Data Visualization

Data Visualization

Data visualization is a technique to present the data in a pictorial or graphical format.

Benefits of Data Visualization are:

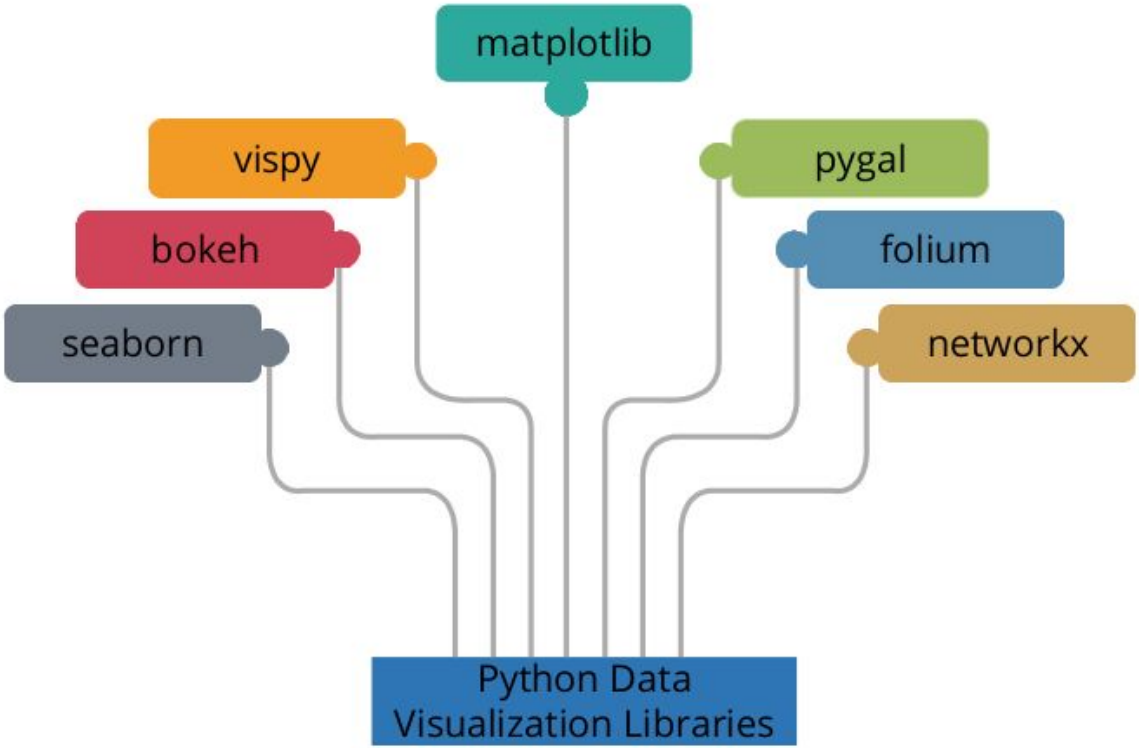
Considerations of Data Visualization

- Ensure the dataset is complete and relevant.This enables the Data Scientist to use the new patterns obtained from the data in the relevant places.
- Ensure that you use appropriate graphical representation to convey the intended message.
- Use efficient visualization techniques that highlight all the data points.

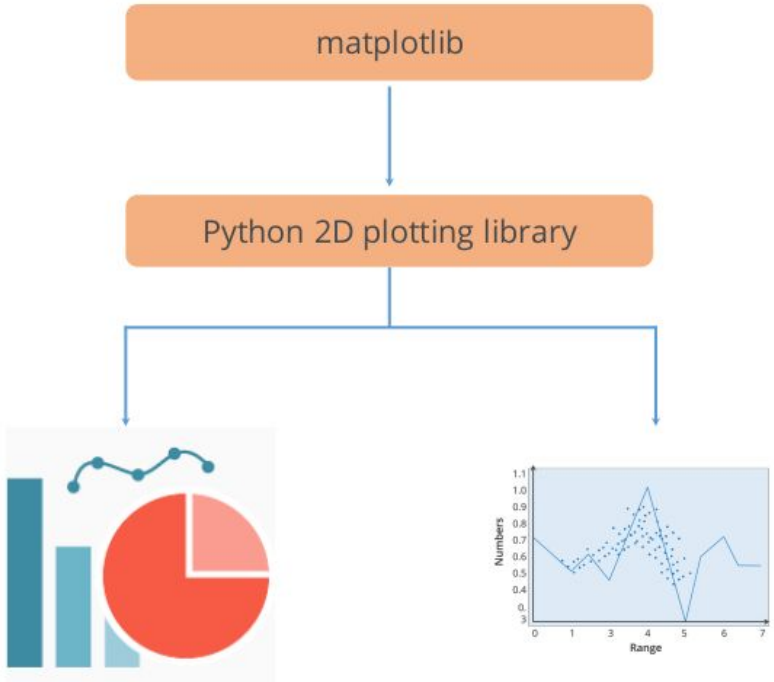
Factors of Data Visualization

- The **visual effect** includes the usage of appropriate shapes, colors, and sizes to represent the analyzed data.
- The **coordinate system** helps organize the data points within the provided coordinates.
- The **data types and scale** choose the type of data, for example, numeric or categorical.
- The informative interpretation helps create visuals in an effective and easily interpretable manner using labels, title, legends, and pointers.

Python Libraries



Python Libraries: matplotlib



matplotlib(Advantages)

Is a multi-platform data visualization tool; therefore, it is fast and efficient.

Can work well with many operating systems and graphics back ends

Has high-quality graphics and plots to print and view a range of graphs

With Jupyter notebook integration, the developers are free to spend their time implementing features

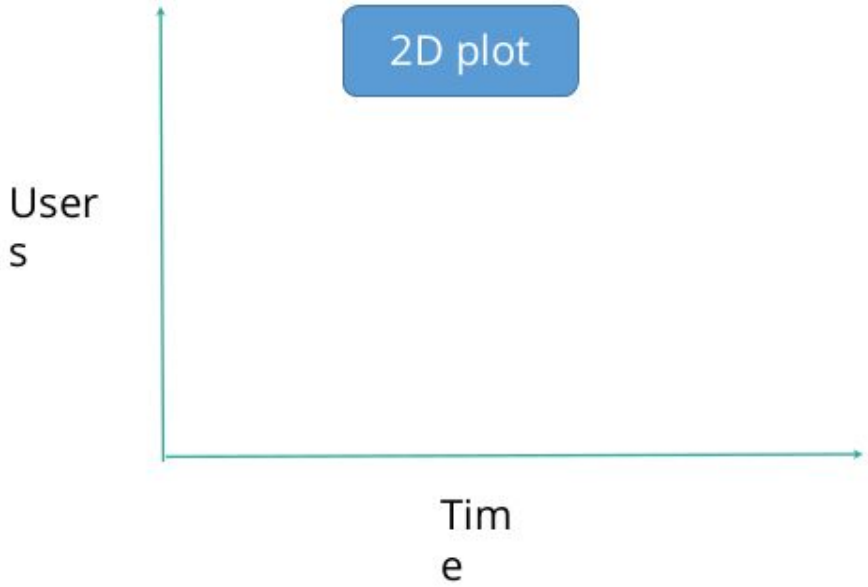
Has large community support and cross platform support as it is an open source tool

Has full control over graphs or plot styles

Let’s Plot

Plot with (X,Y)

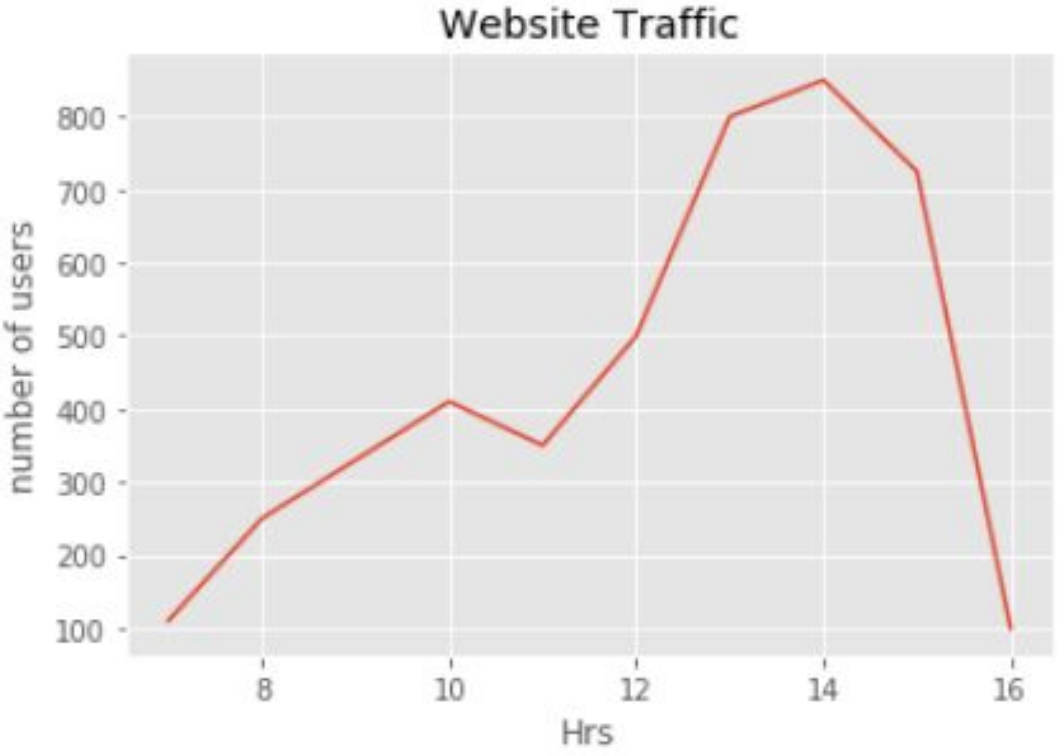
ICFOSS wants to know how many people visit its website in a particular time. This analysis helps it control and monitor the website traffic.



```
#import libraries
import matplotlib.pyplot as plt
from matplotlib import style
%matplotlib inline
```

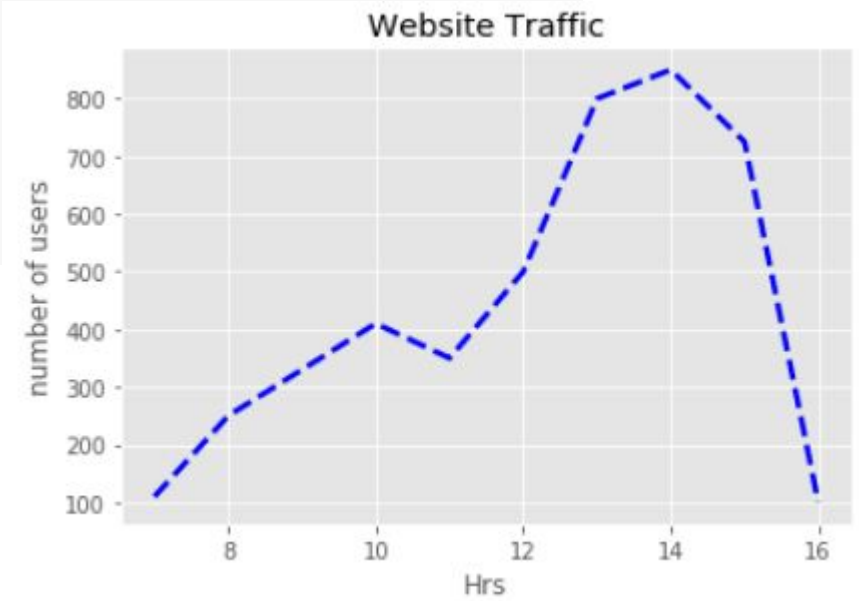
```
# Website traffic data
# Number of users visited on the website
web_customers = [110,250,330,410,350,500,800,850,725,100]
#hourly time distribution
time_hrs = [7,8,9,10,11,12,13,14,15,16]
```

```
#select the style of the plot
style.use('ggplot')
#plot the data(X axis as hrs and Y axis as number of users)
plt.plot(time_hrs,web_customers)
#Set the title of the plot
plt.title('Website Traffic')
#Set label for x axis
plt.xlabel('Hrs')
#Set label for y axis
plt.ylabel('number of users')
plt.show()
```


Website Traffic
number of users
800
700
600
500
400
300
200
100
8
10
12
14
16
Hrs

Controlling Line Patterns and Colors

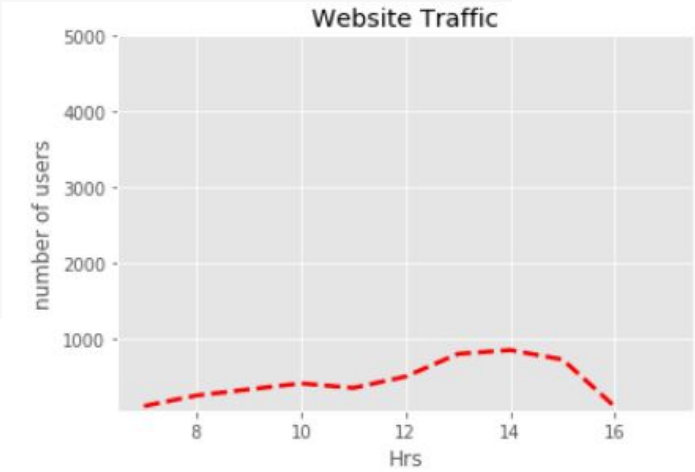
```
#select the style of the plot
style.use('ggplot')
#plot the data(X axis as hrs and Y axis as number of users)
plt.plot(time_hrs,web_customers, color = 'b', linestyle = '--', linewidth=2.5)
#Set the title of the plot
plt.title('Website Traffic')
#Set label for x axis
plt.xlabel('Hrs')
#Set label for y axis
plt.ylabel('number of users')
plt.show()
```



Set Axis, Labels, and Legend Property

Is it possible to set the desired axis to interpret the result. ???

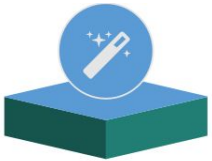
```
#select the style of the plot
style.use('ggplot')
#plot the data(X axis as hrs and Y axis as number of users)
plt.plot(time_hrs,web_customers, color = 'r', linestyle = '--', linewidth=2.5)
plt.axis([6.5, 17.5, 50, 5000])
#Set the title of the plot
plt.title('Website Traffic')
#Set label for x axis
plt.xlabel('Hrs')
#Set label for y axis
plt.ylabel('number of users')
plt.show()
```



Seaborn

Seaborn

Seaborn is a Python visualization library based on matplotlib. It provides a high-level interface to draw attractive statistical graphics.



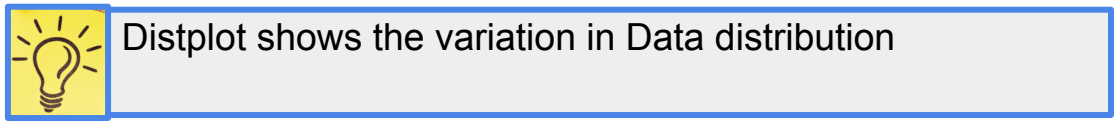
Possesses built-in themes for better visualizations

Has built-in statistical functions which reveal hidden patterns in the dataset.

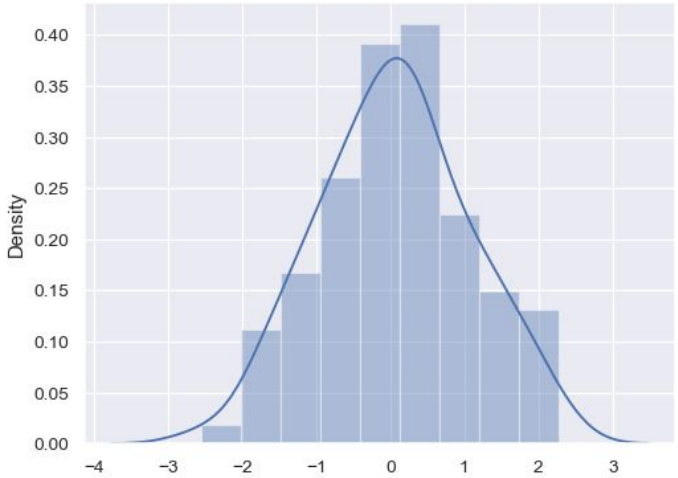
Has functions to visualize matrices of data.

Distribution Plot(distplot)

- Flexibly plot a univariate distribution of observation.
- This function combines the matplotlib **hist** function with the seaborn **kdeplot()** and **rugplot()** functions.
- Distplot represents the histogram and a line in combination to it.

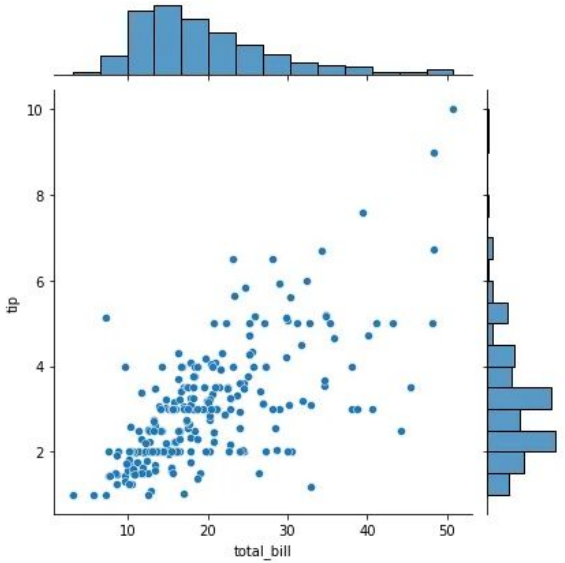


Distplot shows the variation in Data distribution



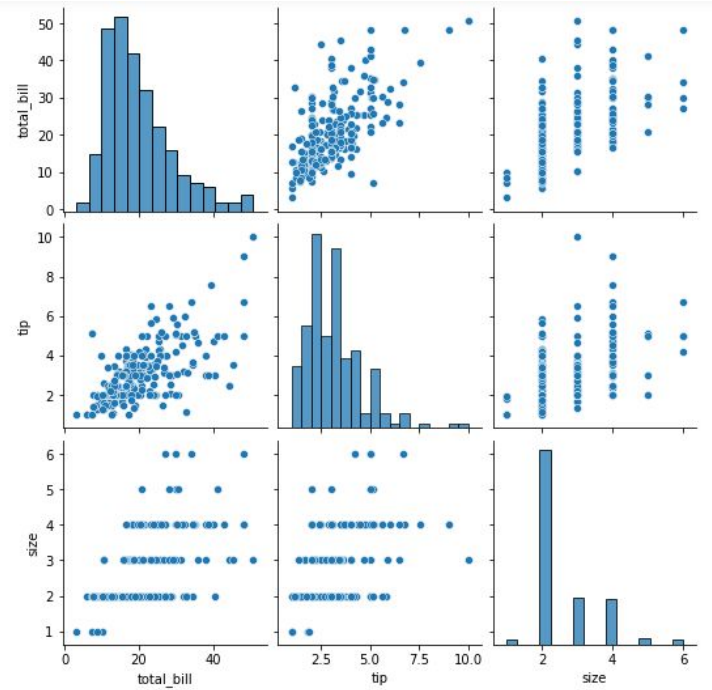
jointplot()

- Represents univariate and bivariate plot in one figure
- It comprises of three plots:
  1. Shows how dependent variable (Y) varies with independent variable(X).
  2. Placed horizontal and shows the distribution of independent variable(X).
  3. Placed vertically and shows the distribution of dependent variable(Y)



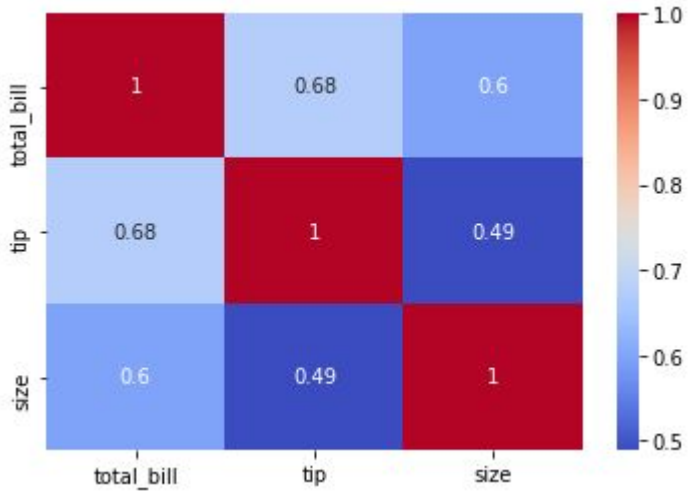
pairplot()

Represents the pairwise relationships in a datasets



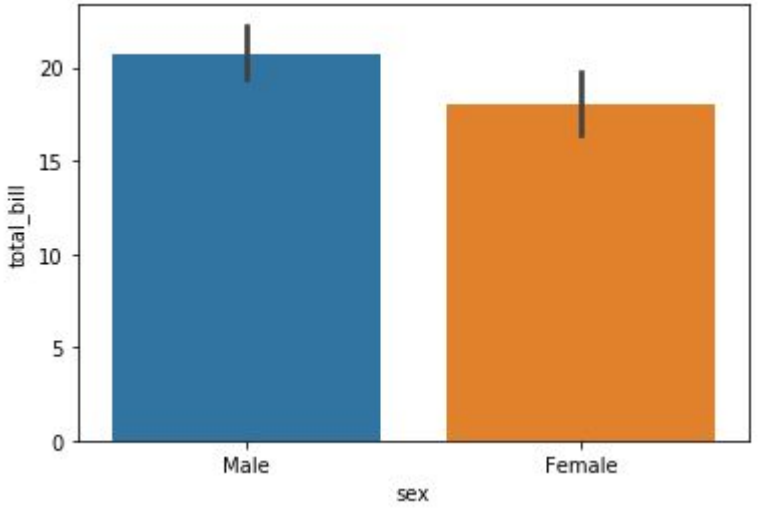
Heatmap

- **Heatmap** is a graphical representation of data that uses a system of color-coding to represent different values.
- **Used to** show user behaviour on specific webpages.



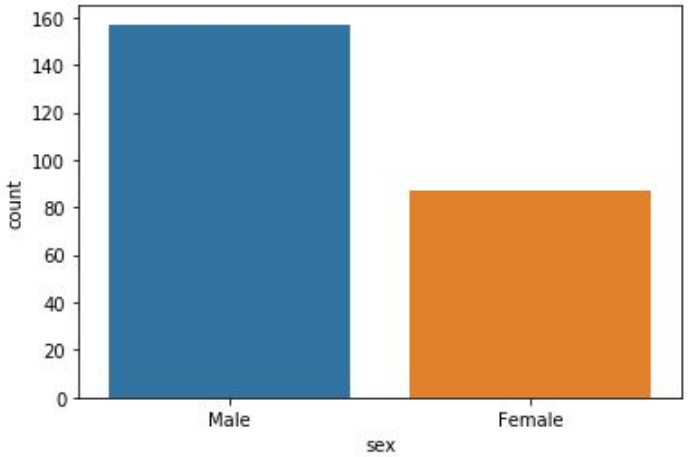
Bar Plot

- Basically used to aggregate the categorical data according to some methods and by default it’s the mean.
- Normally, we choose a categorical column for the x-axis and a numerical column for the y-axis.



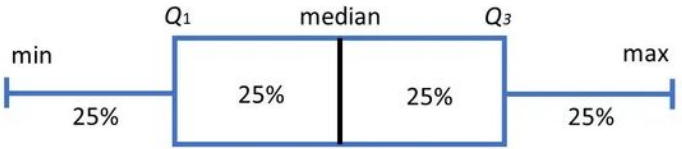
Count Plot

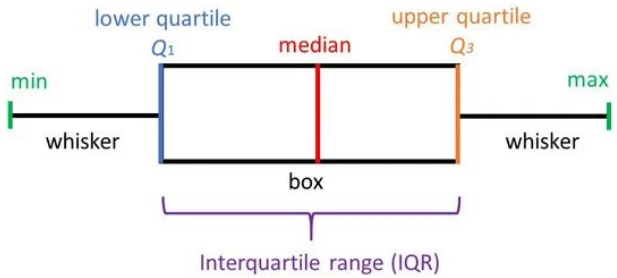
- Show the counts of observations in each categorical bin using bars.
- Count plot can be thought of as a histogram across a categorical, instead of quantitative, variable.

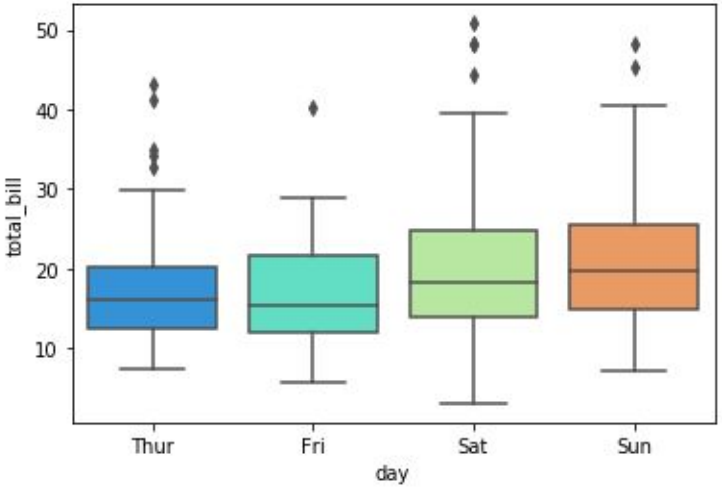


Box Plot

- Representing groups of numerical data through their quartiles.
- Boxplot is also used for detect the outlier in data set.
- A box plot consist of 5 things.
  - **Minimum**
  - **First Quartile or 25%**
  - **Median (Second Quartile) or 50%**
  - **Third Quartile or 75%**
  - **Maximum**
- Small diamond shape of blue box plot is **outlier data**.
- **Whiskers**: Represents scores outside middle 50%.

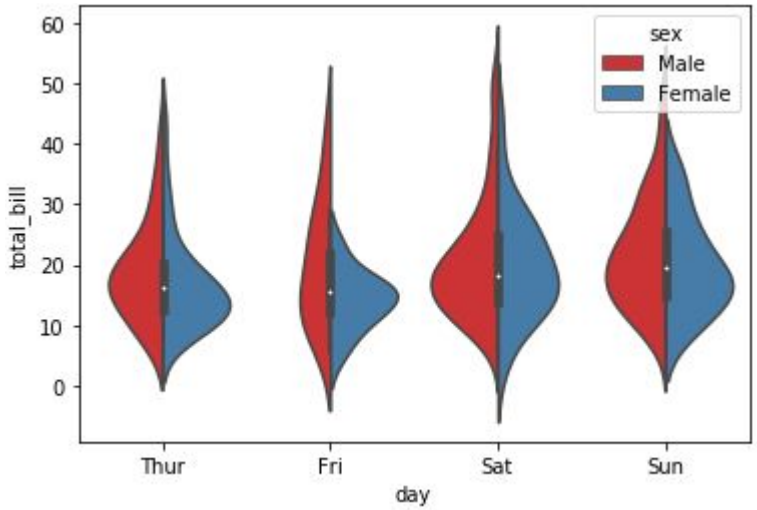


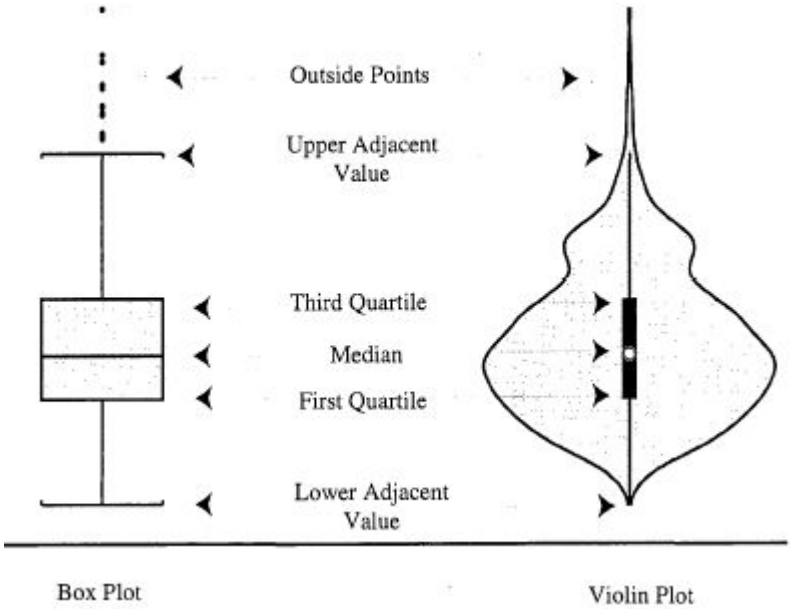




Violin Plot

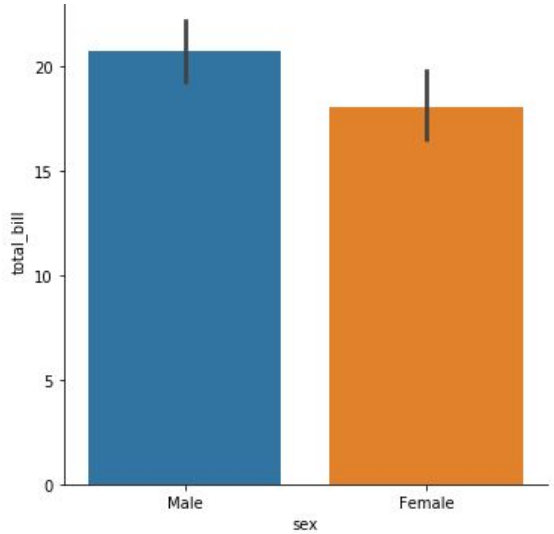
- Violin plots are a method of plotting numeric data.
- Combination of boxplot and kde plot.





Cat Plot

- For plots that involves categorical variables
- Giving access to several types of plots (8 different plots).



Thank You

Kailas E K