IntelliPaat

Data Science with Python

Data Visualization

Agenda

01 Basics of Data Visualization

02 Introduction to Matplotlib

03 Matplotlib Concepts

04 Types of Plots

05 Line and Area Plots

06 Bar Plot

07 Scatter Plot

08 Histogram

09 Box and Violin Plots

10 Image Plot

11 Quiver and Stream Plots

12 Pie Chart

Basics of Data Visualization

Basics of Data Visualization

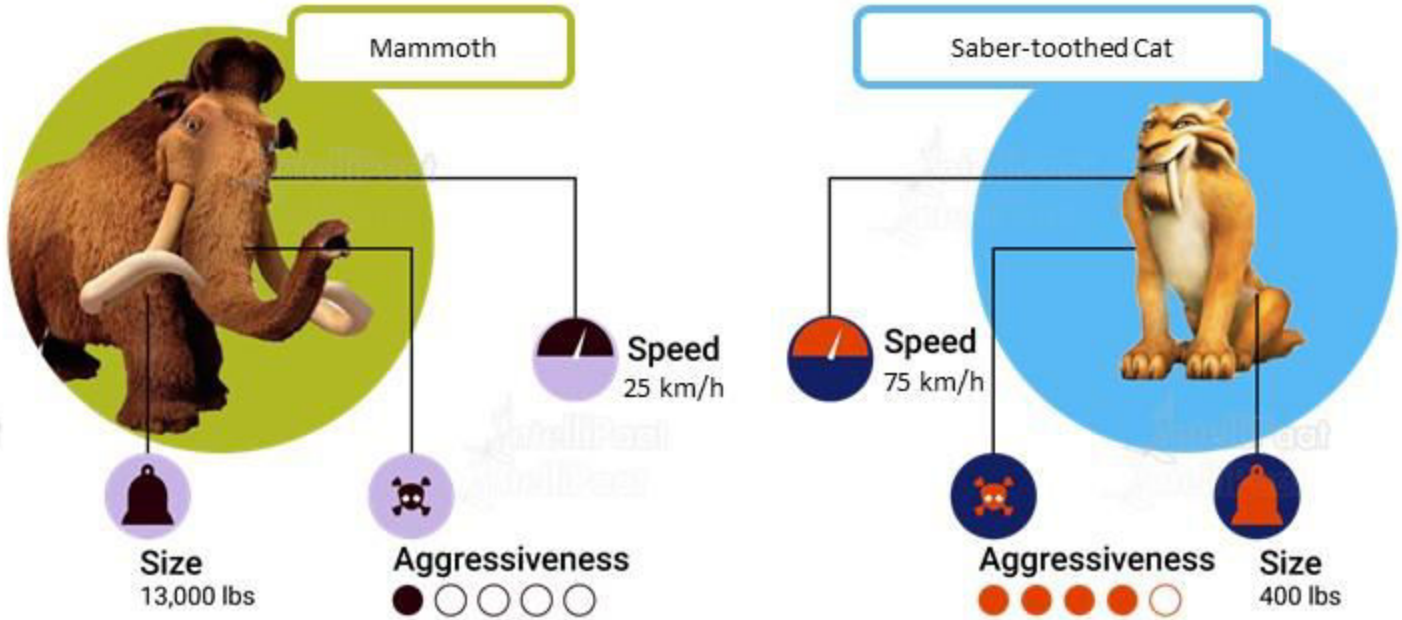
Data visualization is the graphical/pictorial representation of information and data

	S.No	model	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
0	1	Mazda RX4	21.0	6	160.0	110	3.90	2.620	16.460000	0	1	4	4
1	2	Mazda RX4 Wag	21.0	6	160.0	110	3.90	2.875	17.020000	0	1	4	4
2	3	Datsun 710	22.8	4	108.0	93	3.85	2.320	18.610000	1	1	4	1
3	4	Hornet 4 Drive	21.4	6	258.0	110	3.08	3.215	19.440000	1	0	3	1
4	5	Hornet Sportabout	18.7	8	360.0	175	3.15	3.440	17.020000	0	0	3	2
5	6	Valiant	18.1	6	225.0	105	2.76	3.460	17.674828	1	0	3	1
6	7	Duster 360	14.3	8	360.0	245	3.21	3.570	15.840000	0	0	3	4
7	8	Merc 240D	24.4	4	146.7	62	3.69	3.190	20.000000	1	0	4	2
8	9	Merc 230	22.8	4	140.8	95	3.92	3.150	22.900000	1	0	4	2
9	10	Merc 280	19.2	6	167.6	123	3.92	3.440	18.300000	1	0	4	4
10	11	Merc 280C	17.8	6	167.6	123	3.92	3.440	18.900000	1	0	4	4
11	12	Merc 450SE	16.4	8	275.8	180	3.07	4.070	17.400000	0	0	3	3
12	13	Merc 450SL	17.3	8	275.8	180	3.07	3.730	17.600000	0	0	3	3
13	14	Merc 450SLC	15.2	8	275.8	180	3.07	3.780	18.000000	0	0	3	3
14	15	Cadillac Fleetwood	10.4	8	472.0	205	2.93	5.250	17.980000	0	0	3	4
15	16	Lincoln Continental	10.4	8	460.0	215	3.00	5.424	17.820000	0	0	3	4
16	17	Chrysler Imperial	14.7	8	440.0	230	3.23	5.345	17.420000	0	0	3	4
17	18	Fiat 128	32.4	4	78.7	66	4.08	2.200	17.674828	1	1	4	1
18	19	Honda Civic	30.4	4	75.7	52	4.93	1.615	18.520000	1	1	4	2
19	20	Toyota Corolla	33.9	4	71.1	65	4.22	1.835	19.900000	1	1	4	1
20	21	Toyota Corona	21.5	4	120.1	97	3.70	2.465	20.010000	1	0	3	1
21	22	Dodge Challenger	15.5	8	318.0	150	2.76	3.520	16.870000	0	0	3	2
22	23	AMC Javelin	15.2	8	304.0	150	3.15	3.435	17.300000	0	0	3	2
23	24	Camaro Z28	13.3	8	350.0	245	3.73	3.840	15.410000	0	0	3	4
24	25	Pontiac Firebird	19.2	8	400.0	175	3.08	3.845	17.050000	0	0	3	2

Basics of Data Visualization

A Simple Data Visualization Example

Name of Animal	Speed	Aggressiveness	Size
Mammoth	25 km/h	Low	13,000 lbs
Saber-toothed Cat	75 km/h	High	400 lbs



Basics of Data Visualization

Why should we visualize data?

01 We can view changes over time seamlessly using a visual rather than plain data

02 We can easily discover correlations between two or more variables in a visual

03 Using proper visualizing, we can simplify complex information into user-friendly formats

04 Generally, we can tell a better story with a bunch of pictures over time

Basics of Data Visualization

Why should we visualize data?

To understand more about this, let us talk about an English Statistician, Francis Anscombe. He developed **Anscombe's quartet** and showed the importance of visualizing data

Basics of Data Visualization

But how exactly did he show that?

His quartet contains 4 datasets with 11 (x,y) pairs. When we look at the descriptive statistics of these datasets, they are very similar. But when we **convert them into graphs**, their distributions are completely **DIFFERENT**

Seems interesting? Let us dig more into this

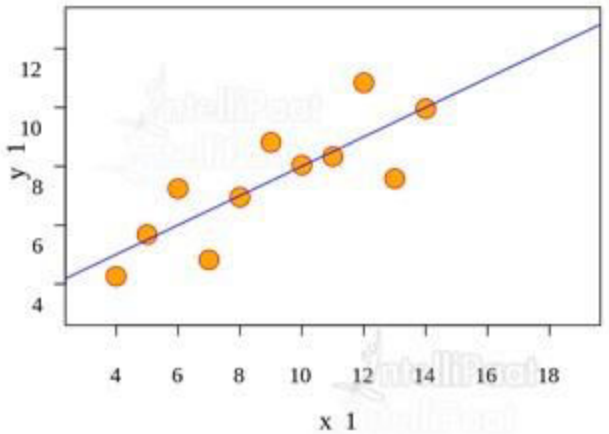
Basics of Data Visualization

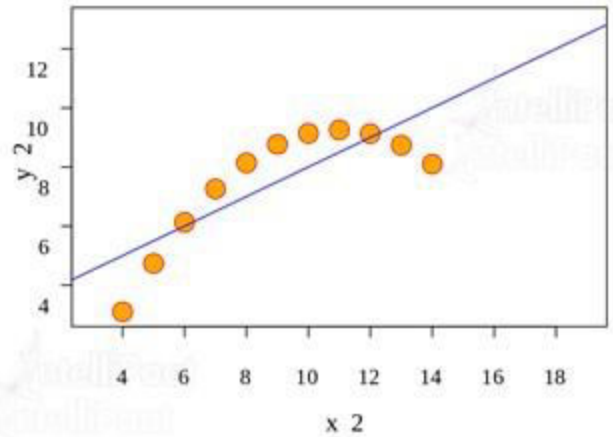
Anscombe's Dataset

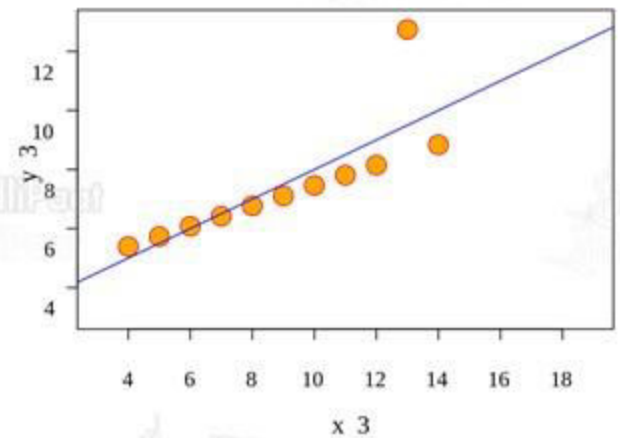
I		II		III		IV	
x	y	x	y	x	y	x	y
10	8,04	10	9,14	10	7,46	8	6,58
8	6,95	8	8,14	8	6,77	8	5,76
13	7,58	13	8,74	13	12,74	8	7,71
9	8,81	9	8,77	9	7,11	8	8,84
11	8,33	11	9,26	11	7,81	8	8,47
14	9,96	14	8,1	14	8,84	8	7,04
6	7,24	6	6,13	6	6,08	8	5,25
4	4,26	4	3,1	4	5,39	19	12,5
12	10,84	12	9,13	12	8,15	8	5,56
7	4,82	7	7,26	7	6,42	8	7,91
5	5,68	5	4,74	5	5,73	8	6,89
SUM	99,00	82,51	99,00	82,51	99,00	82,50	99,00	82,51
AVG	9,00	7,50	9,00	7,50	9,00	7,50	9,00	7,50
STDEV	3,32	2,03	3,32	2,03	3,32	2,03	3,32	2,03

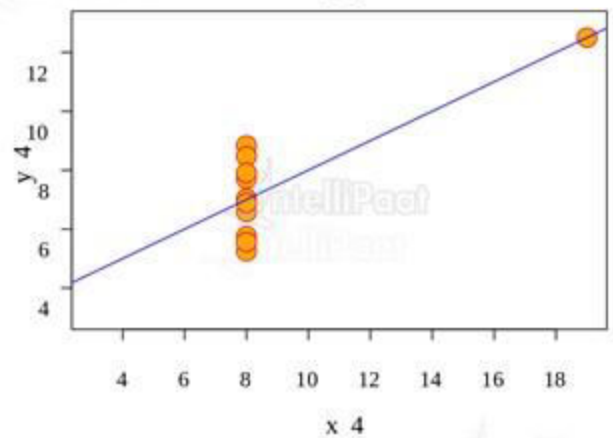
Basics of Data Visualization

Anscombe's Dataset When Graphed



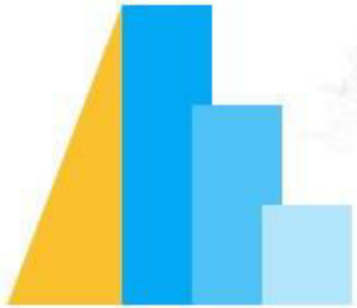






Basics of Data Visualization

These are the popular Python libraries used for data visualization



Seaborn

Introduction to Matplotlib

Introduction to Matplotlib

Matplotlib is the most popular Python library for data visualization used to create 2D plots and graphs with the help of Python scripts that is printing quality

Introduction to Matplotlib

How does Matplotlib help in Data Science?

1. By providing the **pyplot** module that makes it work like MATLAB
2. By making simple, **pre-defined functions** available for visualization
3. By supporting a **variety of graphs and plots**
4. By providing an **object-oriented API**
5. By easily **integrating with Pandas and NumPy**

Matplotlib Concepts

Matplotlib Concepts

There are a few components and functions that we have to understand before starting off with Matplotlib. Let's look at the important Matplotlib functions

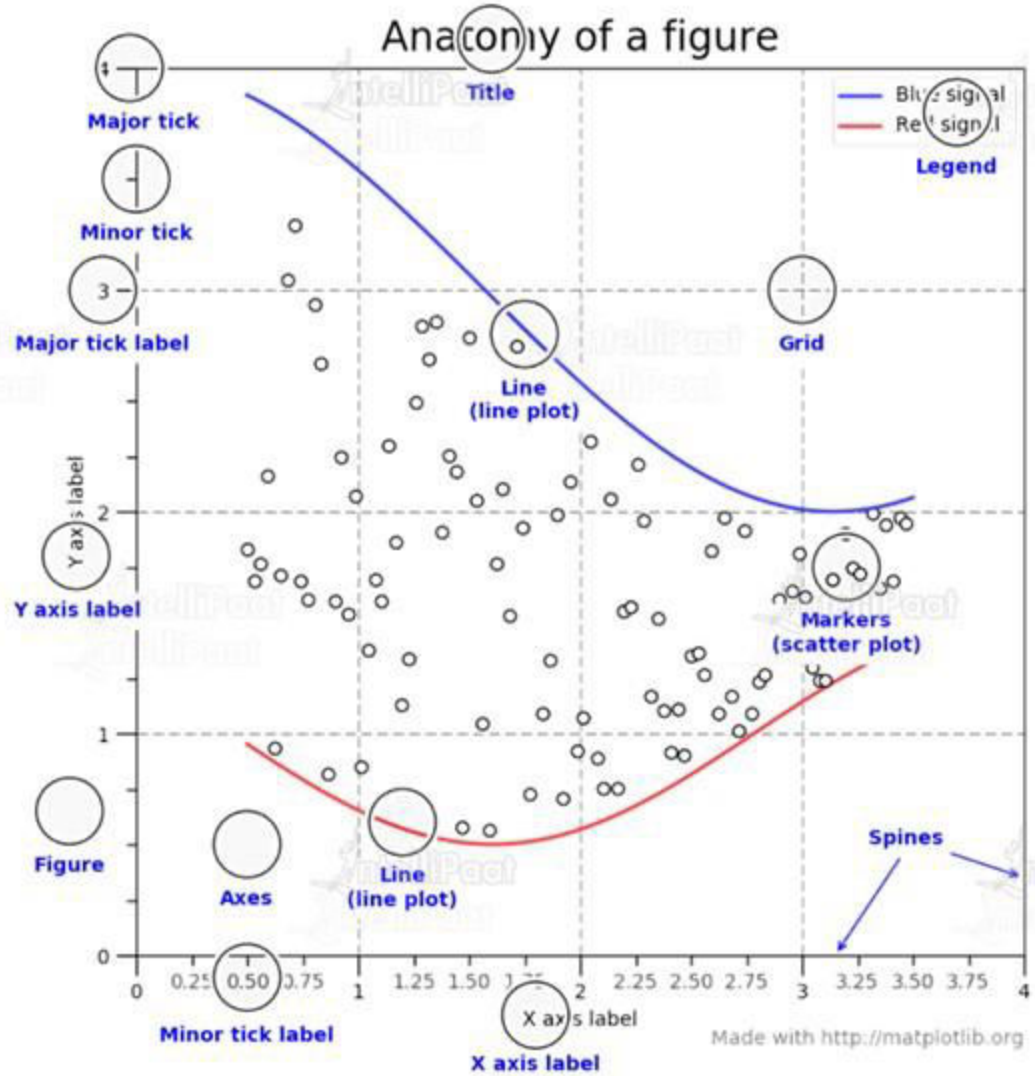
Function	Description
.plot()	Used to plot an array data onto a graph
.show()	Shows the plotted graph as an output
.grid()	Sets the visibility of grids as True or False
.set_title()	Gives a title label to the graph

Matplotlib Concepts

Anatomy of a Matplotlib Figure

Axes class: Axes is the data space and can be identified in the figure. A figure can contain many axes, but a particular axes object can be in only one figure

An axes object has two axis objects, an x-label (**set_xlabel()**) and a y-label (**set_ylabel()**)



Matplotlib Concepts

In the plot() function, we can mention the color, marker, and style of the line

Color Codes

Character	Color
b	Blue
g	Green
r	Red
c	Cyan
m	Magenta
y	Yellow
w	White
k	Black

Marker Codes

Character	Description
.	Point marker
o	Circle marker
x	X marker
D	Diamond marker
H	Hexagon marker
s	Square marker
+	Plus marker

Matplotlib Concepts

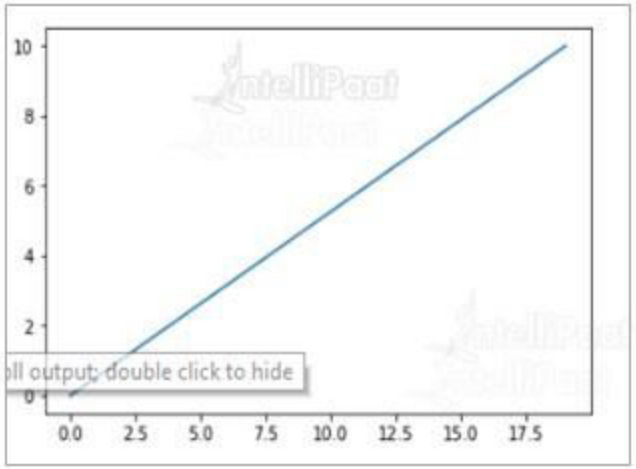
Line Styles

Character	Description
-	Solid line
---	Dotted line
-.	Dash-dot line
:	Dotted line
H	Hexagon marker

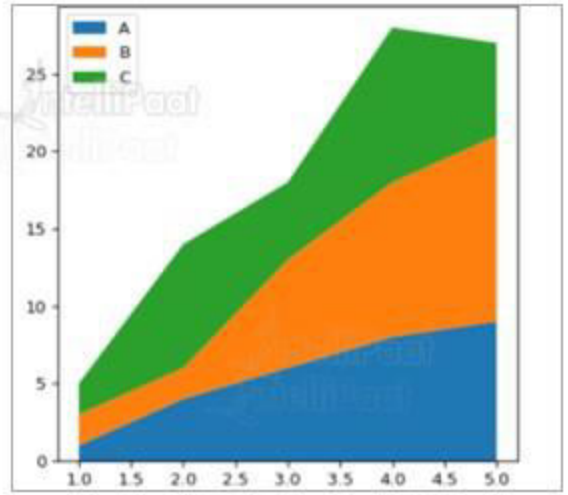
Hands-on: Plots, Grids, Markers, Colors, and Fonts

Types of Plots

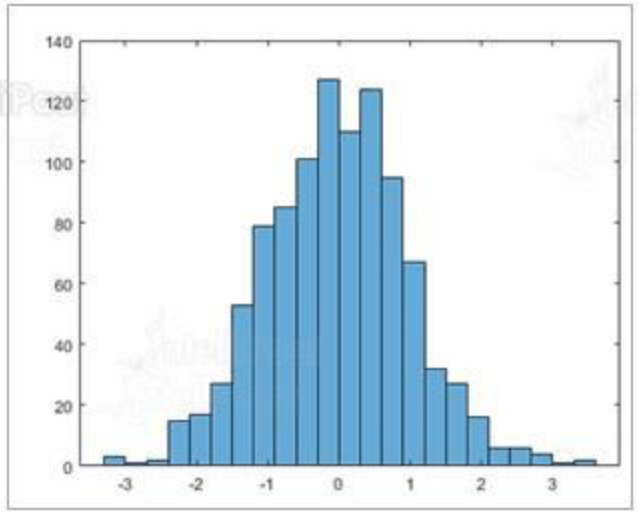
Types of Plots



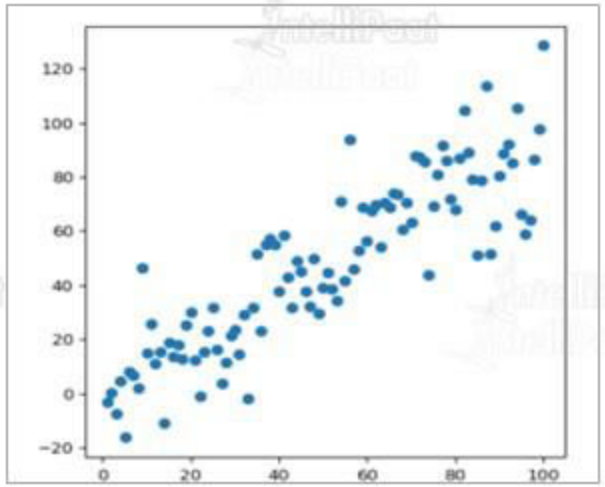
Line Plot



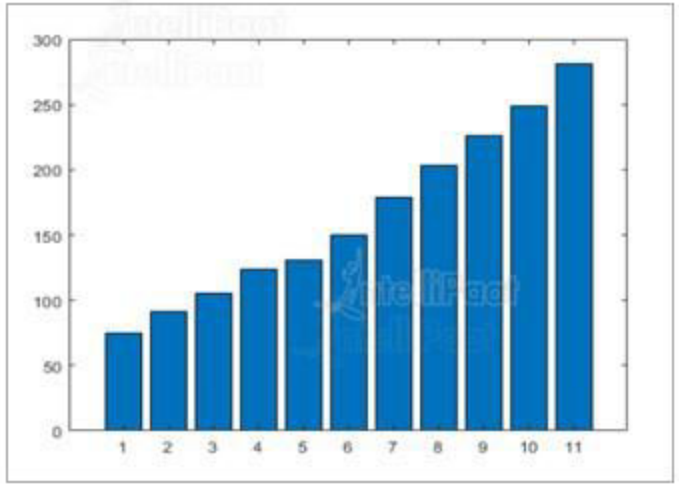
Area Plot



Histogram

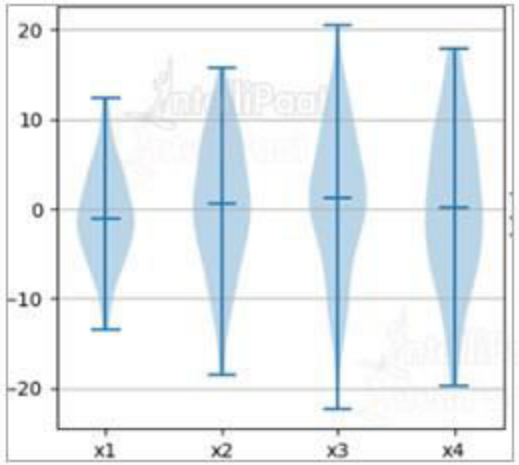


Scatter Plot

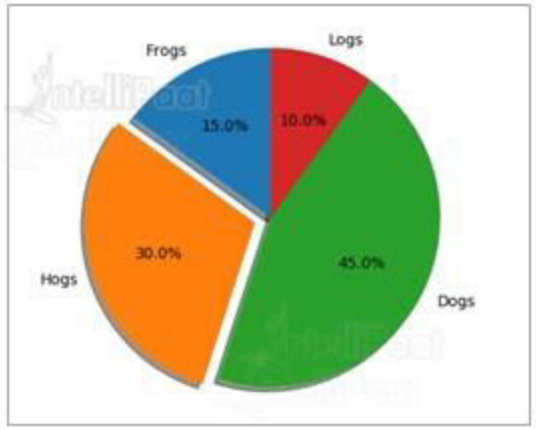


Bar Plot

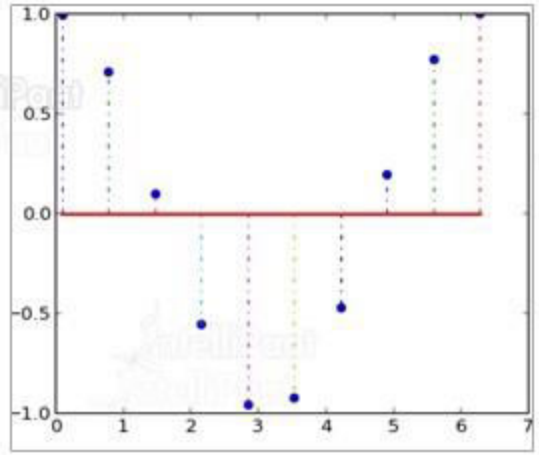
Types of Plots



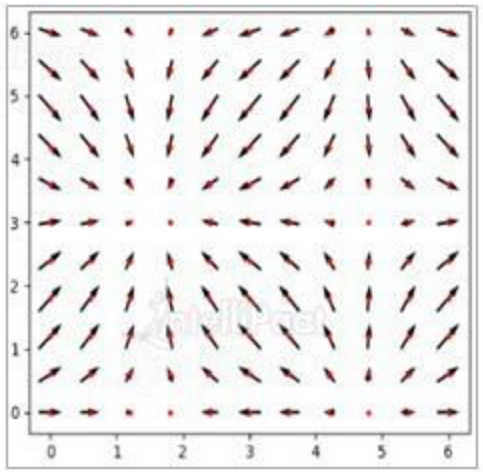
Violin Plot



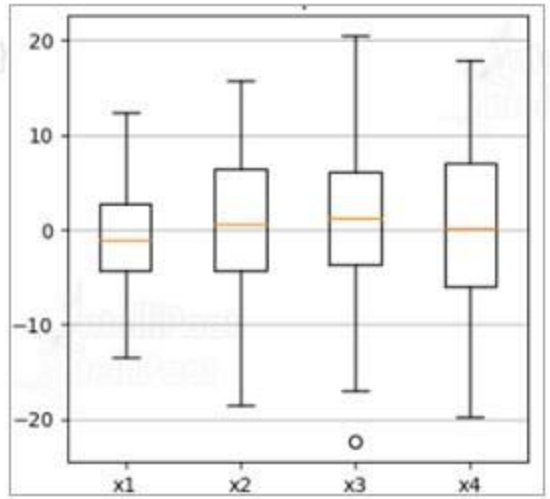
Pie Chart



Stream Plot



Quiver Plot



Box Plot

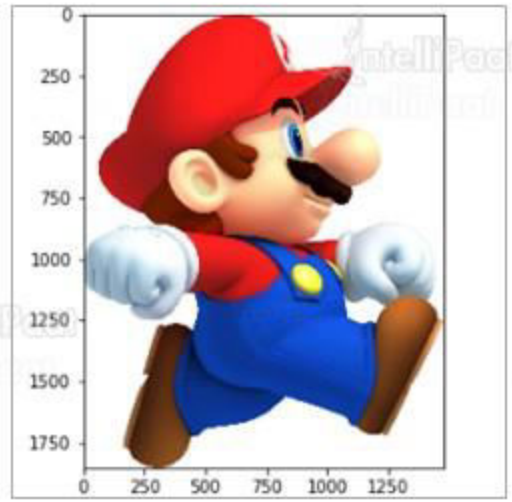


Image Plot

Types of Plots

A few takes on when to choose a specific type of plot

Comparing Values	Distribution of Data	Comparing Continuous Data
1. Bar plot	1. Scatter plot	1. Histogram
2. Line plot	2. Box plot	2. Box plot
3. Pie chart	3. Violin plot	3. Area plot

Line and Area Plots

Line and Area Plots

Line Plot

It is best to use a line plot when comparing fewer than 25 numbers. It is a quick and simple way to organize data

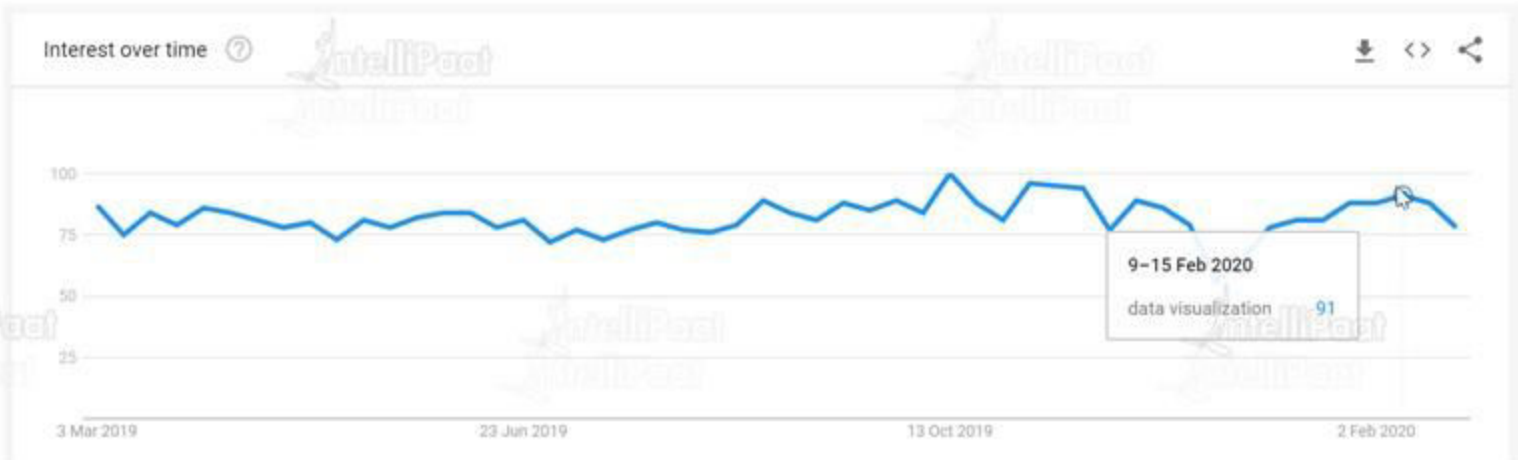
Area Plot

We use an area plot to represent cumulative totals using numbers or percentages over time

Line and Area Plots

When should we choose a line plot?

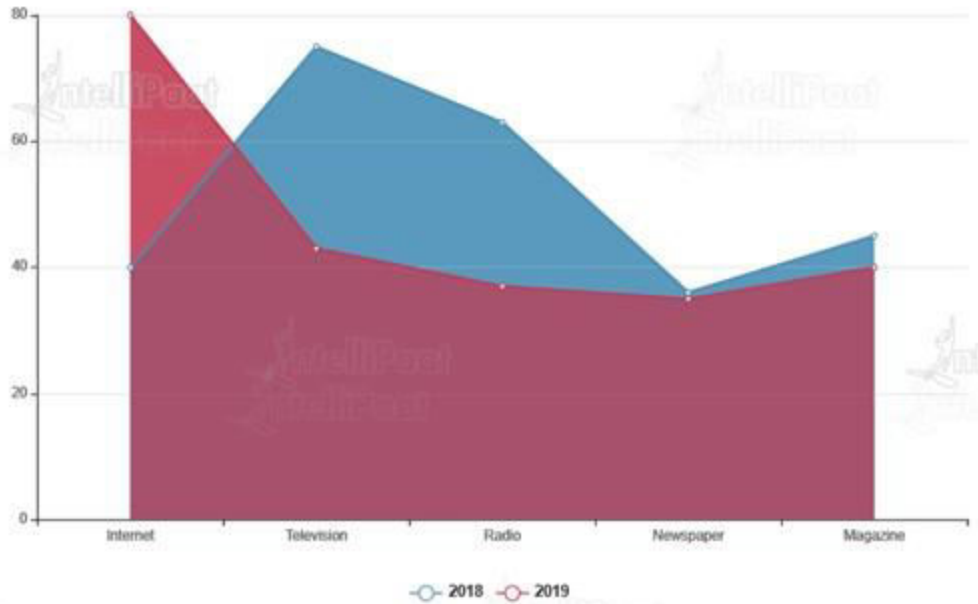
Line plots can be used when we are trying to compare trends over a period of time, e.g., searching a word on Google over time



Line and Area Plots

When should we choose an area plot?

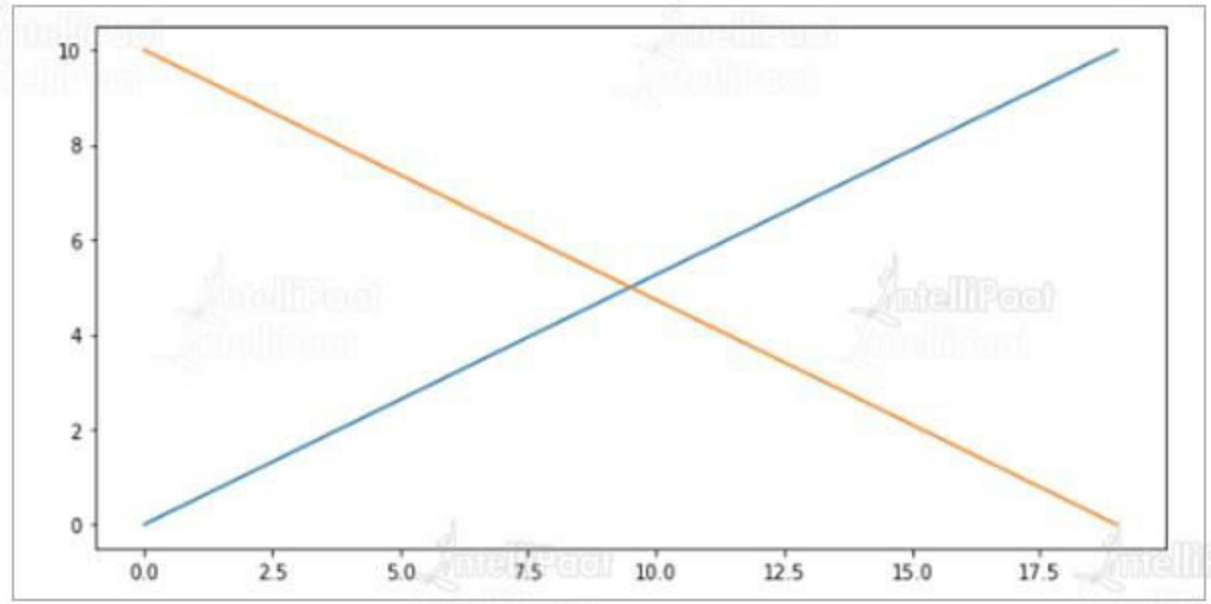
Whenever we have to show multiple variables across time and also check out the data space they take up, we use an area plot



Line and Area Plots

Line Plot

```
In [33]: #import libraries
         import numpy as np
         import matplotlib.pyplot as plt
         %matplotlib inline

         #preparing data
         a = np.linspace(0, 10, 20)
         b = np.linspace(10, 0, 20)
         #Adding figure
         fig=plt.figure(figsize=(10,5))
         #Adding axes
         ax1 = plt.subplot()
         #simple line plot of both a and b
         ax1.plot(a)
         ax1.plot(b)
         #show the plot
         plt.show()
```

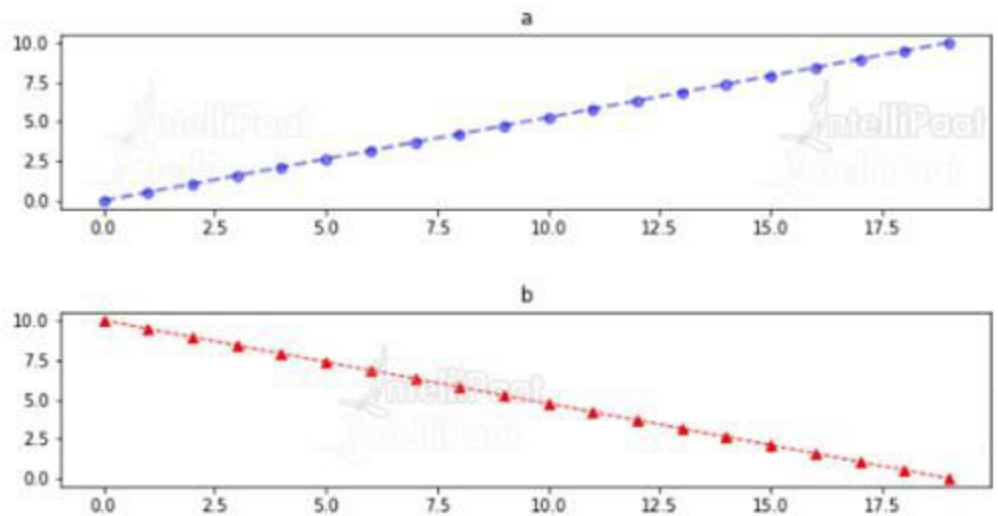


Line and Area Plots

Sub-plotting

Use sub-plotting while comparing two or more plots

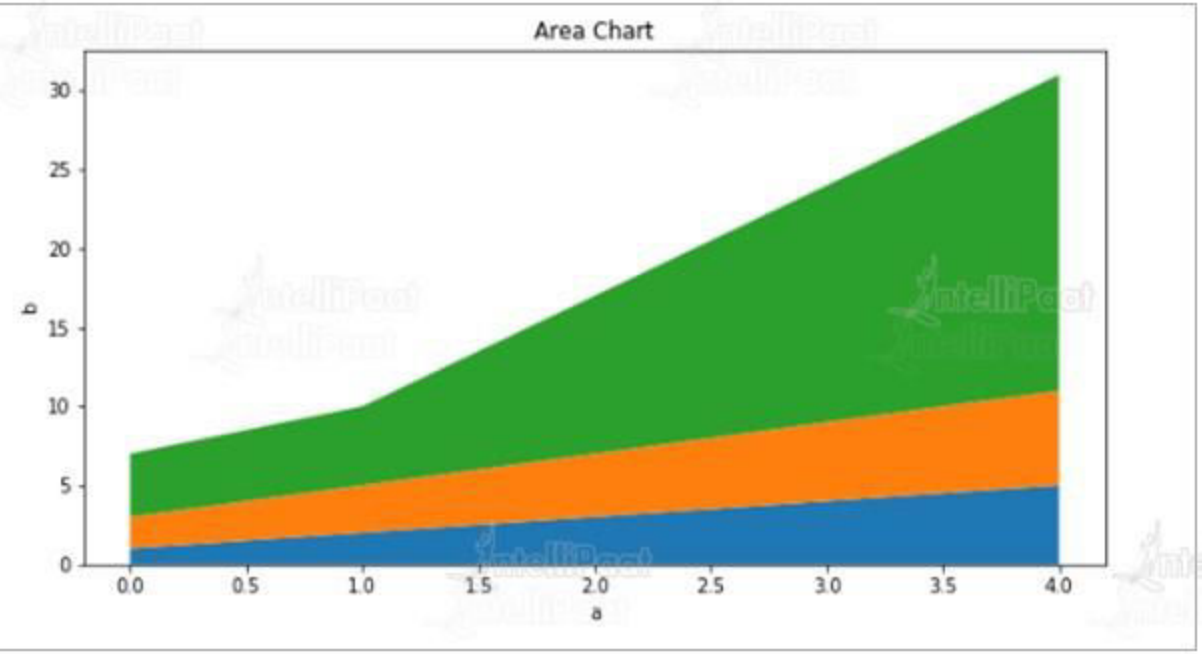
```
In [34]: #importing libraries
         import numpy as np
         import matplotlib.pyplot as plt
         %matplotlib inline
         #preparing data
         a = np.linspace(0, 10, 20)
         b = np.linspace(10, 0, 20)
         #Add figure
         fig=plt.figure(figsize=(10,5))
         #Sub-plotting
         ax1 = plt.subplot(211)    #2 rows 1 column 1st position
         ax2 = plt.subplot(212)    #2 rows 1 column 2nd position
         #Customization- Line Width, Line Style,Line Color, Line Opacity and Marker Options
         ax1.plot(a,linewidth=2.0,linestyle='--',color='b',alpha=0.5,marker='o')
         ax2.plot(b,linewidth=1.0,linestyle='--',color='r',alpha=1,marker='^')
         #setting title of first subplot
         ax1.set(title='a')
         ax2.set(title='b')
         #Adding Space between subplots
         plt.subplots_adjust(left=None, bottom=None, right=None, top=None, wspace=None, hspace=0.6)
         #showing plot
         plt.show()
```



Line and Area Plots

Area Plot

```
In [79]: #import libraries
         import matplotlib.pyplot as plt
         %matplotlib inline
         #prepare data
         a= range(0,5)
         b= [[1,2,3,4,5],[2,3,4,5,6],[4,5,10,15,20]]
         #Add figure
         fig=plt.figure(figsize=(10,5))
         #add axes
         ax1 = plt.subplot()
         #plot the area plot
         ax1.stackplot(a,b)
         #Customization-title
         plt.title('Area Chart')
         #Customization-x-axis label, y-axis label
         plt.xlabel('a')
         plt.ylabel('b')
         #save the plot
         plt.savefig('AreaPlot.png')
         #show
         plt.show()
```



Bar Plot

Bar Plot

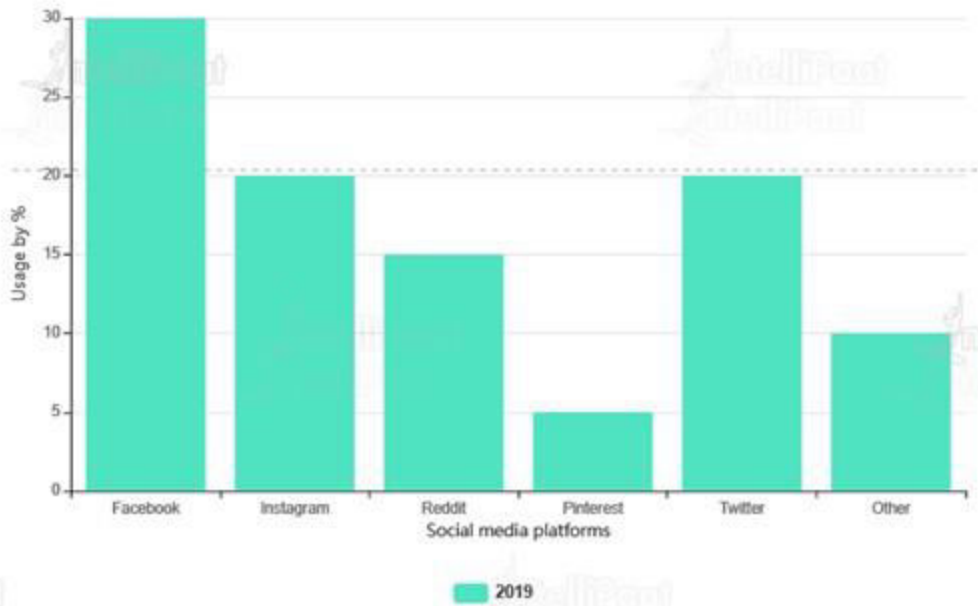
Bar Plot

A bar plot or bar graph is a chart/graph that presents categorical data with rectangular bars of heights or lengths proportional to the values that they represent. The bars can be plotted vertically or horizontally

Bar Plot

When should we choose a bar plot?

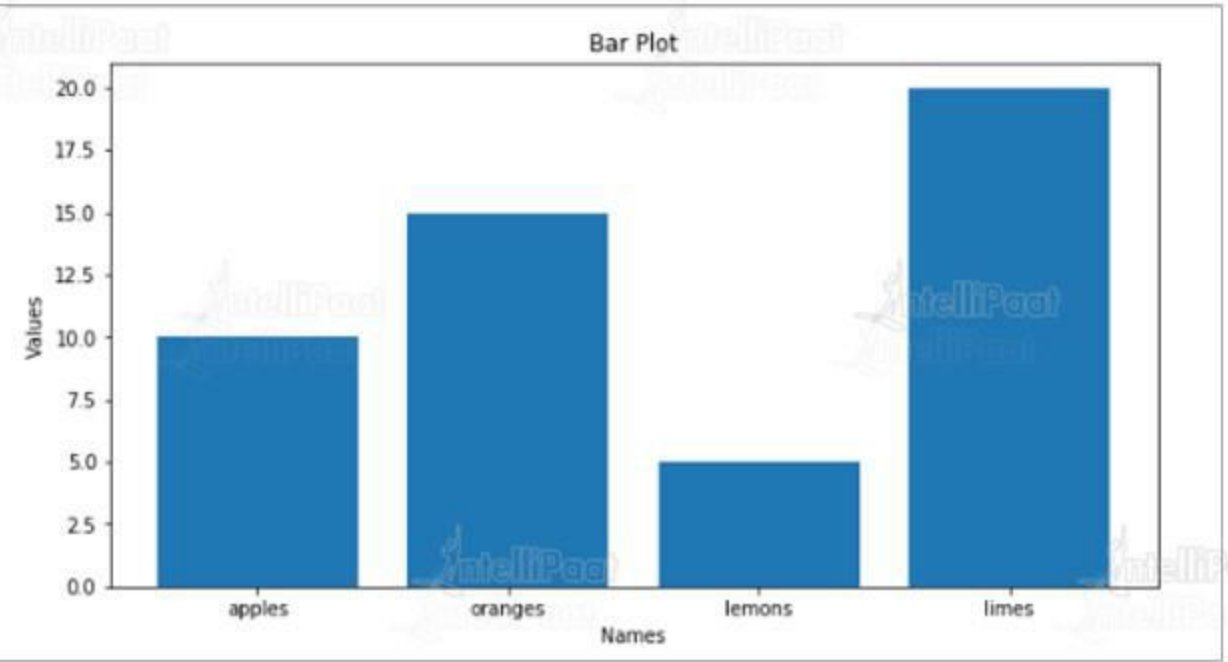
When comparing various categories on the same variable, we use a bar plot, e.g., social media usage by percentage over a year



Bar Plot

Vertical Bar Plot

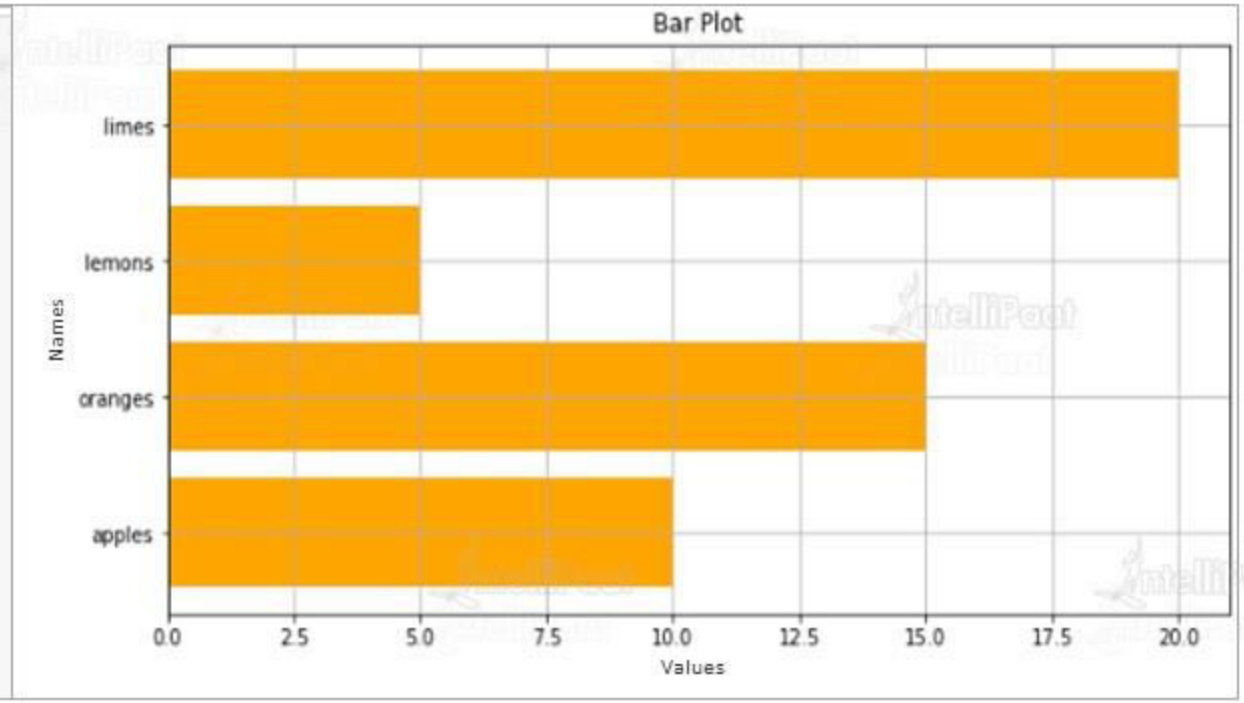
In [37]:
```
#import library
import matplotlib.pyplot as plt
%matplotlib inline
#prepare data
data = {'apples': 10, 'oranges': 15, 'lemons': 5, 'limes': 20}
names = list(data.keys())
values = list(data.values())
#Add figure
fig=plt.figure(figsize=(10,5))
#Sub-plotting
ax1 = plt.subplot()
#plot
ax1.bar(names, values)
#Customization-Title
plt.title('Bar Plot')
#Customization-x-axis label, y-axis label
plt.xlabel('Names')
plt.ylabel('Values')
#showing plot
plt.show()
```



Bar Plot

Horizontal Bar Plot

```
In [41]: #import libraries
         import matplotlib.pyplot as plt
         %matplotlib inline
         #prepare the data
         data = {'apples': 10, 'oranges': 15, 'lemons': 5, 'limes': 20}
         names = list(data.keys())
         values = list(data.values())
         #Add figure
         fig=plt.figure(figsize=(10,5))
         #adding axes
         ax1 = plt.subplot()
         #Customization-alignment, color
         ax1.barh(names, values, align='center', color='orange')
         #Customization-Title
         plt.title('Bar Plot')
         #Customization-x-axis label, y-axis label
         plt.xlabel('Names')
         plt.ylabel('Values')
         #customization-add grid
         plt.grid(True)
         #Save the plot
         plt.savefig('HorizontalBarPlot.png')
         #show plot
         plt.show()
```

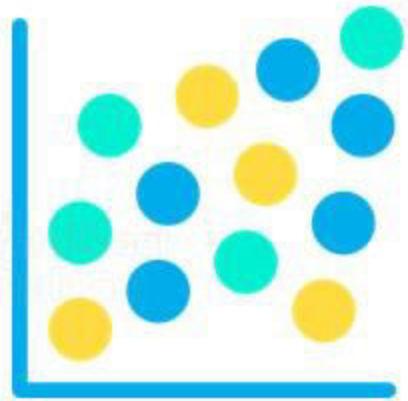


Scatter Plot

Scatter Plot

Scatter Plot

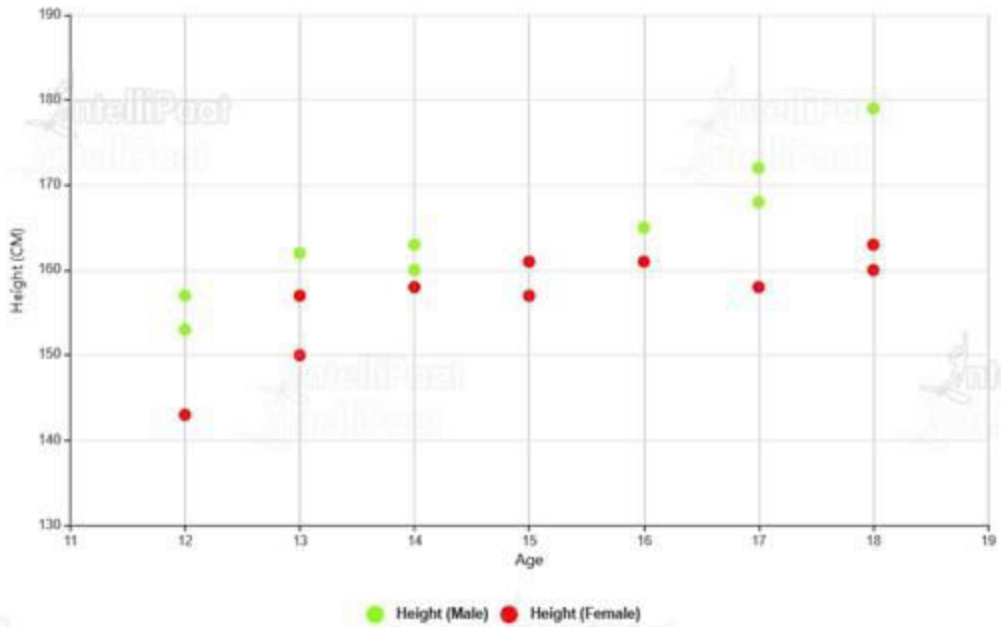
Scatter plots are used to plot data points on an area defined by horizontal and vertical axes in the attempt to show how much one variable is affected by another. It helps in visualizing the correlation



Scatter Plot

When should we choose a scatter plot?

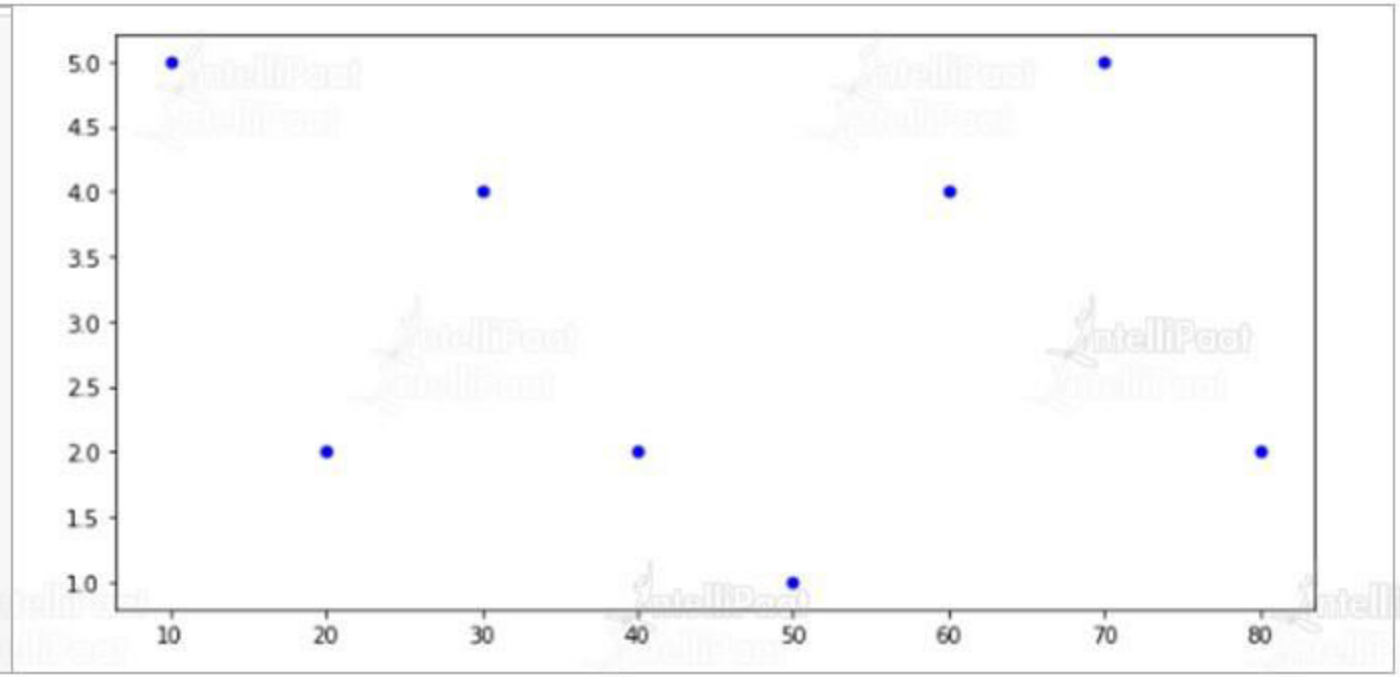
It is effective when we compare one variable across multiple subjects, e.g., ‘Male’ and ‘Female’ heights in correspondence with their ‘Age’



Scatter Plot

Scatter Plot

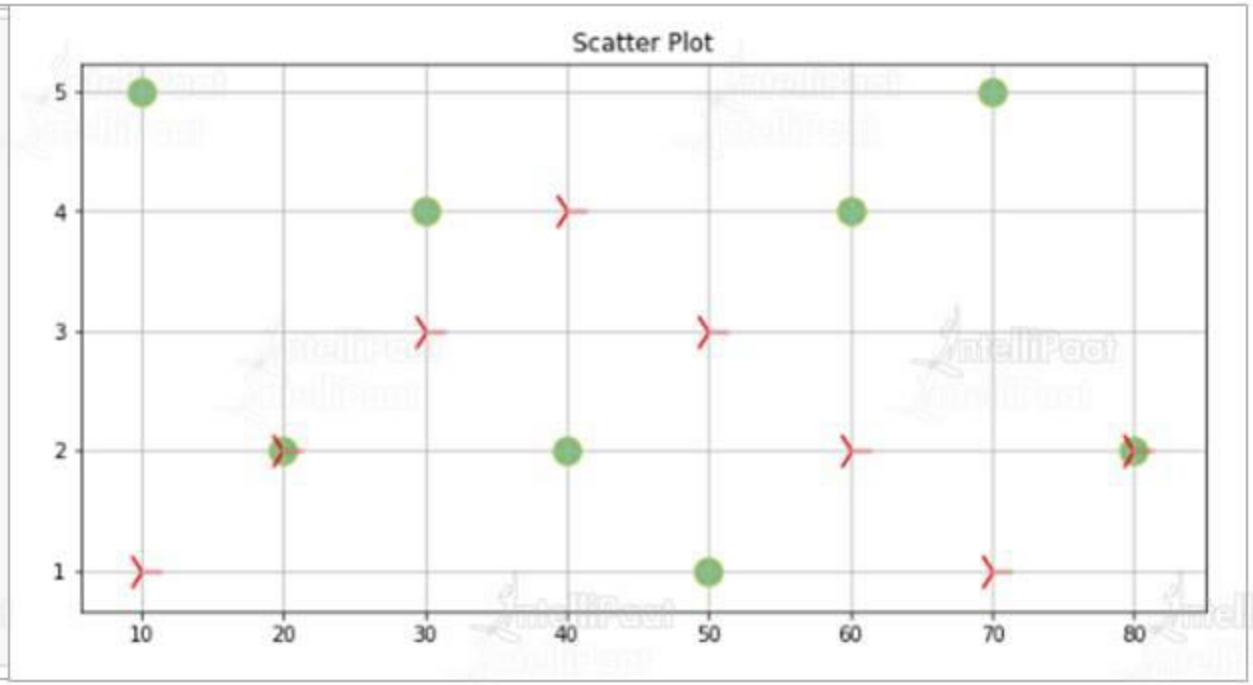
```
In [41]: #import libraries
         import matplotlib.pyplot as plt
         %matplotlib inline
         #prepare the data
         data = {'apples': 10, 'oranges': 15, 'lemons': 5, 'limes': 20}
         names = list(data.keys())
         values = list(data.values())
         #Add figure
         fig=plt.figure(figsize=(10,5))
         #adding axes
         ax1 = plt.subplot()
         #Customization-alignment, color
         ax1.barh(names, values, align='center', color='orange')
         #Customization-Title
         plt.title('Bar Plot')
         #Customization-x-axis label, y-axis label
         plt.xlabel('Names')
         plt.ylabel('Values')
         #customization-add grid
         plt.grid(True)
         #Save the plot
         plt.savefig('HorizontalBarPlot.png')
         #show plot
         plt.show()
```



Scatter Plot

Customized Scatter Plot

```
In [47]: #importing library
         import matplotlib.pyplot as plt
         %matplotlib inline
         #prepare data
         a = [10,20,30,40,50,60,70,80]
         b = [5,2,4,2,1,4,5,2]
         x = [1,2,3,4,3,2,1,2]
         #Add figure
         fig=plt.figure(figsize=(10,5))
         #add axes
         ax1 = plt.subplot()
         #customization-color,size,edgecolors,marker,alpha
         ax1.scatter(a, b, c='g', s=200, edgecolors='y', marker='o', alpha=0.5)
         ax1.scatter(a, x, c='r', s=400, edgecolors='b', marker='4',alpha=1)
         #Customization-Title
         plt.title('Scatter Plot')
         #customization-add grid
         plt.grid(True)
         #Save the plot
         plt.savefig('ScatterPlot.png')
         #show plot
         plt.show()
```



Histogram

Histogram

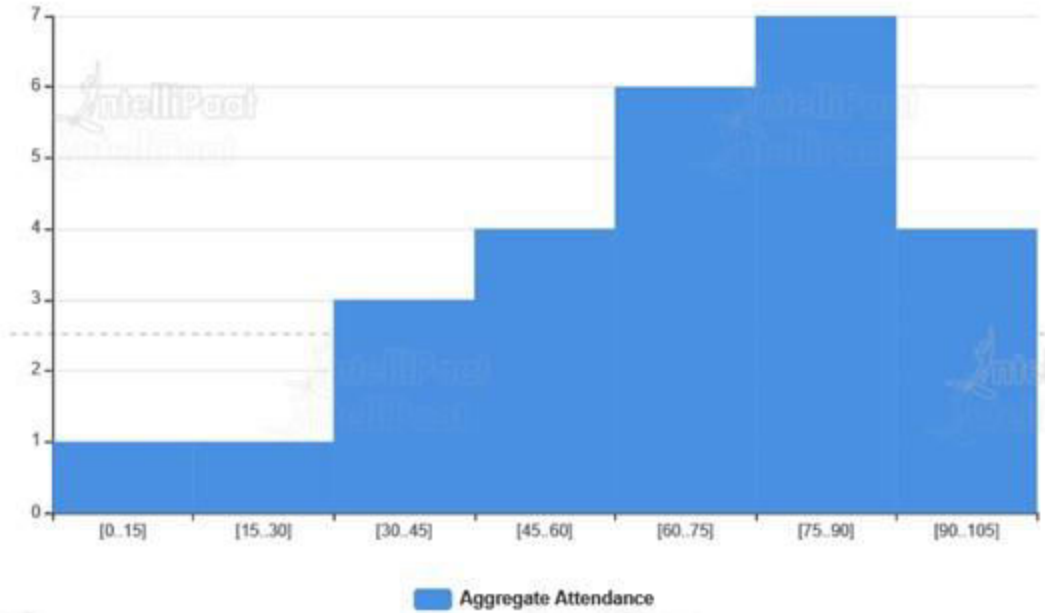
Histogram

Histogram is a plot used to display the frequency of a continuous or discrete variable. The final outcome would be a figure of data distribution

Histogram

When should we choose a histogram?

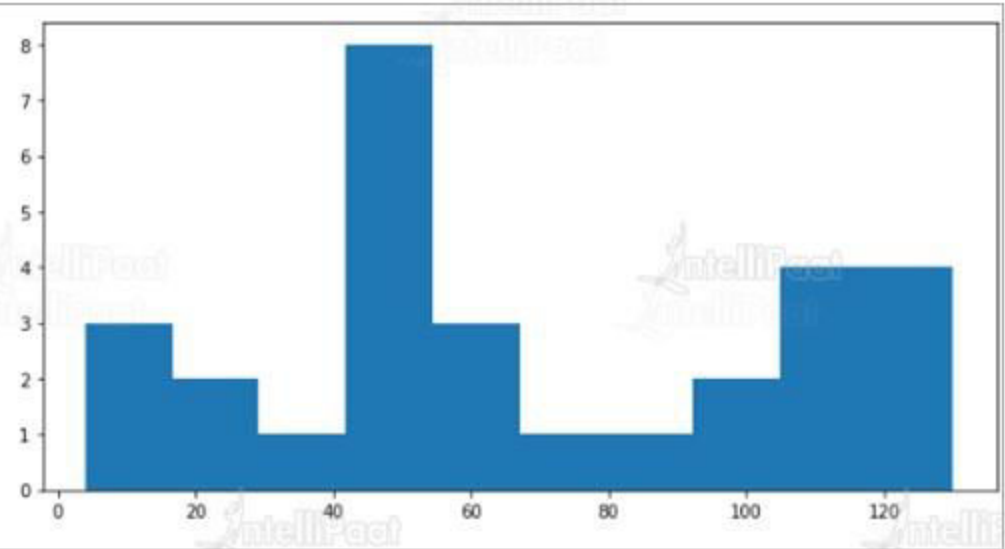
When we are trying to show the distribution of data for a variable, we can go ahead with a histogram, e.g., an aggregate attendance data of a student



Histogram

Histogram

```
#import libraries
import matplotlib.pyplot as plt
%matplotlib inline
#prepare data
number = [12,55,11,62,45,21,22,34,42,42,4,99,102,110,120,121,122,130,111,115,112,80,75,65,54,44,43,42,48]
#Add figure
fig=plt.figure(figsize=(10,5))
#add axes
ax1 = plt.subplot()
#plot and customize
ax1.hist(number, bins=10)
#show plot
plt.show()
```



Box and Violin Plots

Box and Violin Plots

Box Plot

A box plot is very helpful in showing the summary of a dataset in an efficient way; also, it helps us in doing outlier analysis

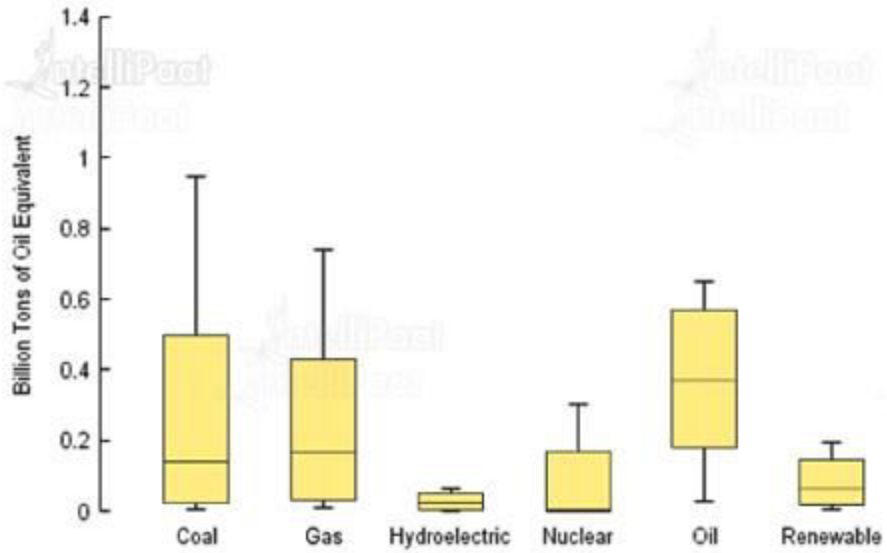
Violin Plot

A violin plot visualizes the distribution of a numeric variable for one or several groups. It is adapted when the amount of data is huge and showing individual observations is impossible

Box and Violin Plots

When should we choose a box plot?

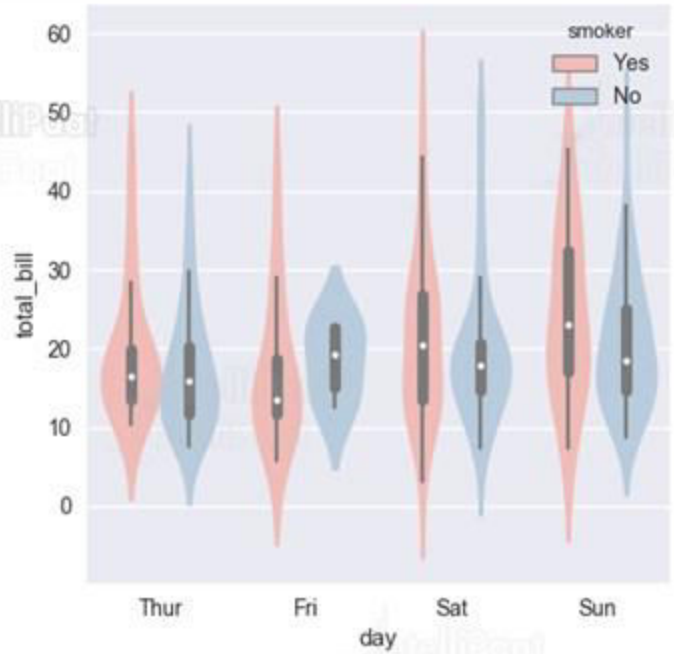
A box plot is a way of summarizing a set of data measured on an interval scale



Box and Violin Plots

When should we choose a violin plot?

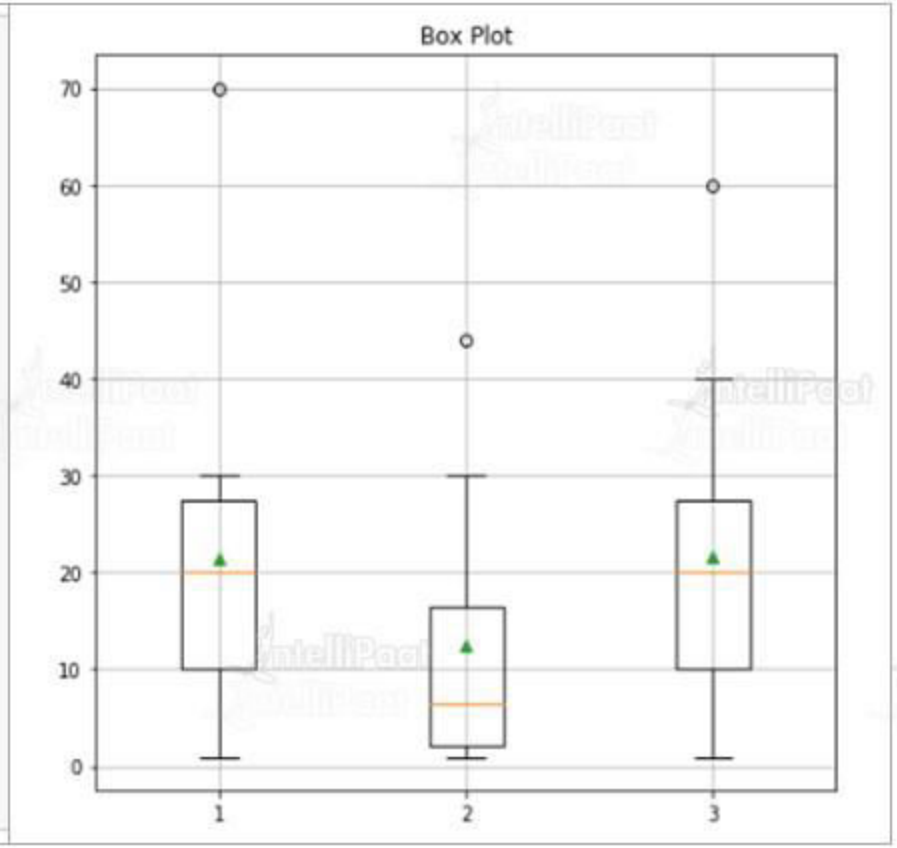
Same as a box plot, but a violin plot is more attractive and can show variations in a better way



Box and Violin Plots

Box Plot

```
In [52]: #import libraries
         import matplotlib.pyplot as plt
         %matplotlib inline
         #data preparation
         total = [20,4,1,30,20,10,20,70,30,10]
         orders = [10,3,1,15,17,2,30,44,2,1]
         discount = [30,20,10,5,20,10,60,20,40,1]
         data = list([total, orders, discount ])
         #Add figure
         fig=plt.figure(figsize=(7,7))
         #add axes
         ax1 = plt.subplot()
         #plot data
         ax1.boxplot(data, showmeans=True)
         #add title
         plt.title('Box Plot')
         #customization-add grid
         plt.grid(True)
         #save the plot
         plt.savefig('BoxPlot.png')
         #show plot
         plt.show()
```



Box and Violin Plots

Violin Plot

```
In [54]: #import libraries
         import matplotlib.pyplot as plt
         %matplotlib inline
         #prepare data
         total = [20,4,1,30,20,10,20,70,30,10]
         orders = [10,3,1,15,17,2,30,44,2,1]
         discount = [30,20,10,5,20,10,60,20,40,1]
         data = list([total, orders, discount ])
         #Add figure
         fig=plt.figure(figsize=(7,7))
         #add axes
         ax1 = plt.subplot(121)
         ax2 = plt.subplot(122)
         ax1.violinplot(data, showmeans=True, showmedians=False)
         ax2.violinplot(data, showmeans=False, showmedians=True)
         #add axes title
         ax1.set_title('Violin Plot-Showing means')
         ax2.set_title('Violin Plot-Showing medians')
         #customization-add grid
         ax1.grid(True)
         ax2.grid(True)
         #save the plot
         plt.savefig('ViolinPlot.png')
         #show the plot
         plt.show()
```

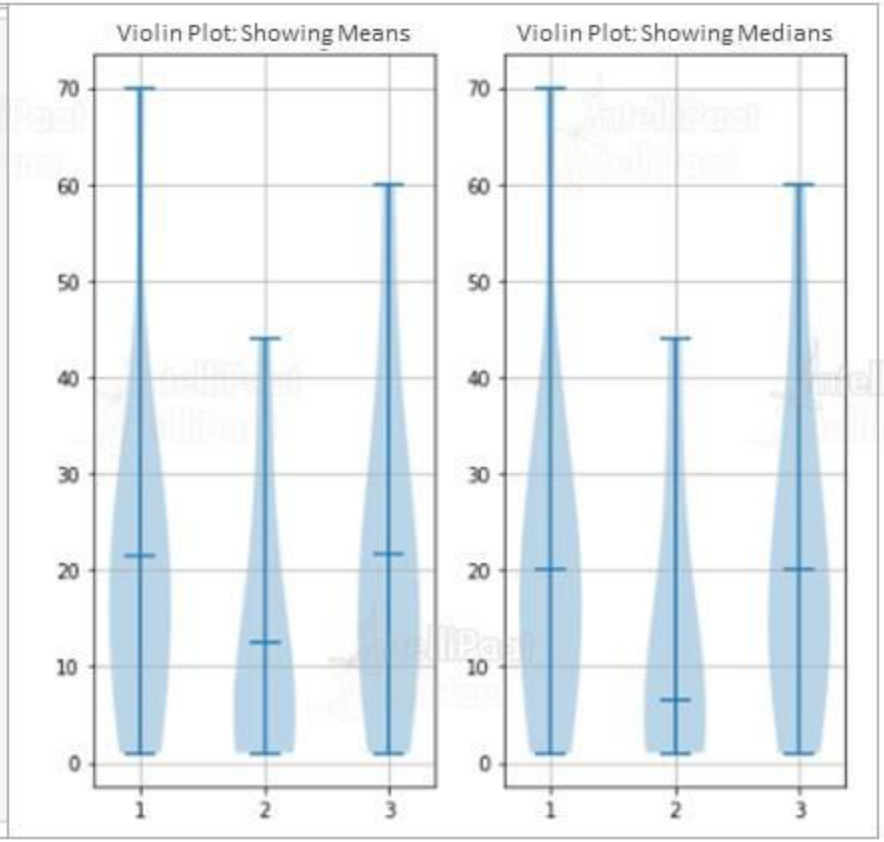


Image Plot

Image Plot

Image Plot

An image plot can be used for image manipulation. We can convert an image into a NumPy array and then use those values to plot the image

Image Plot

When should we choose an image plot?

There is no particular use case. We can use this whenever we want to project our image in various color scales or convert it to a NumPy array

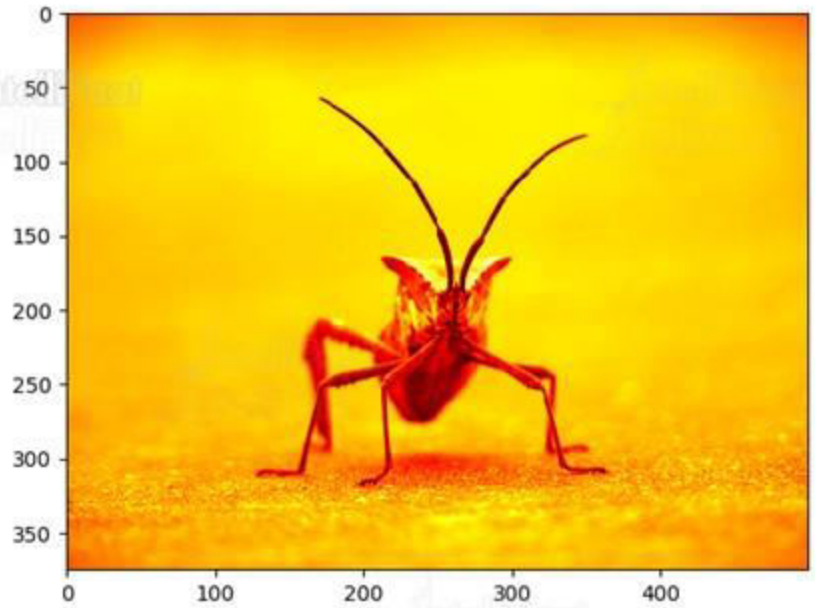


Image Plot

Image Plot

Image to NumPy Array Conversion

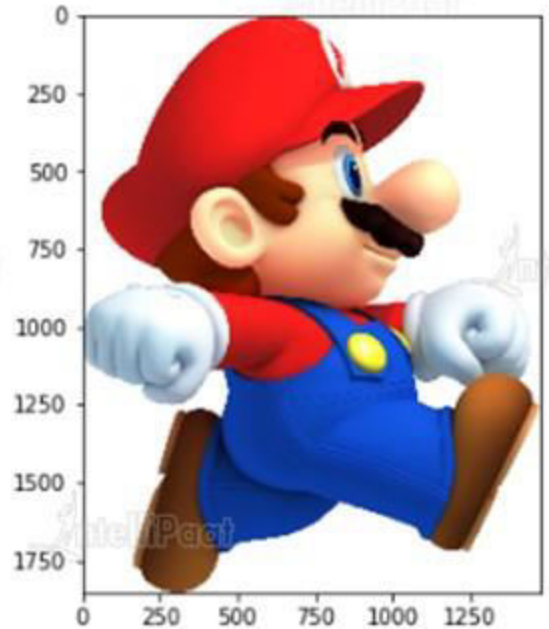
```
[[[255 255 255   0]
  [255 255 255   0]
  [255 255 255   0]
  ...
  [255 255 255   0]
  [255 255 255   0]
  [255 255 255   0]]

 [[255 255 255   0]
  [255 255 255   0]
  [255 255 255   0]
  ...
  [255 255 255   0]
  [255 255 255   0]
  [255 255 255   0]]

 [[255 255 255   0]
  [255 255 255   0]
  [255 255 255   0]
  ...
  [255 255 255   0]
  [255 255 255   0]
  [255 255 255   0]]

 ...
```

Image Plot

Plotting image from a NumPy array

```
In [59]: #import numpy and matplotlib
         #Python Imaging Library
         from PIL import Image
         import numpy as np
         import matplotlib.pyplot as plt
         %matplotlib inline
         #prepare data
         img = Image.open("mario.png")
         #convert to .npy
         arr = np.array(img)
         #Add figure
         fig=plt.figure(figsize=(10,5))
         #add axes
         ax1 = plt.subplot()
         #plot image
         ax1.imshow(arr)
```

```
Out[59]: <matplotlib.image.AxesImage at 0x25d09eb7f60>
```



Quiver and Stream Plots

Quiver and Stream Plots

Quiver Plot

A quiver plot shows vector lines as arrows and is useful in electrical engineering to visualize electrical potential

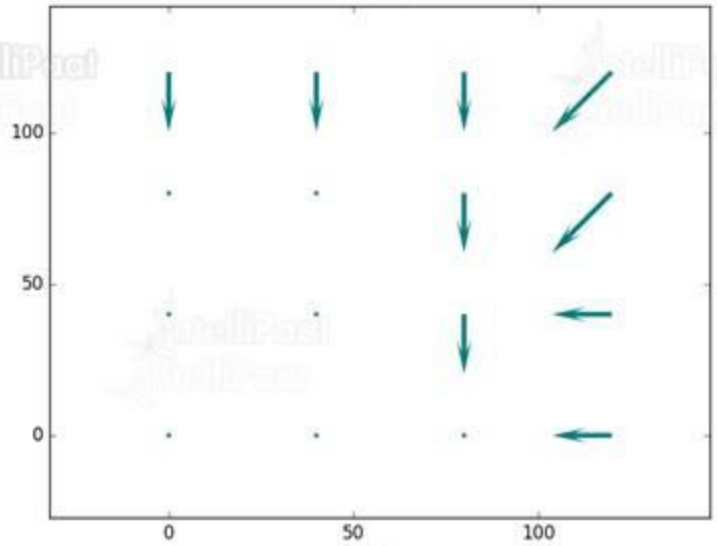
Stream Plot

A stream plot is a type of 2D plot used to show fluid flow and 2D field gradients

Quiver and Stream Plots

When should we choose a quiver plot?

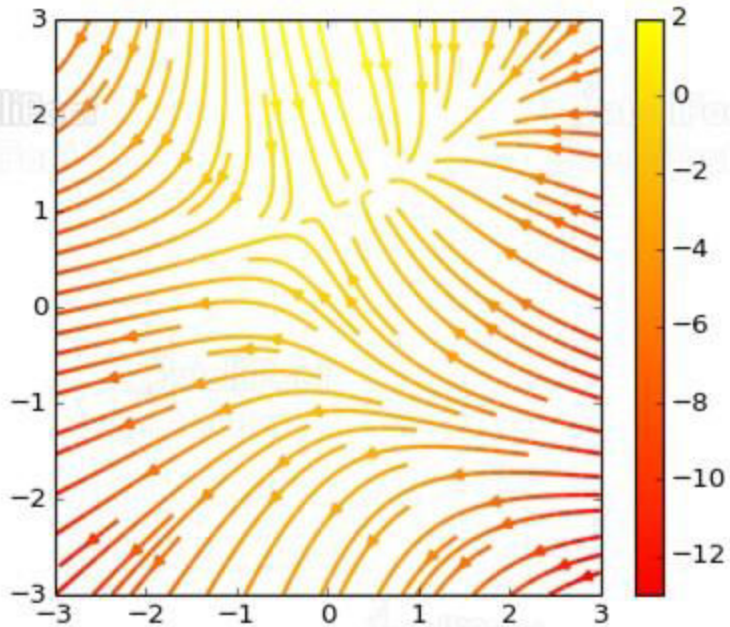
We use it when we want to create a 2-dimensional field of arrows, e.g., making a chart for electrical current fields



Quiver and Stream Plots

When should we choose a stream plot?

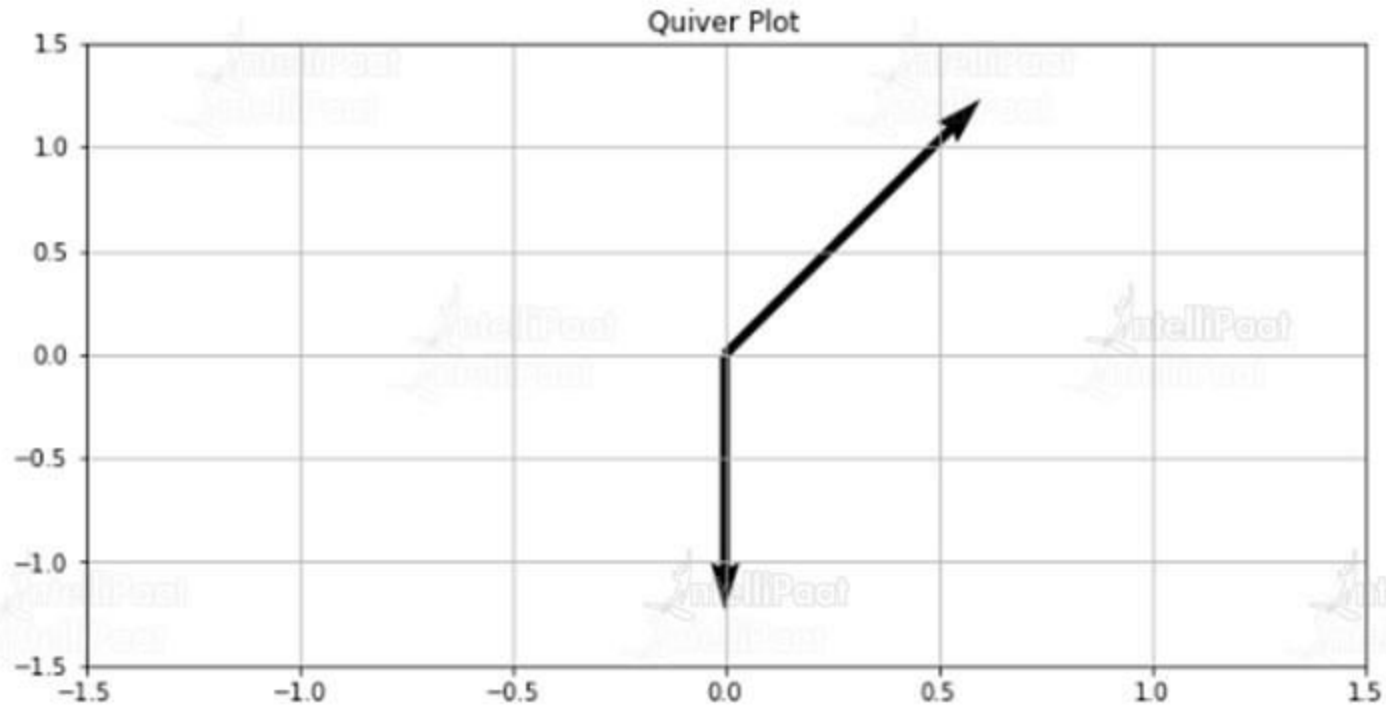
Similar to a quiver plot, but we can use a stream plot for purposes like ocean current plotting or area density using the color legend



Quiver and Stream Plots

Quiver Plot

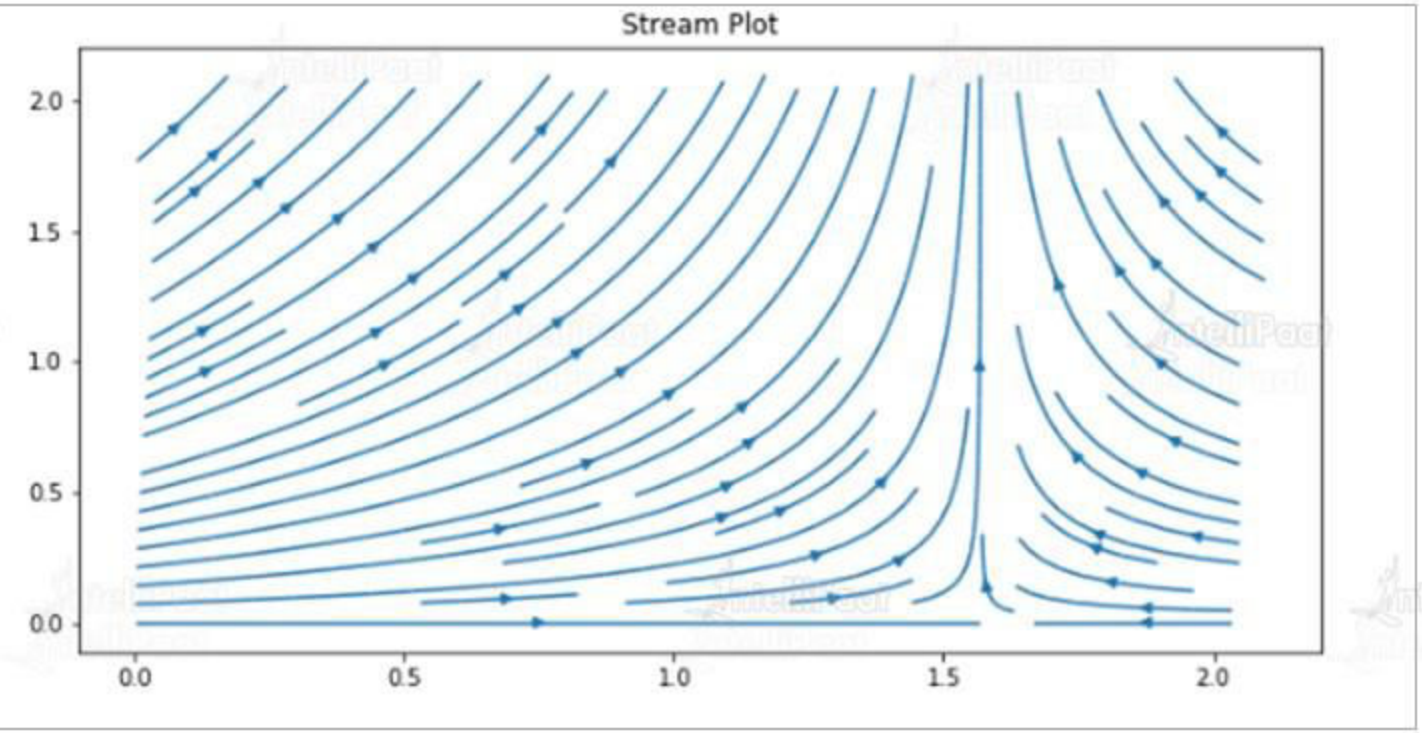
```
In [69]: #import libraries
         import matplotlib.pyplot as plt
         %matplotlib inline
         #prepare data
         x_pos = [0, 0]
         y_pos = [0, 0]
         x_direct = [1, 0]
         y_direct = [1, -1]
         #Add figure
         fig=plt.figure(figsize=(10,5))
         #add axes
         ax1 = plt.subplot()
         #plot
         ax1.quiver(x_pos,y_pos,x_direct,y_direct,scale=5)
         #Changing the scale limits
         ax1.axis([-1.5, 1.5, -1.5, 1.5])
         #Customization-title
         plt.title('Quiver Plot')
         #customization-add grid
         plt.grid(True)
         #save the plot
         plt.savefig('QuiverPlot.png')
         #show plot
         plt.show()
```



Quiver and Stream Plots

Stream Plot

```
In [72]: #import libraries
         import numpy as np
         import matplotlib.pyplot as plt
         %matplotlib inline
         #prepare data
         x = np.arange(0,2.2,0.1)
         y = np.arange(0,2.2,0.1)
         X, Y = np.meshgrid(x, y)
         u = np.cos(X)*y
         v = np.sin(y)*Y
         #Add figure
         fig=plt.figure(figsize=(10,5))
         #add axes
         ax1 = plt.subplot()
         #plot
         ax1.streamplot(X,Y,u,v, density = 1)
         #Customization-title
         plt.title('Stream Plot')
         #save the plot
         plt.savefig('StreamPlot2.png')
         #show plot
         plt.show()
```

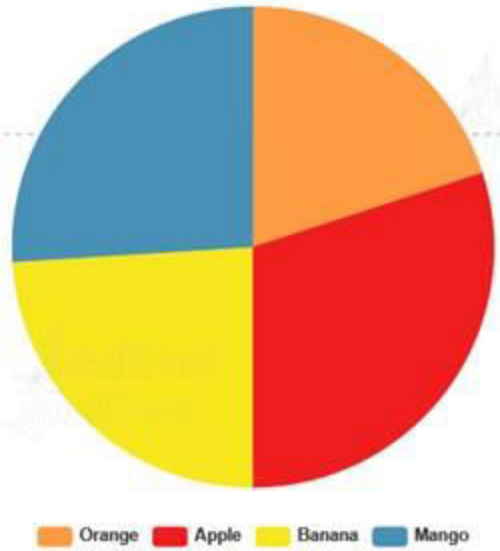


Pie Chart

Pie Chart

When should we choose a pie chart?

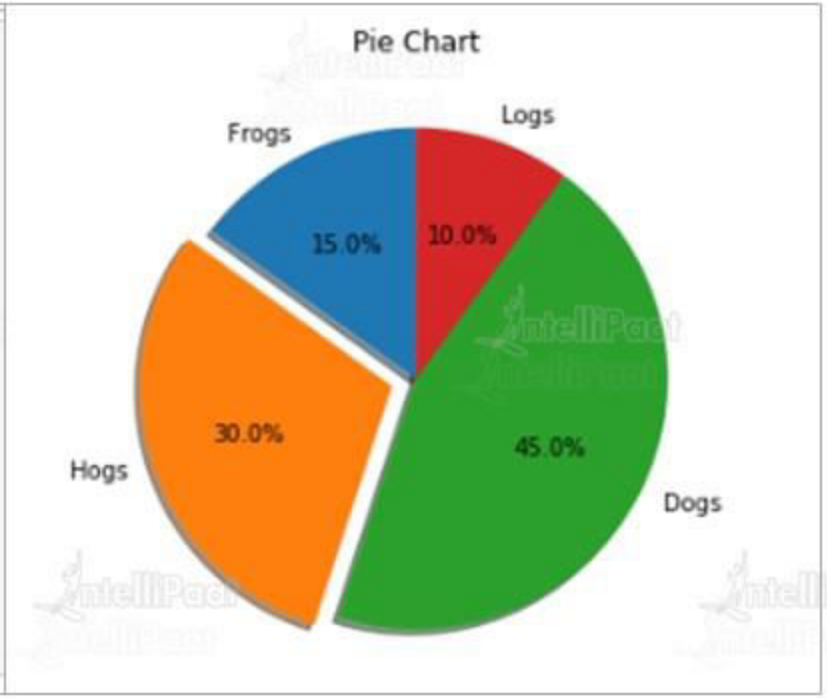
We use it when we want to compare sub-parts of the whole, e.g., to show different types of fruits



Pie Chart

Pie Chart

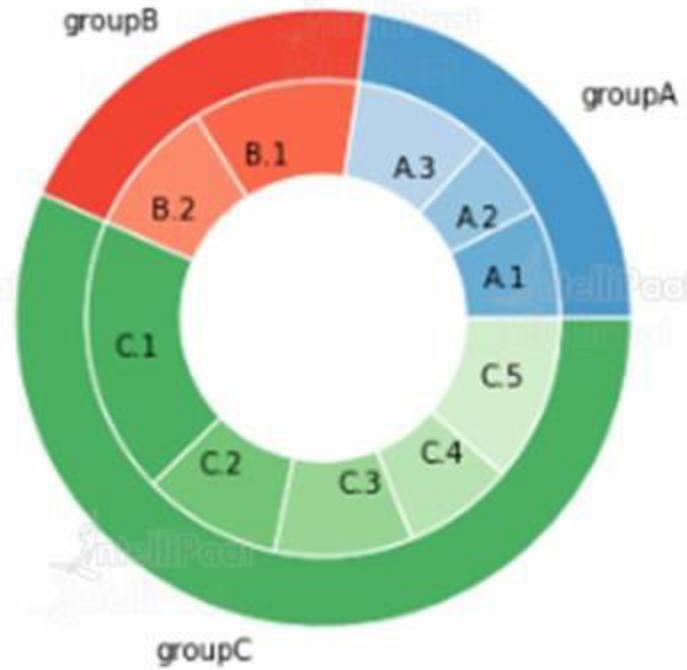
In [74]:
```
#import libraries
import matplotlib.pyplot as plt
%matplotlib inline
#prepare data
labels = ['Frogs', 'Hogs', 'Dogs', 'Logs']
sizes = [15, 30, 45, 10]
#add explode if required or else keep 0
explode = (0, 0.1, 0, 0)
#Add figure
fig=plt.figure(figsize=(10,5))
#add axes
ax1 = plt.subplot()
#plot-sezes, labels, autopercentage, shadow, start-angle=90
ax1.pie(sizes, explode=explode, labels=labels, autopct='%1.1f%%',shadow=True, startangle=90)
#Customization-title
plt.title('Pie Chart')
#save the plot
plt.savefig('PieChart2.png')
#show
plt.show()
```



Pie Chart: Donut Chart

Donut Chart

```
In [7]: # Libraries
import matplotlib.pyplot as plt

# Make data: I have 3 groups and 7 subgroups
group_names=['groupA', 'groupB', 'groupC']
group_size=[12,11,30]
subgroup_names=['A.1', 'A.2', 'A.3', 'B.1', 'B.2', 'C.1', 'C.2', 'C.3', 'C.4', 'C.5']
subgroup_size=[4,3,5,6,5,10,5,5,4,6]
# Create colors
a, b, c=[plt.cm.Blues, plt.cm.Reds, plt.cm.Greens]


# Add figure and axes
fig, ax = plt.subplots()
ax.axis('equal')

#plot first ring
mypie, _ = ax.pie(group_size, radius=1.3, labels=group_names, colors=[a(0.6), b(0.6), c(0.6)] )
# plot Second Ring (Inside)
mypie2, _ = ax.pie(subgroup_size, radius=1.3-0.3, labels=subgroup_names, labeldistance=0.7, colors=[a(0.5), a(0.4), a(0.3)

# Customize
plt.setp( mypie, width=0.3, edgecolor='white')
plt.setp( mypie2, width=0.4, edgecolor='white')
plt.margins(0,0)

#save the plot
plt.savefig('NestedPieChart.png')

# show it
plt.show()
```



Pie Chart: Donut Chart

Pie Chart and Donut Chart

A **pie chart** is used to show percentage or proportional data, which is good for displaying data for around 6 categories or fewer

A **donut chart** can contain more than one data series. Each data series that we plot in a donut chart adds an extra ring to it

Hands-on: Graphs and Charts

India: +91-7847955955

US: 1-800-216-8930 (TOLL FREE)

support@intellipaat.com

24/7 Chat with Our Course Advisor