CHOOSING THE RIGHT TOOL

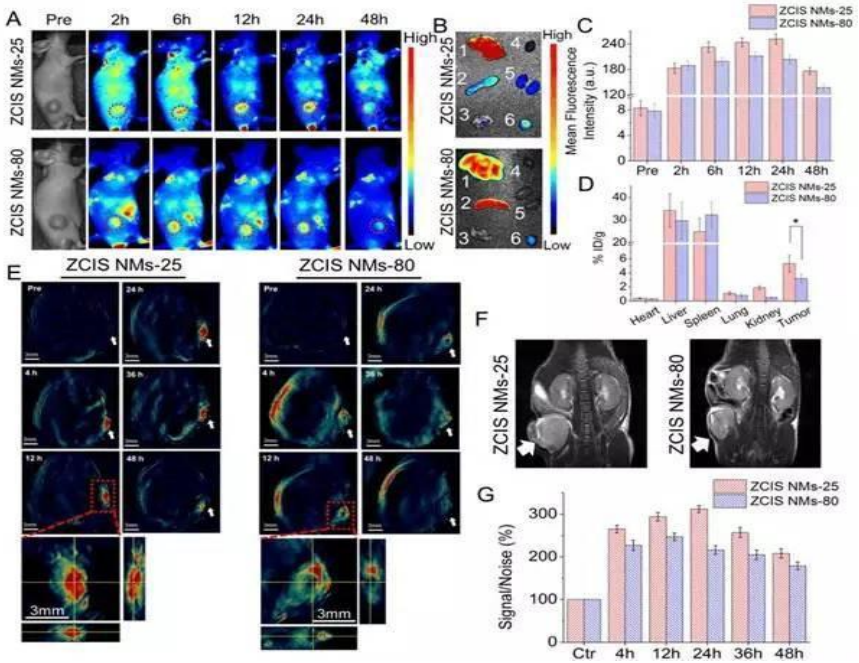
Data visualization tools provide data visualization experts with an easier way to create visual representations of large data sets. These data visualizations can then be used for a variety of purposes: dashboards, annual reports, sales and marketing materials, investor slide decks, and virtually anywhere else information needs to be interpreted immediately. We have already choosen Python as the tool for creating plots and analyzing data. But it's a good time to know about other tools that are available in the market some of them are free and others are paid.

What Do the Best Data Visualization Tools Have in Common?

- Ease of use: There are some incredibly complicated apps available for visualizing data. Some have excellent documentation and tutorials and are designed in ways that feel intuitive to the user.
- Handle large set of values: The best tools can also handle huge sets of data and can even handle multiple sets of data in a single visualization.
- Output charts: The best tools also can output an array of different chart, graph, and map types. Most of the tools below can output both images and interactive graphs.
- Cost: While a higher price tag doesn't necessarily disqualify a tool, the higher price tag has to be justified in terms of better support, better features, and better overall value.

It's worth talking about some interesting types of visualization here which will also dictate the tool to use:

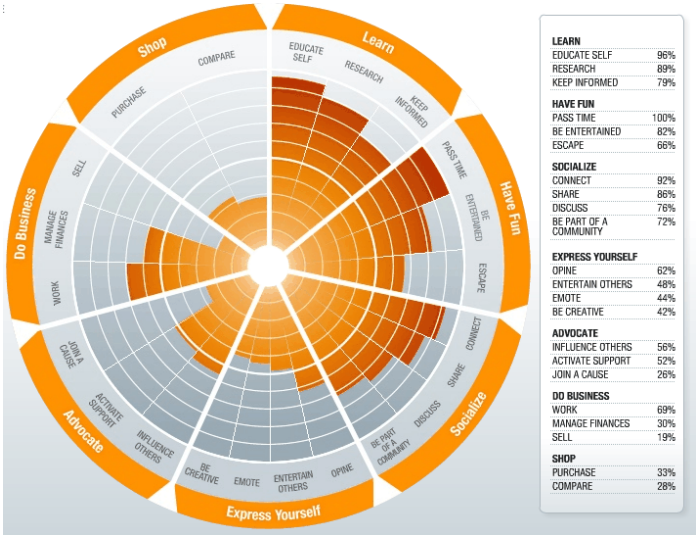
Scientific Visualization



Scientific visualization is an interdisciplinary research and application field in science, focusing on the visualization of three-dimensional phenomena, such as architecture, meteorology, medicine or biological systems. Its purpose is to graphically illustrate scientific data, enabling scientists to understand, explain, and collect patterns from the data.

Information visualization

Information visualization is the study of interactive visual representations of abstract data to enhance human cognition. Abstract data includes both digital and non-digital data such as geographic information and text. Graphics such as histograms, trend graphs, flow charts, and tree diagrams all belong to information visualization, and the design of these graphics transforms abstract concepts into visual information.



Visual Analytics

Visual analytics is a new field that has evolved with the development of scientific visualization and information visualization, with an emphasis on analytical reasoning through an interactive visual interface.

COMMON TOOLS

Tableau

Tableau is a robust tool for visualizing data in a better way. You can connect any database to create understandable visuals. It enables you to share visualization with other people.
Link: https://www.tableau.com

Qlik
Qlik is a data visualization software which is used for converting raw data into knowledge. This software acts like a human brain which works on "association" and can go into any direction to search the answers.
Link: https://www.qlik.com/us

Adaptive Insights
Adaptive Insights is a data visualization tool built to boost your business. It helps you to plan, budget, as well as forecast to make better decisions.
Link: https://www.adaptiveinsights.com

Plotly
Plotly is a tool that helps you to build analytic web apps. This app enables you to export HTML files and images in the report.
Link: https://plot.ly

RAWgraphs
RAWGraphs is a data visualization app which makes the visual representation of any complicated data simple. It works with CSV and TSV (Tab Separated Values). This app helps you to embed charts directly into your web pages.
Link: https://rawgraphs.io

Google Charts
Google Charts is an interactive cloud service that creates graphical charts from the information supplied by users. You can use it to make a simple line chart or complex hierarchical tree.
https://developers.google.com/chart

FusionCharts
FusionCharts is a JavaScript library for data visualization. It offers more than 2000 maps and 100+ charts. This library gives a range of customizable options to build charts from raw data.
Link: https://www.fusioncharts.com

PYTHON DATA VISUALIZATION LIBRARIES

The Python Package Index has libraries for practically every data visualization need—from Pastalog for real-time visualizations of neural network training to Gaze Parser for eye movement research. Some of these libraries can be used no matter the field of application, yet many of them are intensely focused on accomplishing a specific task.
An overview of 10 interdisciplinary Python data visualization libraries we will talk about here:

1. Matplotlib

Matplotlib Python Library is used to generate simple yet powerful visualizations. More than a decade old, it is the most widely-used library for plotting in the Python community. Matplotlib is used to plot a wide range of graphs– from histograms to heat plots. Matplotlob is the first Python data visualization library, therefore many other libraries are built on top of Matplotlib and are designed to work in conjunction with the analysis. Libraries like pandas and matplotlib are "wrappers" over Matplotlib allowing access to a number of Matplotlib's methods with less code.

The versatility of Matplotlib can be used to make visualization types such as:

- Scatter plots
- Bar charts and Histograms
- Line plots
- Pie charts
- Stem plots
- Contour plots
- Quiver plots
- Spectrograms
- You can create grids, labels, legends etc. with ease since everything is easily customizable.

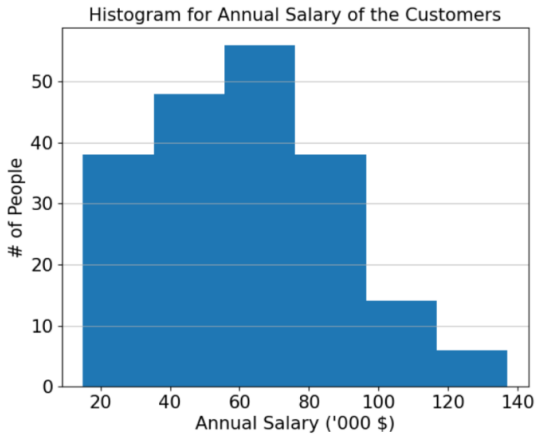
Install the package:

```
pip install matplotlib
```

Example

```
#Histogram using MatPlotLib
import matplotlib.pyplot as plt
import pandas

#Read the data from the web using pandas
url =
"https://raw.githubusercontent.com/swapnilsaurav/Dataset/master/Mall_C
ustomers.csv"
data = pandas.read_csv(url)
#Histogram for Annual Income
#lets create 6 bins
n_bins = 6
plt.hist(data['Annual Income (k$)'], bins = n_bins)
#Add attributes to the plot
plt.grid(axis='y', alpha=0.75)
plt.xlabel("Annual Salary (\'000 $)",fontsize=15)
plt.ylabel('Frequency',fontsize=15)
plt.xticks(fontsize=15)
plt.yticks(fontsize=15)
plt.ylabel('# of People',fontsize=15)
plt.title('Histogram for Annual Salary of the Customers',fontsize=15)

plt.show()
```

Output Graph



2. Seaborn

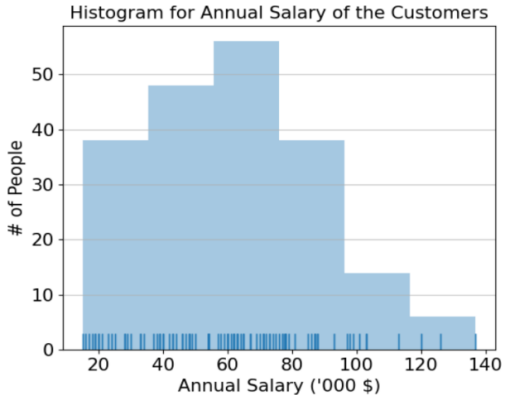
Seaborn is a popular data visualization library that is built on top of Matplotlib. Seaborn's default styles and color palettes are much more sophisticated than Matplotlib. Seaborn puts visualization at the core of understanding any data. Seaborn is a higher-level library- it's easier to generate certain kinds of plots, including heat maps, time series, and violin plots.

Example

Since Seaborn is built on top of matplotlib, you'll need to know matplotlib to tweak Seaborn's defaults. The key difference is Seaborn's default styles and color palettes, which are designed to be more aesthetically pleasing and modern.

```
#Histogram using Seaborn
import matplotlib.pyplot as plt
import seaborn as sns
import pandas

#Read the data from the web using pandas
url =
"https://raw.githubusercontent.com/swapnilsaurav/Dataset/master/Mall_Customers.csv"
data = pandas.read_csv(url)
#Histogram for Annual Income:
data_val = data['Annual Income (k$)']
#lets create 6 bins
n_bins = 6
sns.distplot(data_val, bins=n_bins, kde=False, rug=True);
#Kernel density estimation (KDE)
#rug plot, which draws a small vertical tick at each observation
#Add attributes to the plot
## You can add exactly same attributes as we have seen
### in MatPlotLib example

plt.show()
```

Output Graph



3. ggplot

Ggplot is a Python visualization library based on R's ggplot2 and the Grammar of Graphics. You can construct plots using high-level grammar without worrying about the implementation details. Ggplot operates differently compared to Matplotlib: it lets users' layer components to create a full plot. For example, the user can start with axes, and then add points, then a line, a trend line, etc. The Grammar of Graphics has been hailed as an "intuitive" method for plotting; however seasoned Matplotlib users might need time to adjust to this new mindset.

Example

```
#Histogram using GGPlot
import pandas
import ggplot as gg

#Read the data from the web using pandas
url =
"https://raw.githubusercontent.com/swapnilsaurav/Dataset/master/Mall_Customers.csv"
data_all = pandas.read_csv(url)
#Histogram for Annual Income:
data_val = data_all['Annual Income (k$)']

gg.ggplot(gg.aes(x='Annual Income (k$)'), data=data_val) +
gg.geom_histogram()
```

Note: There could be a problem using ggplot with pandas as some versions may not be compatible with all the versions. In that case, install below version which we know works well with Pandas.

```
pip uninstall ggplot
pip install git+git://github.com/yhat/ggpy.git@9d00182343eccca6486beabd256e8c89fb0c59e8 --no-cache
```

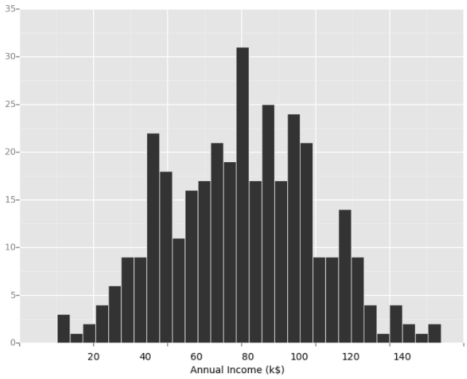
If that doesn't help, Downgrade Pandas

```
pip uninstall pandas
pip install pandas==0.19.2
```

pip install numpy==1.16.4

You might even have to install package *wheel* if asked for. If both option doesn't work, try downgrading your Python version to 3.5.

Output Graph



4. Bokeh

Bokeh, native to Python is also based on The Grammar of Graphics like ggplot. It also supports streaming, and real-time data. The unique selling proposition is its ability to create interactive, web-ready plots, which can easily output as JSON objects, HTML documents, or interactive web applications. Bokeh has three interfaces with varying degrees of control to accommodate different types of users. The topmost level is for creating charts quickly. It includes methods for creating common charts such as bar plots, box plots, and histograms. The middle level allows the user to control the basic building blocks of each chart (for example, the dots in a scatter plot) and has the same specificity as Matplotlib. The bottom level is geared toward developers and software engineers. It has no pre-set defaults and requires the user to define every element of the chart.

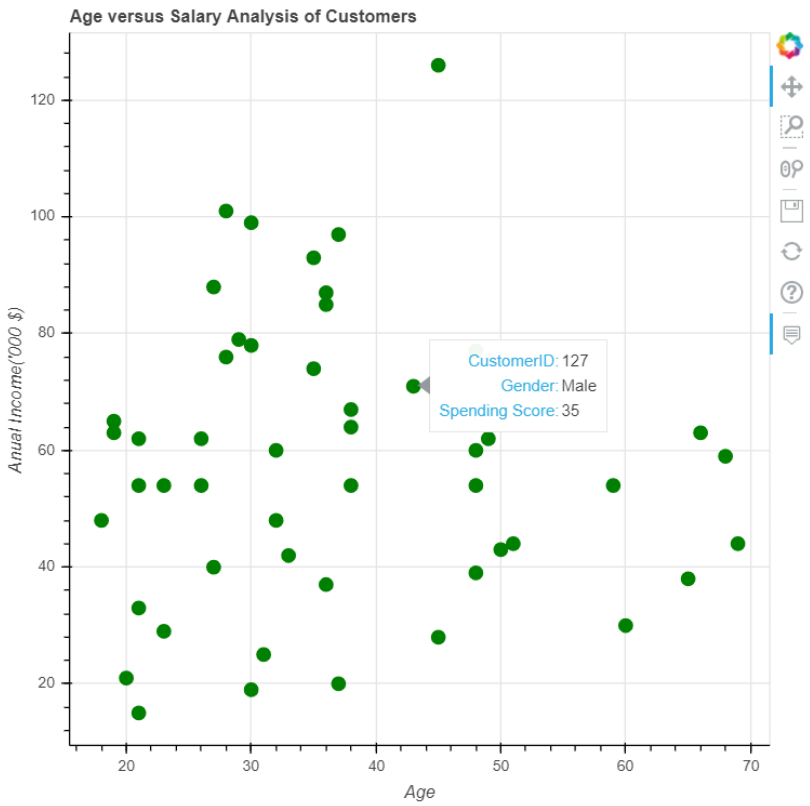
Example

```
import pandas as pd
from bokeh.plotting import figure, output_file, show
from bokeh.models import ColumnDataSource
from bokeh.models.tools import HoverTool

#html file for the plot
output_file('my_first_graph.html')

#Dataset:Mall_Customer.csv
#Read the data from the web using pandas
url =
"https://raw.githubusercontent.com/swapnilsaurav/Dataset/master/Mall_C
ustomers.csv"
data = pd.read_csv(url)
#Print the list of columns
print(data.columns.tolist())

# we randomly sample 50 rows using the dataframe's
# sample method.
sample = data.sample(50)
source = ColumnDataSource(sample)
```

```
#we create our figure object and call the circle
# glyph method to plot our data. This is where the
# source variable that holds our ColumnDataSource
# comes into play.
p = figure()
p.circle(x='Age', y='Annual Income (k$)',
         source=source,
         size=10, color='green')
#Next we add a title and label our axes.
p.title.text = 'Age versus Salary Analysis of Customers'
p.xaxis.axis_label = 'Age'
p.yaxis.axis_label = 'Anual Income(\'000 $)'

#Bokeh allows us to customize our plot by adding
# new interactive tools to it.
hover = HoverTool()
hover.tooltips=[
    ('CustomerID', '@CustomerID'),
    ('Gender', '@Gender'),
    ('Spending Score', '@{Spending Score (1-100)}')
]
#{} for columns with space or bracket
p.add_tools(hover)

show(p)
#HoverTool is particularly useful for data exploration and
interaction.
# HoverTool allows you to set a tooltips property which takes a list
of tuples.
```

Output Graph

Html file is opened with the interactive plot:



Some of the features offered by Bokeh are:

- interactive visualization library
- targets modern web browsers for presentation
- versatile graphics

5. Plotly

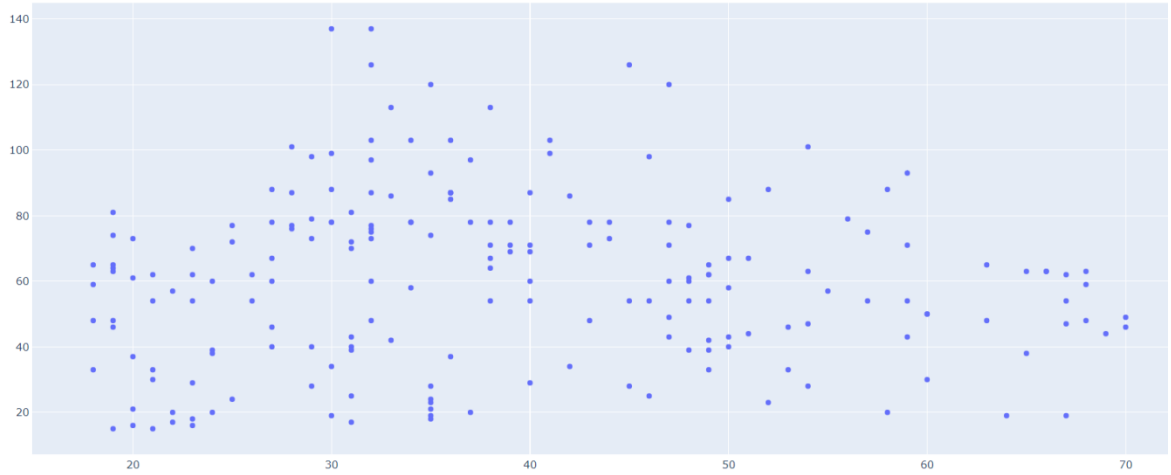
While Plotly is widely known as an online platform for data visualization, very few people know that it can be can be accessed from a Python notebook. Like Bokeh, Plotly's strength lies in making interactive plots, and it offers contour plots, which cannot be found in most libraries.

Example

```
import plotly
import plotly.graph_objs as go
import pandas as pd
url =
"https://raw.githubusercontent.com/swapnilsaurav/Dataset/master/Mall_Customers.csv"
data = pd.read_csv(url)
x=data['Age']
y=data['Annual Income (k$)']
# Create a trace
trace = go.Scatter(
    x = x,
    y = y,
    mode = 'markers'
)
```

```
data = [trace]

# Plot and embed in ipython notebook!
plotly.offline.iplot(data, filename='basic-scatter')
```

Output Graph



Plotly provides the following key features:

- Feature parity with MATLAB/matplotlib graphing
- Online chart editor
- Fully interactive (hover, zoom, pan)

6. Pygal

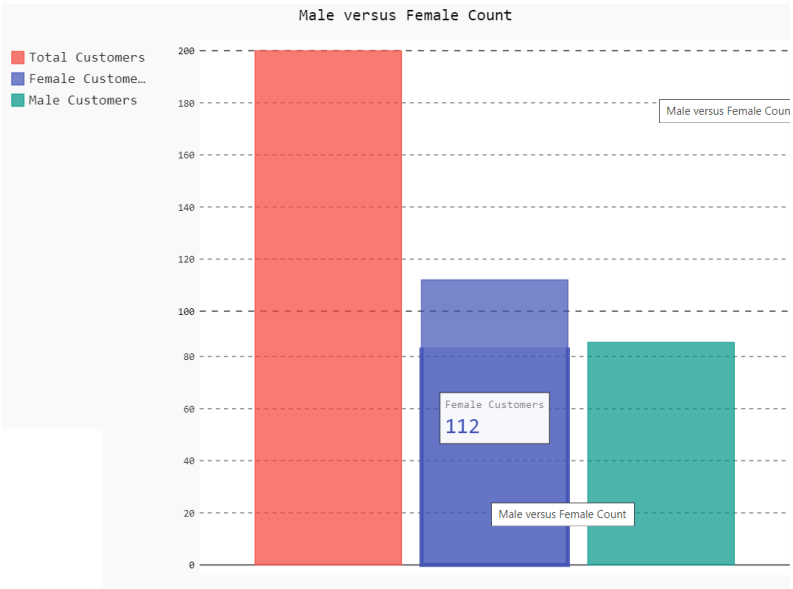
Pygal, like Plotly and Bokeh, offers interactive plots that can be embedded in a web browser. The ability to output charts as SVGs is its prime differentiator. For work involving smaller datasets, SVGs will do just fine. However, for charts with hundreds of thousands of data points, they become sluggish and have trouble rendering. It's easy to create a nice-looking chart with just a few lines of code since each chart type is packaged into a method and the built-in styles are great.

Example

```
import pygal
import pandas as pd
from pygal.style import Style

url =
"https://raw.githubusercontent.com/swapnilsaurav/Dataset/master/Mall_Customers.csv"
data = pd.read_csv(url)
#Total count

tot = len(data['Gender'])
#Total Female Count
fm = len(data[data['Gender'] == 'Female'])
#Total Male Count
```

```
m=len(data[data['Gender'] == 'Male'])
b_chart = pygal.Bar()
b_chart.title = "Male versus Female Count"
b_chart.add("Total Customers", tot)
b_chart.add("Female Customers", fm)
b_chart.add("Male Customers", m)
b_chart.render_in_browser()
```

Output Graph



7. Altair

Altair is a declarative statistical visualization python library based on Vega-Lite. You only need to mention the links between data columns to the encoding channels, such as x-axis, y-axis, color, etc. and the rest of the plotting details are handled automatically. This makes Altair simple, friendly and consistent. It is easy to design effective and beautiful visualizations with a minimal amount of code using Altair.

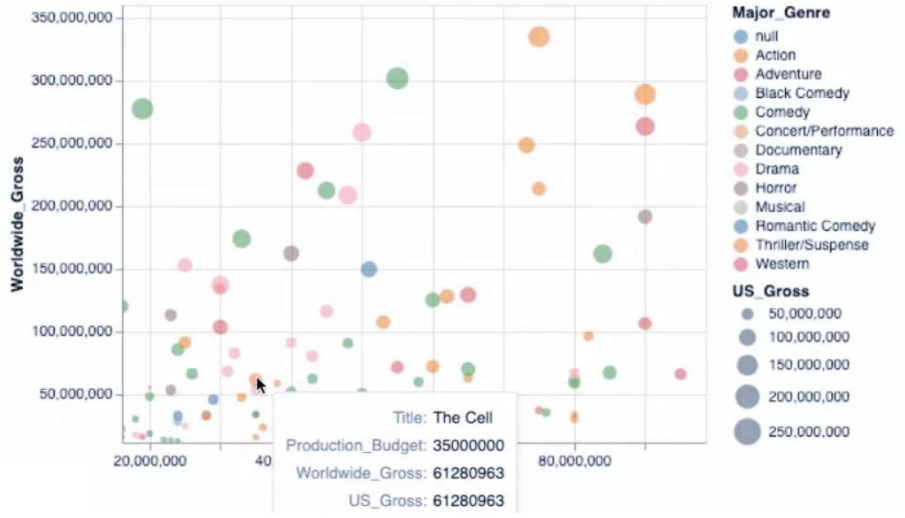
Example

In this example, we will be working with the Movies Dataset from the Vega dataset. This or similar dataset related to movies are famous in the data science community, especially in the context of Recommendation Engines. In this section, we will use it for visualizations with Altair. The dataset contains information related to movies such as the title of the movie, how much money did the movie gross in America and worldwide, along with the production budget, genre, ratings from IMDB and Rotten Tomatoes.

```
#Install the Altair package: pip install altair
#Install the package with the dataset: pip install vega_datasets
import pandas as pd
import altair as alt

# Importing the Vega Dataset
```

```
from vega_datasets import data as vega_data
movies_df = pd.read_json(vega_data.movies.url)

def extract_year(value):
    ''' Extract Year from a given date'''
    return pd.to_datetime(value, format='%b %d %Y').year

# Checking the type of data that we get
print("movies_df is of the type: ", type(movies_df))
print("movies_df: ", movies_df.shape)

#List the columns we have:
print(movies_df.columns)

#Extract year from the Release date
movies_df["Year"] = movies_df["Release Date"].apply(extract_year)

#Preparing the data
movies_2000 = movies_df[movies_df["Year"] == 2000]

ch = alt.Chart(movies_2000).mark_point(filled=True).encode(
    alt.X('Production_Budget'),
    alt.Y('Worldwide_Gross'),
    alt.Size('US_Gross'),
    alt.Color('Major_Genre'),
    alt.OpacityValue(0.7),
    tooltip = [alt.Tooltip('Title'),
               alt.Tooltip('Production_Budget'),
               alt.Tooltip('Worldwide_Gross'),
               alt.Tooltip('US_Gross')
              ]
).interactive()
import altair_viewer
altair_viewer.display(ch, open_browser=True)
```

Output Graph



8. Geoplotlib

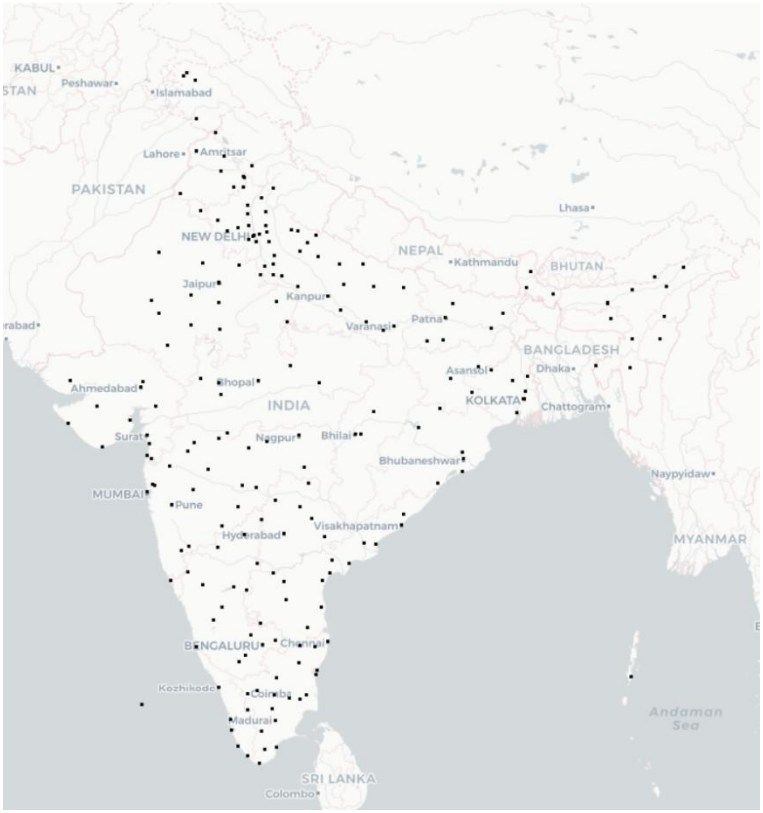
Geoplotlib is a toolbox used for plotting geographical data and map creation. It can be used to create a variety of map-types, like choropleths, heatmaps, and dot density maps. Pyglet (an object-oriented programming interface) is required to be installed to use Geoplotlib. Geoplotlib reduces the complexity of designing visualizations by providing a set of in-built tools for the most common tasks such as density visualization, spatial graphs, and shape files. Since most Python data visualization libraries don't offer maps, it's good to have a library dedicated to them.

Example

Note: We will use simple geospatial dataset of Indian cities and use it for our analysis

```
import pandas as pd
import geoplotlib
from geoplotlib.utils import read_csv

url =
"https://raw.githubusercontent.com/swapnilsaurav/Dataset/master/india_
places.csv"
data = pd.read_csv(url)
print(data)
#Make sure your data following columns:
### lat - which has latitude details
### lon - this has longitude details
geoplotlib.dot(data, color=[0,0,0], point_size=1.5)
geoplotlib.show()
```

Output Graph



9. Missingno

Dealing with missing data is cumbersome. The completeness of a dataset can be gauged quickly with Missingno, rather than painstakingly searching through a table. The user can filter and sort data based on completion or spot correlations with a heat map or a dendrogram.

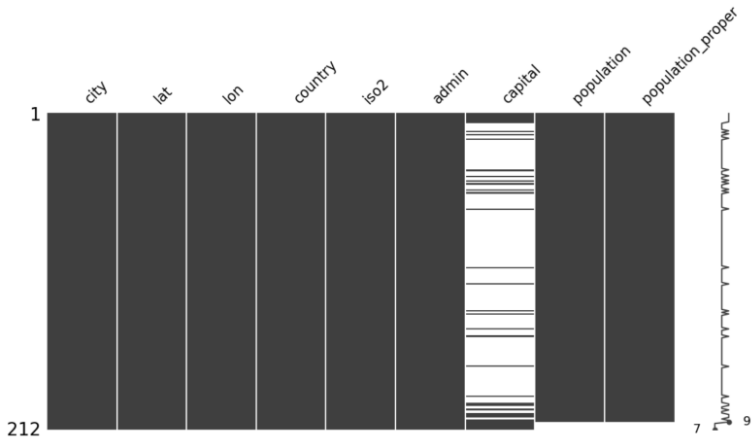
Example

```
import pandas as pd
import matplotlib.pyplot as plt
import missingno as msno
url =
"https://raw.githubusercontent.com/swapnilsaurav/Dataset/master/india_
places.csv"
data = pd.read_csv(url)
print(data)
# Visualize missing values as a matrix
msno.matrix(data)
#display the graph
plt.show()
#Visualize using bar graph
msno.bar(data)
#Save the graph as .png
plt.savefig('comparison.png')
#display the graph
```

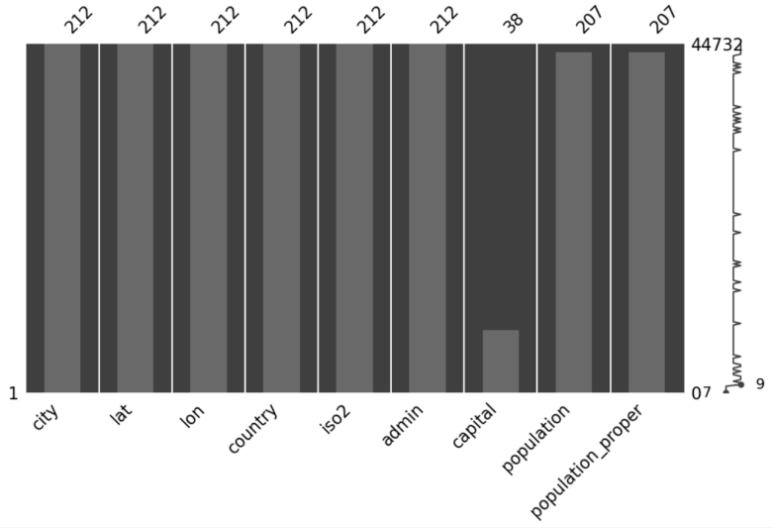
```
plt.show()
```

Output Graph

Matrix graph:



Bar graph:



Above graph shows that all the columns have 212 records except capital which has only 38 values.

10. Leather

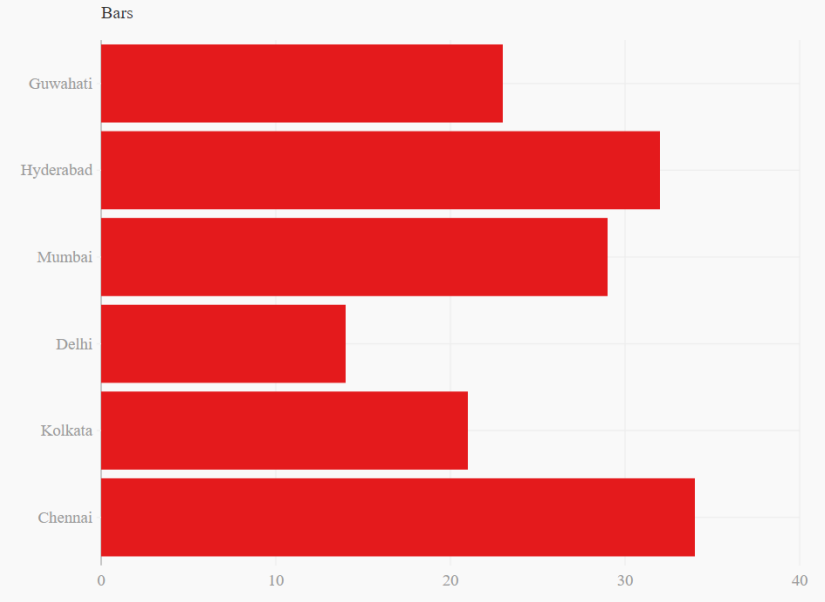
Leather is designed to work with all data types and produces charts such as SVGs, so that they can be scaled without losing image quality. Leather's creator, Christopher Groskopf, puts it best: "Leather is the Python charting library for those who need charts now and don't care if they're perfect." Since this library is relatively new, some of the documentation is still in progress. The charts that can be made are pretty basic—but that's the intention.

There is a wide range of visualization tools, with a huge diversity, depending on the focus of the task at hand available for Python. This is reflected in the sheer number of libraries available. It is imperative for the users to bear in mind the differences between the approaches and their implications before zeroing in on a particular approach.

Example

```
#Sample Bar graph
import leather
data = [
     (23, 'Guwahati'),
     (32, 'Hyderabad'),
     (29, 'Mumbai'),
     (14, 'Delhi'),
     (21, 'Kolkata'),
     (34, 'Chennai')
]
chart = leather.Chart('Bars')
chart.add_bars(data)
chart.to_svg('charts/city_temp.svg')
```

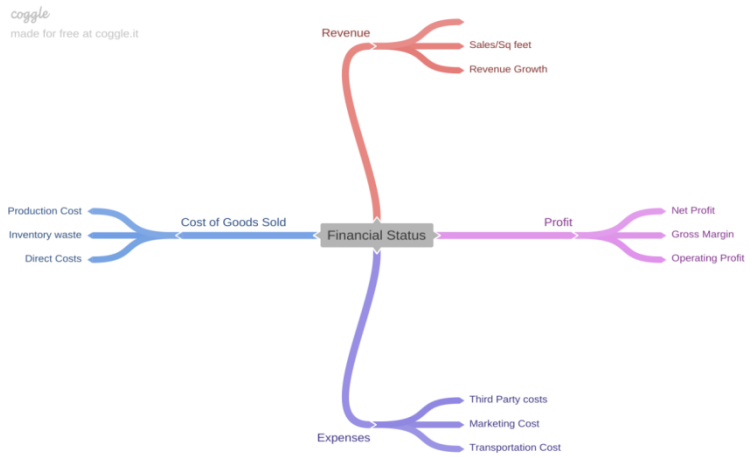
Output Graph



NON-PYTHON TOOLS OUTPUT

Brainstorm Plot by Coggle

Website: coggle.it



CURATED LISTS OF TOOLS

As we have seen, there are many many excellent data visualization tools available for us to use. Challenge is to know all of them. Though we have discussed some of them in this chapter but there are many more which we can use for our purpose after exploring. How do we know them? The following two curated lists are great places to explore some of the awesome tools:

Curated List of Visualization Tools
Maintained by Interactive Things, a design and technology studio based in Zurich, Switzerland.
Link: http://selection.datavisualization.ch/

Visualising Data - Resources List
Curated by Andy Kirk, UK freelance data visualization specialist.
https://www.visualisingdata.com/resources/

With this, we end our discussion on tools here. At the end of the day, you can use any tool but keep just one thing in mind: use the tool that gives you the power to create a great story!