Data Visualization Libraries in Python

“It is easy to lie with statistics. It is hard to tell the truth without it.”

-Andrejs Dunkels, Mathematician

Common times to make graphics

- The **beginning** of a project
  - Exploring and understanding data
  - Preliminary Results
- The **middle** of a project
  - Explore model assumptions
  - Examine model output
- The **end** of a project
  - Publications
  - Presentations to policy and decision makers
  - Communicating results to non-technical audiences

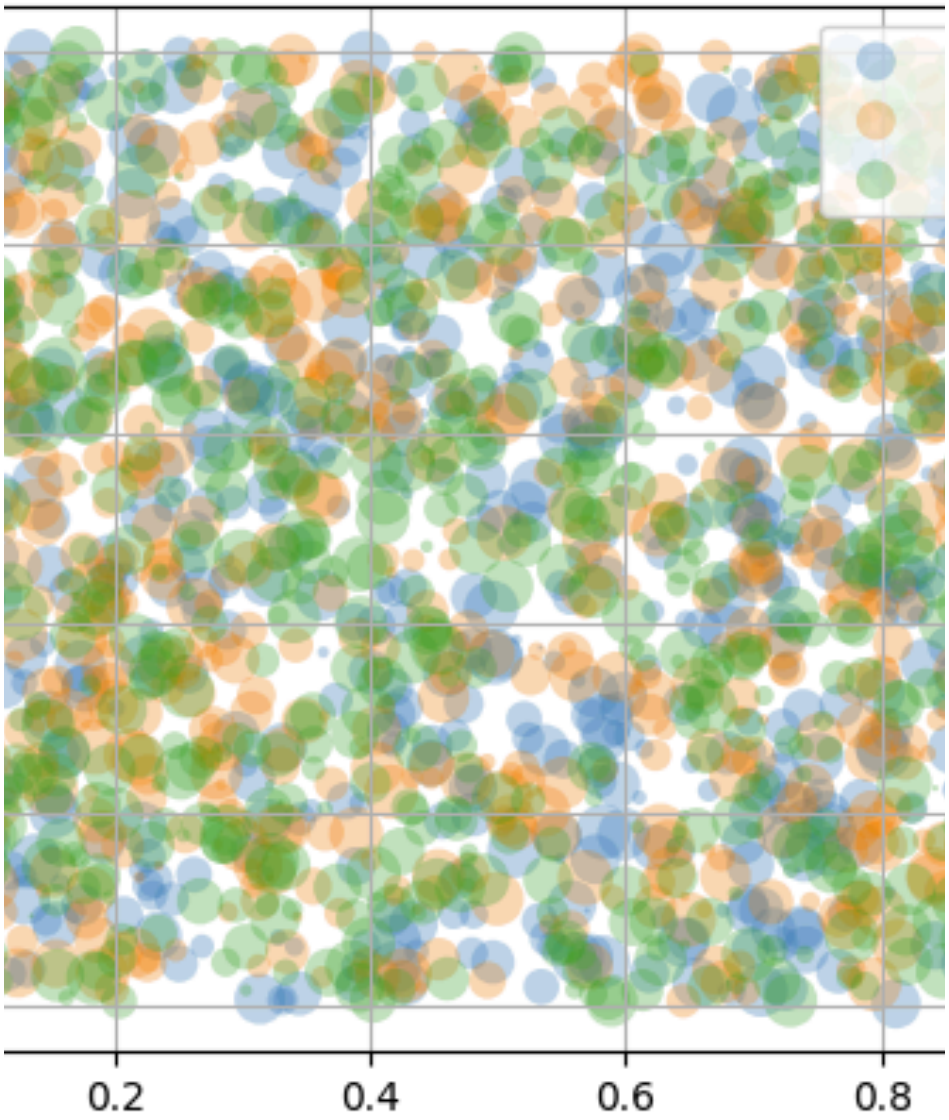
Common times to make graphics

- The **beginning** of a project
  - Exploring and understanding data
  - Preliminary Results
- The **middle** of a project
  - Explore model assumptions
  - Examine model output
- The **end** of a project
  - Publications
  - Presentations to policy and decision makers
  - Communicating results to non-technical audiences

Different tools are needed for each part

Graphics Libraries

- Some libraries are better for the beginning and some are better for the end
- The following are just a few of the graphics libraries available in python



- Matplotlib
  - The grandfather of python visualizations
  - Initial release: 2003 (this predates pandas!)
  - Originally designed to mimic MATLAB graphics
  - Very versatile and customizable
  - Syntax can be a bit clunky
  - https://matplotlib.org/stable/gallery/index.html

seaborn

- Seaborn
  - Builds on Matplotlib
  - Provides a high-level interface for statistical graphics
  - Easy to make nice-looking graphics
  - Can only create plot available in documentation
  - https://seaborn.pydata.org/examples/index.html

- Pandas
  - Uses Matplotlib on the backend
  - Limited scope (graphics is not the main purpose of pandas)
  - Easy to make through `.plot()` calls
  - https://pandas.pydata.org/pandas-docs/stable/user_guide/visualization.html

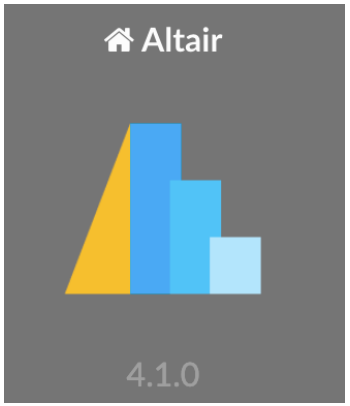
plotnine

- Plotnine
  - Grammar of Graphics for python
  - Based on ggplot2
  - Great option for those who know ggplot2 from R
  - https://plotnine.readthedocs.io/en/stable/gallery.html


```
from plotnine import ggplot, geom_point, aes, stat_smooth, facet_wrap
from plotnine.data import mtcars

(ggplot(mtcars, aes('wt', 'mpg', color='factor(gear)'))
 + geom_point()
 + stat_smooth(method='lm')
 + facet_wrap('~gear'))
```

- Altair



  - Also a Grammar of Graphics for python
  - Based on Vega and Vega-Lite
  - Great option for those who know ggplot2 from R, but syntax is slightly different
  - https://altair-viz.github.io/gallery/index.html

plotly | Graphing Libraries

- Plotly
  - Data visualization library focused on interactive graphics
  - Not python specific -- libraries exist for R, Julia, Javascript, etc.
  - Plotly Express is a fast easy option for quick EDA interactive visualizations
  - Isn't for dashboards, but works with dash (open-source library from the same company) for dashboard

https://plotly.com/python/ https://plotly.com/python/plotly-express/#gallery https://dash.gallery/Portal/

- Bokeh
  - Another common data visualization library for interactive graphics
  - Can also be used for dashboards
  - http://docs.bokeh.org/en/latest/docs/gallery.html

Streamlit

- Streamlit
  - Pure python library that can easily create shareable web apps
  - Can also be used for simple dashboards
  - Apps can be launched from code in a GitHub repo
  - https://streamlit.io/

Matplotlib

As self-respecting pythonistas, we MUST know the basics of matplotlib