



Data Visualization, Interaction and Dashboard

11/15/2021

Data Science Academy: season II week 4
Instructors: Oren Livne, Jiangang Hao, & Qiwei He

Learning Goals

Week 4: Visualization and Dashboarding

- Matplotlib, seaborn
- Ipywidgets, plotly
- Dashboard: voila/streamlit
- Demo of use cases at ETS
- Lab:
 - Create plots using matplotlib, seaborn, plotly and ipywidgets based on the provided data
 - Create a voila and a streamlit dashboard based on the provided data.
- Homework/mini project: complete the dashboards in the lab session.





Data Visualization

Steps to Visualization

- Understand the nature of your data
 - Categorical/continuous
 - Sparse
- Understand your goal
 - Audience
 - Production/exploration?
 - Interactivity required?
- Plan the types of visualizations you'll create
 - Scatter plot, distribution, dendrogram, etc.
- Choose a visualization framework(s)
- Do it!

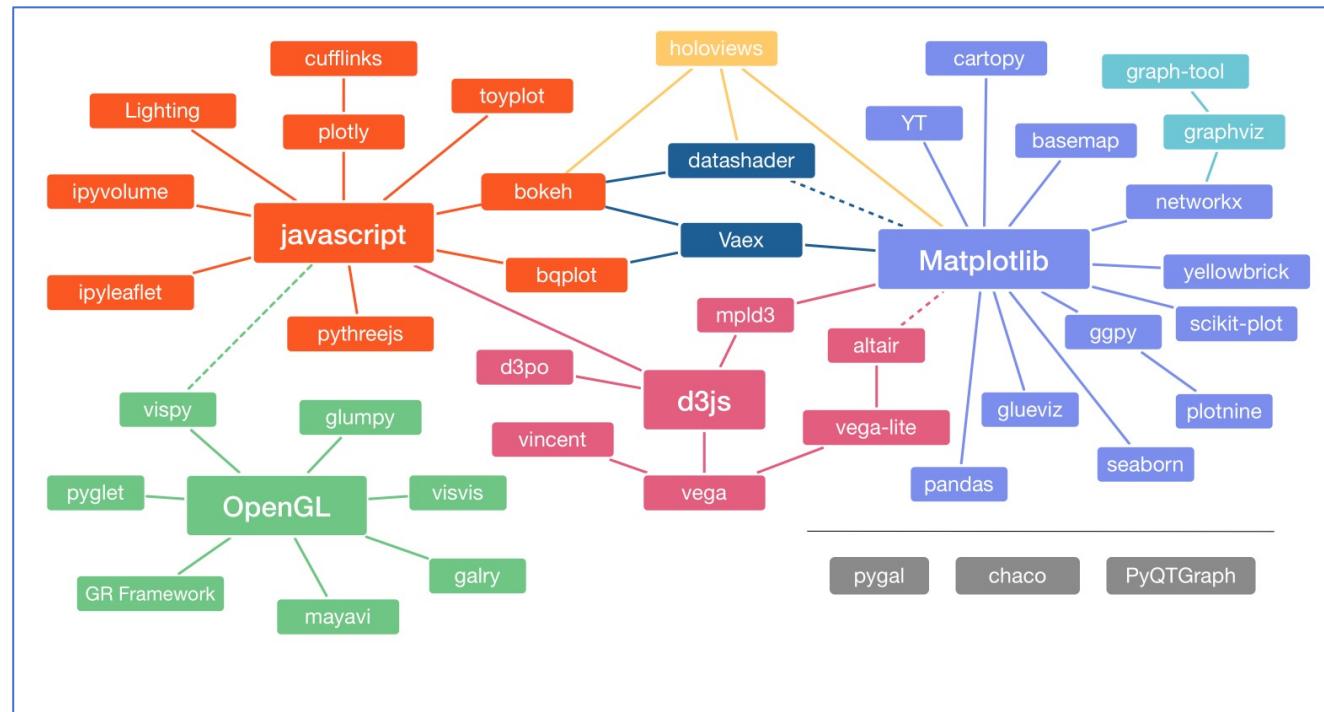


<https://chart.guide/topics/chartguide-poster-4-0/>



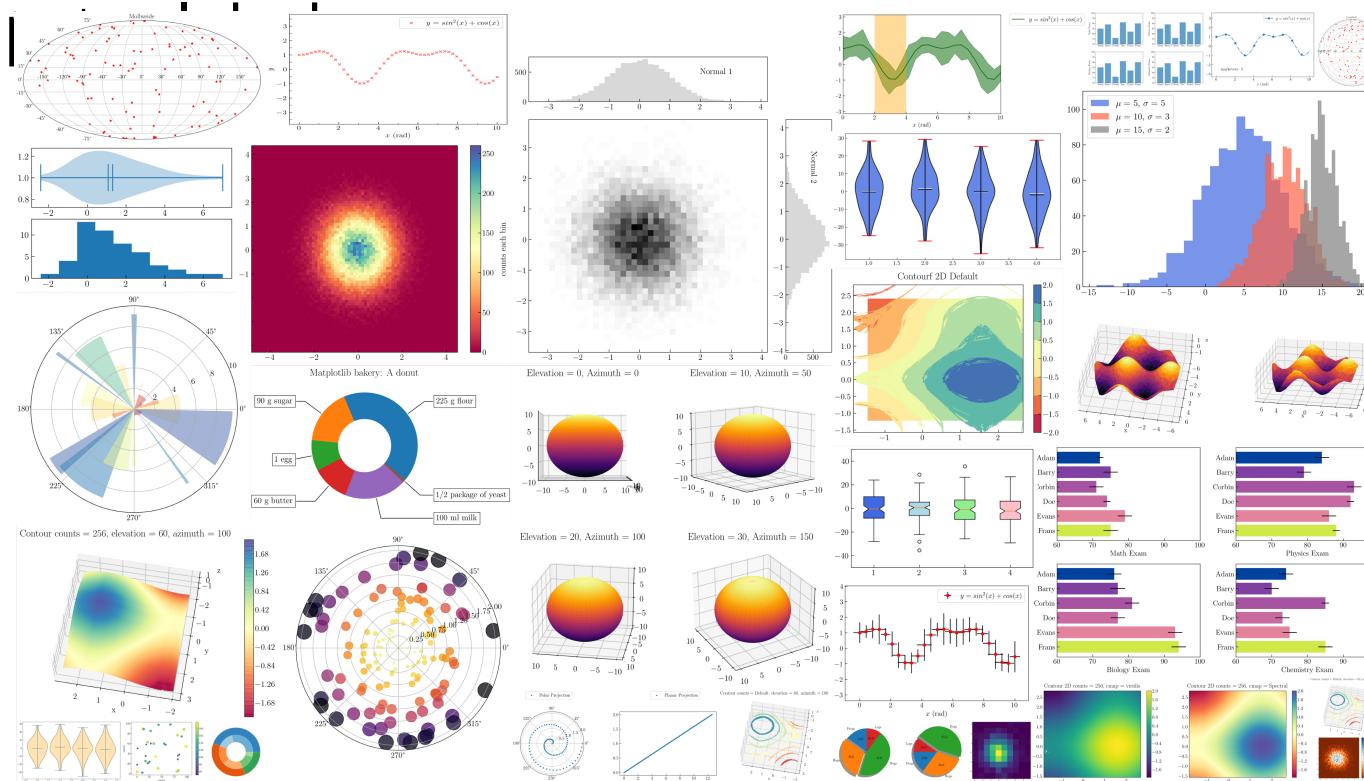
Visualization in Python

- PyViz is your gateway: www.pyviz.org
- Visualization paradigms



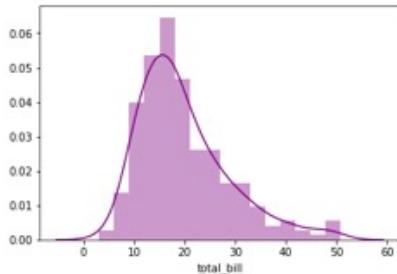
Matplotlib

- jupyter lab demo
- Seaborn - jui

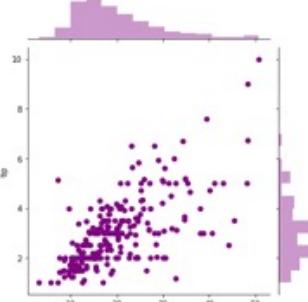


Seaborn: built on top of matplotlib

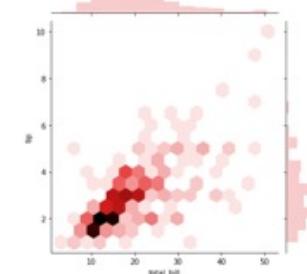
- jupyter lab demo



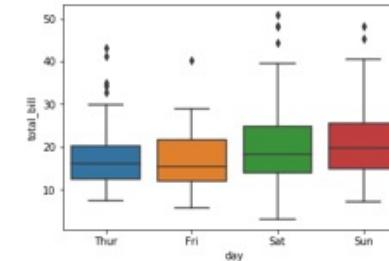
distplot



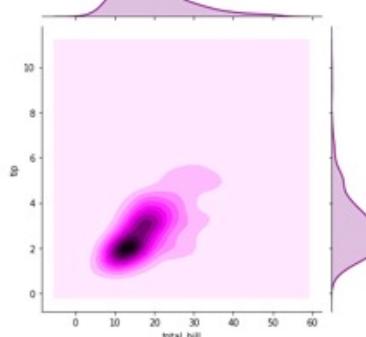
Jointplot



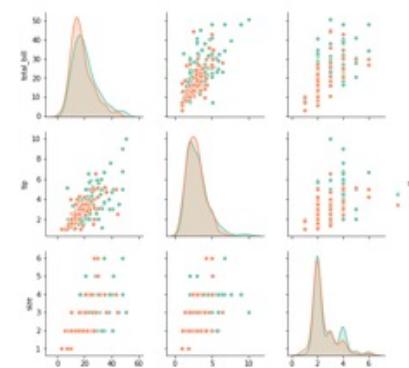
Hexplots



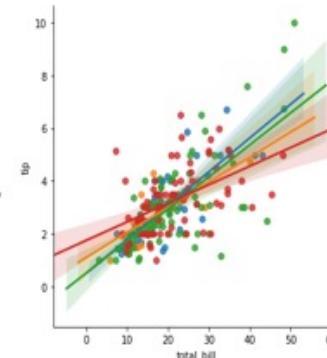
Boxplots



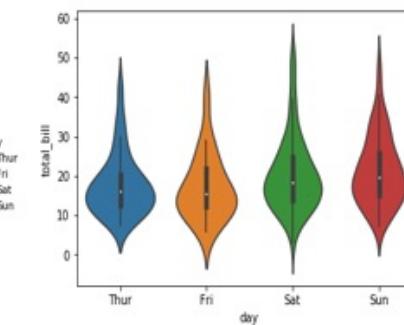
KDE Plot



Pair Plots



LM Plots



Violin Plots

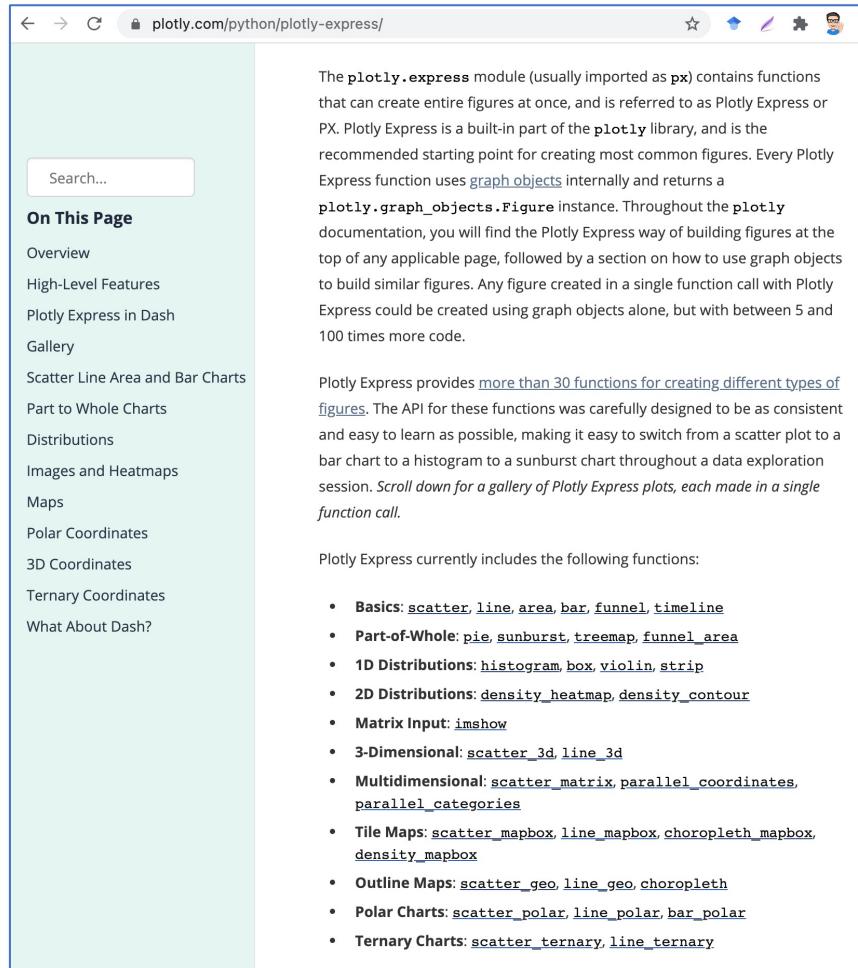




Interactive Visualization



Plotly Express and ipywidgets



The screenshot shows the official Plotly Express documentation page at plotly.com/python/plotly-express/. The page includes a search bar, a sidebar titled "On This Page" with links to various chart types like Scatter Line Area and Bar Charts, and a main content area. The main content describes the `plotly.express` module and its functions for creating different types of plots.

On This Page

- Overview
- High-Level Features
- Plotly Express in Dash
- Gallery
- Scatter Line Area and Bar Charts
- Part to Whole Charts
- Distributions
- Images and Heatmaps
- Maps
- Polar Coordinates
- 3D Coordinates
- Ternary Coordinates
- What About Dash?

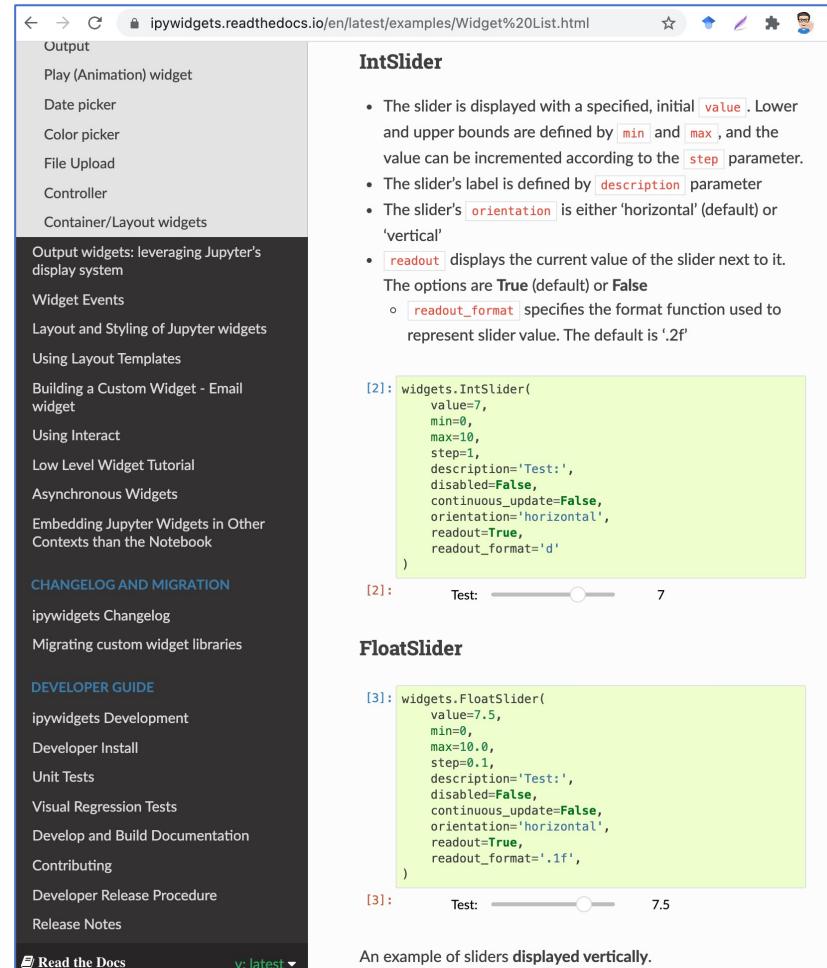
Plotly Express

The `plotly.express` module (usually imported as `px`) contains functions that can create entire figures at once, and is referred to as Plotly Express or PX. Plotly Express is a built-in part of the `plotly` library, and is the recommended starting point for creating most common figures. Every Plotly Express function uses `graph_objects` internally and returns a `plotly.graph_objects.Figure` instance. Throughout the `plotly` documentation, you will find the Plotly Express way of building figures at the top of any applicable page, followed by a section on how to use graph objects to build similar figures. Any figure created in a single function call with Plotly Express could be created using graph objects alone, but with between 5 and 100 times more code.

Plotly Express provides [more than 30 functions for creating different types of figures](#). The API for these functions was carefully designed to be as consistent and easy to learn as possible, making it easy to switch from a scatter plot to a bar chart to a histogram to a sunburst chart throughout a data exploration session. *Scroll down for a gallery of Plotly Express plots, each made in a single function call.*

Plotly Express currently includes the following functions:

- **Basics:** `scatter`, `line`, `area`, `bar`, `funnel`, `timeline`
- **Part-of-Whole:** `pie`, `sunburst`, `treemap`, `funnel_area`
- **1D Distributions:** `histogram`, `box`, `violin`, `strip`
- **2D Distributions:** `density_heatmap`, `density_contour`
- **Matrix Input:** `imshow`
- **3-Dimensional:** `scatter_3d`, `line_3d`
- **Multidimensional:** `scatter_matrix`, `parallel_coordinates`, `parallel_categories`
- **Tile Maps:** `scatter_mapbox`, `line_mapbox`, `choropleth_mapbox`, `density_mapbox`
- **Outline Maps:** `scatter_geo`, `line_geo`, `choropleth`
- **Polar Charts:** `scatter_polar`, `line_polar`, `bar_polar`
- **Ternary Charts:** `scatter_ternary`, `line_ternary`



The screenshot shows the ipywidgets documentation page at ipywidgets.readthedocs.io/en/latest/examples/Widget%20List.html. The page lists various widget types and includes code examples and interactive sliders.

Output

- Play (Animation) widget
- Date picker
- Color picker
- File Upload
- Controller
- Container/Layout widgets

Output widgets: leveraging Jupyter's display system

Widget Events

Layout and Styling of Jupyter widgets

Using Layout Templates

Building a Custom Widget - Email widget

Using Interact

Low Level Widget Tutorial

Asynchronous Widgets

Embedding Jupyter Widgets in Other Contexts than the Notebook

CHANGELOG AND MIGRATION

[ipywidgets Changelog](#)

[Migrating custom widget libraries](#)

DEVELOPER GUIDE

[ipywidgets Development](#)

[Developer Install](#)

[Unit Tests](#)

[Visual Regression Tests](#)

[Develop and Build Documentation](#)

[Contributing](#)

[Developer Release Procedure](#)

[Release Notes](#)

IntSlider

- The slider is displayed with a specified, initial `value`. Lower and upper bounds are defined by `min` and `max`, and the value can be incremented according to the `step` parameter.
- The slider's label is defined by `description` parameter
- The slider's `orientation` is either 'horizontal' (default) or 'vertical'
- `readout` displays the current value of the slider next to it. The options are `True` (default) or `False`
 - `readout_format` specifies the format function used to represent slider value. The default is `'.2f'`

[2]: `widgets.IntSlider(
 value=7,
 min=0,
 max=10,
 step=1,
 description='Test:',
 disabled=False,
 continuous_update=False,
 orientation='horizontal',
 readout=True,
 readout_format='d'
)`

[2]: Test:  7

FloatSlider

[3]: `widgets.FloatSlider(
 value=7.5,
 min=0,
 max=10.0,
 step=0.1,
 description='Test:',
 disabled=False,
 continuous_update=False,
 orientation='horizontal',
 readout=True,
 readout_format='.1f',
)`

[3]: Test:  7.5

An example of sliders displayed vertically.

Demos



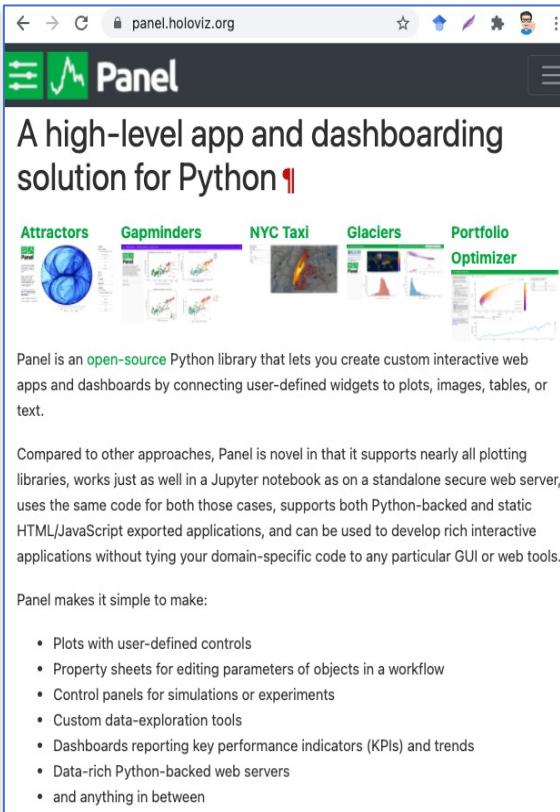


Dashboarding

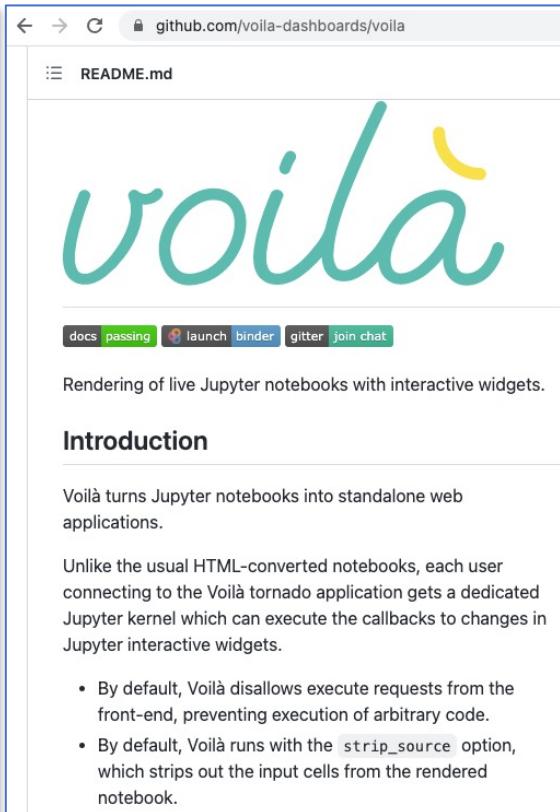
Major Dashboard Tools in Python



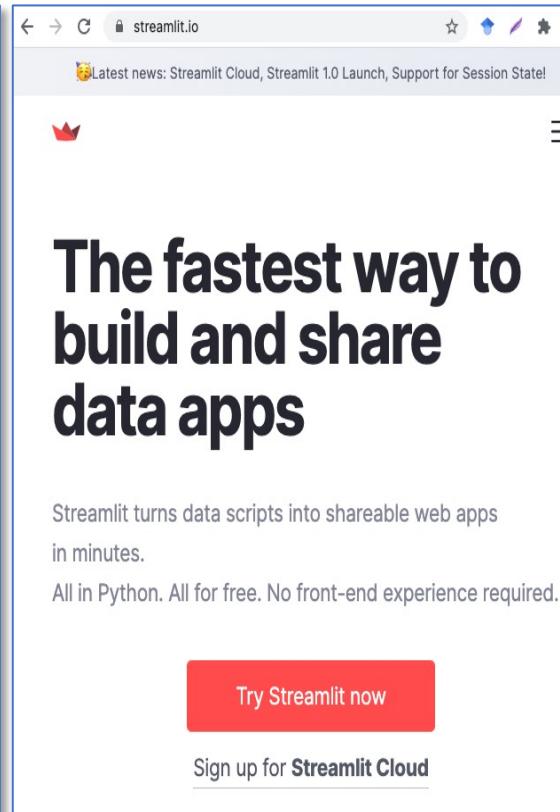
The screenshot shows the official Plotly Dash website. At the top, there's a navigation bar with links for 'Dash' (highlighted), 'Low-Code Development', and 'Deployment & Scaling'. Below this, a section titled 'Overview of Dash & Dash Apps' discusses how Dash apps provide a point-and-click interface for Python, R, and Julia. It highlights the ability to build complex Python analytics into business decision-making tools. A callout at the bottom points to 'Dash Enterprise' for enterprise deployment and scaling.



The screenshot shows the Panel library website. It features a large title 'Panel' with a subtitle 'A high-level app and dashboarding solution for Python'. Below this are five thumbnail images of different dashboard examples: 'Attractors', 'Gapminders', 'NYC Taxi', 'Glaciers', and 'Portfolio Optimizer'. A descriptive paragraph explains that Panel is an open-source Python library for creating custom interactive web apps and dashboards. It compares Panel to other approaches, noting its support for various plotting libraries and its ability to work in Jupyter notebooks or as a standalone web server. A list of features follows, including support for user-defined controls, property sheets, and various types of dashboards.



The screenshot shows the GitHub page for the Voilà library. The main feature is a large, stylized 'voilà' logo. Below it are links for 'docs', 'passing', 'launch binder', 'gitter', and 'join chat'. A brief description states that Voilà turns Jupyter notebooks into standalone web applications. It contrasts this with standard HTML conversion, mentioning that each user gets a dedicated Jupyter kernel. A bulleted list details specific features:Disallowing execute requests from the front-end, running with strip_source, and stripping input cells.



The screenshot shows the Streamlit website. It features a large heading 'The fastest way to build and share data apps'. Below this is a brief description: 'Streamlit turns data scripts into shareable web apps in minutes. All in Python. All for free. No front-end experience required.' At the bottom, there are two calls-to-action: a red button for 'Try Streamlit now' and a link for 'Sign up for Streamlit Cloud'.



Use Cases

- Voila
 - Simple example
 - Covid insight
- Streamlit
 - Simple example
 - GRE log safari
 - GRE session deep dive
 - EPCAL analytics





Hands-on Lab

Lab

- Create visualizations using matplotlib, seaborn, plotly and ipywidgets using the dataset
 - Histogram, scatter plot, pie, others
 - Create one voila dashboard (homework if not completed in the lab)
 - Create one streamlit dashboard (homework if not completed in the lab)



Assignments

- Matplotlib:
 - Exercises 1.1, 2.1, 2.2 in Matplotlib tutorial notebooks.
 - Read and run the examples in the examples folder.

