Sinhgad Institutes

Name of the Student:_____**Roll no:**_____

CLASS:- T.E.[I.T] **Division: A**

Course: - 2019

Subject: 314457: Data Science and Big Data Analytics Laboratory

PART_ B _Assignment No. 04

Marks: _____/10

Date of Performance: ___/____/_____ **Sign with Date: _____**

Part- B

ASSIGNMENT NO: 04

TITLE:

To Visualize the data using Python libraries.

AIM:

To Visualize the data using Python libraries matplotlib, seaborn by plotting the graphs for assignment no. 2 and 3 (Group B)

OBJECTIVE:

1. To understand and apply the Analytical concept of Big data using Python.
2. To understand different data visualization techniques for Big Data.

Software used: IDLE Shell 3.9.6 (Python 3.9.6)

THEORY:

R - Pie Charts

In R the pie chart is created using the pie() function which takes positive numbers as a vector input. The additional parameters are used to control labels, color, title etc.

Syntax

The basic syntax for creating a pie-chart using the R is −

pie(x, labels, radius, main, col, clockwise)

Following is the description of the parameters used −

- x is a vector containing the numeric values used in the pie chart. Labels is used to give description to the slices.
- radius indicates the radius of the circle of the pie chart.(value between −1 and +1).
- main indicates the title of the chart.
- col indicates the color palette.
- clockwise is a logical value indicating if the slices are drawn clockwise or anti clockwise.

Example

A very simple pie-chart is created using just the input vector and labels. The below script will create and save the pie chart in the current R working directory.

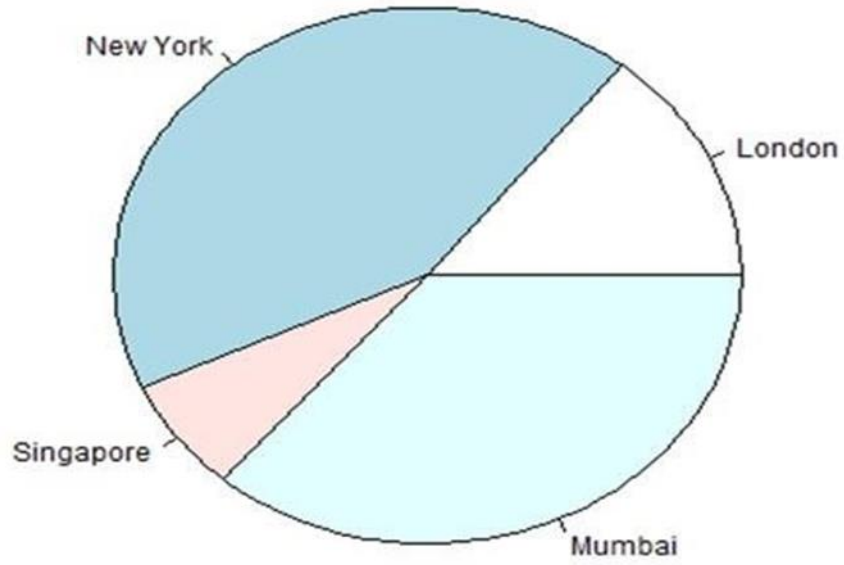
Create data for the graph. x <- c(21, 62, 10, 53)

labels <- c("London", "New York", "Singapore", "Mumbai")

Give the chart file a name. png(file = "city.jpg")

Plot the chart. pie(x,labels) # Save the file. dev.off()

When we execute the above code, it produces the following result –



R - Bar Charts

A bar chart represents data in rectangular bars with length of the bar proportional to the value of the variable. R uses the function barplot() to create bar charts. R can draw both vertical and horizontal bars in the bar chart. In bar chart each of the bars can be given different colors.

Syntax

The basic syntax to create a bar-chart in R is –

barplot(H, xlab, ylab, main, names.arg, col)

Following is the description of the parameters used –

- H is a vector or matrix containing numeric values used in bar chart.
- xlab is the label for x axis.
- ylab is the label for y axis.
- main is the title of the bar chart.
- names.arg is a vector of names appearing under each bar.

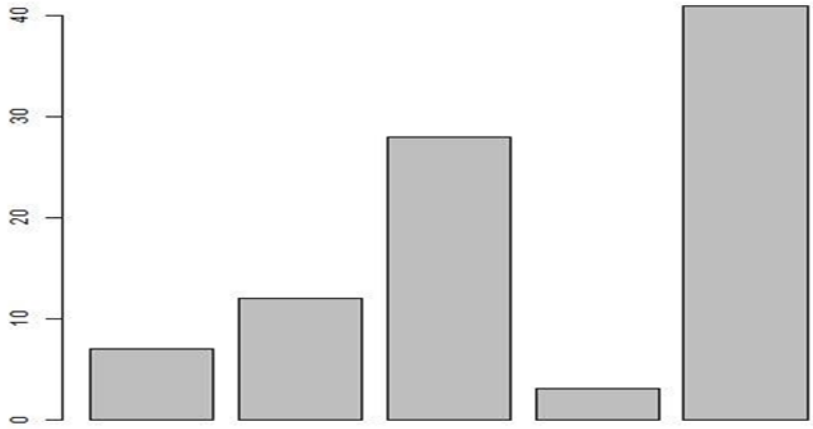
- col is used to give colors to the bars in the graph.

Example

A simple bar chart is created using just the input vector and the name of each bar. The below script will create and save the bar chart in the current R working directory.

```
# Create the data for the chart.
H <- c(7,12,28,3,41)
# Give the chart file a name. png(file = "barchart.png") # Plot the bar chart.
barplot(H)
# Save the file. dev.off()
```

When we execute the above code, it produces the following result –



R – Boxplots

Boxplots are a measure of how well distributed is the data in a data set. It divides the data set into three quartiles. This graph represents the minimum, maximum, median, first quartile and third quartile in the data set. It is also useful in comparing the distribution of data across data sets by drawing boxplots for each of them.

Boxplots are created in R by using the boxplot() function.

Syntax

The basic syntax to create a boxplot in R is –

```
boxplot(x, data, notch, varwidth, names, main)
```

Following is the description of the parameters used –

- x is a vector or a formula.
- data is the data frame.
- notch is a logical value. Set as TRUE to draw a notch.
- varwidth is a logical value. Set as true to draw width of the box proportionate to

the sample size.

- names are the group labels which will be printed under each boxplot. main is used to give a title to the graph.

Example

We use the data set "mtcars" available in the R environment to create a basic boxplot. Let's look at the columns "mpg" and "cyl" in mtcars.

input <- mtcars[,c('mpg','cyl')] print(head(input))

When we execute above code, it produces following result –

mpg cyl Mazda RX4 21.0 6

Mazda RX4 Wag 21.0 6

Datsun 710 22.8 4

Hornet 4 Drive 21.4 6

Hornet Sportabout 18.7 8

Valiant 18.1 6

Creating the Boxplot

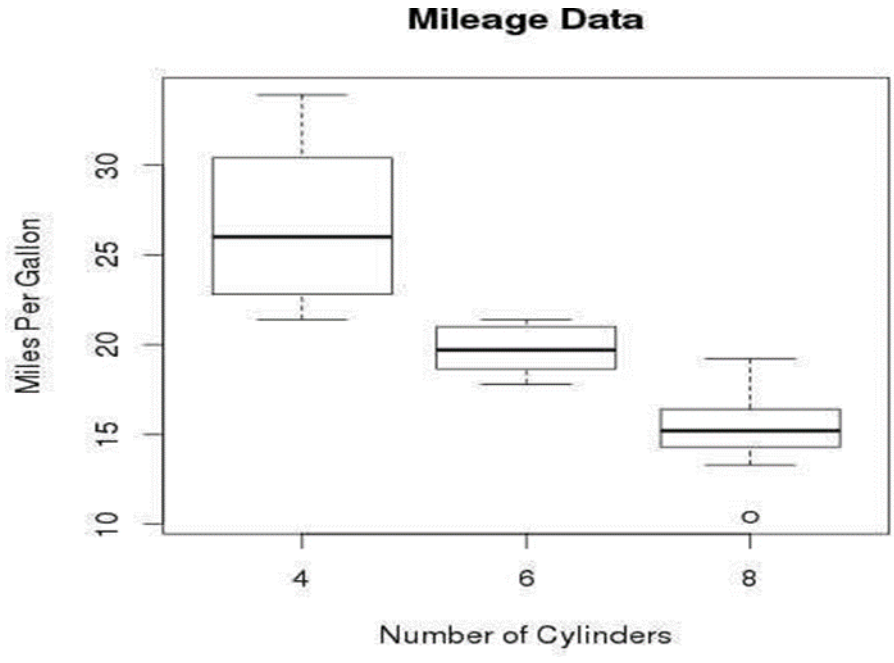
The below script will create a boxplot graph for the relation between mpg (miles per gallon) and cyl (number of cylinders).

Give the chart file a name. png(file = "boxplot.png")

Plot the chart. boxplot(mpg ~ cyl, data = mtcars, xlab = "Number of Cylinders", ylab = "Miles Per Gallon", main = "Mileage Data")

Save the file. dev.off()

When we execute the above code, it produces the following result –



R - Histograms

A histogram represents the frequencies of values of a variable bucketed into ranges. Histogram is similar to bar chat but the difference is it groups the values into continuous ranges. Each bar in histogram represents the height of the number of values present in that range.

R creates histogram using hist() function. This function takes a vector as an input and uses some more parameters to plot histograms.

Syntax

The basic syntax for creating a histogram using R is −

hist(v,main,xlab,xlim,ylim,breaks,col,border)

Following is the description of the parameters used −

- v is a vector containing numeric values used in histogram.
- main indicates title of the chart.
- col is used to set color of the bars.
- border is used to set border color of each bar.
- xlab is used to give description of x-axis.
- xlim is used to specify the range of values on the x-axis.
- ylim is used to specify the range of values on the y-axis.
- breaksis used to mention the width of each bar.

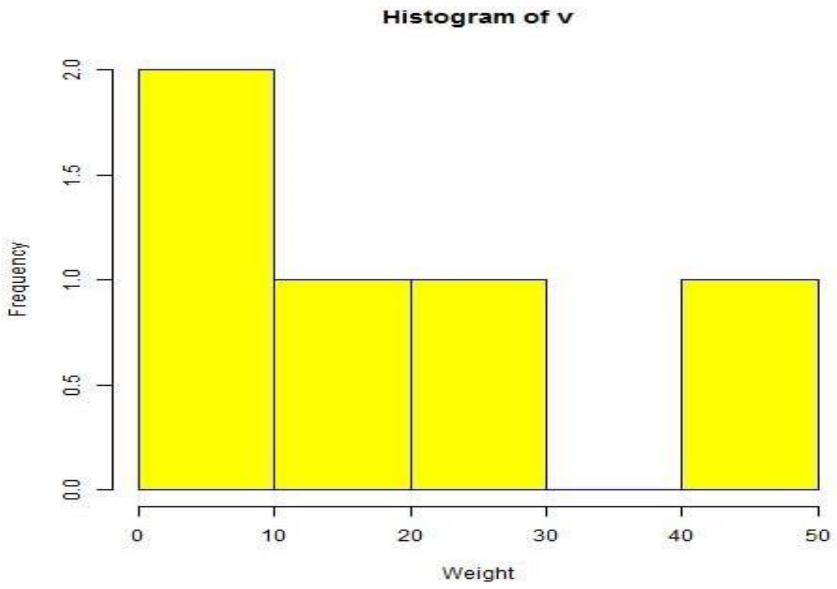
Example

A simple histogram is created using input vector, label, col and border parameters.

The script given below will create and save the histogram in the current R working directory.

```
# Create data for the graph.
v <- c(9,13,21,8,36,22,12,41,31,33,19)
# Give the chart file a name. png(file = "histogram.png")
# Create the histogram. hist(v,xlab = "Weight",col = "yellow",border = "blue")
# Save the file.
dev.off()
```

When we execute the above code, it produces the following result –



R - Line Graphs

A line chart is a graph that connects a series of points by drawing line segments between them. These points are ordered in one of their coordinate (usually the x-coordinate) value. Line charts are usually used in identifying the trends in data.

The plot() function in R is used to create the line graph.

Syntax

The basic syntax to create a line chart in R is −

plot(v,type,col,xlab,ylab)

Following is the description of the parameters used −

- v is a vector containing the numeric values.
- type takes the value "p" to draw only the points, "l" to draw only the lines and "o" to draw both points and lines.
- xlab is the label for x axis.
- ylab is the label for y axis.
- main is the Title of the chart.
- col is used to give colors to both the points and lines.

Example

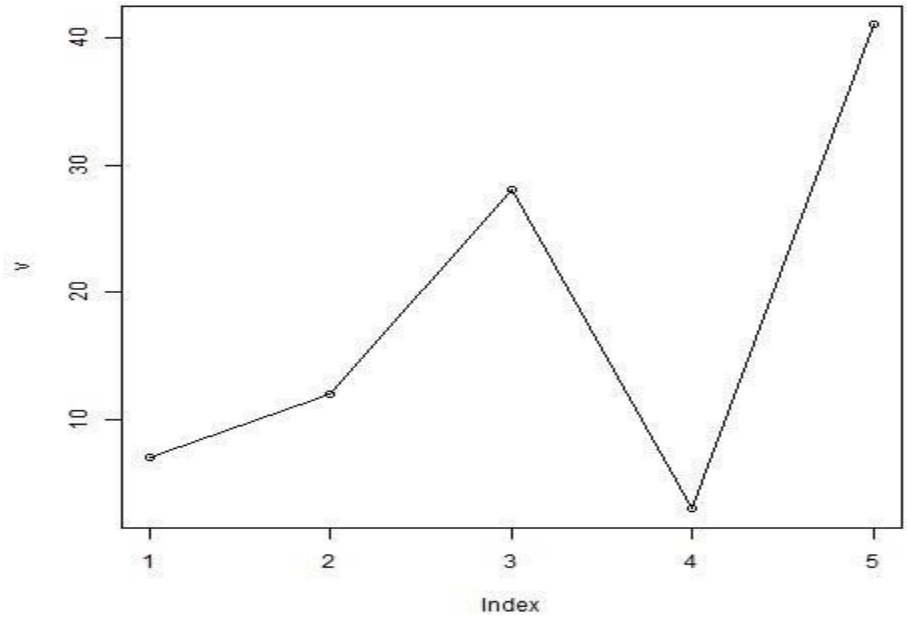
A simple line chart is created using the input vector and the type parameter as "O". The below script will create and save a line chart in the current R working directory.

Create the data for the chart. v <- c(7,12,28,3,41)

Give the chart file a name. png(file = "line_chart.jpg") # Plot the bar chart. plot(v,type = "o")

Save the file. dev.off()

When we execute the above code, it produces the following result –



R – Scatter plots

Scatterplots show many points plotted in the Cartesian plane. Each point represents the values of two variables. One variable is chosen in the horizontal axis and another in the vertical axis. The simple scatterplot is created using the plot() function.

Syntax

The basic syntax for creating scatter plot in R is –

plot(x, y, main, xlab, ylab, xlim, ylim, axes)

Following is the description of the parameters used –

- x is the data set whose values are the horizontal coordinates.
- y is the data set whose values are the vertical coordinates.
- main is the tile of the graph.
- xlab is the label in the horizontal axis.
- ylab is the label in the vertical axis.
- xlim is the limits of the values of x used for plotting.
- ylim is the limits of the values of y used for plotting.
- axesindicates whether both axes should be drawn on the plot.

Example

We use the data set "mtcars" available in the R environment to create a basic scatterplot. Let's use the columns "wt" and "mpg" in mtcars.

input <- mtcars[,c('wt','mpg')] print(head(input))

When we execute the above code, it produces the following result –

wt mpg Mazda RX4 2.620 21.0

Mazda RX4 Wag 2.875 21.0

Datsun 710 2.320 22.8

Hornet 4 Drive 3.215 21.4

Hornet Sportabout 3.440 18.7

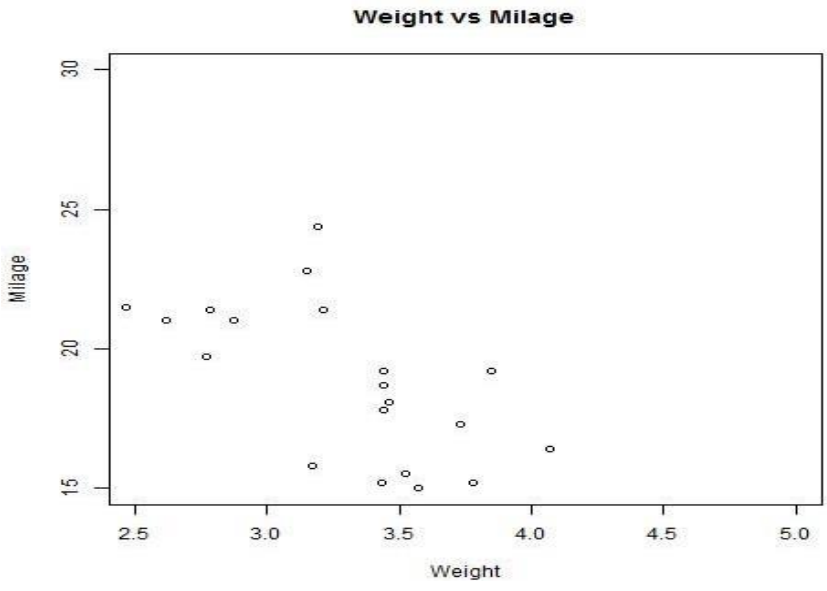
Valiant 3.460 18.1

Creating the Scatter plot

The below script will create a scatterplot graph for the relation between wt(weight) and mpg(miles per gallon).

```
# Get the input values.
input <- mtcars[,c('wt','mpg')]
# Give the chart file a name. png(file = "scatterplot.png")
# Plot the chart for cars with weight between 2.5 to 5 and mileage between 15 and 30.
plot(x = input$wt,y = input$mpg, xlab =
"Weight", ylab = "Milage", xlim = c(2.5,5), ylim = c(15,30),
main = "Weight vs Milage")
# Save the file. dev.off()
```

When we execute the above code, it produces the following result –



Scatter plot Matrices

When we have more than two variables and we want to find the correlation between one variable versus the remaining ones we use scatterplot matrix. We use pairs() function to create matrices of scatterplots.

Syntax

The basic syntax for creating scatterplot matrices in R is −

pairs(formula, data)

Following is the description of the parameters used −

formula represents the series of variables used in pairs.

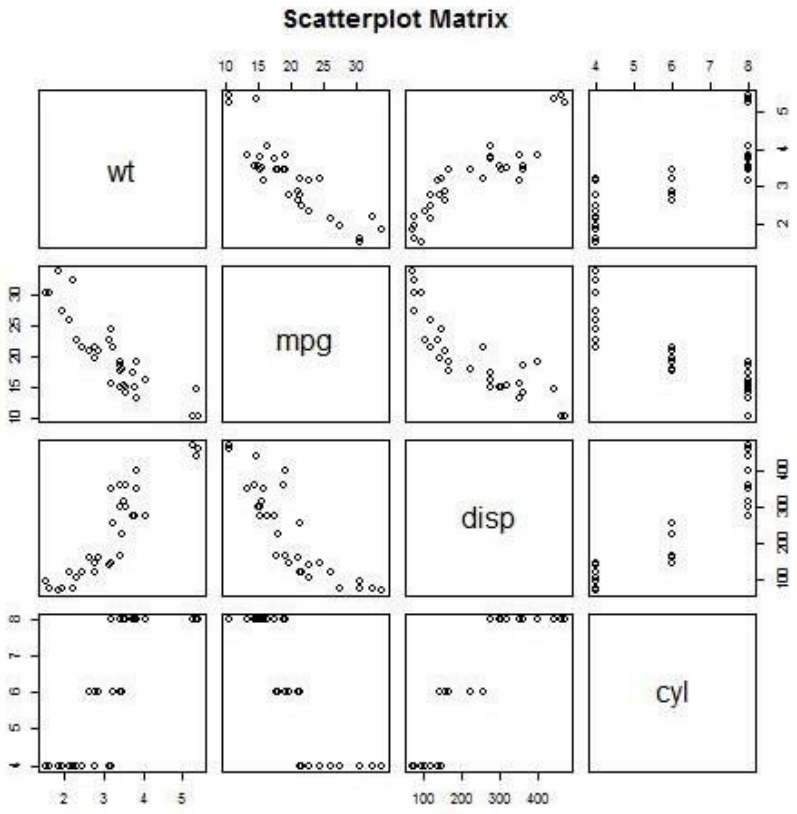
data represents the data set from which the variables will be taken.

Example

Each variable is paired up with each of the remaining variable. A scatterplot is plotted for each pair.

```
# Give the chart file a name.
png(file = "scatterplot_matrices.png")
# Plot the matrices between 4 variables giving 12 plots. # One variable with 3 others and total 4 variables.
pairs(~wt+mpg+disp+cyl,data = mtcars,    main = "Scatterplot Matrix")
# Save the file. dev.off()
```

When the above code is executed we get the following output.



CONCLUSION: After the study of this assignment we have learnt Visualize the data using R/Python by plotting the graphs.

Write Short Answers for Following Questions :

1. How to create a Histogram?
2. How to create a Bar Chart?
3. How to create a Stacked Bar Chart?
4. How to create a Box Plot?

Viva Questions :

1. How to create an Area Chart?
2. How to create a Heat Map?
3. How to create a Correlogram?
4. How to plot a geographical map?
5. How to plot the entire data in a single command?

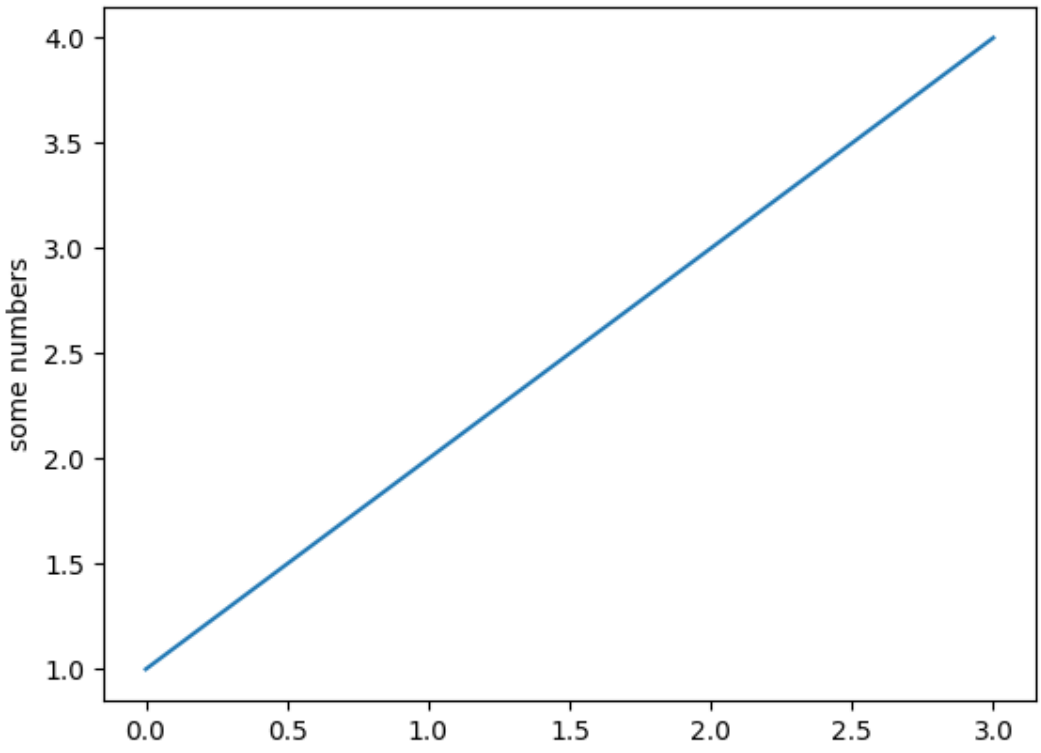
Python Program

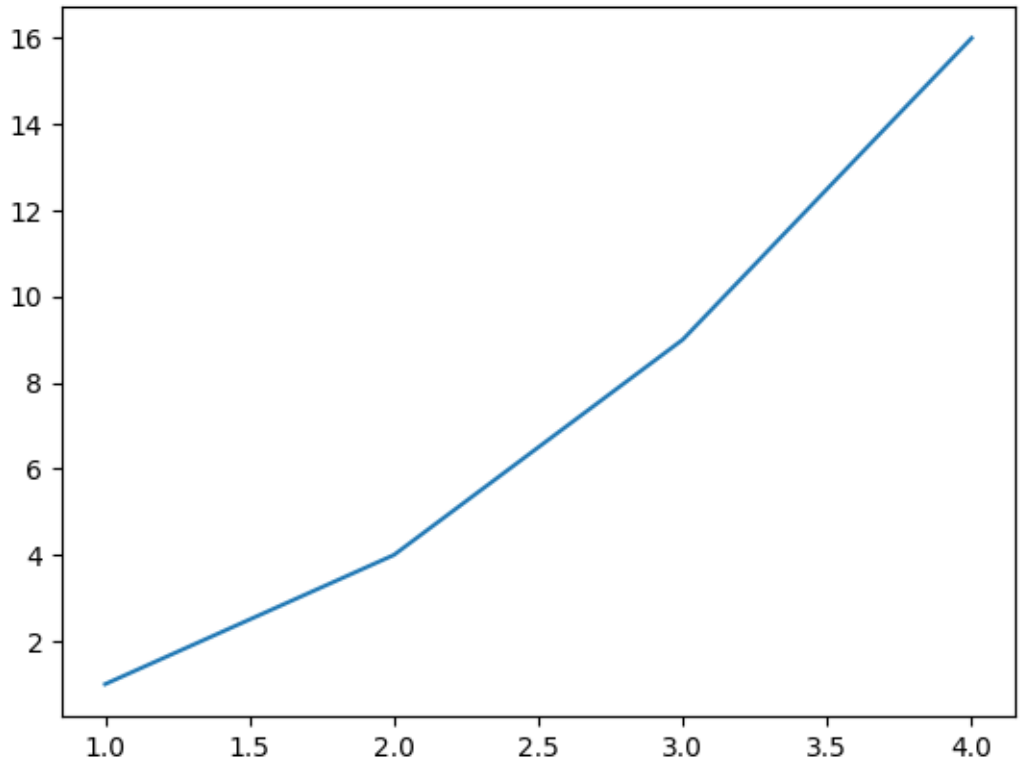
 # plot1

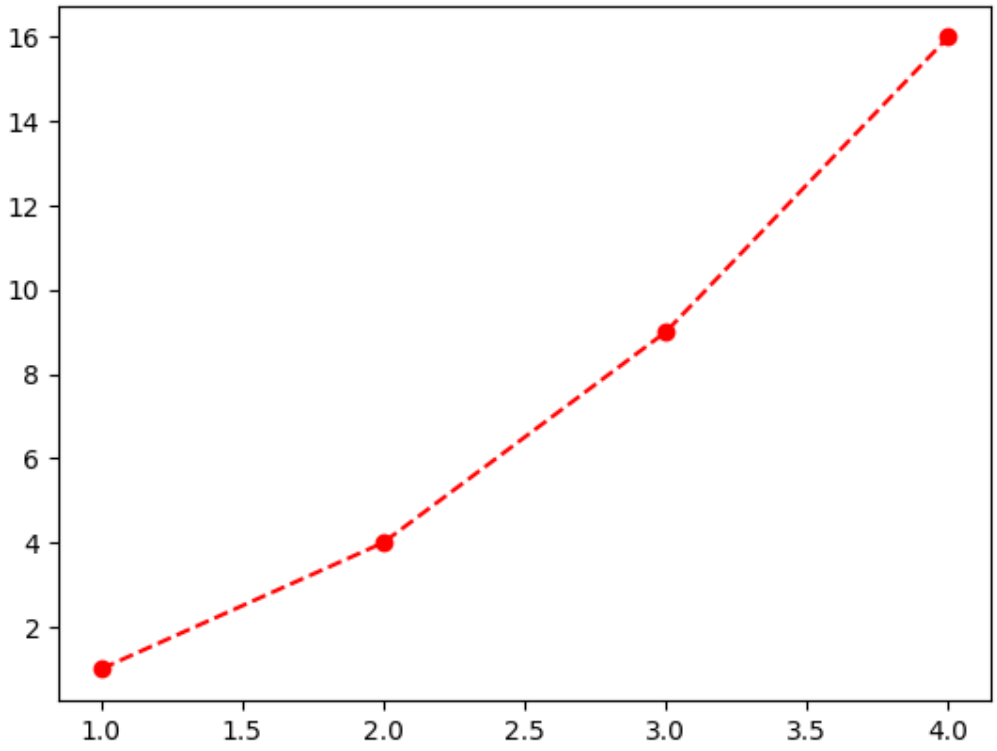
```
import matplotlib.pyplot as plt
plt.plot([1, 2, 3, 4])
plt.ylabel('some numbers')
plt.show()

plt.plot([1, 2, 3, 4], [1, 4, 9, 16])
plt.show()

plt.plot([1, 2, 3, 4], [1, 4, 9, 16], 'ro')
plt.plot([1, 2, 3, 4], [1, 4, 9, 16], 'r--')
plt.show()
```

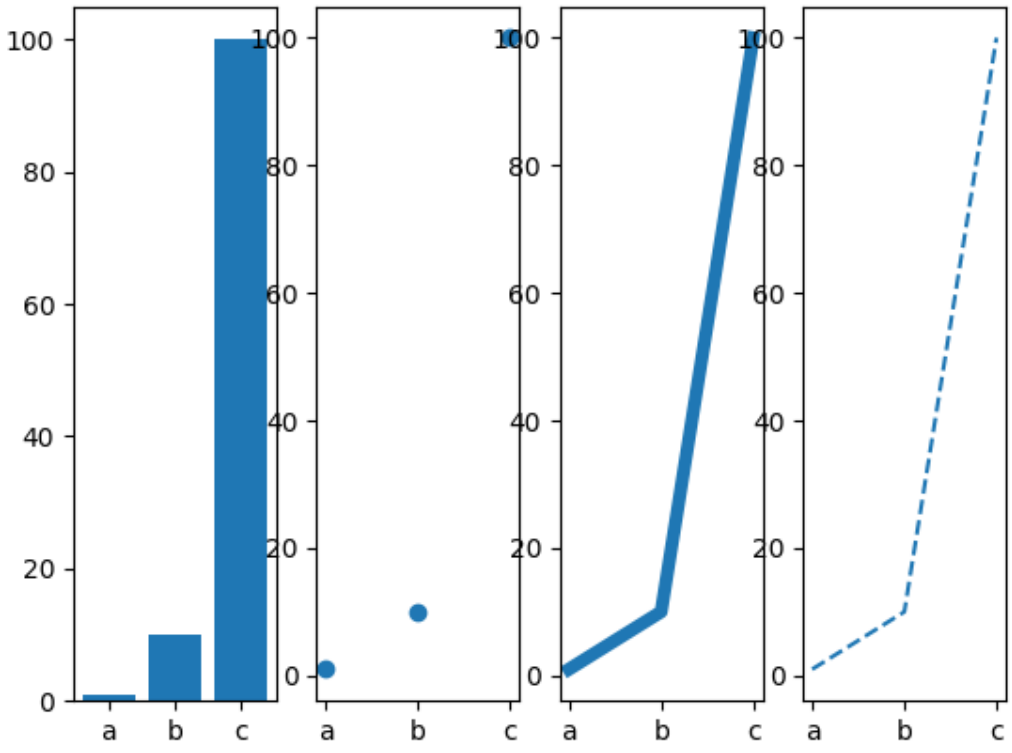



16
14
12
10
8
6
4
2
1.0
1.5
2.0
2.5
3.0
3.5
4.0


16
14
12
10
8
6
4
2
1.0
1.5
2.0
2.5
3.0
3.5
4.0

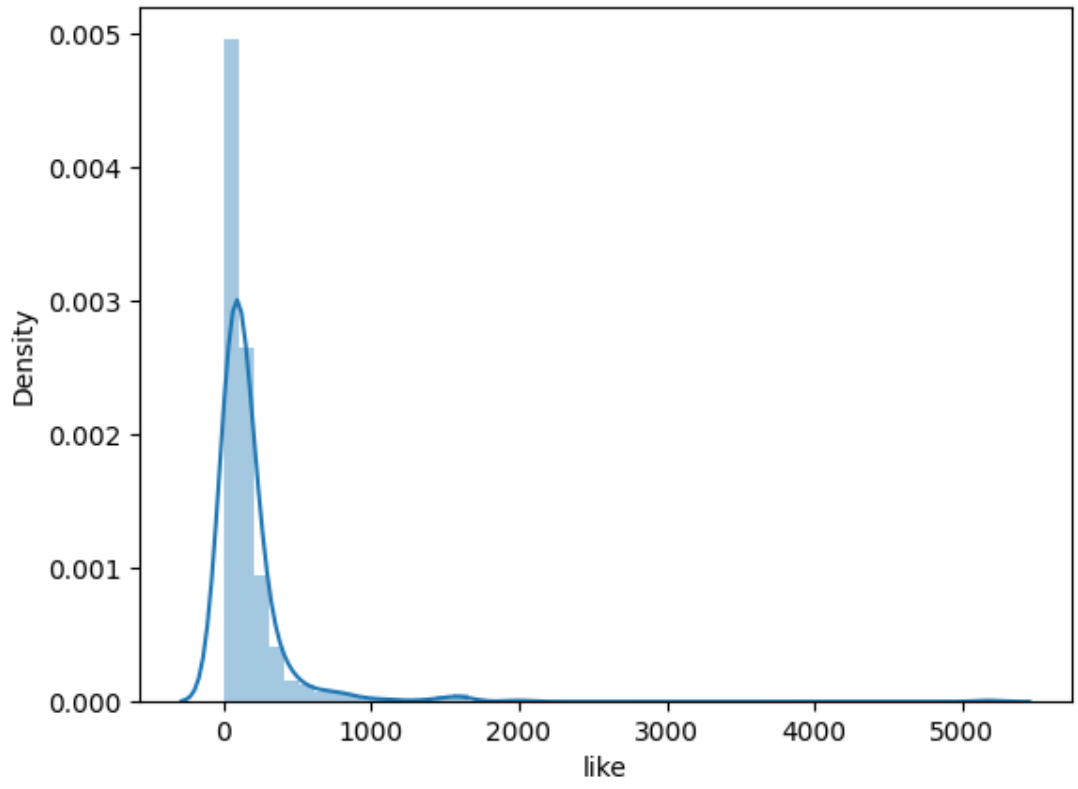
plot2

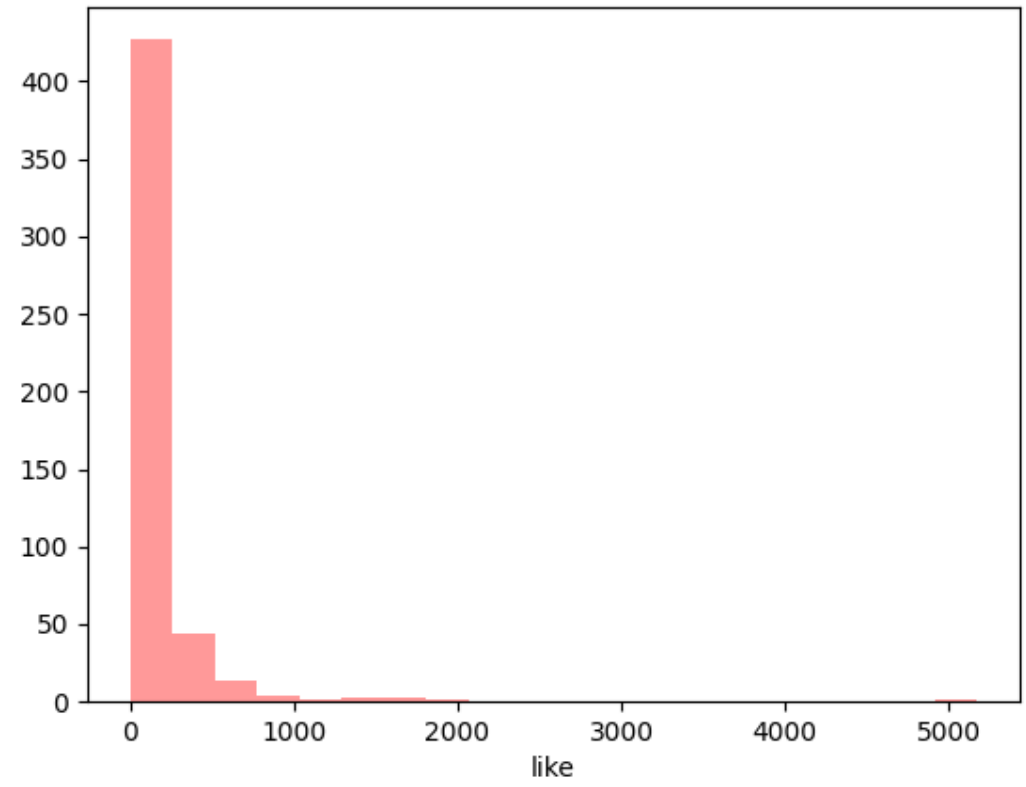
```
import matplotlib.pyplot as plt
names = ['a', 'b', 'c']
values = [1, 10, 100]
plt.subplot(141)
plt.bar(names, values)
plt.subplot(142)
plt.scatter(names, values)
plt.subplot(143)
plt.plot(names, values, linewidth=5.0)
plt.subplot(144)
plt.plot(names, values, '--')
plt.show()
```

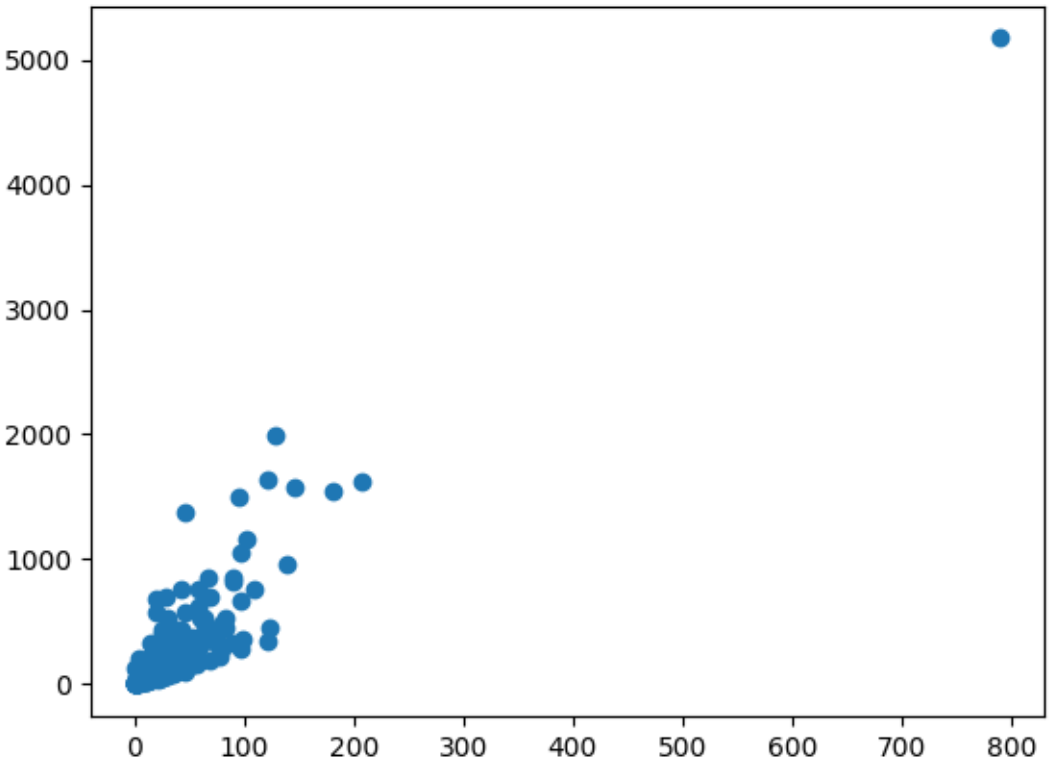


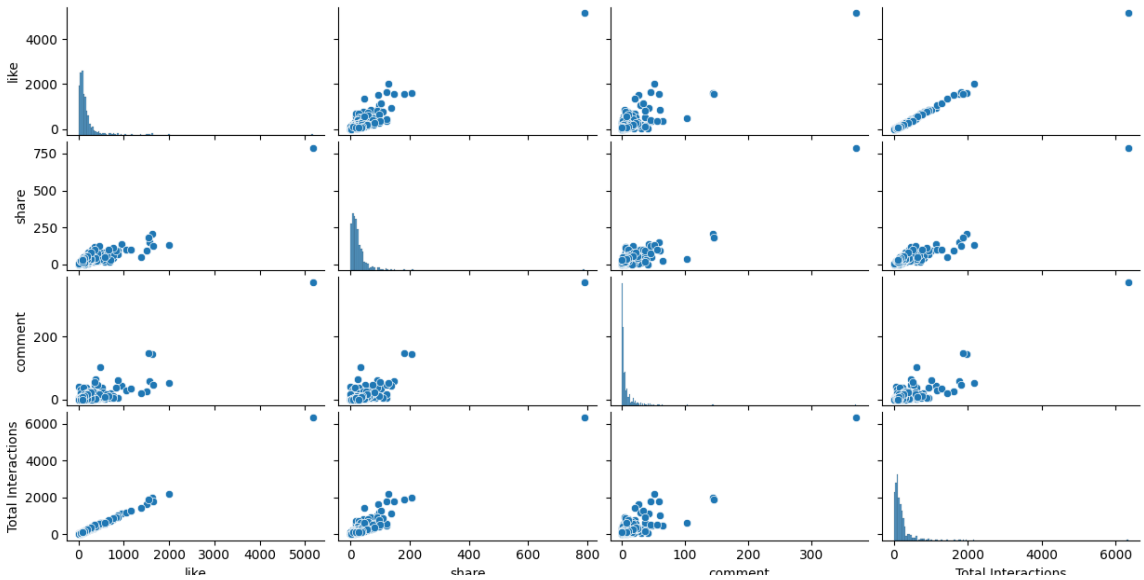
Plot3

```
import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd
pstore = pd.read_csv("dataset_Facebook.csv")
pstore.head(10)
#Create a distribution plot for rating
sns.distplot(pstore.like)
#Adding dark background to the graph
plt.style.use("dark_background")
#Change the number of bins
#sns.distplot(pstore.like, bins=20, kde = False, color='red')
#Plotting the scatter plot
#plt.scatter(pstore.share, pstore.like)
# Plotting the same thing now using a jointplot
sns.pairplot(pstore[['like', 'share', 'comment','Total Interactions']])
plt.show()
```



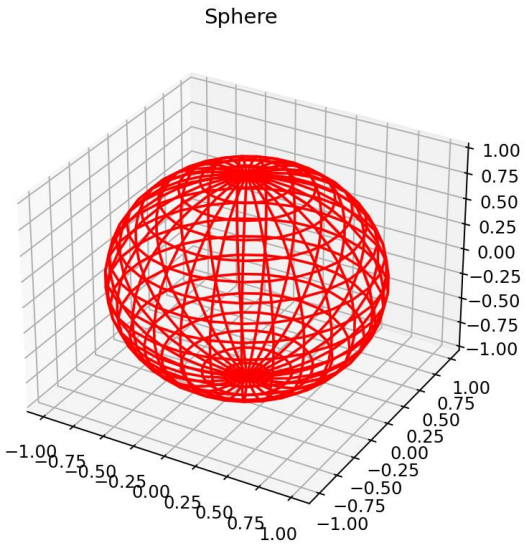

400
350
300
250
200
150
100
50
0
0
1000
2000
3000
4000
5000
like


5000
4000
3000
2000
1000
0
0
100
200
300
400
500
600
700
800



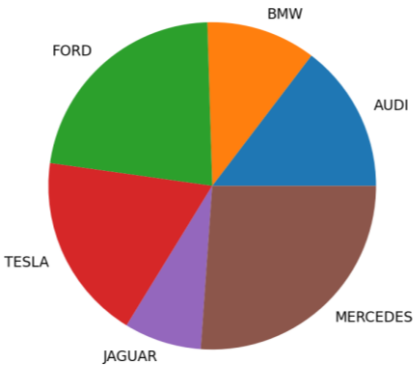
 # 3D plot

```
import matplotlib.pyplot as plt
import numpy as np
fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')
u, v = np.mgrid[0:2 * np.pi:30j, 0:np.pi:20j]
x = np.cos(u) * np.sin(v)
y = np.sin(u) * np.sin(v)
z = np.cos(v)
ax.plot_wireframe(x, y, z, color="red")
ax.set_title("Sphere")
plt.show()
```



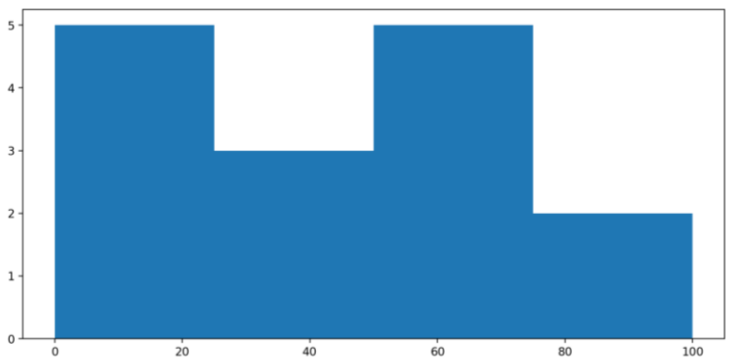
#PI Plot

```
import matplotlib.pyplot as plt
import numpy as np
# Creating dataset
cars = ['AUDI', 'BMW', 'FORD', 'TESLA', 'JAGUAR', 'MERCEDES']
data = [23, 17, 35, 29, 12, 41]
# Creating plot
fig = plt.figure(figsize =(10, 7))
plt.pie(data, labels = cars)
plt.show()
```



#Histrogram Plot

```
import matplotlib.pyplot as plt
import numpy as np
# Creating dataset
a = np.array([22, 87, 5, 43, 56, 73, 55, 54, 11, 20, 51, 5, 79, 31, 27])
# Creating histogram
fig, ax = plt.subplots(figsize =(10, 7))
ax.hist(a, bins = [0, 25, 50, 75, 100])
plt.show()
```



 # Box Plot

```
import matplotlib.pyplot as plt
import numpy as np
# Creating dataset
np.random.seed(10)
data = np.random.normal(100, 20, 200)
 fig = plt.figure(figsize =(10, 7))
 # Creating plot
plt.boxplot(data)
# show plot
plt.show()
```