# Statistics Practical Implementations in Python

**By : Ambarish Singh**

## Measure of Central Tendency

1. **Mean**
2. **Median**
3. **Mode**

In [5]:

```
## DataSet lists For Practices:-

ages = [23,24,32,45,12,43,67,45,32,56,32,120]
```

In [6]:

```
## Finding Mean & Median via Numpy Library.

import numpy as np
print(np.mean(ages))
print(np.median(ages))
```

```
44.25
37.5
```

In [7]:

```
## We can also find Mean & Median via Statistics Library.

import statistics
print(statistics.mean(ages))
print(statistics.median(ages))
```

```
44.25
37.5
```

In [8]:

```
## Findind Mode via Statistics Library.

import statistics
statistics.mode(ages)
```

Out[8]:

```
32
```

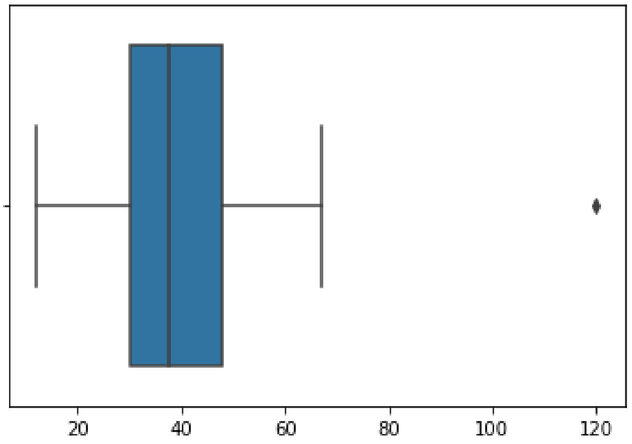
In [10]:

```
## For Finding Outlier, We Basically Use Boxplox.

import seaborn as sns
sns.boxplot(ages)
```

```
C:\Users\Lenovo\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pas
s the following variable as a keyword arg: x. From version 0.12, the only valid positiona
l argument will be `data`, and passing other arguments without an explicit keyword will r
esult in an error or misinterpretation.
  warnings.warn(
```

Out[10]:

```
<AxesSubplot:>
```




# 5 Number Summary

In [12]:

```
# To Find q1 And q3 in Dataset...
## Where, q1 is 25 Percentile And q3 is 75 percentile.

import numpy as np
q1, q3 = np.percentile(ages,[25,75])
```

In [23]:

```
## Here we are finding the value of q1 and q3.

print( f"q1 is {q1} and q3 is {q3}")
```

```
q1 is 30.0 and q3 is 47.75
```

In [24]:

```
# To check outlier [Lower Fence - Higher Fence]
# Where IQR is Inter Quantile Range.

IQR = q3 -q1
lower_fence = q1 - 1.5*(IQR)
higher_fence = q3 + 1.5*(IQR)
print(f"Lower Fence is {lower_fence} and Higher Fence is {higher_fence}")
```

```
Lower Fence is 3.375 and Higher Fence is 74.375
```

# Measure Of Dispersion

### 1. Variance

### 2. Standard Deviation

In [26]:

```
## In Statistics Library Variance are Comes via sample Variances Formuala
statistics.variance(ages)
```

Out[26]:

```
795.2954545454545
```

In [27]:

```
np.var(ages, axis = 0)   ### In Numpy Library Variance are comes via Population Variances
Formula.
```

Out[27]:

```
729.0208333333334
```

In [28]:

```python
## We are Approaching Manual Step to Find Population Variance

def variance(data):
    n = len(ages)

    ## mean of the data
    mean = sum(data)/n

    ## Variance
    deviation = [(x - mean) ** 2 for x in data]
    variance = sum(deviation)/n    ## here we are using Population Variance formula
    return variance
```

In [39]:

```python
variance(ages)
```

Out[39]:

```
729.0208333333334
```

In [46]:

```python
## We are Approaching Manual Step to Find Sample Variance

def variance(data):
    n = len(ages)

    ## mean of the data
    mean = sum(data)/n

    ## Variance
    deviation = [(x - mean) ** 2 for x in data]
    variance = sum(deviation)/(n-1)    ## here we are using Sample Variance Formula.
    return variance
```

In [47]:

```python
variance(ages)
```

Out[47]:

```
795.2954545454545
```

In [50]:

```python
# Now, We are doing Manual Approaching for Finding Variances with Degree of Freedom (dof)
method.
## Where dof = 0 means it is Population variances.
## Where dof = 1 means it is sample variances.


def variance(data, dof = 0):
    n = len(ages)

    ## mean of the data
    mean = sum(data)/n

    ## Variance
    deviation = [(x - mean) ** 2 for x in data]
    variance = sum(deviation)/(n- dof)    ## here we are using Sample Variance Formula.
    return variance
```

In [53]:

```python
## here we are finding Population Variance with dof =0
variance(ages,dof =0)
```

Out[53]:

```
729.0208333333334
```

In [54]:

```
## here we are finding Sample Variance with dof =1.
variance(ages,dof =1)
```

Out[54]:

```
795.2954545454545
```

In [57]:

```
## In Statistics Library , we are finding sample Variance as below :-
statistics.variance(ages)
```

Out[57]:

```
795.2954545454545
```

In [58]:

```
## In Statistics Library,  we are finding Population Variance as below :-
statistics.pvariance(ages)
```

Out[58]:

```
729.0208333333334
```

In [59]:

```
## For Finding Standard Deviation in statistics we use Maath Library.

import math
math.sqrt(statistics.pvariance(ages))
```

Out[59]:

```
27.000385799712813
```

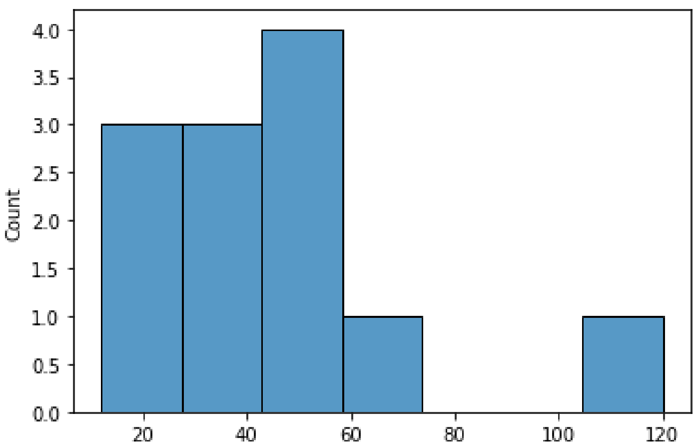
# Histograms And PDF

In [60]:

```
import seaborn as sns
sns.histplot(ages)
```

Out[60]:

```
<AxesSubplot:ylabel='Count'>
```



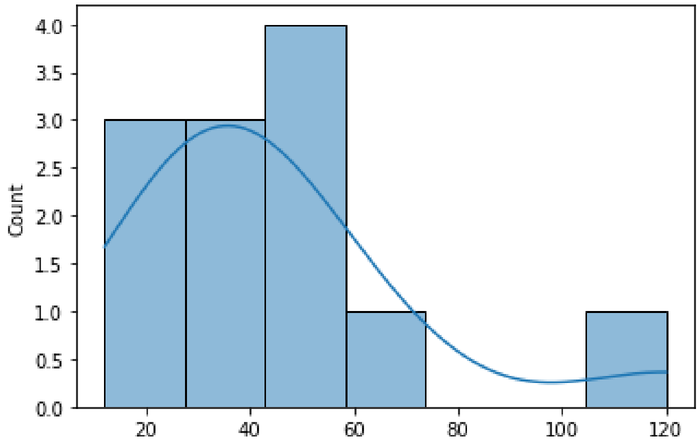

In [61]:

```
import seaborn as sns
```

```
sns.histplot(ages, kde = True)    ## Where Kde = kernal density Estimators
```

Out[61]:

<AxesSubplot:ylabel='Count'>




In [62]:

```
## Now, we check practical with some dataset:-

df = sns.load_dataset('iris')
```

In [64]:

```
## head() basically used for top 5 data present in any dataset
df.head()
```

Out[64]:

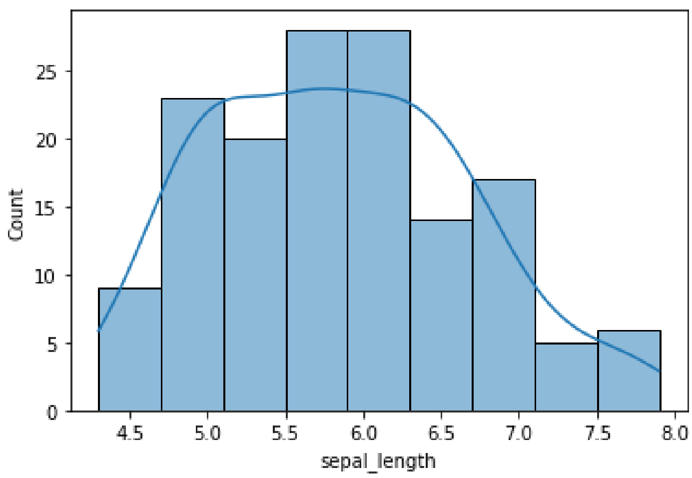
| | sepal_length | sepal_width | petal_length | petal_width | species |
|---|---|---|---|---|---|
| **0** | 5.1 | 3.5 | 1.4 | 0.2 | setosa |
| **1** | 4.9 | 3.0 | 1.4 | 0.2 | setosa |
| **2** | 4.7 | 3.2 | 1.3 | 0.2 | setosa |
| **3** | 4.6 | 3.1 | 1.5 | 0.2 | setosa |
| **4** | 5.0 | 3.6 | 1.4 | 0.2 | setosa |

In [65]:

```
sns.histplot(df['sepal_length'], kde = True)
```

Out[65]:

<AxesSubplot:xlabel='sepal_length', ylabel='Count'>



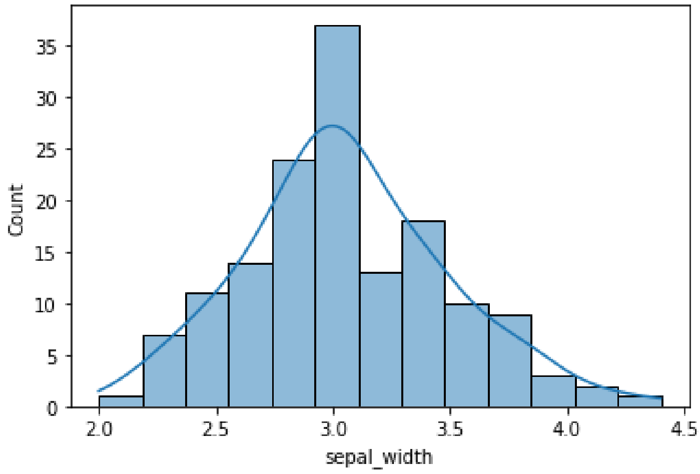

In [66]:

```
sns.histplot(df['sepal_width'], kde = True)
```

```
sns.histplot(df['sepal_width'], kde = True)
```

Out[66]:

```
<AxesSubplot:xlabel='sepal_width', ylabel='Count'>
```

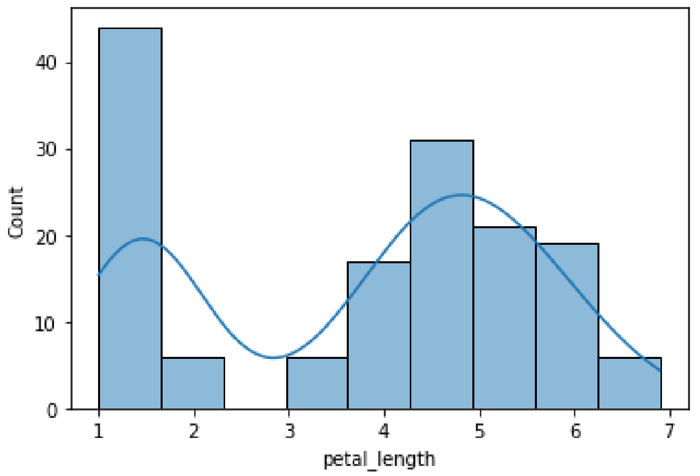



In [67]:

```
sns.histplot(df['petal_length'], kde = True)
```

Out[67]:

```
<AxesSubplot:xlabel='petal_length', ylabel='Count'>
```

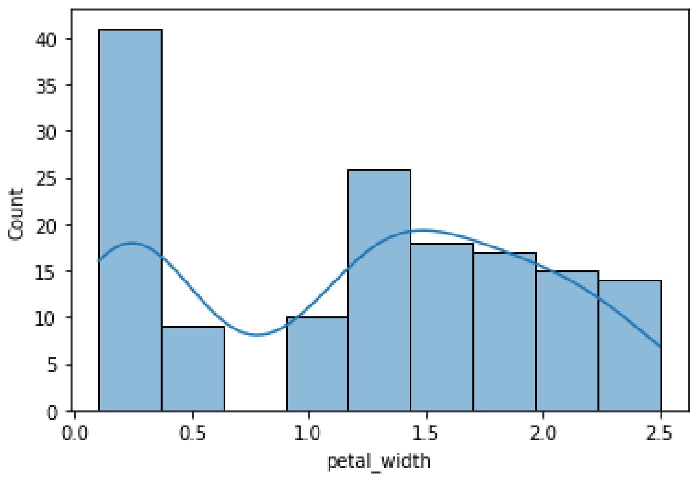



In [68]:

```
sns.histplot(df['petal_width'], kde = True)
```

Out[68]:

```
<AxesSubplot:xlabel='petal_width', ylabel='Count'>
```




In [69]:

```
## How to Create a Normal Distributed Data with the help of Numpy
```

```
## How to Create a Normal Distributed Data with the help of Numpy.

s = np.random.normal(0.5,0.2,1000)
```

In [70]:

```
s
```

Out[70]:

```
array([ 0.45726166,  0.42797827,  0.75504363,  0.37780416,  0.51916142,
        0.76873836,  0.45200607,  0.64708364,  0.08185767,  0.11583081,
        0.50753477,  0.66183224,  0.46699945,  0.69853326,  0.54687729,
        0.20558024,  0.69618226,  0.74501161,  0.68362094,  0.84726852,
        0.62365809,  0.29367494,  0.29087443,  0.25961136,  0.59982994,
        0.58404715,  0.07635905,  0.61173964,  0.57805451,  0.16147041,
        0.60472929,  0.45814231,  0.24068949,  0.42610612,  0.43366094,
        0.35888597,  0.49800564,  0.46136094,  0.4250343 ,  0.2569341 ,
        0.60262731,  0.64904908,  0.23951696,  0.30754576,  0.78876669,
        0.68980504,  0.83161402,  0.75981102,  0.70135736,  0.89598111,
        0.43630588,  0.43374143,  0.35797888,  0.50138788,  0.17170505,
        0.95620988,  0.23469047, -0.12545276,  0.34873769,  0.53418545,
        0.53603763,  0.73325963,  0.68096595,  0.37956325,  0.6232914 ,
        0.48133676,  0.29279513,  0.15941578,  0.7000451 ,  0.52682546,
        0.1680029 ,  0.71680862,  0.43059225,  0.36206931,  0.23680548,
        0.41247866,  0.7756392 ,  0.59262678,  0.41999462,  0.71684884,
        0.56079628,  0.21681823,  0.63965053,  0.62047284,  0.7945499 ,
        0.76688525,  0.42013411,  0.5593568 ,  0.26386038,  0.49427892,
        0.48854792,  0.38118932,  0.63040884,  0.79604236,  0.80611587,
        0.32924939,  0.48974497,  0.58068612,  0.6926996 ,  0.60553884,
        0.52284011,  0.82669614,  0.20220034,  0.43390101,  0.4593637 ,
        0.64705727,  0.75886679,  0.47198353,  0.22267467,  0.3842157 ,
        0.58348336,  0.63861711,  0.74760243,  0.61321714,  0.4890482 ,
        0.72824371,  0.87522741,  0.60397076,  0.79227398,  0.42988857,
        0.43886364,  0.63023817,  0.66255897,  0.61171981,  0.46950599,
        0.82462903,  0.17285309,  0.29619585,  0.61931531,  0.30509549,
        0.17481254,  0.48280156,  0.37149723,  0.39146785,  0.60564877,
        0.53765102,  0.40739161,  0.70018128,  0.56037101,  0.2641016 ,
        0.37658519,  0.59695179,  0.65674656,  0.33691004,  0.53265299,
        0.59387941,  0.66778153,  0.39542271,  0.60768524,  0.5690194 ,
        0.69843035,  0.49079137,  0.9494366 ,  0.3732435 ,  0.23505335,
        0.32709434,  0.54922839,  0.59338006,  0.38124549,  0.38438634,
        0.70469774,  0.36692305,  0.40849928,  0.69296505,  0.64822485,
        0.81820953,  0.41941489,  0.62927117,  0.62546858,  0.92944833,
        0.2423439 ,  0.48274694,  0.41022519,  0.69774058,  0.28123422,
        0.63446217,  0.32850254,  0.37437141,  0.60264969,  0.39987539,
        0.39063701,  0.58635966,  0.44483253,  0.92231542,  0.57439629,
        0.50884667,  0.67486172,  0.90696573,  0.42799385,  0.63938025,
        0.51423494,  0.70894329,  0.69791395,  0.42100306,  0.61264409,
        0.40217151,  0.57629807,  0.41584624,  0.54304749,  0.30088105,
        0.34641691,  0.5255447 ,  0.5211275 ,  0.56921008,  0.72110052,
        0.50124943,  0.32314278,  0.84821845,  0.88243608,  0.79082626,
        0.47014612,  0.96498254,  0.64454466,  0.17409724,  0.52477067,
        0.65652443,  0.29372824,  0.20997974,  0.43358239,  0.22157127,
        0.29438012,  0.57413344,  0.62670175,  0.32083842,  0.59687737,
        0.74445507,  0.34001193,  0.42804173,  0.33340416,  0.3959652 ,
        0.54753295,  0.40207344,  0.53757842,  0.49904609,  0.82687131,
        0.73622079,  0.53734312,  0.66819828,  0.31038659,  0.61079559,
        0.406207  ,  0.53552843, -0.18581805,  0.28778545,  0.72013818,
        0.28728252,  0.81129198,  0.43476258,  0.45330989,  0.48860134,
        0.42437227,  0.45881206,  0.55486488,  0.31309368,  0.31463242,
        0.62032315,  0.44764595,  0.75971949,  0.6619347 ,  0.15172928,
        0.47935263,  0.75238082,  0.47774365,  0.60721845,  0.31855619,
        0.24189336,  0.18171096,  0.58967887,  0.5721065 ,  0.24600824,
        0.69733486,  0.41626531,  0.80011423,  0.71007183,  0.55889897,
        0.59284253,  0.73456273,  0.55006599,  0.58563767,  0.65984728,
        0.53537491,  0.53156605,  0.72280729,  0.51536714,  0.47878253,
        0.51723776,  0.69412363,  0.66283081,  0.26572335,  0.42548253,
        0.60167539,  0.31813347,  0.61195284,  0.69215215,  0.87757633,
        0.67721957,  0.39576139,  0.52298198,  0.79139559,  0.15780111,
        0.13083963,  0.37008553,  0.37660688,  0.72899004,  0.31291207,
        0.59687839,  0.53423228,  0.55054171,  0.20175649,  0.34080295,
        0.06821121,  0.33729212,  0.56461265,  0.45946193,  0.42139347,
```

```
        0.48395967,  0.4585269 ,  0.33528059,  0.58206864,  0.3069747 ,
       -0.00451497,  0.61661685,  0.25177508,  0.12358593,  0.41023921,
        0.31915325,  0.34587621,  0.55963138,  0.4845251 ,  0.01569421,
        0.47111712,  0.61515794,  0.61875486,  0.3261212 ,  0.54315982,
        0.24802247,  0.61387953,  0.82276864,  0.43993498,  0.54022984,
        0.86799974,  0.61814916,  1.01455743,  0.66730133,  0.64620922,
        0.54554804,  0.45017019,  0.55997709,  0.28908693,  0.45656227,
        0.57409843,  0.42770665,  0.55263364,  0.12992142,  0.30451228,
        0.03450785,  0.32456639,  0.51892863,  0.53409086,  0.59765586,
        0.42826427,  0.48080684,  0.68790104,  0.43494453,  0.43994652,
        0.42172915,  0.79540775,  0.94490614,  0.65373464,  0.68741264,
        0.93214216,  0.31716397,  0.71579003,  0.54617466,  0.32710782,
        0.39403087,  0.52334206,  0.67001582,  0.65764417,  0.60289002,
        0.70507411,  0.76033488,  0.46791031,  0.36817936,  0.1185852 ,
        0.51941318,  0.39305883,  0.51335399,  0.27477258,  0.64811299,
        0.25394928,  0.56688392,  0.63635198,  0.54704953,  0.6910623 ,
        0.14833783,  0.64477271,  0.45574053,  0.88880067,  0.66767082,
        0.25112394,  0.02153016,  0.57367009,  0.795627  ,  0.73986553,
        0.40387524,  0.80166865,  0.55988996,  0.58511279,  0.49135349,
        0.50806795,  0.40379468,  0.56818495,  0.38823301,  0.44827342,
        0.39502335,  0.59414129,  0.58460625,  0.56126899,  0.4208631 ,
        0.29880334,  0.67631377,  0.66505908,  0.4762677 ,  0.56255336,
        0.28724446,  0.40316263,  0.25017549,  0.68639259,  0.36236415,
        0.61068798,  0.68753244,  0.71235491,  0.41050094,  0.53623644,
        0.32658791,  0.43765397,  0.45211485,  0.59165806,  0.52116022,
        0.41230704,  0.47563445,  0.75628385,  0.58140984,  0.48470469,
        0.41481569,  0.59184976,  0.456876  ,  0.7732573 ,  0.25339895,
        0.54521821,  0.72402255,  0.64478528,  0.52380772,  0.51979594,
        0.45410424,  0.84851085,  0.45952791,  0.28816407,  0.22926703,
        0.71325052,  0.85531776,  0.26384954,  0.5571725 ,  0.18718323,
        0.74212579,  0.77176251,  0.28999978,  0.46211085,  0.40893806,
        0.30756895,  0.55824838,  0.28733483,  0.35888505,  0.61902103,
        0.4993297 ,  0.91820964,  0.49058377,  0.50626629,  0.49300829,
        0.26462739,  0.32319369,  0.67195186,  0.43584153,  0.6174328 ,
        0.74840097,  0.91042404,  0.15260317,  0.34958287,  0.25127774,
        0.59481644,  0.77171614,  0.55046731,  0.56307936,  0.60990975,
        0.40774276,  0.71908505,  0.51188495,  0.70287512,  0.67680469,
        0.32363012,  0.68261836,  0.69105226,  0.51297586,  0.49289459,
        0.58629408,  0.23833967,  0.71960123,  0.66760766,  0.20044003,
        0.52736269,  0.48899729,  0.25908394,  0.37472912,  0.92874082,
        0.55232395,  0.54748546,  0.57860042,  0.28137814,  0.52705216,
        0.40316007,  0.83707607,  0.28996811,  0.66759829,  0.32382273,
        0.53603747,  0.35056312,  0.38934547,  0.66223299,  0.20580616,
        0.63207883,  0.28976019,  0.58170124,  0.04047104,  0.60739827,
        0.64769514,  0.71366291,  0.43672415,  0.67857708,  0.47295937,
        0.15451382,  0.40475612,  0.54435478,  0.55944166,  0.34913906,
        0.35010821,  0.59936618,  0.50927028,  0.60978863,  0.47085451,
        0.42017578,  0.5897955 ,  0.60300128,  0.33160796,  0.75514736,
        0.39365752,  0.32108033,  0.89218369,  0.24018929,  0.43035542,
        0.62774258,  0.44533681,  0.89026163,  0.3741731 ,  0.53841622,
        0.49593727,  0.26955868,  0.55565485,  0.80989639,  0.07329071,
        0.64627624,  0.54080245,  0.38403605,  0.81055187,  0.89897553,
        0.41227574,  0.63429069,  0.62991778,  0.6575427 ,  0.77773453,
        0.62622308,  0.48875985,  0.42915461,  0.17149674,  0.47677191,
        0.33975986,  0.26253853,  0.50570792,  0.27960126,  0.03834298,
        0.66324826,  0.51362995,  0.4994596 ,  0.6720061 ,  0.76812237,
        0.753158  ,  1.08129028,  0.47808061,  0.63879161,  0.28362403,
        0.53265446,  0.34711367,  0.37976737,  0.04699729,  0.69915934,
        0.48045068,  0.85413669,  0.16678755,  0.30873203,  0.53842748,
        0.4954186 ,  0.70061697,  0.84626615,  0.14425868,  0.20636242,
        0.19488441,  0.45929427,  0.49720392,  0.37978322,  0.88348728,
        0.50838838,  0.68531834,  0.44406715,  0.49739405,  0.48500314,
        0.60837269,  0.62510742,  0.54616807,  0.58337134,  0.44831011,
        0.86297816,  0.22612841,  0.35540339,  0.65471009,  0.37348811,
        0.78254754,  0.62405378,  0.39916745,  0.55803147,  0.57311802,
        0.53874608,  0.72764572,  0.67354474,  0.8393321 ,  0.33262817,
        0.75326978,  0.20029857,  0.51231891,  0.16060132,  0.94445553,
        0.42318296,  0.82028203,  0.30988918,  0.55012459,  0.16306654,
        0.74259505,  0.26627634,  0.19184513,  0.44422953,  0.21795308,
        0.19187389,  0.52205473,  0.6943523 ,  0.50771795,  0.90842507,
        0.56352208,  0.80372771,  0.29839909,  0.87376001,  0.50740368,
        0.64445538,  0.87769576,  0.70409311,  0.42865279,  0.38760179,
```

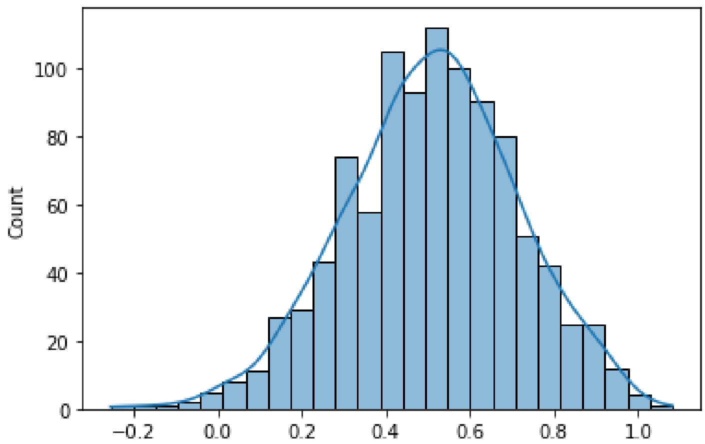
```
        0.29090178,  0.55472098,  0.66282692,  0.30117454,  0.49297507,
        0.23058185,  0.5199069 ,  0.06701473,  0.84423686,  0.58916534,
        0.42863179,  0.97467306,  0.25188355,  0.41363461,  0.19042934,
        0.50948434,  0.2801032 ,  0.65500205,  0.45845769,  0.4928895 ,
        0.69861493,  0.39686519,  0.61688589,  0.7057442 ,  0.47549625,
        0.14538352, -0.25435948,  0.44144478,  0.48221511,  0.53578563,
        0.55052695,  0.53645793,  0.25282753,  0.74240781,  0.67776816,
        0.72830292,  0.48727339,  0.5488339 ,  0.47854495,  0.47073195,
        0.55796367, -0.06423784,  0.44359354,  0.43829667,  0.3538747 ,
        0.38460967,  0.90764915,  0.22822923,  0.46952592,  0.88914693,
        0.7884336 ,  0.83611853,  0.53968519,  0.12733713,  0.39881805,
        0.42685306,  0.86310412,  0.68098652,  0.5797865 ,  0.50641256,
        0.54072448,  0.81796241,  0.3290825 ,  0.45010514,  0.6765732 ,
        0.55713213,  0.54275691,  0.31178742,  0.33079072,  0.50445903,
        0.31957879,  0.45248178,  0.76465262,  0.53065387,  0.32587887,
        0.43749827,  0.99341597,  0.7175874 ,  0.28396952,  0.41981466,
        0.41857073,  0.55611789,  0.57307487,  0.56814482,  0.5204143 ,
        0.45161587,  0.24623824,  0.55498252,  0.50714628,  0.41271604,
        0.71213027,  0.60976532,  0.5616449 ,  0.35267834,  0.55285943,
        0.49490099,  0.56840242,  0.59150322,  0.61678928,  0.6254859 ,
        0.71302068,  0.1472909 ,  0.41885378,  0.6474539 ,  0.50294615,
        0.43606891,  0.34495812,  0.33774896,  0.51615324,  0.34597114,
       -0.0146788 ,  0.42586584,  0.56765358,  0.61130459,  0.40283864,
        0.0965338 ,  0.70360779,  0.76710491,  0.45383436,  0.36092769,
        0.44182079,  0.48837328,  0.68405513,  0.60423802,  0.67262492,
        0.61907778,  0.20927029,  0.48802229,  0.30276425,  0.54231656,
        0.80537964,  0.53406654,  0.23963371,  0.49468925,  0.25691946,
        0.35329966,  0.44931817,  0.4119401 ,  0.15386038,  0.70731443,
        0.37527677,  0.45499002,  0.78407329,  0.39211911,  0.3065562 ,
        0.34726894,  0.53401561,  0.26619282,  0.70089247,  0.64565637,
        0.68417633,  0.72186676,  0.80065798,  0.44444562,  0.61701141,
        0.16894704,  0.17257751,  0.38819994,  0.59201962,  0.59722766,
        0.47303279,  0.42360756,  0.32392958,  0.44678644,  0.76504954,
        0.51845479,  0.46145523,  0.51636433,  0.51944255,  0.1302311 ,
        0.53344601,  0.567473  ,  0.24565205,  0.54368153,  0.74108519,
        0.41550848,  0.22834207,  0.68341198,  0.3210783 ,  0.37372921,
        0.92482301,  0.61970749,  0.5288567 ,  0.08224764,  0.63566643,
        0.56574055,  0.78862135,  0.32540628,  0.7000287 ,  0.64332763,
        0.72565973,  0.51750974,  0.92514787,  0.42195915,  0.58145221,
        0.57597288,  0.32919596,  0.49562278,  0.40087892,  0.32792153,
        0.73587396,  0.54492343,  0.51069372,  0.5382638 ,  0.90983008,
        0.63099454,  0.83253068,  0.87195117,  0.44305183,  0.50933596,
        0.17569773,  0.26303945,  0.82060271,  0.3838317 ,  0.52375995,
        0.28247888,  0.54983214,  0.41377738,  0.38999848,  0.78338551,
        0.33365737,  0.65753191,  0.37649211,  0.41805593,  0.60745925,
        0.19953312,  0.83122211,  0.20218839,  0.1336818 ,  0.76726428,
        0.6766193 ,  0.71637762,  0.50737217,  0.43695642,  0.17558831,
        0.58466421,  0.4662823 ,  0.70502881,  0.89232931,  0.80353223,
        0.61682144,  0.32490246,  0.54475546,  0.41214313,  0.56039055,
        0.50115174,  0.48004787,  0.42020044,  0.50768809,  0.70097338,
        0.18120982,  0.48097886,  0.60338321,  0.26968114,  0.88911387,
        0.57986447,  0.65002787,  0.65210429,  0.57719824,  0.40488593,
        0.6122815 ,  0.43014789,  0.28602586,  0.57448923,  0.1983315 ,
        0.19840933,  0.89475295,  0.61418258,  0.54419776,  0.72365591,
        0.33304277,  0.63661063,  0.81324478,  0.68031054,  0.14285249,
        0.43492765,  0.71471008,  0.00171871,  0.10560521,  0.84168057,
        0.32714273,  0.80493678,  0.35829641,  0.41744278,  0.41477778,
        0.76419351,  0.0094769 ,  0.77920207,  0.28459638,  0.01771988,
        0.81149316,  0.07477097,  0.63572231, -0.0192333 ,  0.91736001,
        0.57358388,  0.48784435,  0.41649497,  0.55353294,  0.82255711,
        0.46421916,  0.49862718,  0.52462708,  0.36832615,  0.65956124,
        0.47108847,  0.65550511,  0.29168258,  0.77578801,  0.62602314,
        0.53007176,  0.6189141 ,  0.98419809,  0.78040766,  0.45319585,
       -0.08478568,  0.94572126,  0.54690887,  0.03163383,  0.57363574,
        0.7510476 ,  0.4842176 ,  0.42652436,  0.89163488,  0.3757975 ])
```

In [71]:

```
## Histogram Plot for Normal Distribution dataset
sns.histplot(s, kde = True)
```

Out[71]:

<AxesSubplot:ylabel='Count'>




# Other Distribution

## Log Normal Distribution, Power Law distribution

In [78]:

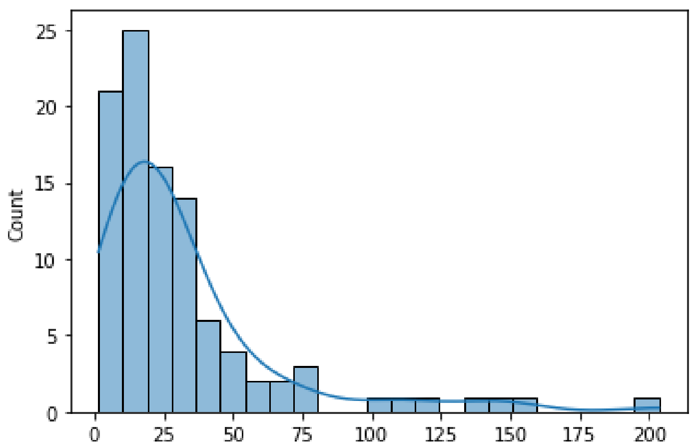
```
## To Find Log Normal Distributions:-

mu, sigma = 3., 1.  ## Where mu is Means and sigma is Standard deviation.

s = np.random.lognormal(mu, sigma, 100)
```

In [79]:

```
## Histogram Plot for Log Normal Distribution dataset

sns.histplot(s, kde = True)
```

Out[79]:

<AxesSubplot:ylabel='Count'>

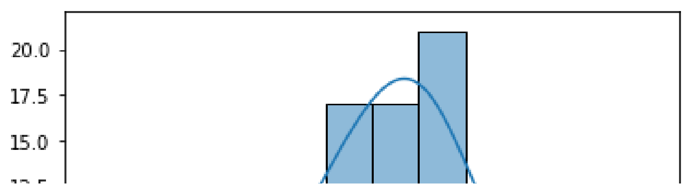


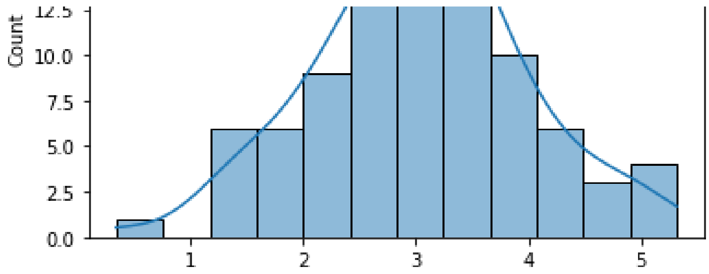

In [80]:

```
sns.histplot(np.log(s), kde = True)
```

Out[80]:

<AxesSubplot:ylabel='Count'>





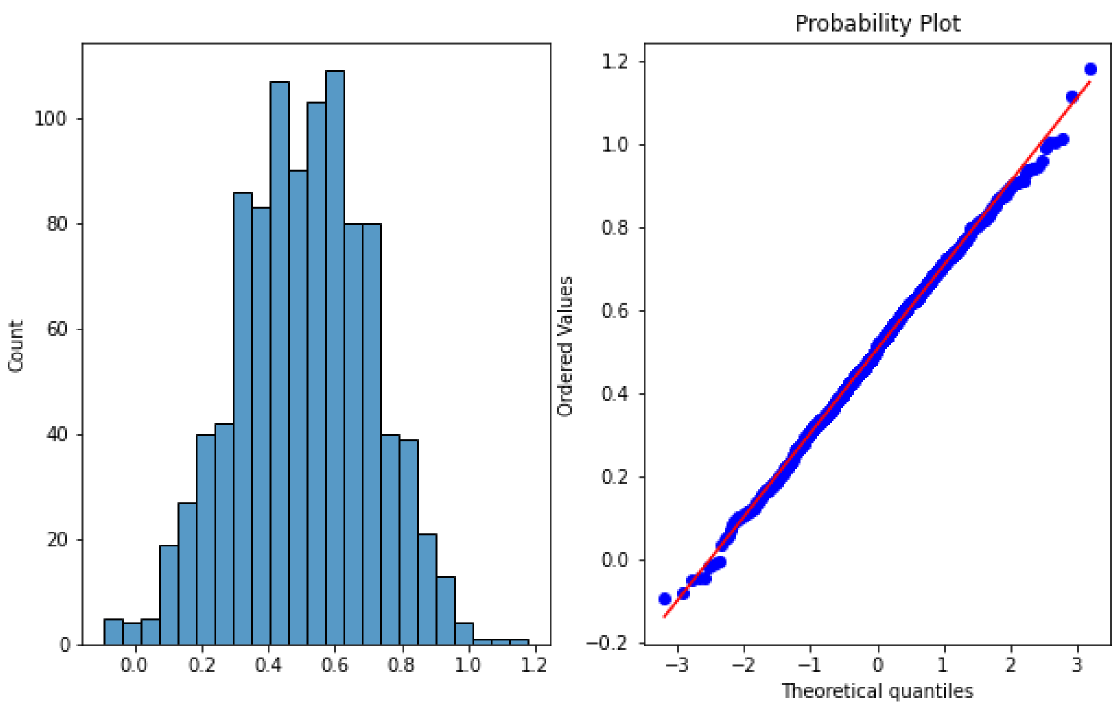

# Check Whether Distribution is Normal Distribution or not.

In [83]:

```
# If you want to check whether feature is Guassian or Normal distribution .
## Q-Q Plot

import matplotlib.pyplot as plt
import scipy.stats as stat
import pylab

def plot_data(sample):
    plt.figure(figsize = (10,6))
    plt.subplot(1,2,1)
    sns.histplot(sample)
    plt.subplot(1,2,2)
    stat.probplot(sample, dist='norm', plot = pylab)
    plt.show()
```

In [84]:

```
## Creating a Normal Distributed data:-

s = np.random.normal(0.5, 0.2, 1000)
plot_data(s)    ## Here, in below Result data comes in a straight line of Probability Plot
, so it is a Normal Distribution.
```

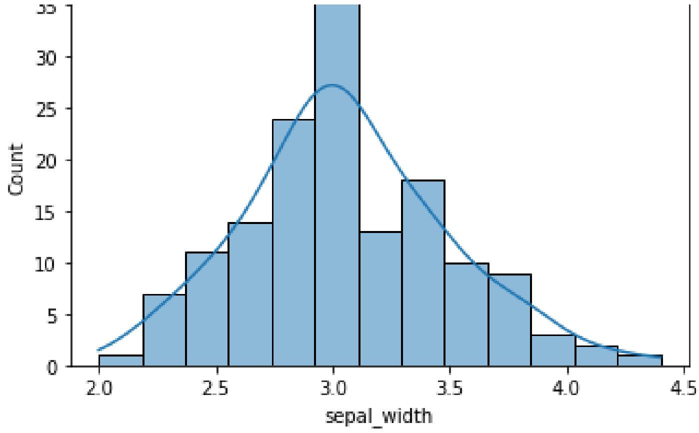



In [86]:

```
sns.histplot(df['sepal_width'], kde= True)
```

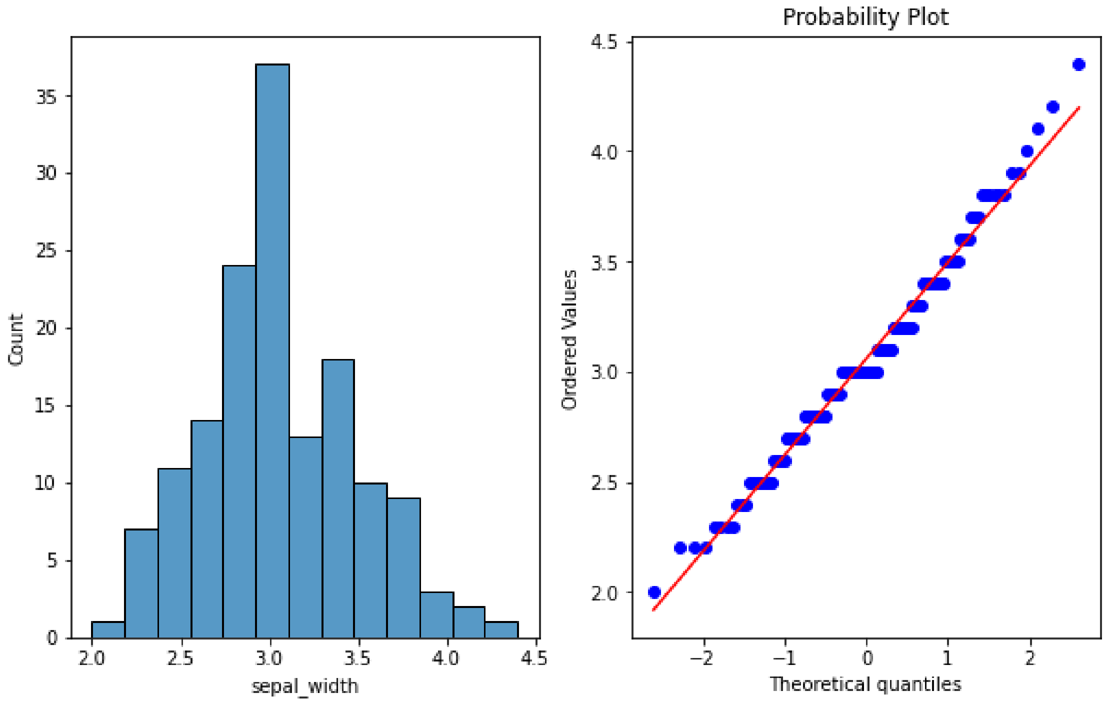
Out[86]:

<AxesSubplot:xlabel='sepal_width', ylabel='Count'>




In [87]:

```
## Here, in below Result data comes in a straight line of Probability Plot, so it is a No
rmal Distribution.

plot_data(df['sepal_width'])
```




# Check Whether Distribution is Log Normal Distribution or not.

In [92]:

```
## To Find Log Normal Distributions:-

mu, sigma = 3., 1.   ## Where mu is Means and sigma is Standard deviation.

sample = np.random.lognormal(mu, sigma, 100)
```

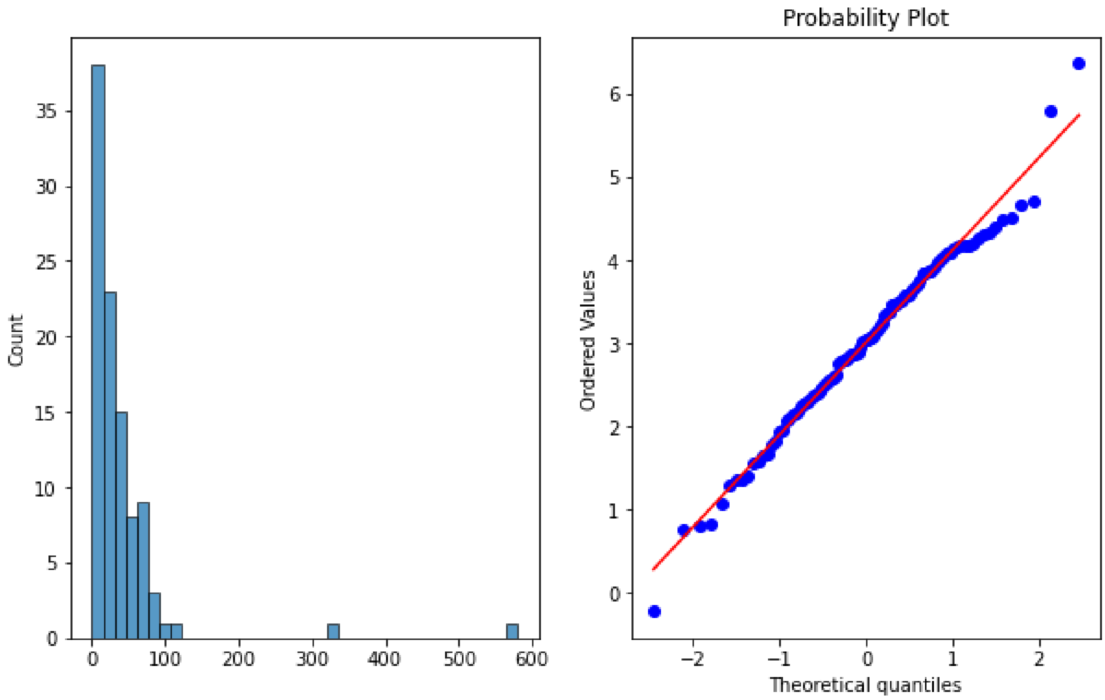
In [93]:

```
## If you want to check whether feature is Log Normal distribution .


import matplotlib.pyplot as plt
import scipy.stats as stat
import pylab

def plot_data(sample):
    plt.figure(figsize = (10,6))
    plt.subplot(1,2,1)
```

```
    sns.histplot(sample)
    plt.subplot(1,2,2)
    stat.probplot(np.log(sample), dist='norm', plot = pylab)
    plt.show
```

In [94]:

```
## Here, in below Result data comes in a straight line of Probability Plot, so it is a  L
og Normal Distribution.

plot_data(sample)
```




# Pearson and Sperman Rank Correlation

In [95]:

```
## For Practicing we are taking Tips Dataset:-
df = sns.load_dataset('tips')
```

In [96]:

```
df.head()
```

Out[96]:

| | total_bill | tip | sex | smoker | day | time | size |
|---|---|---|---|---|---|---|---|
| **0** | 16.99 | 1.01 | Female | No | Sun | Dinner | 2 |
| **1** | 10.34 | 1.66 | Male | No | Sun | Dinner | 3 |
| **2** | 21.01 | 3.50 | Male | No | Sun | Dinner | 3 |
| **3** | 23.68 | 3.31 | Male | No | Sun | Dinner | 2 |
| **4** | 24.59 | 3.61 | Female | No | Sun | Dinner | 4 |

In [97]:

```
import pandas as pd
```

In [98]:

```
## In Pandas Dafault, Corr() takes Pearson Correlation:-
df.corr()
```

Out[98]:

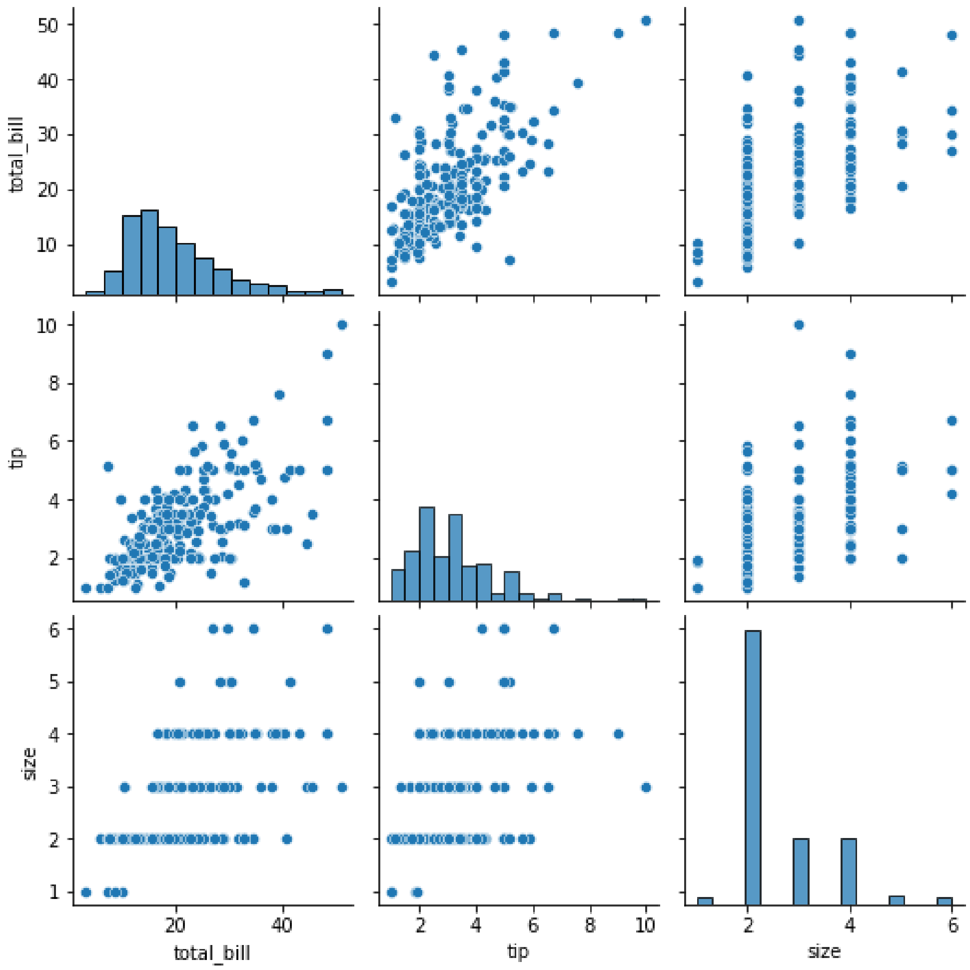
| | total_bill | tip | size |
|---|---|---|---|
| **total_bill** | 1.000000 | 0.675734 | 0.598315 |
| **tip** | 0.675734 | 1.000000 | 0.489299 |
| **size** | 0.598315 | 0.489299 | 1.000000 |

In [99]:

```
## To Check Pearson Correlation in Diagramatically format, we use.
sns.pairplot(df)
```

Out[99]:

```
<seaborn.axisgrid.PairGrid at 0x1edc25ea550>
```




# Thank You