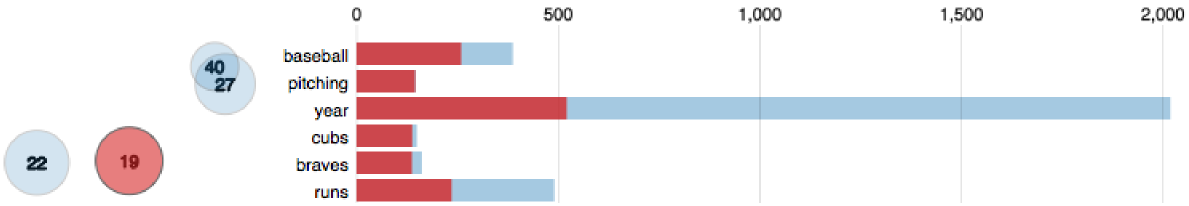
Python Data Visualization Libraries

pyLDAvis

Python library for interactive topic model visualization.



pyLDAvis is designed to help users interpret the topics in a topic model that has been fit to a corpus of text data. The package extracts information from a fitted LDA topic model to inform an interactive web-based visualization.

The visualization is intended to be used within an IPython notebook but can also be saved to a stand-alone HTML file for easy sharing.

Checkout the link below for the jupyter notebook implementation:

https://nbviewer.org/github/bmabey/pyLDAvis/blob/master/notebooks/pyLDAvis_overview.ipynb

1. Matplotlib

Matplotlib is the most popular and widely-used Python package for creating advanced data visualizations. lets you create static, interactive, and dynamic representations.

Python 2-D plotting package **incredible level of flexibility,** can build **interactive 2D** graphs using matplotlib, such as line graphs, scatter graphs, bar graphs, etc. Although matplotlib **does not include built-in support for 3D graphs,** it does include an alternative toolkit – **mplot3d – that can be used to create 3D graphs.**

PROS:

- Relatively easy to understand for beginners.
- It offers high-quality photos and plots in multiple formats, including png, pdf, etc.
- This library controls numerous aspects of an image, including image color, image size, etc.

CONS:

- not be ideal for time series data because it requires importing all helper classes for the year, month, week, and day formatters.
- **library is low-level and requires extra code to generate the visualization.**
- **inconvenient when dealing with several datasets**

2. Seaborn

It's a high-level interface for creating aesthetically appealing and valuable statistical visuals, crucial for studying and comprehending data. This Python library is closely linked with NumPy and pandas data structures.

Seaborn conducts the relevant semantic mapping and statistical aggregation internally to produce relevant charts.

PROS:

- It **generates a dynamic and informative plot** to represent your data
- can use Seaborn to visualize data without worrying about the internal details
- The **'kind' property** inside this library enables you **to switch to any other data representation format.**

CONS:

- the **customization options are limited.**
- **Interactive visualizations are rare** in this library.

3. Ggplot

Based on the ggplot2 implementation for the R programming language. It also lets you **combine many types of data visualization components or layers into a single visualization**. After mentioning which variables to map to specific aesthetics in the plot, ggplot takes care of the rest, allowing the user to analyze the visualizations rather than design them.

ggplot differs from Matplotlib as it a**llows users to stack components to build a complete plot**. You can start with axes and then add points, a line, a trend line, and so on. **For heavily customized graphics, ggplot isn't precisely the best choice.**

Key Features of ggplot

- **declarative-style library** closely integrated with Pandas.can create visualizations directly from your Pandas dataframe.
- ggplo**t isn't designed for highly customized visualizations.**
- Using the pip install command in the Python environment lets you install ggplot because it is an open-source module.
- **keeping your data in a Data Frame is essential** when using ggplot.

Pros of using ggplot

- For beginners working with ggplot for the first time, the ggplot documentation is **simple and easy to follow.**
- ggplot has a save method if you need to exhibit your plots or discuss your insights with other collaborators.

Cons of using ggplot

- Some features, such as **creating maps with theme_map, are not available** in ggplot.
- ggplot **does not let you create highly customized visualizations**.

4. Plotly

It is a web-based data visualization tool built on top of the Plotly JavaScript library (plotly.js).

Contour plots are also available in Plotly, making it slightly different from other data visualization frameworks.

You can use Plotly **to generate web-based data visualisations** presented in Jupyter notebooks or web apps using Dash or saved as standalone HTML files.

Key Features of Plotly

- Plotly allows you **to share plots with the public without revealing your code.**
- The syntax is simple, as all graphs use the same parameters.
- **3D plots with a variety of interactive tools** are available in Plotly.

Pros of using Plotly

- **Hover tool capabilities** in Plotly allow us to spot outliers or anomalies in massive numbers of sample points.
- It allows you to **personalize your graphs** in an infinite number of ways, making your plots more exciting and understandable to all others.

Cons of using Plotly

- It's challenging to keep up with many Plotly tools (Chart Studio, Express, etc.) and out-of-date documentation.
- there are no easy steps to link graphs to the same source dataset.

5. Bokeh

It generates **detailed images with a high level of interaction** for various datasets, big and small.

It allows for **creating adaptable graphics** with high-performance interactivity **across big or streaming datasets.**

To **generate interactive plots** for modern web browsers that one can use in interactive online applications, HTML pages, or JSON objects.

There are three layers in Bokeh for creating data visualizations.

The first level: constructing data plots

the second level: controls the plot's essential building components.

The third level: gives complete flexibility over chart creation with no pre-set settings.

Key Features of Bokeh

- Bokeh supports **active plot interaction** such as zooming, panning, selecting, etc.
- It provides a low-level interface with **additional flexibility** to **customize charts**.
- Bokeh can also convert Matplotlib, ggplot.py, and seaborn charts and plots.
- It helps **to create interactive plots** that can be saved in both PNG and SVG formats.
- Bokeh generates **output in various forms**, including HTML, notebook, etc.

Pros of using Bokeh

- **Linking plots is easier** with the help of Bokeh.
- Bokeh can be used as a high-level or low-level interface, allowing it to produce many of the same graphs as Matplotlib but with **fewer lines of code and higher resolution.**

Cons of using Bokeh

- Because Bokeh is a middle-level package, it often takes less code than Matplotlib but more code to produce the same plot as Seaborn, Altair, or Plotly.
- It doesn't support 3D graphic functions and has limited interactivity options to work with.
- Before you construct any plot, you must first define the output mode, including notebook, server, and web browser modes.

6. Altair

It is based on Vega and Vega-Lite, declarative languages for generating, preserving, and sharing interactive data visualization designs.

Altair can generate attractive data visualizations of plots with little scripting

Key Features of Altair

- Because Altair **creates plots in a declarative form,** it's **simple and easy to move through visualizations** and experiments quickly when using this package.
- **Filtering data** is another great feature
- You can also use Altair to provide **dynamic filtering,** and it allows you to use a shared filter to connect many plots.

Pros of using Altair

- It's easy to use and results in more attractive and compelling visualizations.
- Although the coding structure remains the same, multiple plots can be produced simply by altering the mark attribute.
- Altair **helps you to better understand your data** by **supporting data transformations**, such as using the count, min, and max aggregator functions.

Cons of using Altair

- **Cannot create 3D visualizations.**
- Altair, like many high-level visualization frameworks, i**sn't wholly customizable**
- Altair **lacks some of the major plots, such as boxplots.**

7. Pygal

For creating interactive plots that you can embed in a web browser. It's an excellent library for working with smaller datasets, and it generates. C**reating charts with hundreds of thousands of data points may be challenging,** and it may be difficult to deliver results in such instances.

One can use pip to install the Pygal library. Pygal makes plotting simple, and it supports a variety of chart kinds, including line, bar, histogram, radar, box, treemap, etc.

Key Features of Pygal

- This module may be **used to build dynamic and interactive graphs on a web page** using typical Python web interfaces like flask and Django.
- It can make SVG presentations easier to work with interactive files.

- Pygal is an excellent choice **for small web apps that require quick and efficient graphs.**

Pros of using Pygal

- With **minimal coding,** you can create distinctive and visually stunning graphs.
- One can **export the charts and graphs in various formats**, including SVG, PNG, Etree, and others.

Cons of using Pygal

- SVGs will suffice as long as you're working with smaller datasets. However, **working with massive datasets** containing thousands of data points **makes visualization challenging** and inefficient.

8. Gleam

Gleam is a Python library that allows you to create interactive online visualizations of data without needing to know HTML or JS. Gleam combines everything into a web interface that allows anyone to interact with your data in real-time. It makes it easier for you to explain and interpret your data to others. The Shiny package in R was the inspiration behind the development of the Gleam library.

Gleam is used for making interactive visualizations that include pages, panels, and buttons. These interactive web visualizations are also completely web-integrated, meaning they can be embedded in anything from a website to an endpoint.

Key Features of Gleam

- It allows you to turn analysis into interactive web apps using only python scripts, minimizing the need-to-know other languages such as HTML, CSS, or javascript.
- It works with nearly all other Python data visualization libraries.
- You can add fields to a plot after it's been created so that users can filter and sort data.

REFERENCES:

https://pyldavis.readthedocs.io/en/latest/readme.html

https://www.projectpro.io/article/python-data-visualization-libraries/543

For video tutorials:

Matplotlib Tutorial 2021

Seaborn Tutorial 2021

Plotly Tutorial 2021

Rachel Brynsvold - Literary Analysis via NLP: Topic Modeling Project Gutenberg (PyTexas 2017)

Ep 1 - Python Bokeh Installation & Circle plot