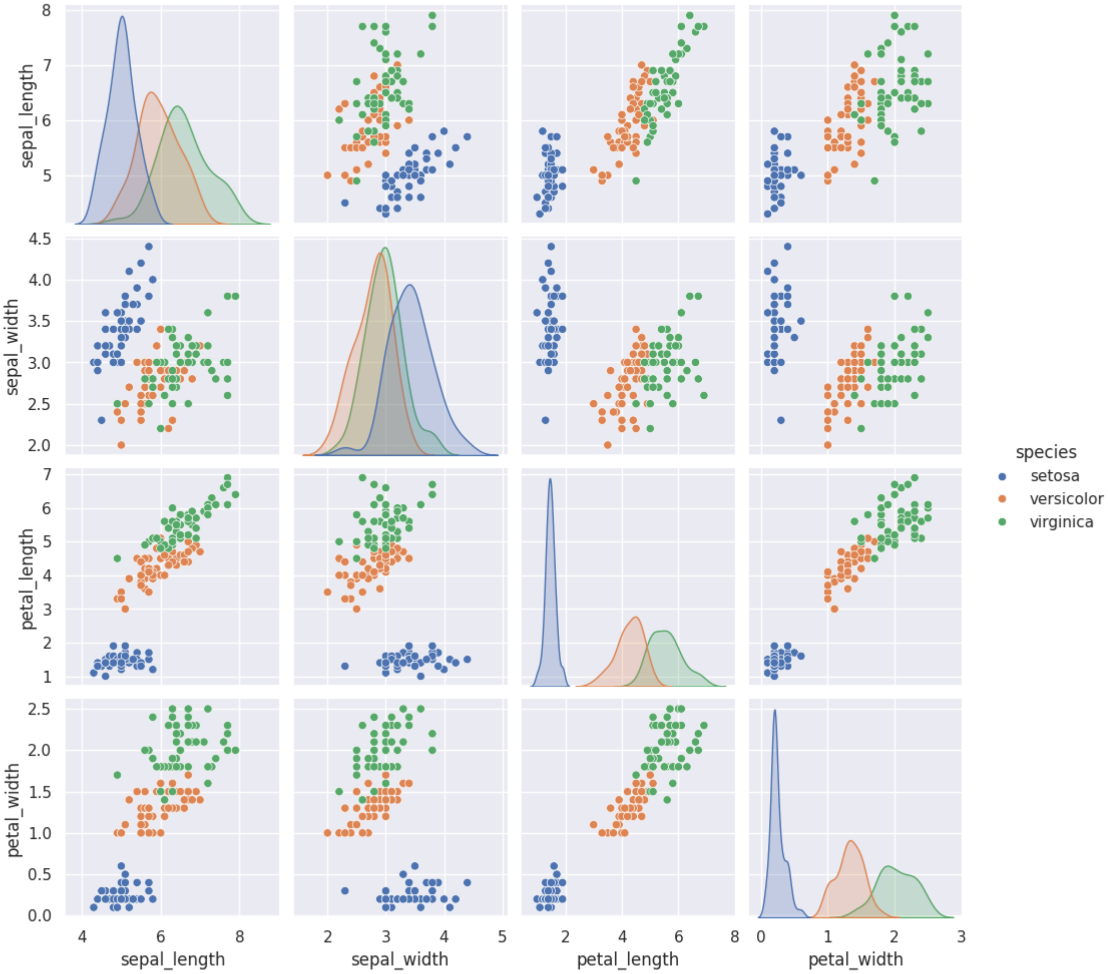
```
Text(0.5, 1.0, 'Correlation matrix (iris)')
```



1.8 Pairplot for multivariate exploration

Pairwise relationships with hue by species. This can be slow on large datasets.

```
<seaborn.axisgrid.PairGrid at 0x7fd5a4d1af90>
```



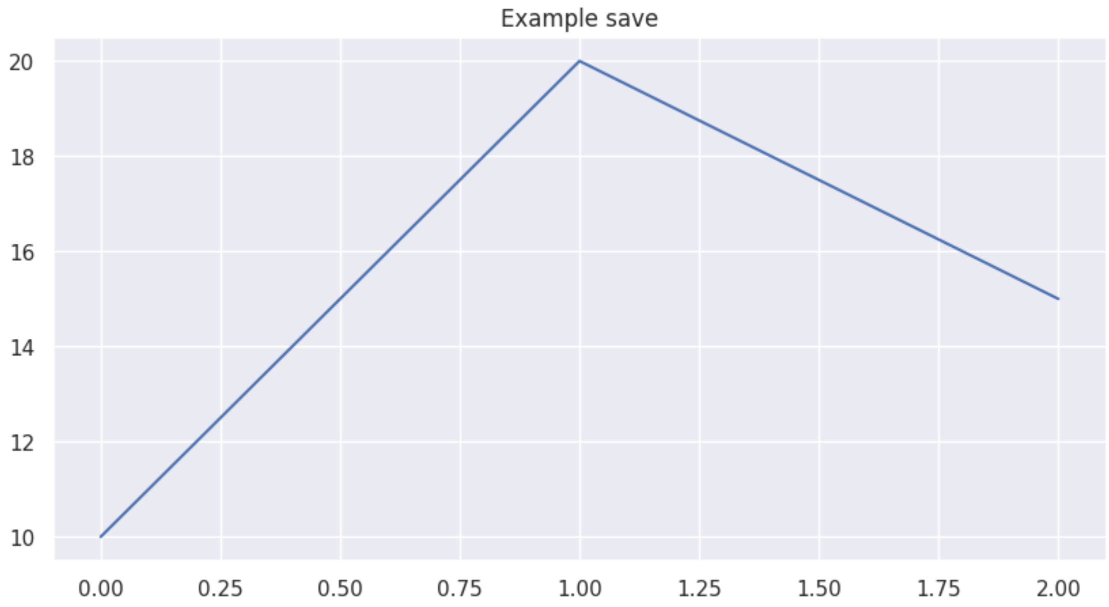
1.9 Interactive plots with Plotly Express

Interactive scatter (tips) and an interactive time series (flights).

1.10 Saving figures and exporting

Examples of saving static Matplotlib figures and exporting Plotly to HTML.

```
saved example_plot.png
saved interactive_tips.html
```

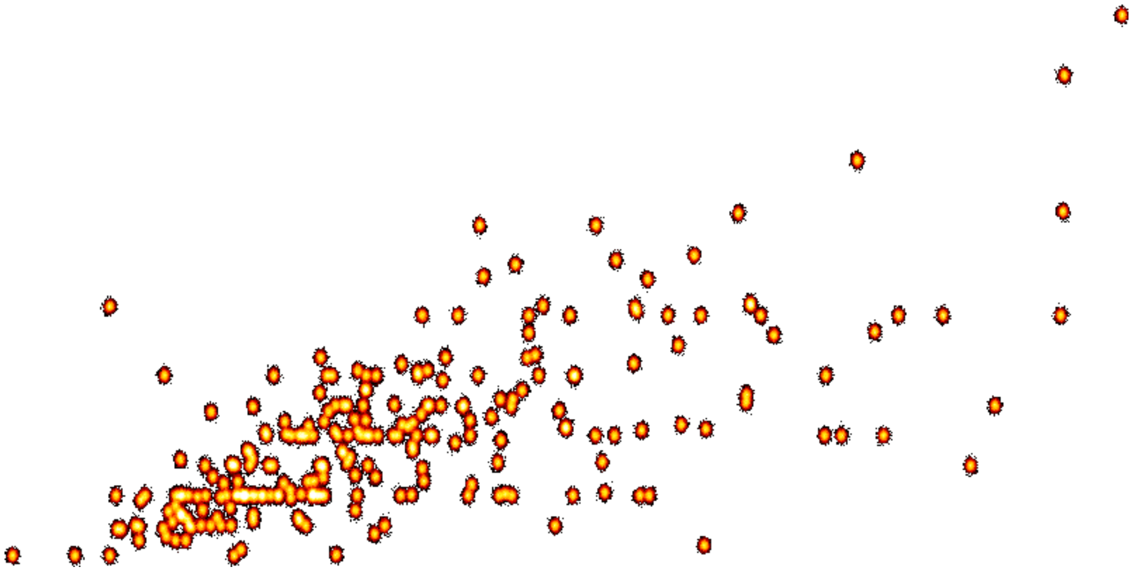


1.11 Notes on performance and large datasets

For very large point clouds consider hexbin, datashader, or downsampling. `plt.hexbin` and `sns.kde` are useful for dense plots.

1.12 Datashader example (for very large datasets)

This cell demonstrates how to use `datashader` to render large scatter clouds efficiently. If `datashader` is not installed in your environment the cell will fall back to a Matplotlib `hexbin` plot and print installation instructions.



1.13 Animated plots (Matplotlib FuncAnimation)

This example creates a simple animated sine wave using Matplotlib's FuncAnimation and displays it inline using HTML/JS. If your environment does not support JS rendering, you can save the animation to MP4/GIF (may require ffmpeg).

```
 <IPython.core.display.HTML object>
```

```
Saved animation as MP4: sine_animation.mp4
```

1.14 Requirements and how to run

Required Python packages: numpy, pandas, matplotlib, seaborn, plotly, statsmodels. Install with: `pip install numpy pandas matplotlib seaborn plotly statsmodels`

Recommended quick steps to run locally: 1. Create and activate a virtual environment (recommended): `python -m venv .venv` On Linux/macOS: `source .venv/bin/activate` On Windows (PowerShell): `./.venv/Scripts/Activate.ps1` 2. Install dependencies: `pip install -r requirements.txt` or the pip line above. 3. Start Jupyter: `jupyter lab` or `jupyter notebook` and open this file: `/home/sysadmin/Notebooks/data-visualization-demo.ipynb`.

Notes: - Plotly inline rendering may raise an error if `nbformat` or notebook MIME renderers are missing; the notebook already falls back to writing interactive HTML files and showing links. - To export animations to MP4/GIF you may need `ffmpeg` installed on your system. - For very large datasets consider using datashader (optional) or downsampling before plotting.

1.15 Conclusion and next steps

This notebook provided quick examples across Matplotlib, Seaborn, and Plotly. Next steps: experiment with custom palettes, GridSpec layouts, animations, or datashader for very large datasets.

To run locally: in a terminal run `jupyter lab` or `jupyter notebook` and open this file: `/home/sysadmin/Notebooks/data-visualization-demo.ipynb`.

1.16 Exporting this notebook to PDF

This section demonstrates how to produce a PDF copy of this notebook programmatically using `nbconvert`. The cell below first attempts a direct PDF export (which commonly requires a LaTeX installation such as TeX Live). If that fails it falls back to exporting an HTML file which you can open in a browser and print/save as PDF.

Note: the code assumes this notebook file is named `data-visualization-demo.ipynb` and is in the current working directory. If your notebook has a different name or path, update the `nb` variable in the code cell.

```
Running: /home/sysadmin/Projects/data-visualization-demo/.venv/bin/jupyter
nbconvert --to pdf data-visualization-demo.ipynb
[NbConvertApp] Converting notebook data-visualization-demo.ipynb to pdf
/home/sysadmin/Projects/data-visualization-
```

```
demo/.venv/share/jupyter/nbconvert/templates/latex/display_priority.j2:32:
UserWarning: Your element with mimetype(s)
dict_keys(['application/vnd.plotly.v1+json']) is not able to be represented.
  ((*- endblock -*))
[NbConvertApp] Support files will be in data-visualization-demo_files/
[NbConvertApp] Making directory ./data-visualization-demo_files
[NbConvertApp] Writing 77240 bytes to notebook.tex
[NbConvertApp] Building PDF
[NbConvertApp] Running xelatex 3 times: ['xelatex', 'notebook.tex', '-quiet']
[NbConvertApp] Running bibtex 1 time: ['bibtex', 'notebook']
[NbConvertApp] WARNING | bibtex had problems, most likely because there were no
citations
[NbConvertApp] PDF successfully created
[NbConvertApp] Writing 668080 bytes to data-visualization-demo.pdf

PDF exported to: /home/sysadmin/Projects/data-visualization-demo/data-
visualization-demo.pdf
```

1.16.1 Notes and alternatives

- Direct PDF export via `nbconvert --to pdf` typically uses a LaTeX intermediate; installing a TeX distribution (TeX Live on Linux) is often required: e.g., `sudo apt install texlive-xetex texlive-fonts-recommended texlive-latex-recommended`.
- If you prefer not to install LaTeX, export to HTML (`nbconvert --to html`) and then: open in a browser and Print -> Save as PDF, or use a headless browser (Chromium/Chrome) to print-to-pdf programmatically (e.g., Puppeteer/Playwright).
- Example command-line alternative (if you have chromium): `chromium --headless --disable-gpu --print-to-pdf=output.pdf data-visualization-demo.html`
- If you want, I can also add an automated Chromium-based HTML->PDF step (requires installing Chromium) or a Playwright/Pyppeteer example—tell me which fallback you'd prefer.

1.17 Automated Chromium HTML -> PDF (optional)

The cell below will look for a Chromium/Chrome binary on your PATH and, if found, will run it in headless mode to convert the previously-exported HTML (`data-visualization-demo.html`) to a PDF. This is useful when LaTeX is not installed and you want a fully automated HTML->PDF step.

If you don't have Chromium/Chrome installed, the cell will print install suggestions for common Linux distributions.

```
Running: /snap/bin/chromium --headless --no-sandbox --disable-gpu --print-to-
pdf=/home/sysadmin/Projects/data-visualization-demo/data-visualization-
demo.chromium.pdf /home/sysadmin/Projects/data-visualization-demo/data-
visualization-demo.html
[24320:24320:1112/184948.008100:ERROR:dbus/object_proxy.cc:573] Failed to call
method: org.freedesktop.DBus.Properties.GetAll: object_path=
/org/freedesktop/UPower/devices/DisplayDevice:
org.freedesktop.DBus.Error.ServiceUnknown: The name org.freedesktop.UPower was
```

```
not provided by any .service files
1125049 bytes written to file /home/sysadmin/Projects/data-visualization-
demo/data-visualization-demo.chromium.pdf

Chromium produced PDF: /home/sysadmin/Projects/data-visualization-demo/data-
visualization-demo.chromium.pdf
```