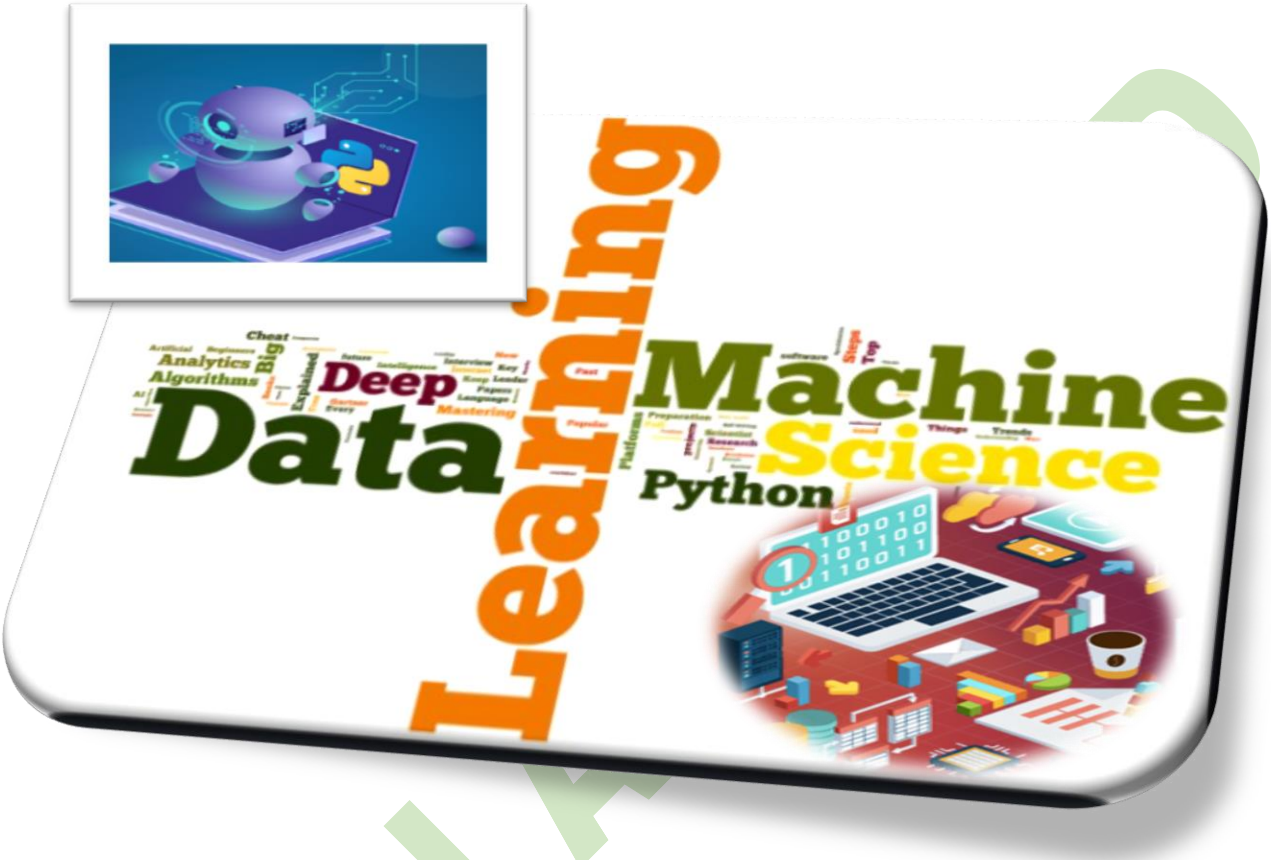
Data science and Machine learning with Python



Designed by Abdur Rahman Joy - MCSD, MCPD, MCSE, MCTS, OCJP, Sr. Technical Trainer for VFX at IDB BISW (Scholarship program), and C#.net, R, Scala, Kotlin, JAVA, Android/IOS/Windows Mobile Apps, SQL server, Azure, Oracle, SharePoint Development, AWS , CEH, KALI Linux, Python, Data Science, Machine Learning ,Software Testing, Graphics, Multimedia and Game Developer at Joy Infosys and other premises like BITM, SkillsJob, PNTL, Leads Training and New Horizon inc , Cell #: +880-1712587348, email: jspaonline@gmail.com. Web URL: http://www.joyinfosys.com/me.

Data Visualization

Introduction

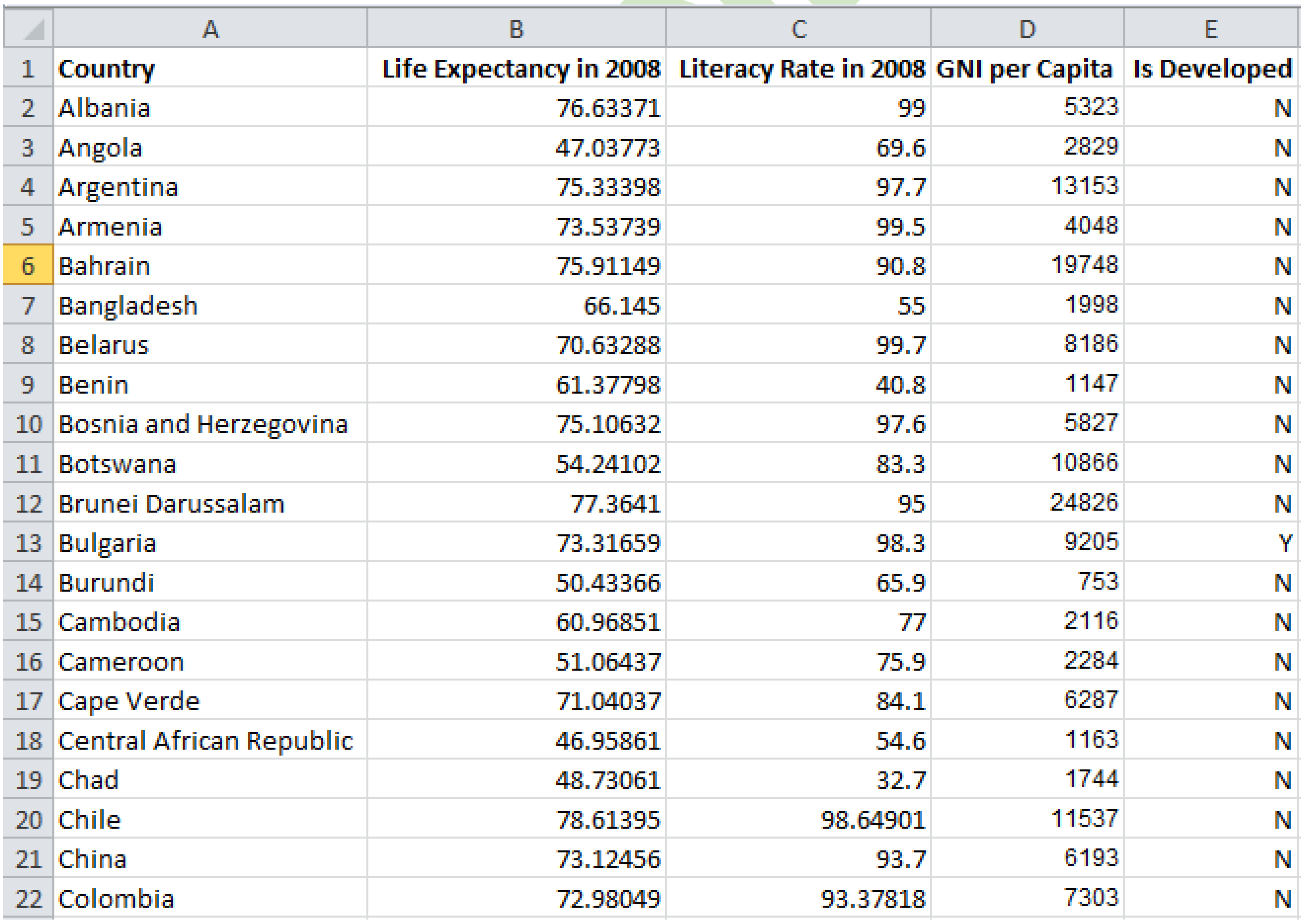
- Data visualization is the process of converting raw data into easily understandable pictorial representation, that enables fast and effective decisions.
- It is both an Art and a Science.
- Data visualization is a strategy where we represent the quantitative information in a graphical form.

Why Data Visualization?

Now the question is that why we visualize data? So, the answer is that the pictorial form of data is easily understandable rather than huge numbers of numerical data.

You might heard that **"A picture is worth a thousand words"** ,as a result data visualization is so important.Therefore the key points are –

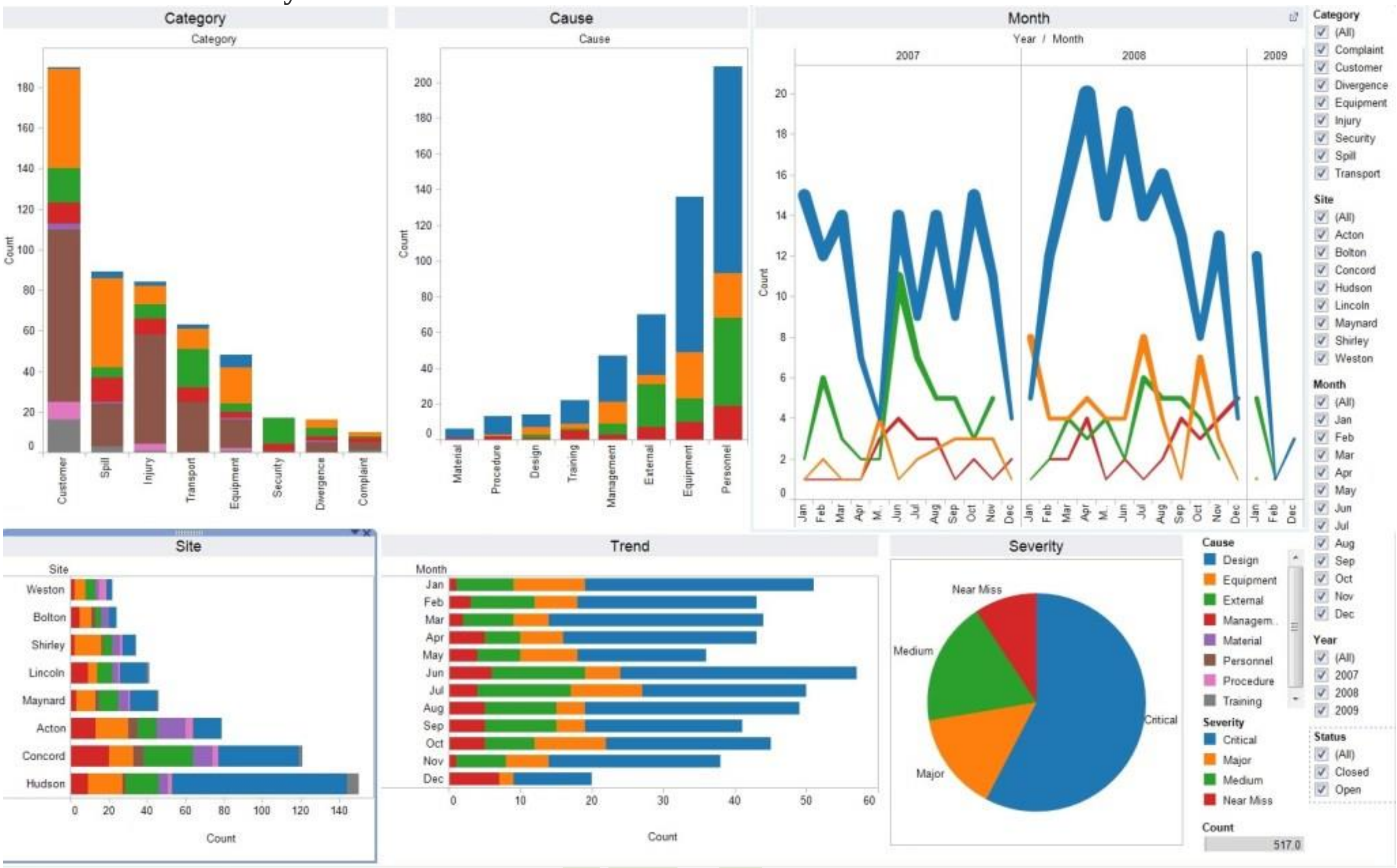
- Data visualization is the first step of analysis work.
- It gives intuitive understanding of data.
- Helps you to see data in certain meaningful patterns.
- Visual representations enhances the human cognitive process.



	A	B	C	D	E
1	**Country**	**Life Expectancy in 2008**	**Literacy Rate in 2008**	**GNI per Capita**	**Is Developed**
2	Albania	76.63371	99	5323	N
3	Angola	47.03773	69.6	2829	N
4	Argentina	75.33398	97.7	13153	N
5	Armenia	73.53739	99.5	4048	N
6	Bahrain	75.91149	90.8	19748	N
7	Bangladesh	66.145	55	1998	N
8	Belarus	70.63288	99.7	8186	N
9	Benin	61.37798	40.8	1147	N
10	Bosnia and Herzegovina	75.10632	97.6	5827	N
11	Botswana	54.24102	83.3	10866	N
12	Brunei Darussalam	77.3641	95	24826	N
13	Bulgaria	73.31659	98.3	9205	Y
14	Burundi	50.43366	65.9	753	N
15	Cambodia	60.96851	77	2116	N
16	Cameroon	51.06437	75.9	2284	N
17	Cape Verde	71.04037	84.1	6287	N
18	Central African Republic	46.95861	54.6	1163	N
19	Chad	48.73061	32.7	1744	N
20	Chile	78.61395	98.64901	11537	N
21	China	73.12456	93.7	6193	N
22	Colombia	72.98049	93.37818	7303	N

Designed by Abdur Rahman Joy - MCSD, MCPD, MCSE, MCTS, OCJP, Sr. Technical Trainer for VFX at IDB BISW (Scholarship program), and C#.net, R, Scala, Kotlin, JAVA, Android/IOS/Windows Mobile Apps, SQL server, Azure, Oracle, SharePoint Development, AWS , CEH, KALI Linux, Python, Data Science, Machine Learning ,Software Testing, Graphics, Multimedia and Game Developer at Joy Infosys and other premises like BITM, SkillsJob, PNTL, Leads Training and New Horizon inc , Cell #: +880-1712587348, email: jspaonline@gmail.com. Web URL: http://www.joyinfosys.com/me.

Data Visualization Python



Data Visualization Python

Let’s take an example, we see in the above two pictures, first we have data in the numerical form and then next is pictorial representation of data.

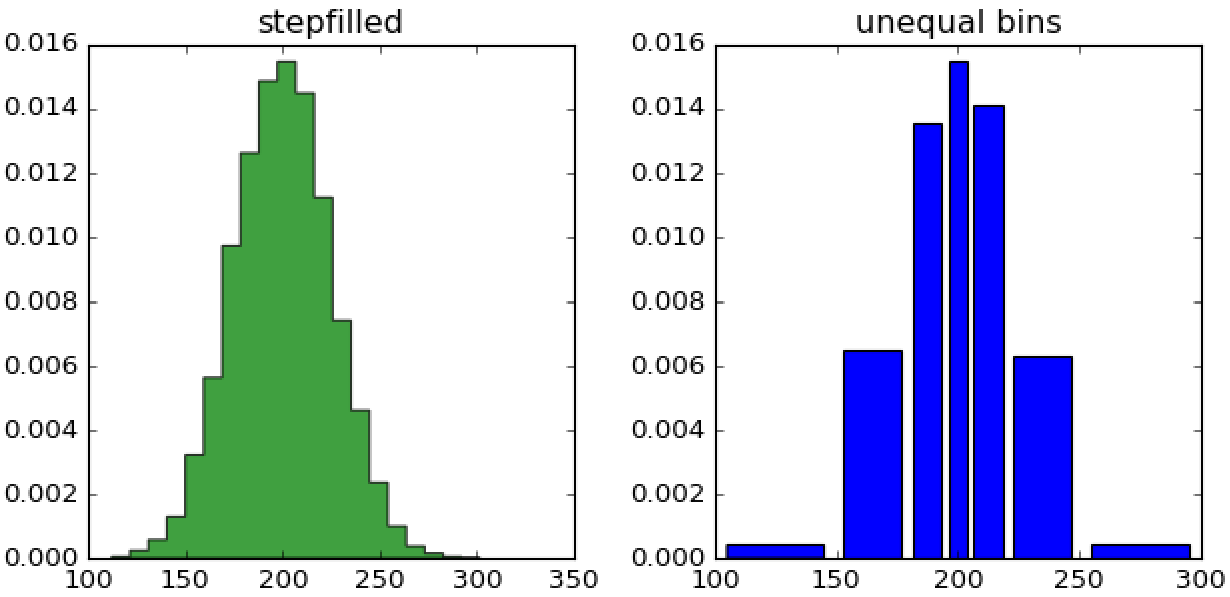
So, now let me know that whether it is easy to analyze the numbers or visualization, definitely the visualization. Human brain works faster when we see a picture rather than data, so that is the main concept we go for data visualization.

Benefits of Data Visualization

- Data visualization allow users to see several different perspectives of data.
- Data visualization makes it possible to interpret vast amounts of data.
- It offers ability to note expectations in data.
- Exploring trends within a database through visualization by letting analysts navigate through data and visually orient themselves to the patterns in the data.

Useful Python Data Visualization Libraries for Any Discipline:

matplotlib



Two histograms (matplotlib)

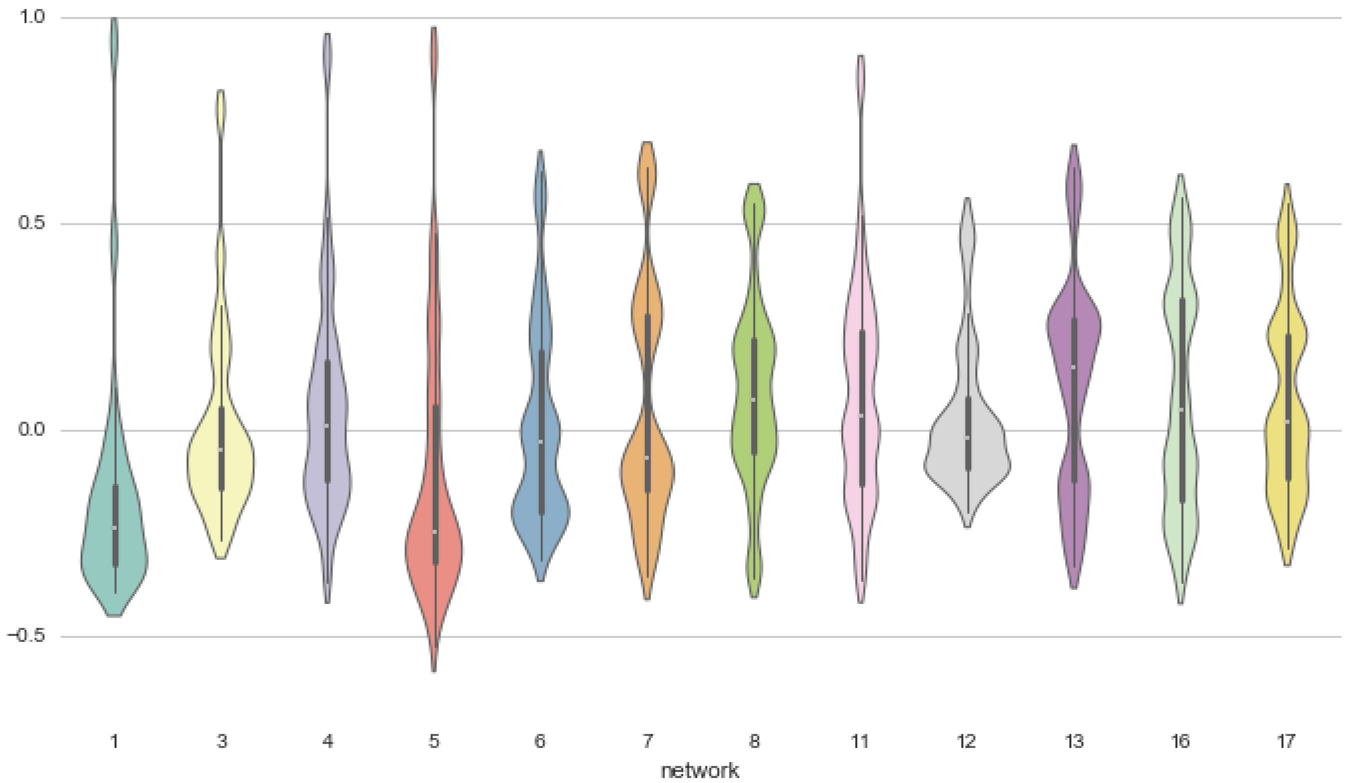
matplotlib is the O.G. of Python data visualization libraries. Despite being over a decade old, it's still the most widely used library for plotting in the Python community. It was designed to closely resemble MATLAB, a proprietary programming language developed in the 1980s.

Because matplotlib was the first Python data visualization library, many other libraries are built on top of it or designed to work in tandem with it during analysis. Some libraries like pandas and Seaborn are “wrappers” over matplotlib. They allow you to access a number of matplotlib’s methods with less code.

While matplotlib is good for getting a sense of the data, it's not very useful for creating publication-quality charts *quickly* and *easily*. As Chris Moffitt points out in his overview of Python visualization tools, matplotlib “is extremely powerful but with that power comes complexity.”

matplotlib has long been criticized for its default styles, which have a distinct 1990s feel. The upcoming release of matplotlib 2.0 promises many new style changes to address this problem.

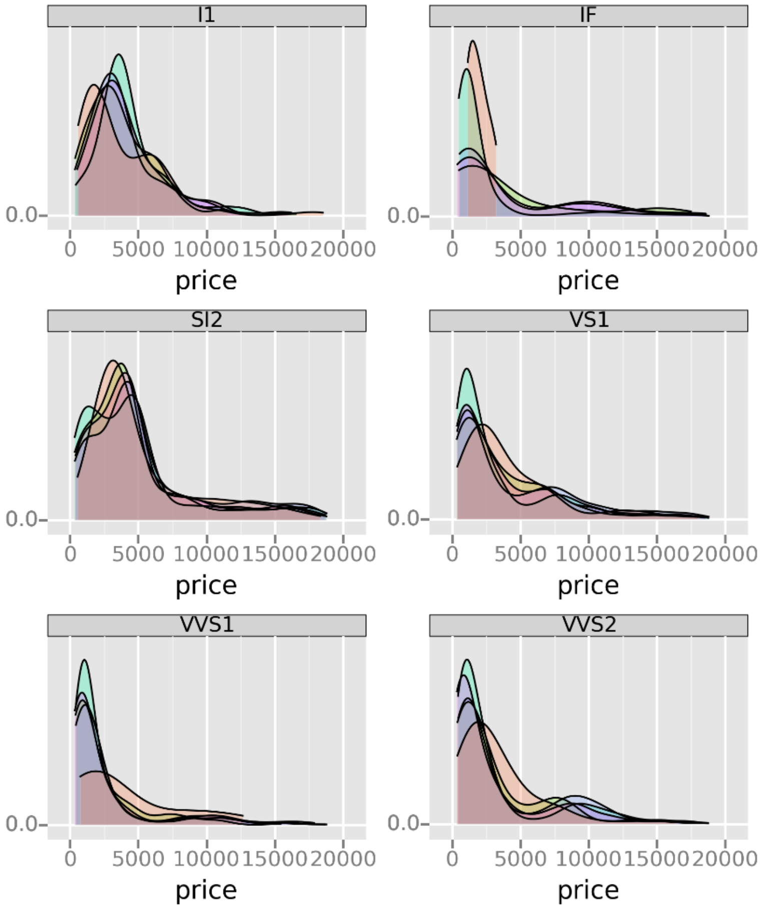
Seaborn



Violinplot (Michael Waskom)

Seaborn harnesses the power of matplotlib to create beautiful charts in a few lines of code. The key difference is Seaborn's default styles and color palettes, which are designed to be more aesthetically pleasing and modern. Since Seaborn is built on top of matplotlib, you'll need to know matplotlib to tweak Seaborn's defaults.

ggplot



Small multiples (ŷhat)

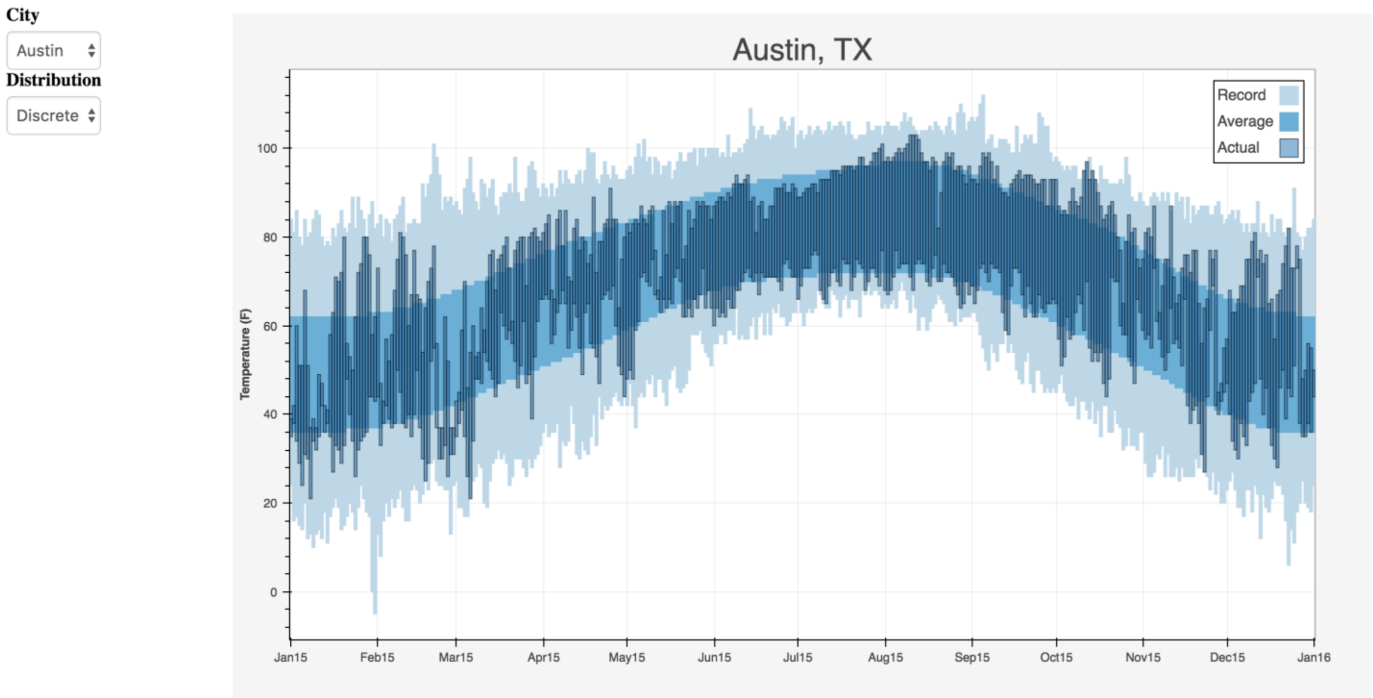
ggplot is based on ggplot2, an R plotting system, and concepts from *The Grammar of Graphics.* ggplot operates differently than matplotlib: it lets you layer components to create a complete plot.

For instance, you can start with axes, then add points, then a line, a trendline, etc. Although *The Grammar of Graphics* has been praised as an “intuitive” method for plotting, seasoned matplotlib users might need time to adjust to this new mindset.

According to the creator, ggplot isn't designed for creating highly customized graphics. It sacrifices complexity for a simpler method of plotting.

ggplot is tightly integrated with pandas, so it's best to store your data in a DataFrame when using ggplot.

Bokeh



City: Austin

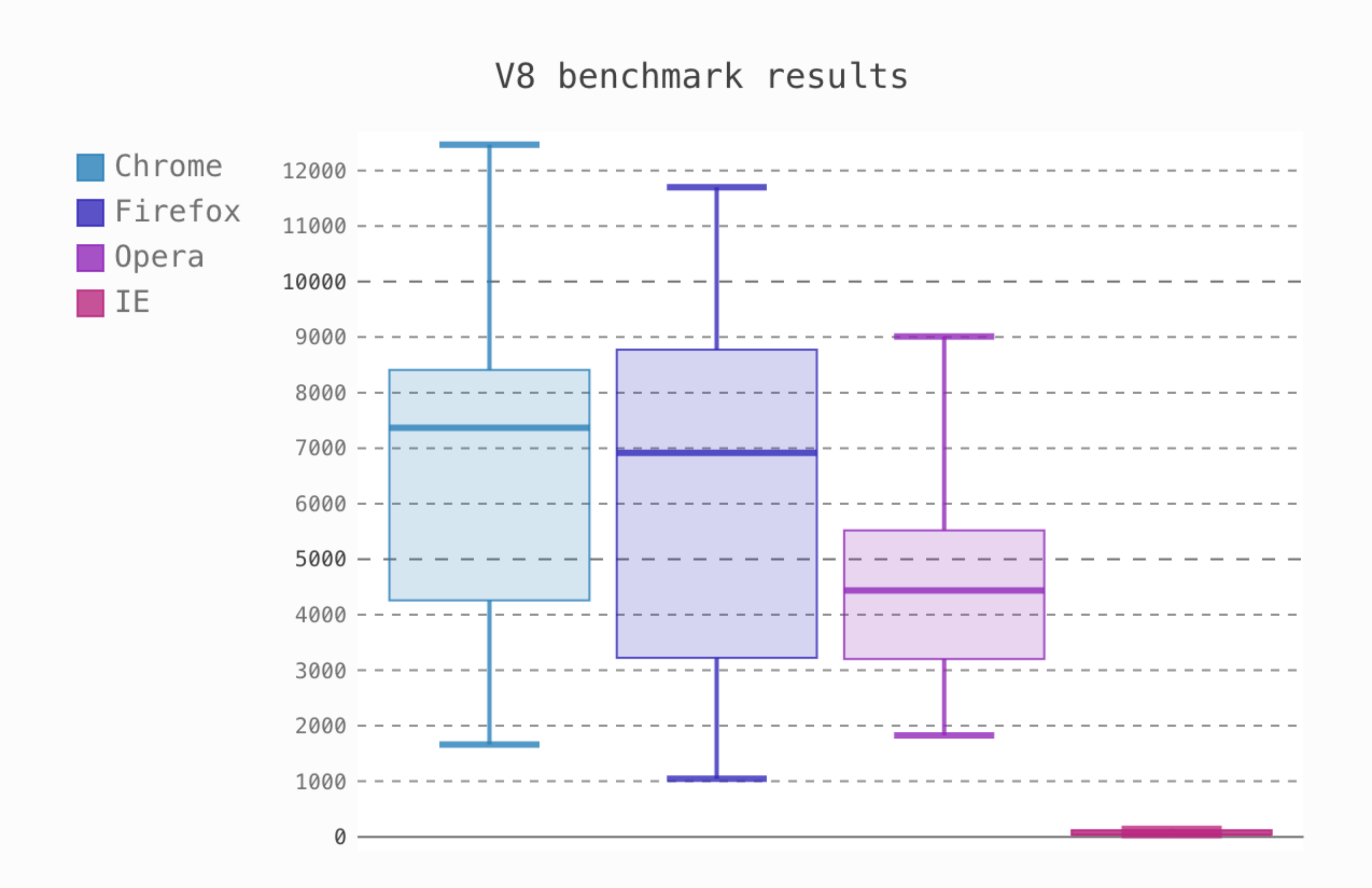
Distribution: Discrete

Interactive weather statistics for three cities (Continuum Analytics)

Like ggplot, Bokeh is based on *The Grammar of Graphics*, but unlike ggplot, it's native to Python, not ported over from R. Its strength lies in the ability to create interactive, web-ready plots, which can be easily output as JSON objects, HTML documents, or interactive web applications. Bokeh also supports streaming and real-time data.

Bokeh provides three interfaces with varying levels of control to accommodate different user types. The highest level is for creating charts quickly. It includes methods for creating common charts such as bar plots, box plots, and histograms. The middle level has the same specificity as matplotlib and allows you to control the basic building blocks of each chart (the dots in a scatter plot, for example). The lowest level is geared toward developers and software engineers. It has no pre-set defaults and requires you to define every element of the chart.

pygal

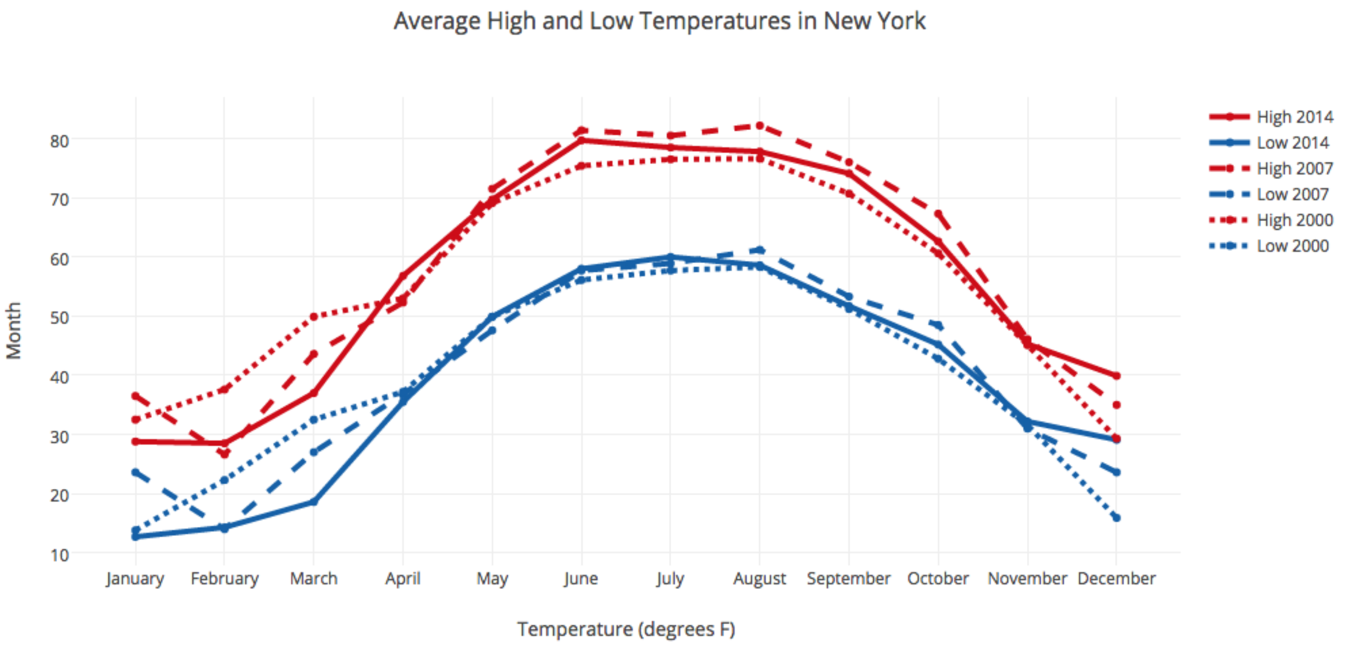


Box plot (Florian Mounier)

Like Bokeh and Plotly, pygal offers interactive plots that can be embedded in the web browser. Its prime differentiator is the ability to output charts as SVGs. As long as you're working with smaller datasets, SVGs will do you just fine. But if you're making charts with hundreds of thousands of data points, they'll have trouble rendering and become sluggish.

Since each chart type is packaged into a method and the built-in styles are pretty, it's easy to create a nice-looking chart in a few lines of code.

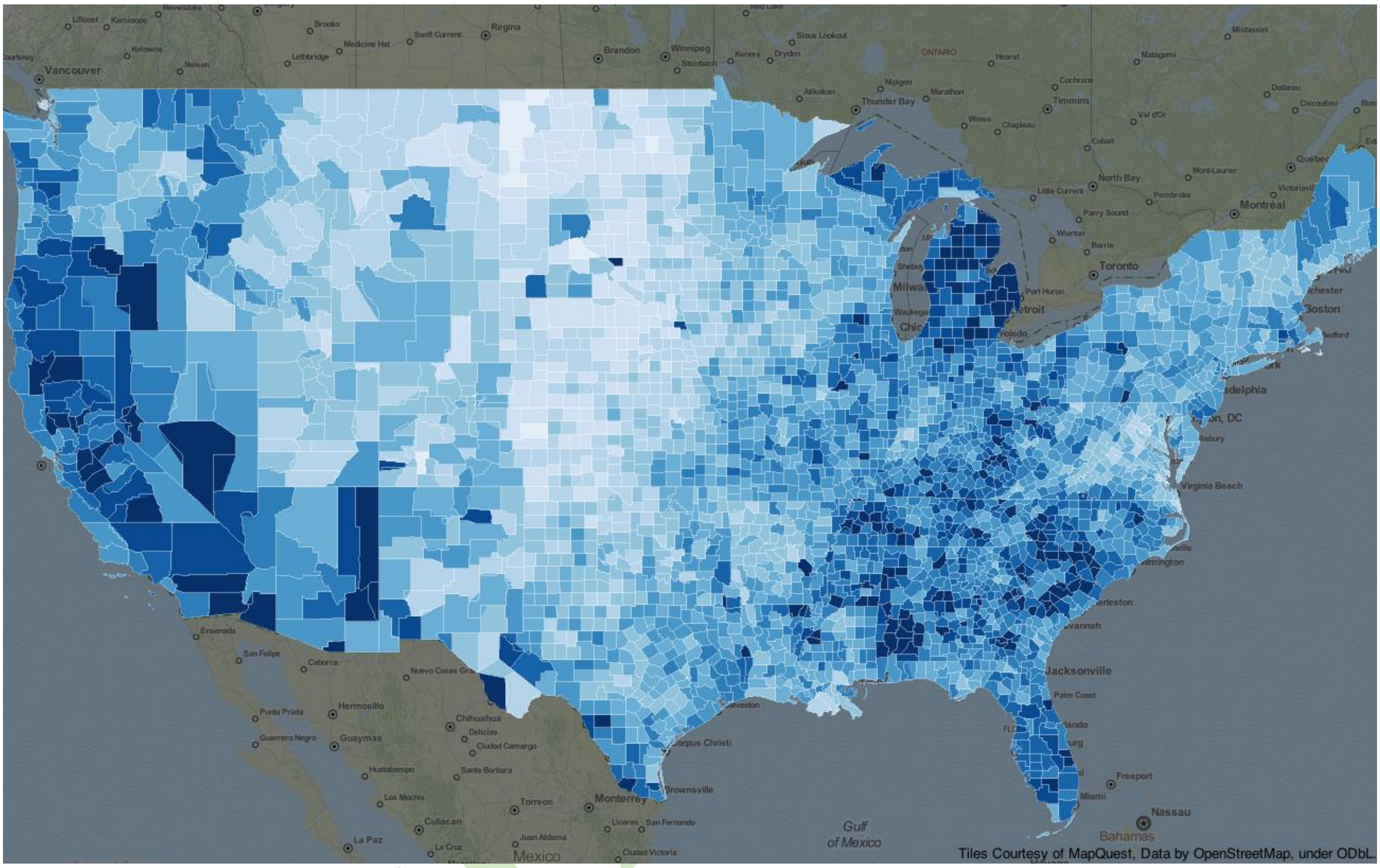
Plotly



Line plot (Plotly)

You might know Plotly as an online platform for data visualization, but did you also know you can access its capabilities from a Python notebook? Like Bokeh, Plotly's forte is making interactive plots, but it offers some charts you won't find in most libraries, like contour plots, dendograms, and 3D charts.

geoplotlib

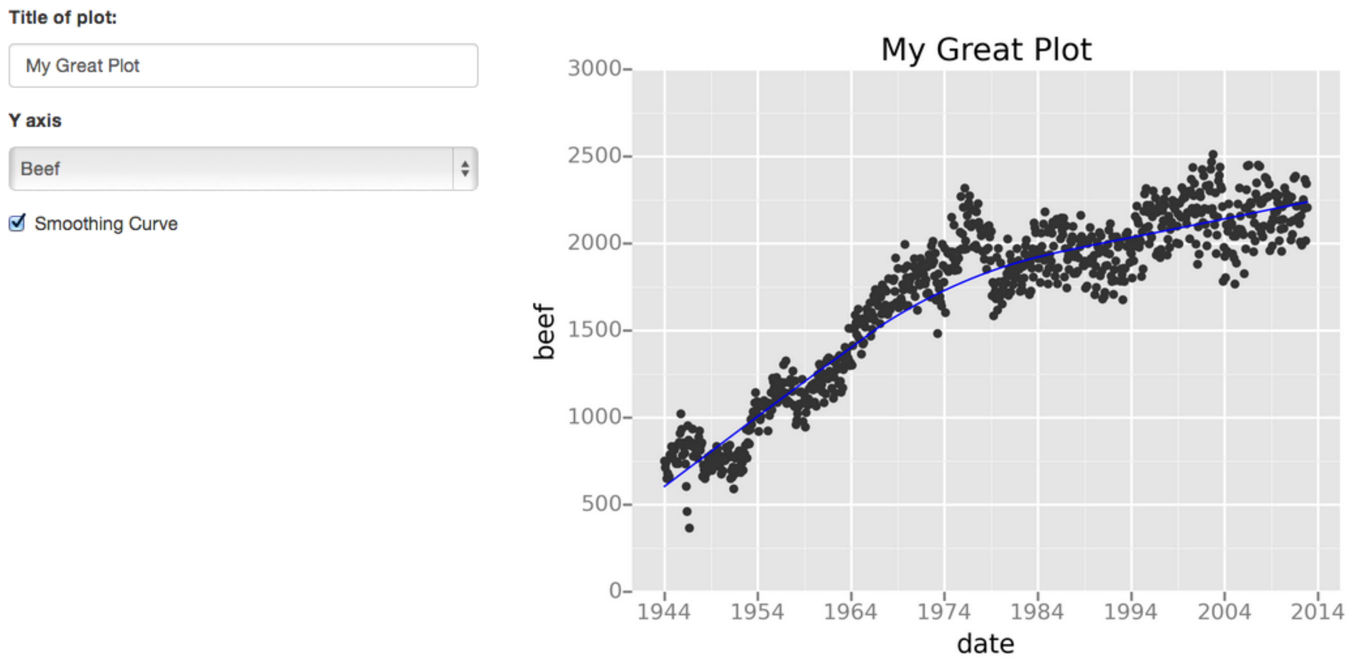


Tiles Courtesy of MapQuest, Data by OpenStreetMap, under ODbL.

Choropleth (Andrea Cuttone)

geoplotlib is a toolbox for creating maps and plotting geographical data. You can use it to create a variety of map-types, like choropleths, heatmaps, and dot density maps. You must have Pyglet (an object-oriented programming interface) installed to use geoplotlib. Nonetheless, since most Python data visualization libraries don't offer maps, it's nice to have a library dedicated solely to them.

Gleam



Title of plot: My Great Plot

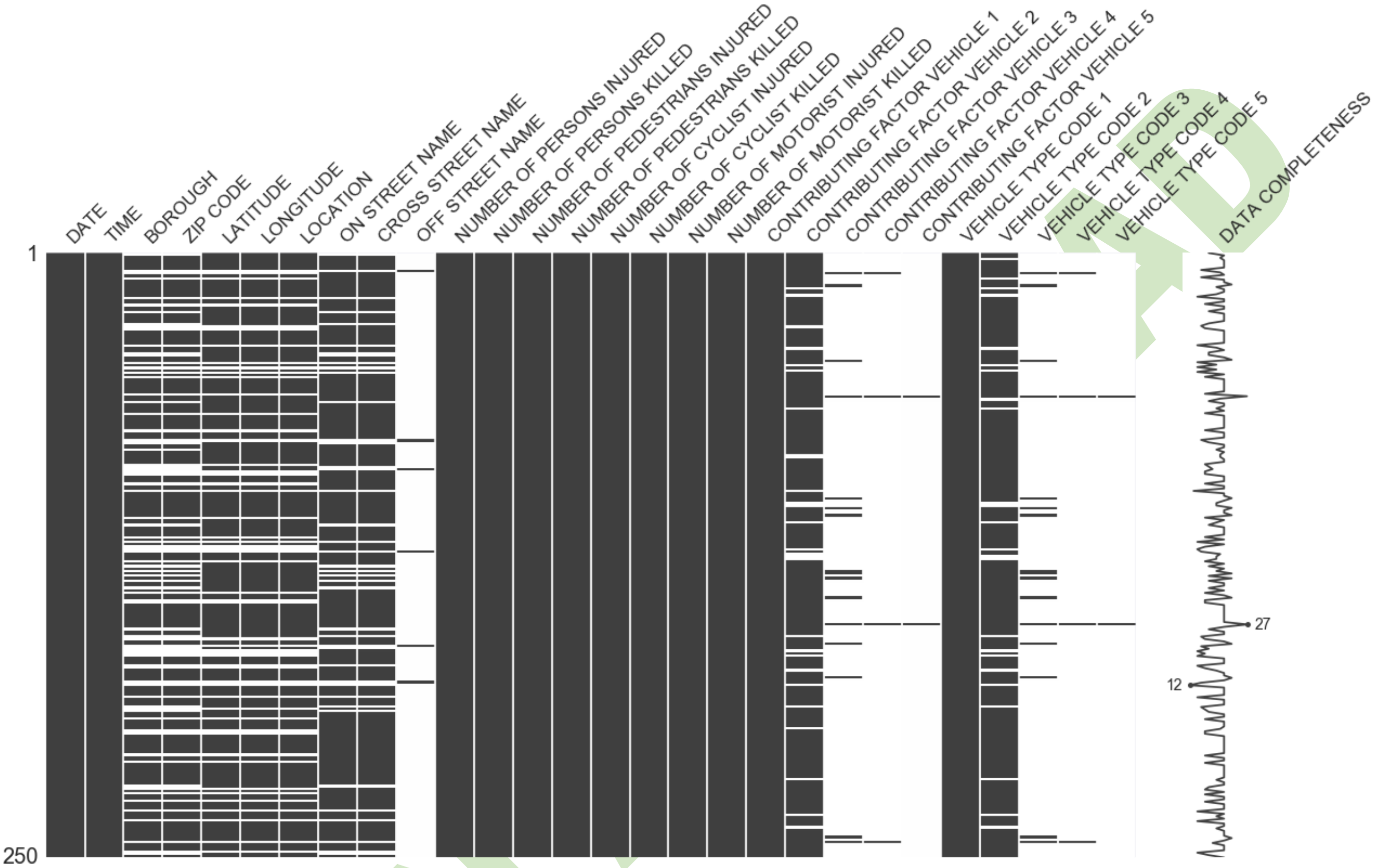
Y axis: Beef

☑ Smoothing Curve

Scatter plot with trend line (David Robinson)

Gleam is inspired by R's Shiny package. It allows you to turn analyses into interactive web apps using only Python scripts, so you don't have to know any other languages like HTML, CSS, or JavaScript. Gleam works with any Python data visualization library. Once you've created a plot, you can build fields on top of it so users can filter and sort data.

missingno



Nullity matrix (Aleksey Bilogur)

Dealing with missing data is a pain. missingno allows you to quickly gauge the completeness of a dataset with a visual summary, instead of trudging through a table. You can filter and sort data based on completion or spot correlations with a heatmap or a dendrogram.

Leather

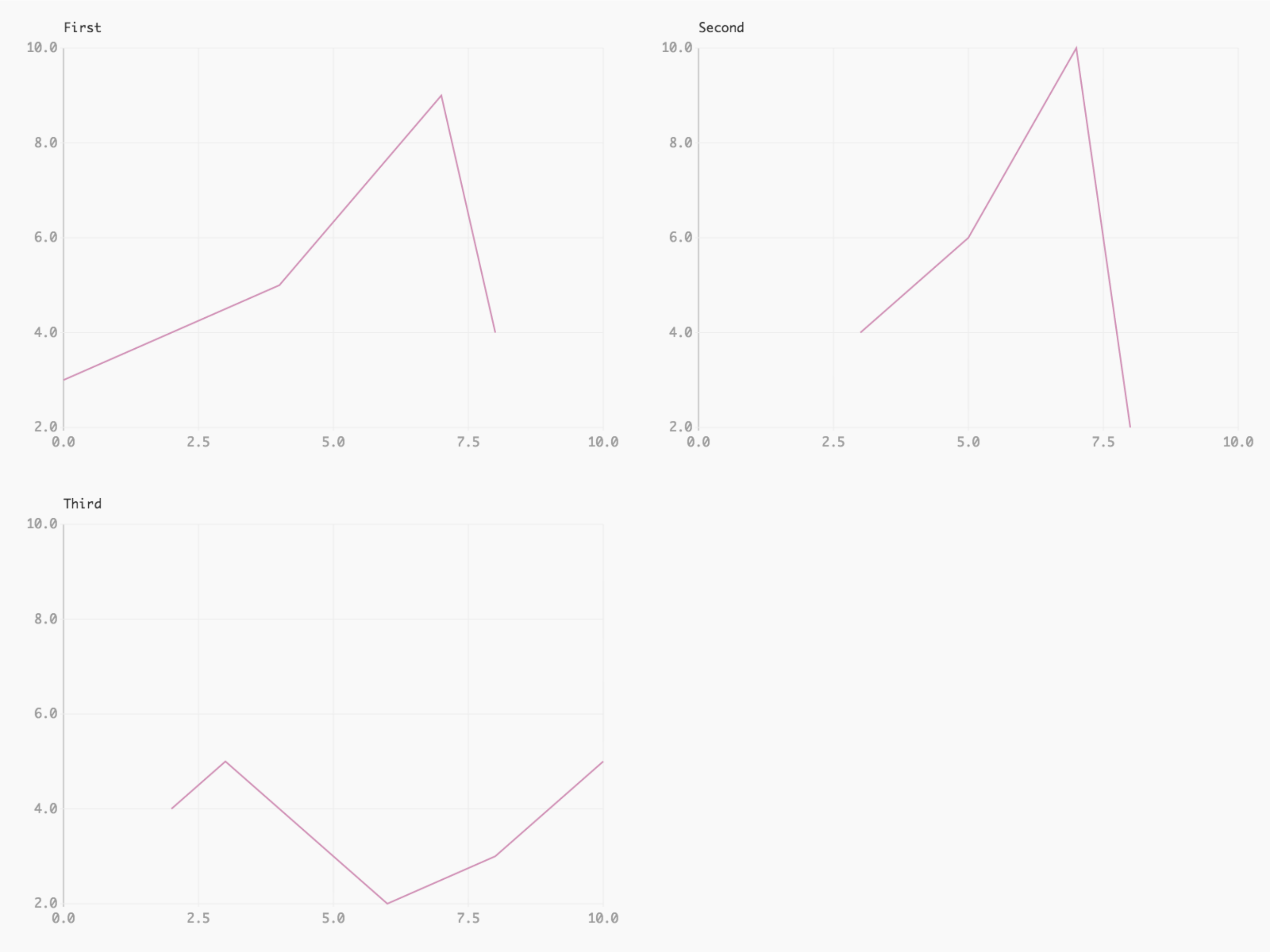


Chart grid with consistent scales (Christopher Groskopf)

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.” It's designed to work with all data types and produces charts as SVGs, so you can scale them without losing image quality. Since this library is relatively new, some of the documentation is still in progress. The charts you can make are pretty basic—but that's the intention.

Data Visualization Techniques

Some of the popular techniques are:

- Pie chart
- Bar graph
- Histogram
- Wordle or tag cloud
- Tree map
- Scatter plot
- Line chart
- Bubble chart etc.

Data Visualization Python Tutorial

Python provides many libraries for data visualization like matplotlib, seaborn, ggplot, Bokeh etc.Here i am using the most popular **matplotlib** library.So let’s a look on matplotlib.

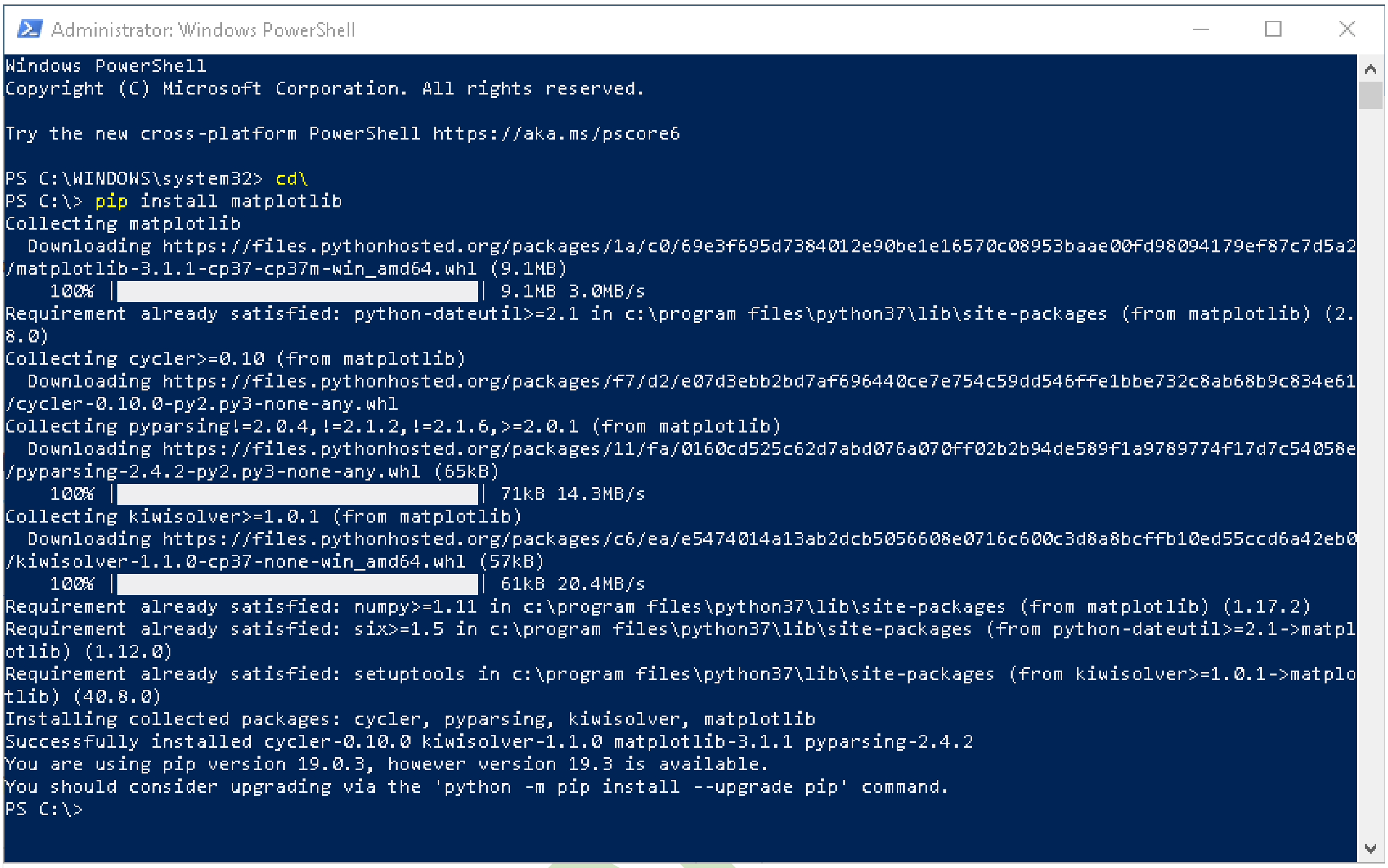
Matplotlib

- Matplotlib library is a graph plotting library of python.
- Using matplotlib we can plot different scatter plots, line graphs, bar graphs, pie chart and histograms .
- Using these plots we can visualize our data.
- It provides an object-oriented APIs for embedding plots into applications using general-purpose GUI toolkits like Tkinter, wxPython, Qt, or GTK+.

Installing Matplotlib

Run the following command for installing matplotlib –

```
pip install matplotlib
```



```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\WINDOWS\system32> cd\
PS C:\> pip install matplotlib
Collecting matplotlib
  Downloading https://files.pythonhosted.org/packages/1a/c0/69e3f695d7384012e90be1e16570c08953baae00fd98094179ef87c7d5a2
/matplotlib-3.1.1-cp37-cp37m-win_amd64.whl (9.1MB)
    100% |████████████████████████████████| 9.1MB 3.0MB/s
Requirement already satisfied: python-dateutil>=2.1 in c:\program files\python37\lib\site-packages (from matplotlib) (2.
8.0)
Collecting cycler>=0.10 (from matplotlib)
  Downloading https://files.pythonhosted.org/packages/f7/d2/e07d3ebb2bd7af696440ce7e754c59dd546ffe1bbe732c8ab68b9c834e61
/cycler-0.10.0-py2.py3-none-any.whl
Collecting pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.1 (from matplotlib)
  Downloading https://files.pythonhosted.org/packages/11/fa/0160cd525c62d7abd076a070ff02b2b94de589f1a9789774f17d7c54058e
/pyparsing-2.4.2-py2.py3-none-any.whl (65kB)
    100% |████████████████████████████████| 71kB 14.3MB/s
Collecting kiwisolver>=1.0.1 (from matplotlib)
  Downloading https://files.pythonhosted.org/packages/c6/ea/e5474014a13ab2dcb5056608e0716c600c3d8a8bcffb10ed55ccd6a42eb0
/kiwisolver-1.1.0-cp37-none-win_amd64.whl (57kB)
    100% |████████████████████████████████| 61kB 20.4MB/s
Requirement already satisfied: numpy>=1.11 in c:\program files\python37\lib\site-packages (from matplotlib) (1.17.2)
Requirement already satisfied: six>=1.5 in c:\program files\python37\lib\site-packages (from python-dateutil>=2.1->matpl
otlib) (1.12.0)
Requirement already satisfied: setuptools in c:\program files\python37\lib\site-packages (from kiwisolver>=1.0.1->matplo
tlib) (40.8.0)
Installing collected packages: cycler, pyparsing, kiwisolver, matplotlib
Successfully installed cycler-0.10.0 kiwisolver-1.1.0 matplotlib-3.1.1 pyparsing-2.4.2
You are using pip version 19.0.3, however version 19.3 is available.
You should consider upgrading via the 'python -m pip install --upgrade pip' command.
PS C:\>
```

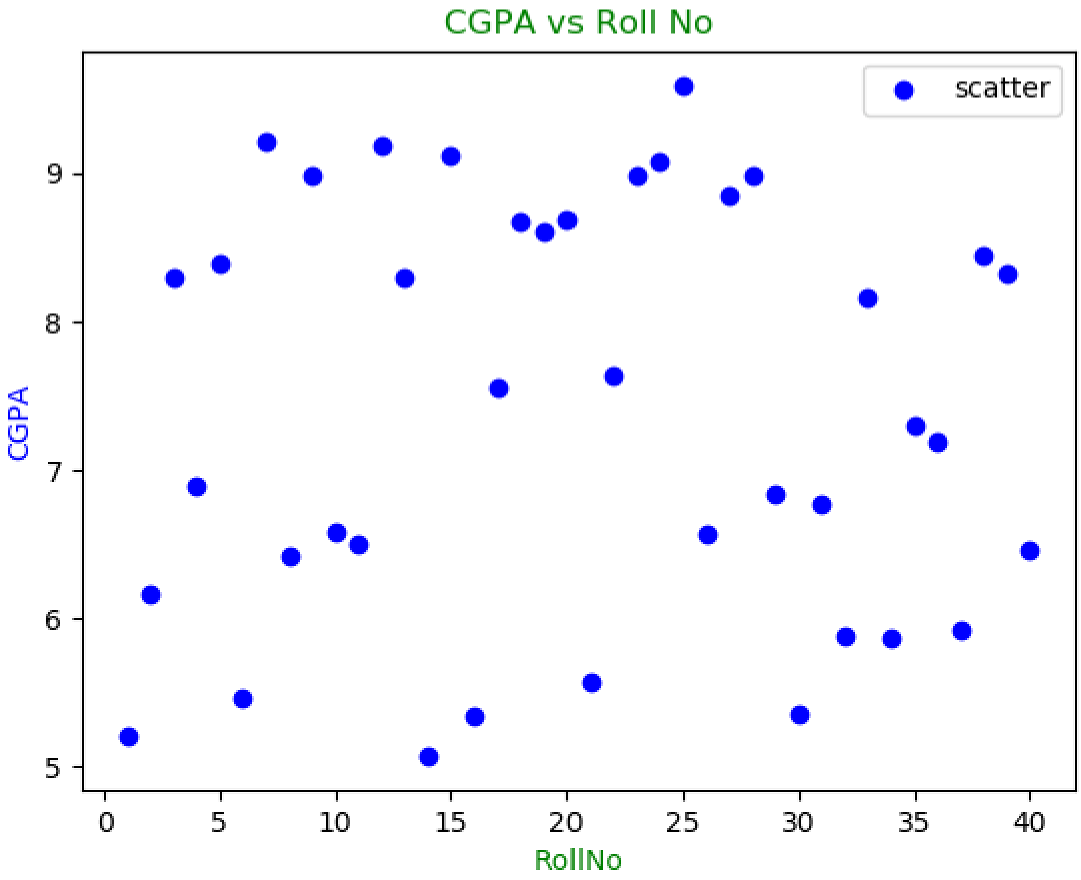
Creating different visualizations

We can create different types of visualization using matplotlib.Now learn them one by one.

#1 Scatter Plot

First of all we create a scatter plot.Here we are using pyplot .**Pyplot provides the state-machine interface to the underlying plotting library in matplotlib.** This means that figures and axes are implicitly and automatically created to achieve the desired plot

```
import matplotlib.pyplot as pt  #importing matplotlib pt is an alias of pyplot
import pandas as pd             #importing pandas , pd is an elias of pandas

 #If we want to read something from dataset or csv file we use read_csv() method.
data= pd.read_csv("cgpa.csv")

data= data.head(40)      # head() method is used to select some elements of dataset

#here i am plotting rollno vs cgpa so x-axis will be rollno and y-axis will be cgpa
#color of scatter is blue and we also include label as scatter.
#scatter() method is used to plot the scatter

pt.scatter(data["rollno"],data["cgpa"],color="blue",label="scatter")
pt.xlabel("RollNo",color="green") #xlabel() defines the label of x-axis
pt.ylabel("CGPA",color="blue") #ylabel() defines the label of x-axis
pt.title("CGPA vs Roll No",color="green") #title() is used to give title of this scatter plot

pt.show()
```

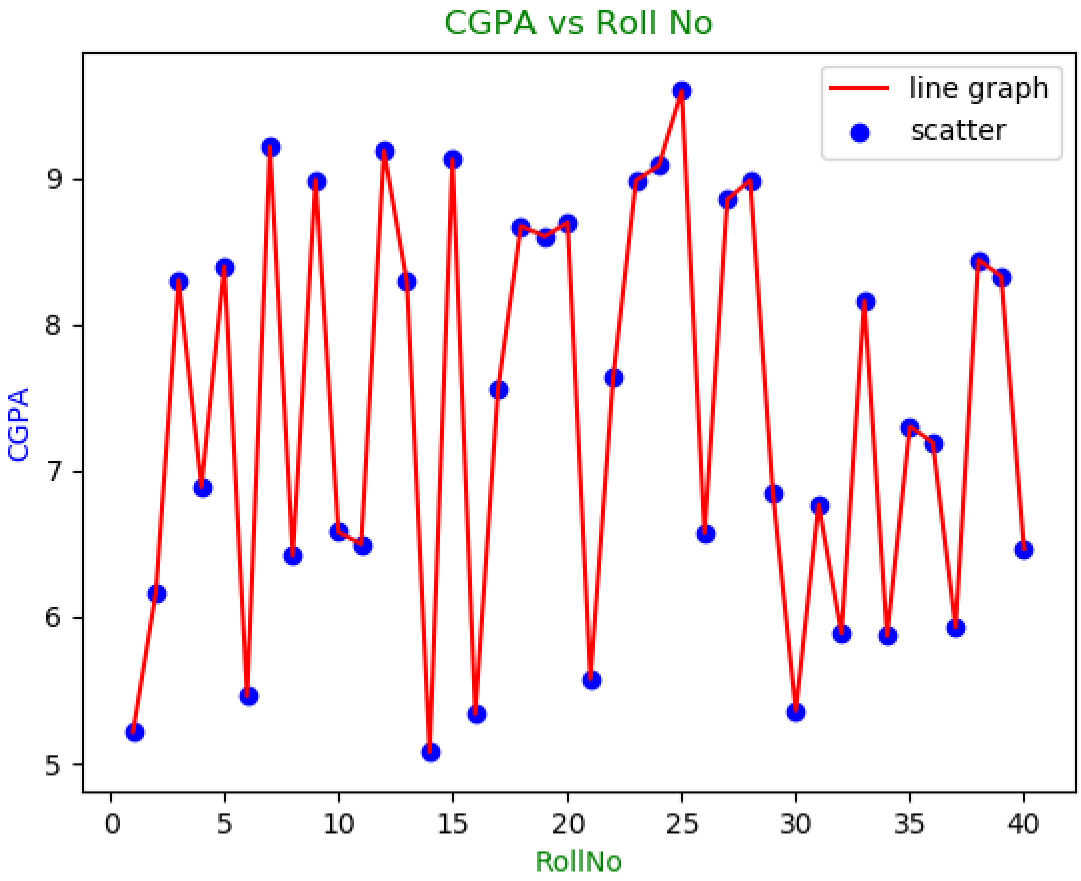


Finally that is looking pretty cool. Now if we want to include line graph into this we just have to add two line in the above code.

```
import matplotlib.pyplot as pt  #importing matplotlib pt is an alias of pyplot
import pandas as pd           #importing pandas , pd is an elias of pandas

 #If we want to read something from dataset or csv file we use read_csv() method.
data= pd.read_csv("cgpa.csv")

data= data.head(40)     # head() method is used to select some elements of dataset

#here i am plotting rollno vs cgpa so x-axis will be rollno and y-axis will be cgpa
```

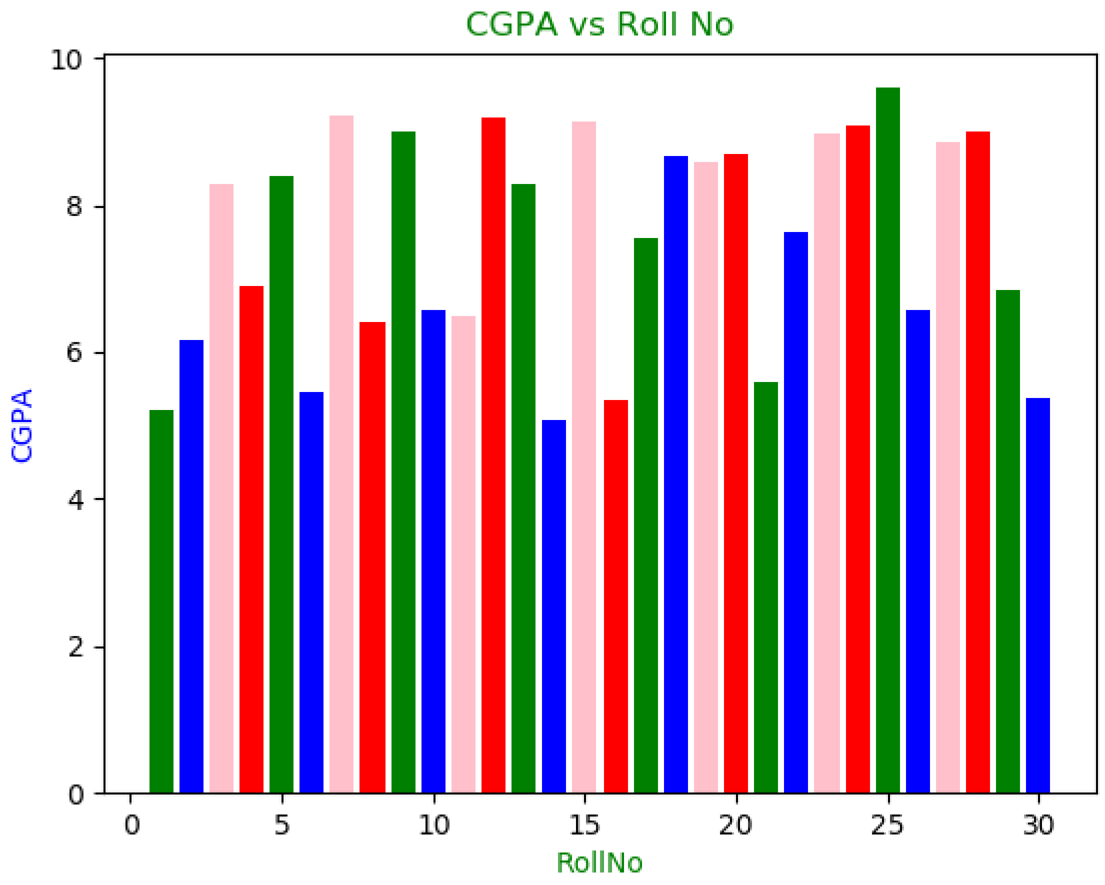
```
#color of scatter is blue and we also include label as scatter.
#scatter() method is used to plot the scatter

pt.scatter(data["rollno"],data["cgpa"],color="blue",label="scatter")

pt.xlabel("RollNo",color="green") #xlabel() defines the label of x-axis
pt.ylabel("CGPA",color="blue") #ylabel() defines the label of x-axis
pt.title("CGPA vs Roll No",color="green") #title() is used to give title of this scatter plot

pt.plot(data["rollno"],data["cgpa"],color="red",label="line graph") #plot() is used to create line graph
pt.legend() # Calling&nbsp;legend()&nbsp;with no arguments automatically fetches the legend handles
and their associated labels

pt.show()
```

So the output will be



```
import matplotlib.pyplot as pt
import pandas as pd

data = pd.read_csv("cgpa.csv")
data= data.head(30)
#bar() method is used to plot a bar graph
#Here i am taking a list of colors to showing graph attractive

pt.bar(data["rollno"],data["cgpa"],color=["green","blue","pink","red"])

pt.xlabel("RollNo",color="green")
pt.ylabel("CGPA",color="blue")
pt.title("CGPA vs Roll No",color="green")
pt.show()
```

Hence the output will be as –



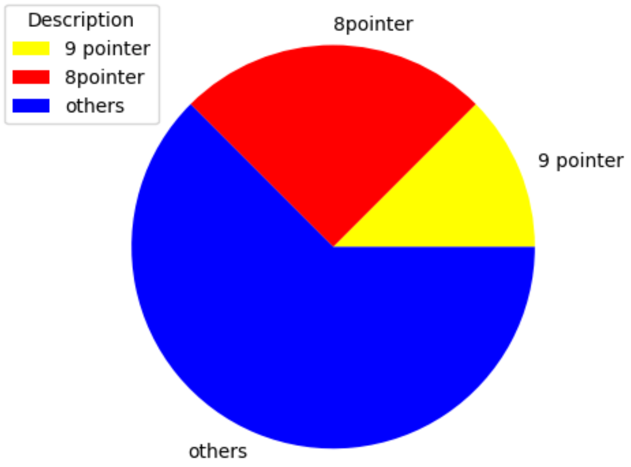
#3 Pie Charts

Now let’s learn about pie charts.Pie charts can be drawn using the function **pie()** in the pyplot module. The below python code example draws a pie chart using the pie()function.

Let’s assume we gonna see how many students got 9 points and how many students got 8 points,7 points and so on.Pie chart will be the best option to visualize these kinds of data.

```
import matplotlib.pyplot as pt
import pandas as pd

data = pd.read_csv("cgpa.csv")
data = data.head(30)


x=len(data[data.cgpa>=9])  #students having cgpa over 9 point
x1=len(data[(data.cgpa>=8) & (data.cgpa<9)])  #students got over 8 point but less than 9 point
x2=len(data[data.cgpa<8])    #students having cgpa less than 8 point

pt.axis('equal')  #for making pie chart circular,that makes major axis and minor axis equal

#Here we need a list of values that are simply x,x1 and x2
#colors specify a list of colors in pie chart
#In order to specify labels we use labels attribute

pt.pie([x,x1,x2],colors=['yellow','red','blue'],labels=['9 pointer','8 pointer','others'])

pt.legend(title='Description')  # to shown the labels as legends

pt.show()
```

The output will be as shown below –



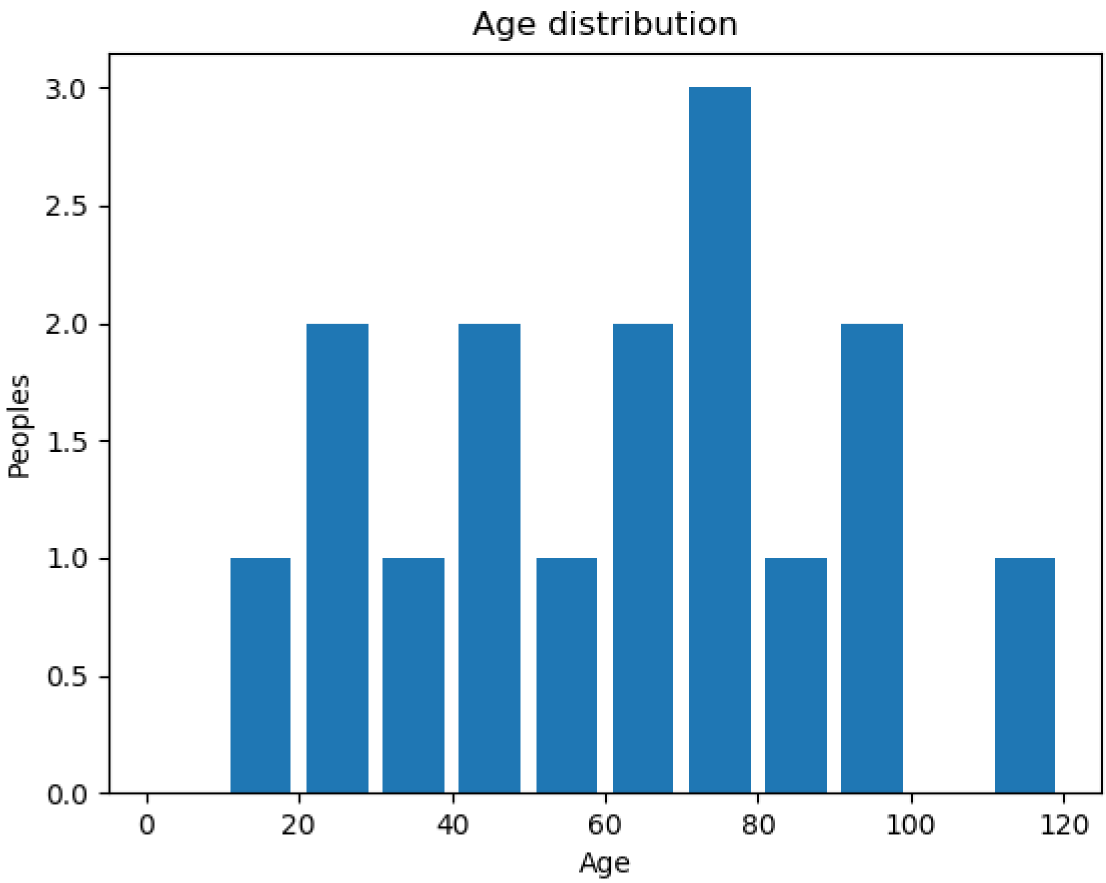
#4 Histogram

Now we will discuss about histogram.It is an estimate of the probability distribution of a continuous variable (quantitative variable) and was first introduced by **Karl Pearson**. It is a kind of bar graph.

Here i am taking an example of age distribution in a particular place.

```
import matplotlib.pyplot as pt
import pandas as pd

age_data = [20,30,54,66,70,10,45,65,77,99,120,130,29,40,80,75,90] #List of ages of peoples in a particular place
bins=[0,10,20,30,40,50,60,70,80,90,100,110,120] #The bins are usually specified as consecutive, non-overlapping intervals of a variable.

#hist() is used to draw histogram

pt.hist(age_data,bins,histtype='bar', rwidth=0.8)
pt.title('Age distribution')
pt.xlabel('Age')
pt.ylabel('People')
pt.show()
```

So the output of the above code is –

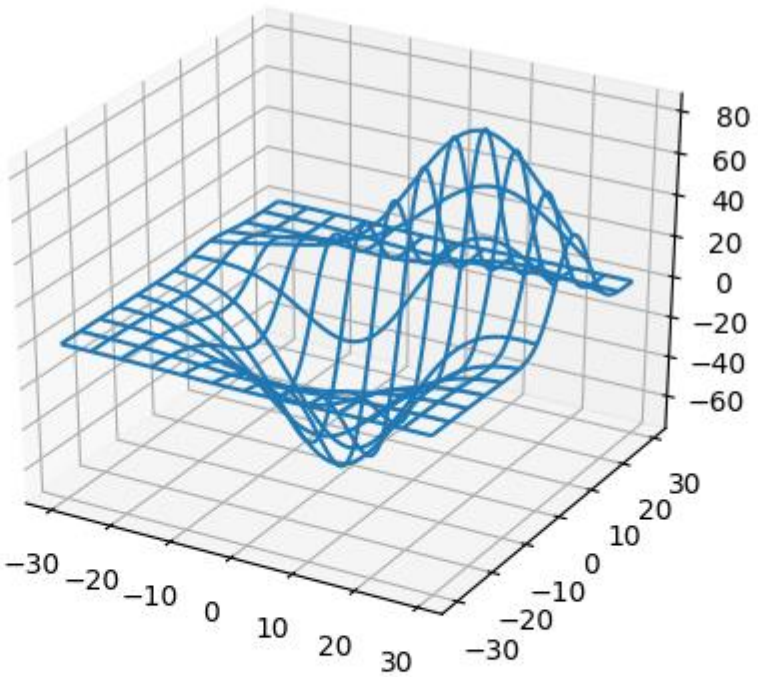


#5 3D Plane wire frame Graph Chart

A **wire-frame graph chart** is a visual presentation of a 3-dimensional (3D) or physical object used in 3D computer graphics.**plot_wireframe()** method is used to plot a wire frame.So, the code is as follows –

```
import matplotlib.pyplot as pt
from mpl_toolkits.mplot3d import axes3d # it allow us make to 3d axis

#figure() function create the figure
fig = pt.figure(figsize=(10,10)) # this sets the figure size here it is 10 by 10

ax = fig.add_subplot(111,projection='3d')  # projection='3d' is used to create Axes3D object

x,y,z = axes3d.get_test_data(0.05) #x,y,z are the axes

#rstride is row spacing and cstride is column spacing
#plot_wireframe() plot a wireframe
ax.plot_wireframe(x,y,z, rstride=10 ,cstride=10)

pt.show()
```

And the output of this code is



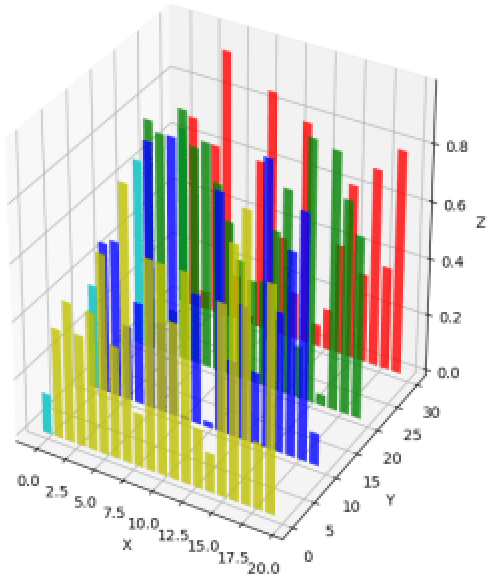
#6 3D Bar Plots

3D bar draws a three-dimensional bar graph.

```
import matplotlib.pyplot as pt
from mpl_toolkits.mplot3d import axes3d
import numpy as np

fig = pt.figure(figsize=(10,10))
ax = fig.add_subplot(111,projection='3d')

for c,z in zip(['r','g','b','y'],[30,20,10,0]):
   xs = np.arange(20) # x location of bars
   ys = np.random.rand(20) # y location of bars

# you can provide either a single color or an array.
# Here the first bar of each set will be cyan

cs = [c]*len(xs)
cs[0] = 'c'

# bar() function is used to draw bars
#zdir means Which direction to use as z (‘x’, ‘y’ or ‘z’) when plotting a 2D set.

 ax.bar(xs,ys,zs=z ,zdir ='y',color=cs, alpha=0.8)
```

```
# Set the labels
ax.set_xlabel('X')
ax.set_ylabel('Y')
ax.set_zlabel('Z')

pt.show()
```

The above code will give this output


0.8
0.6
0.4
0.2
0.0
Z
30
25
20
15
10
5
0
Y
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0
X