```
Author   -Shashank Chhoker
Github   -https://github.com/Shashank975
LinkedIn -https://www.linkedin.com/in/shashankchhoker07
Kaggle   -https://www.kaggle.com/shashankchhoker
```

Implementing Statistics Practically

Measure of Central Tendency

1. Mean
2. Median
3. Mode

The terms mean, median, mode, and range describe properties of statistical distributions. In statistics, a distribution is the set of all possible values for terms that represent defined events. The value of a term, when expressed as a variable, is called a random variable.

```
In Machine Learning (and in mathematics) there are often three values that interests us:

Mean - The average value.
Median - The mid point value.
Mode - The most common value.
```

In [1]:
```
#Importing Required libraries
import numpy as np
import pandas as pd
import statistics
import seaborn as sns
import matplotlib.pyplot as plt
import scipy.stats as stat
import pylab
```

In [2]:
```
ages=[23,24,32,45,12,43,67,45,32,56,32,120]
```

In [3]:
```
print(np.mean(ages))
```

```
44.25
```

In [4]:
```
print(np.median(ages))
```

```
37.5
```

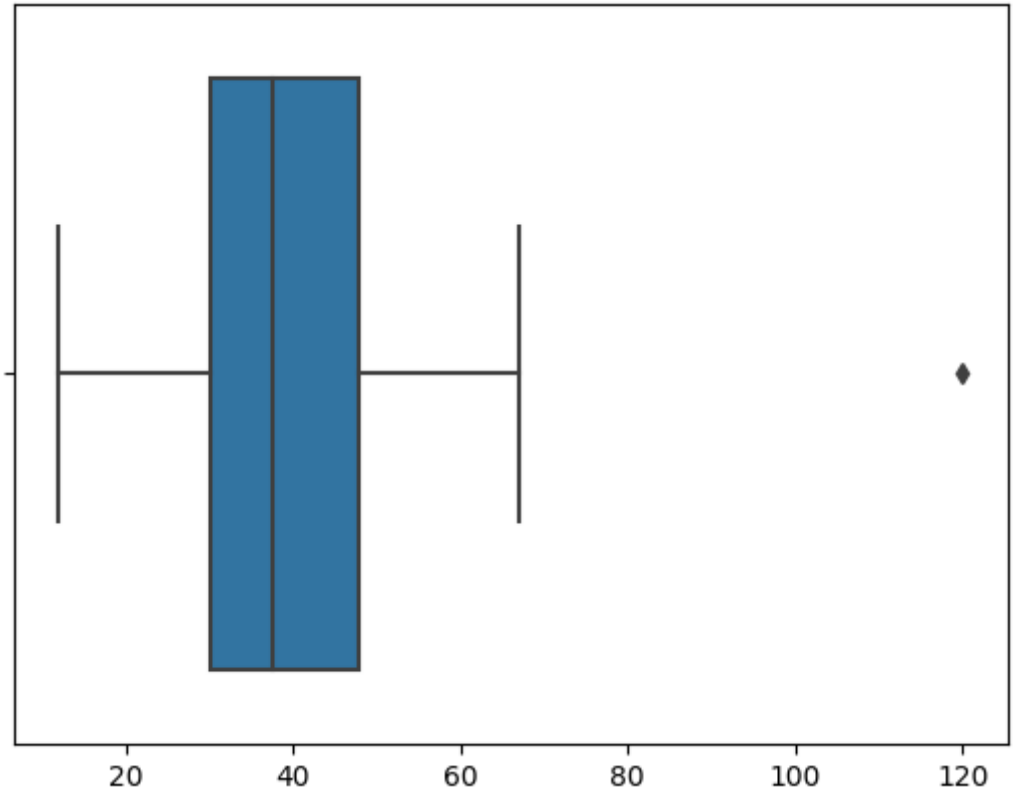
In [5]:
```
print(statistics.mode(ages))
print(statistics.mean(ages))
print(statistics.median(ages))
```

```
32
44.25
37.5
```

```
This code utilizes Seaborn to create a horizontal boxplot showcasing the age distribution and potential outliers
within the sample data.
--> we can easily Detect the outlier here .Which is at the point 120 on the Right side.
```

In [6]:
```
sns.boxplot(x=ages, orient="h")
```

Out[6]: <Axes: >



-:5 Number Summary:-

```
-->Now By Using the 5 Number Summary we are finding the Outliers.
```

In [7]:
```
q1,q3=np.percentile(ages,[25,75])
```

In [8]:
```
print(q1)
```

```
30.0
```

In [9]:
```
print(q3)
```

```
47.75
```

```
-->This code calculates the 25th and 75th percentiles, then computes the interquartile range (IQR). Using the IQR,
it defines lower and upper bounds to identify outliers that fall outside these bounds. The identified outliers are
printed. Replace the data list with your dataset.
```

In [10]:
```
## To Check Outliers [Lower Fence - Higher Fence ]
```

In [11]:
```
IQR=q3-q1
lower_fence=q1-1.5*(IQR)
higher_fence=q3+1.5*(IQR)
print("Lower Fence is =",lower_fence,"And Higher Fence is =",higher_fence)
print(" ")
print ("This tell us about the  outlier .The value which is Lower than 3.375 and value Higher than 74.375 is consider
```

```
Lower Fence is = 3.375 And Higher Fence is = 74.375

This tell us about the  outlier .The value which is Lower than 3.375 and value Higher than 74.375 is consider to be
the outliers.
```

-:Measure of Dispersion:-

```
-->Measures of dispersion, also known as measures of variability or spread, provide information about how the data points in a dataset are spread out from the central tendency (mean, median, etc.). These measures help to quantify the degree of variability or diversity within a dataset
```

```
-->Range: Difference between the maximum and minimum values, sensitive to outliers.

-->Interquartile Range (IQR): Range between the 75th and 25th percentiles, less affected by outliers.

-->Variance: Average of squared differences from the mean, providing overall spread.

-->Standard Deviation: Square root of variance, indicates average deviation from the mean.

-->Coefficient of Variation (CV): Ratio of standard deviation to mean, measures relative variability.

-->Mean Absolute Deviation (MAD): Average of absolute differences from the mean.

-->Percentiles: Values indicating specific data points within a distribution.

These measures help understand the spread and distribution of data points in a dataset.
```

```
1.Variance
2.Standard Deviation
```

In [12]:
```
statistics.variance(ages)
```

Out[12]: 795.2954545454545

In [13]:
```
np.var(ages,axis=0)# It basically uses the Formula of Sample Variance.
```

Out[13]: 729.0208333333334

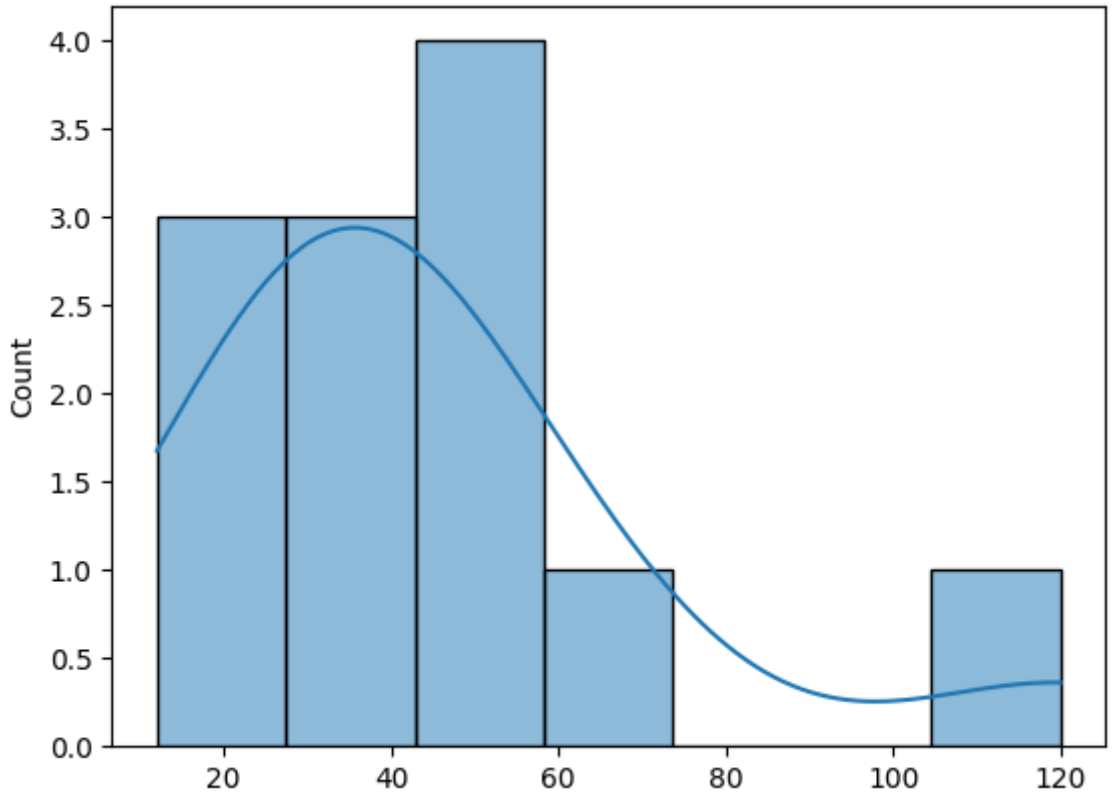
In [14]:
```
statistics.pvariance(ages) # Similer like --> np.var(ages,axis=0)
```

Out[14]: 729.0208333333334

-:Histogram and PDF:-

In [15]:
```
sns.histplot(ages,kde=True)#Probability Dinsity function
```

Out[15]: <Axes: ylabel='Count'>



In [16]:
```
#practice on Dataset
#Here we loading A " IRIS" Dataset
```

In [17]:
```
df=sns.load_dataset('iris')
```

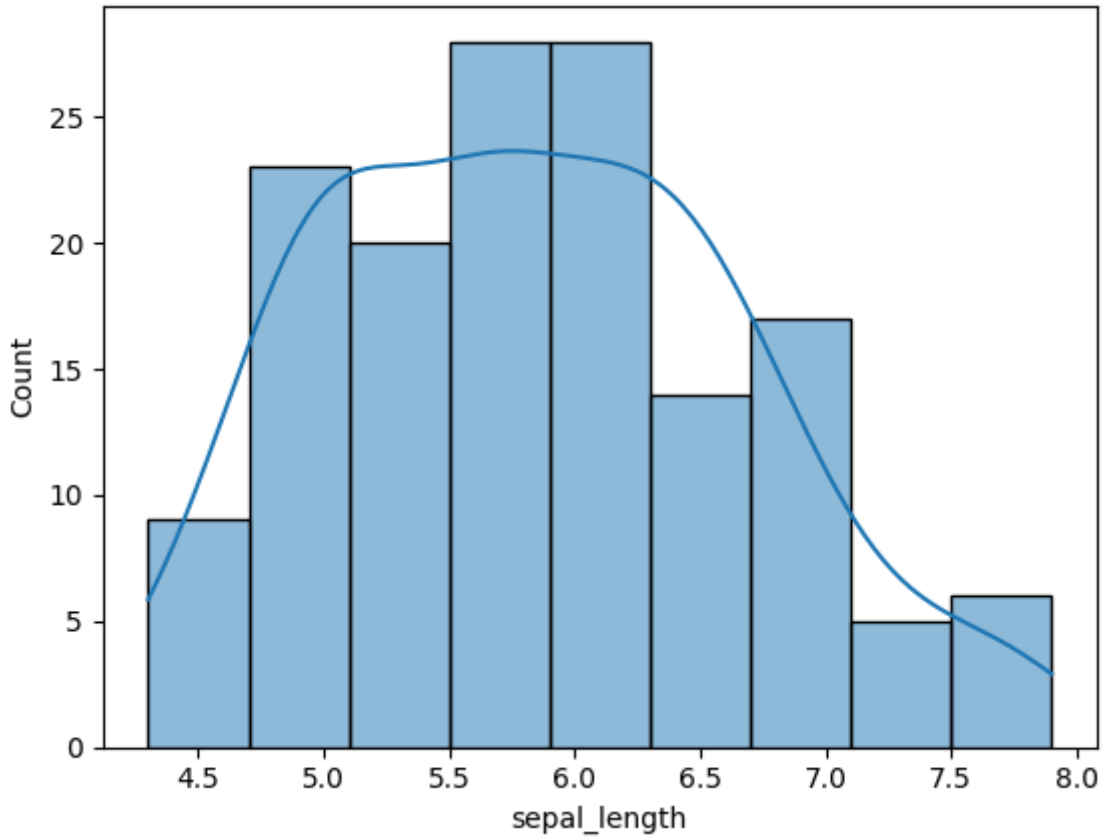
In [18]:
```
df.head(10)
```

Out[18]:

	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
5	5.4	3.9	1.7	0.4	setosa
6	4.6	3.4	1.4	0.3	setosa
7	5.0	3.4	1.5	0.2	setosa
8	4.4	2.9	1.4	0.2	setosa
9	4.9	3.1	1.5	0.1	setosa

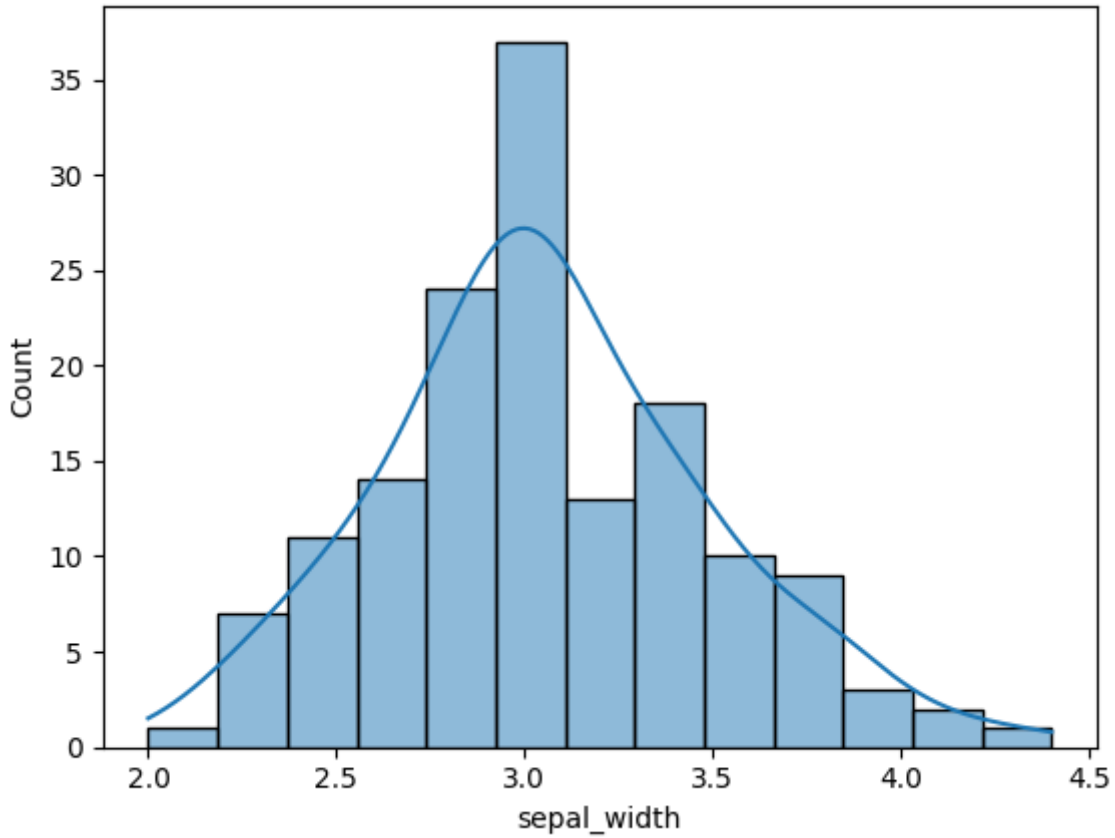
In [19]:
```
sns.histplot(df['sepal_length'],kde=True)
```

Out[19]:
```
<Axes: xlabel='sepal_length', ylabel='Count'>
```



In [20]:
```
sns.histplot(df['sepal_width'],kde=True)# this graph is the fine Example of Normal Distribution .
```

Out[20]:
```
<Axes: xlabel='sepal_width', ylabel='Count'>
```

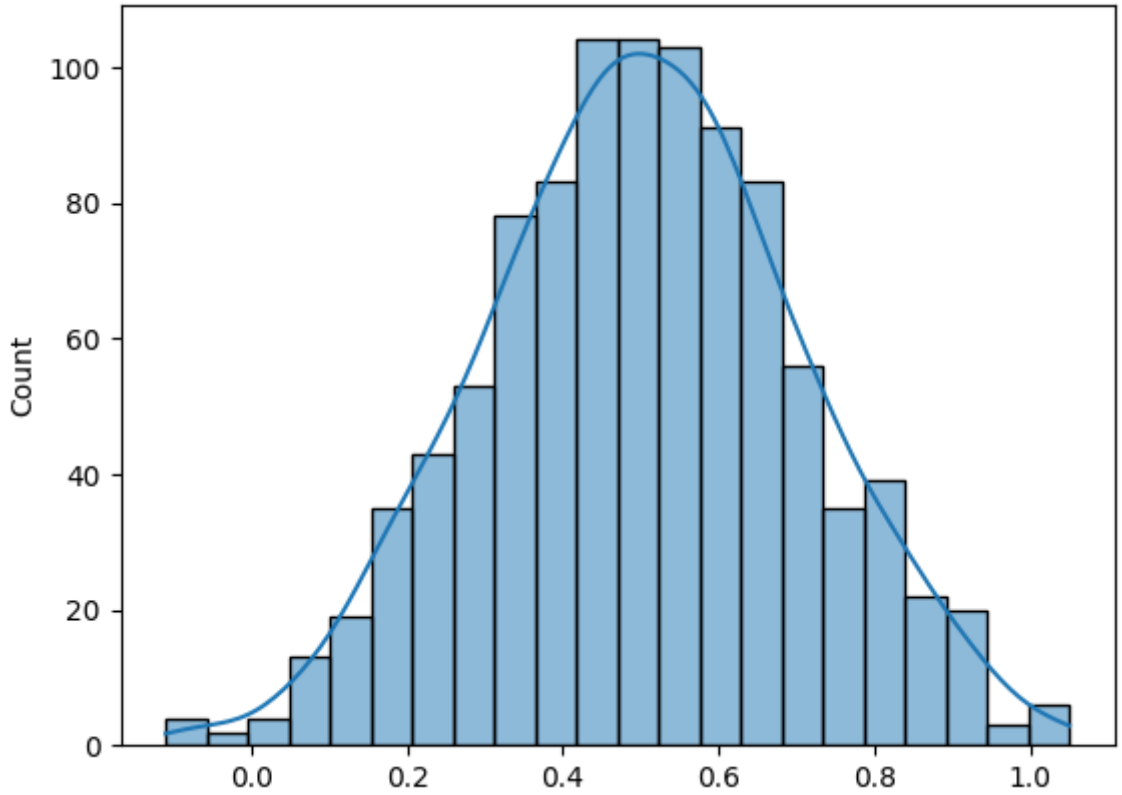


Create a Normal Distributed Dataset

```
-->A normal distribution, also called Gaussian distribution, is a symmetric bell-shaped pattern commonly seen in
various natural and social phenomena. It's described by a mean and standard deviation. Many real-world
measurements like heights and errors follow this pattern, making it crucial for statistical analysis and modeling.
```

In [21]:
```
s=np.random.normal(0.5,0.2,1000)
sns.histplot(s,kde=True)
```

Out[21]: <Axes: ylabel='Count'>

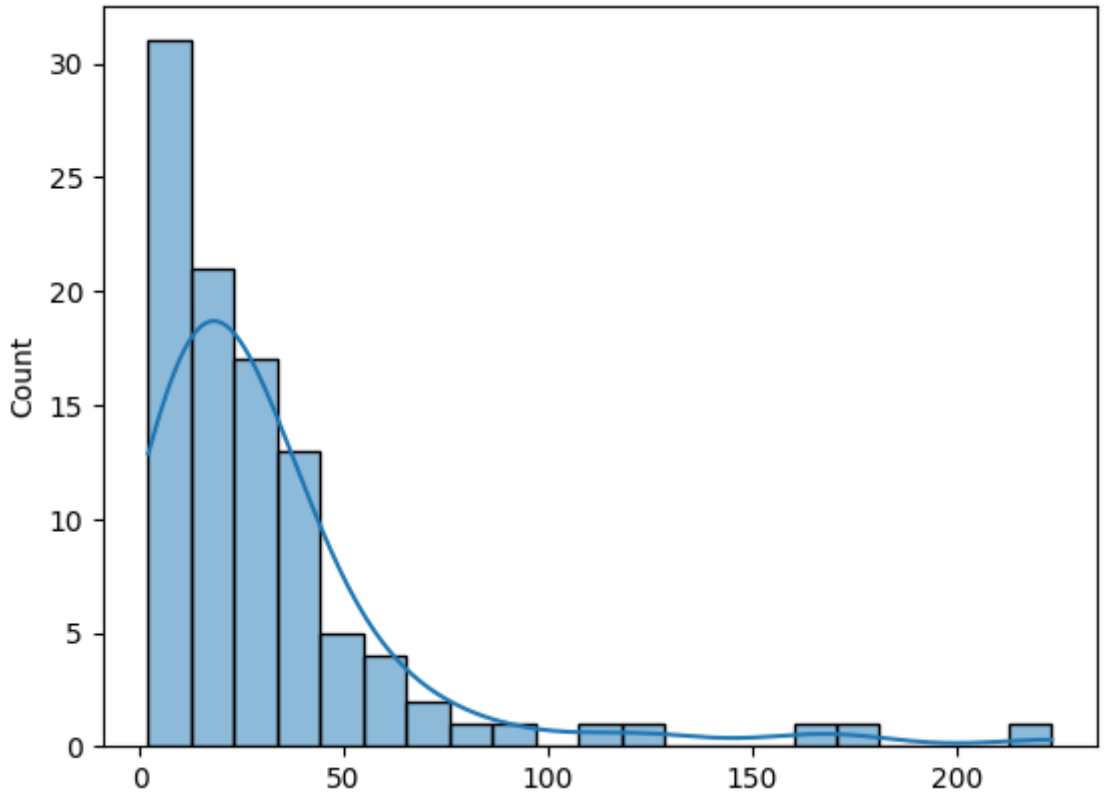


In [22]:
```
#Creating a Log Normal Distribution
```

```
-->A lognormal distribution describes data where the logarithm of the values follows a normal distribution. It's
used for positive, multiplicative quantities like stock prices or income, and results in a right-skewed
distribution.
```

In [23]:
```
mu,sigma =3.,1
s=np.random.lognormal(mu,sigma,100)
sns.histplot(s,kde=True)
```

Out[23]: <Axes: ylabel='Count'>

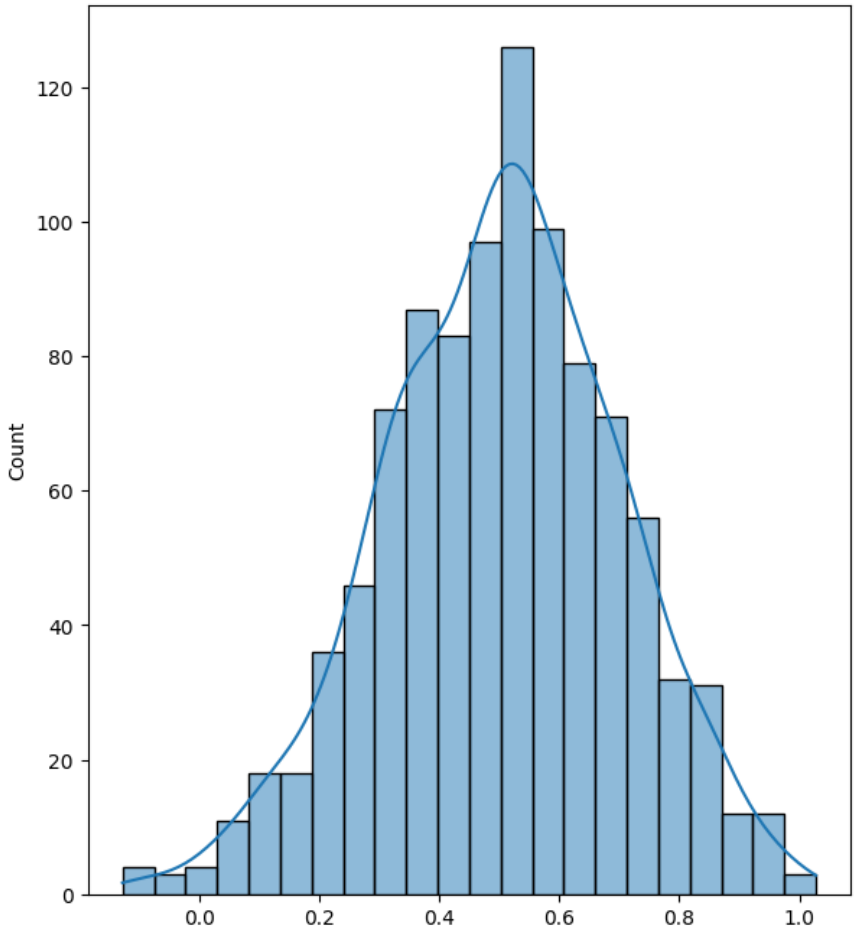


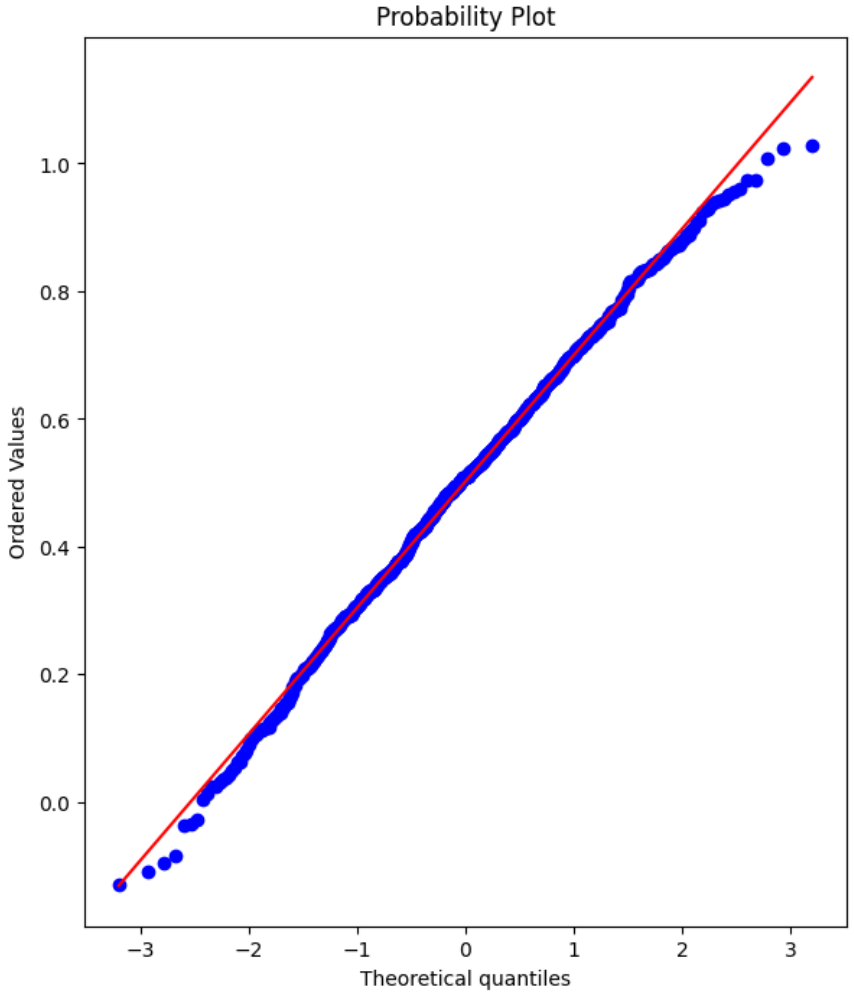
Check Weather a Distribution is Normal or Not?

```
-->A Q-Q plot is a visual tool to check if data follows a specific theoretical distribution, like the normal distribution. If the points on the plot form a straight line, the data is likely normally distributed. Deviations suggest non-normality.
```

In [24]:
```
import matplotlib.pyplot as plt
import scipy.stats as stat
import pylab
def plot_data(sample):
    plt.figure(figsize=(15,8))
    plt.subplot(1,2,1)
    sns.histplot(sample,kde=True)
    plt.subplot(1,2,2)
    stat.probplot(sample,dist='norm',plot=pylab)
    plt.show

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

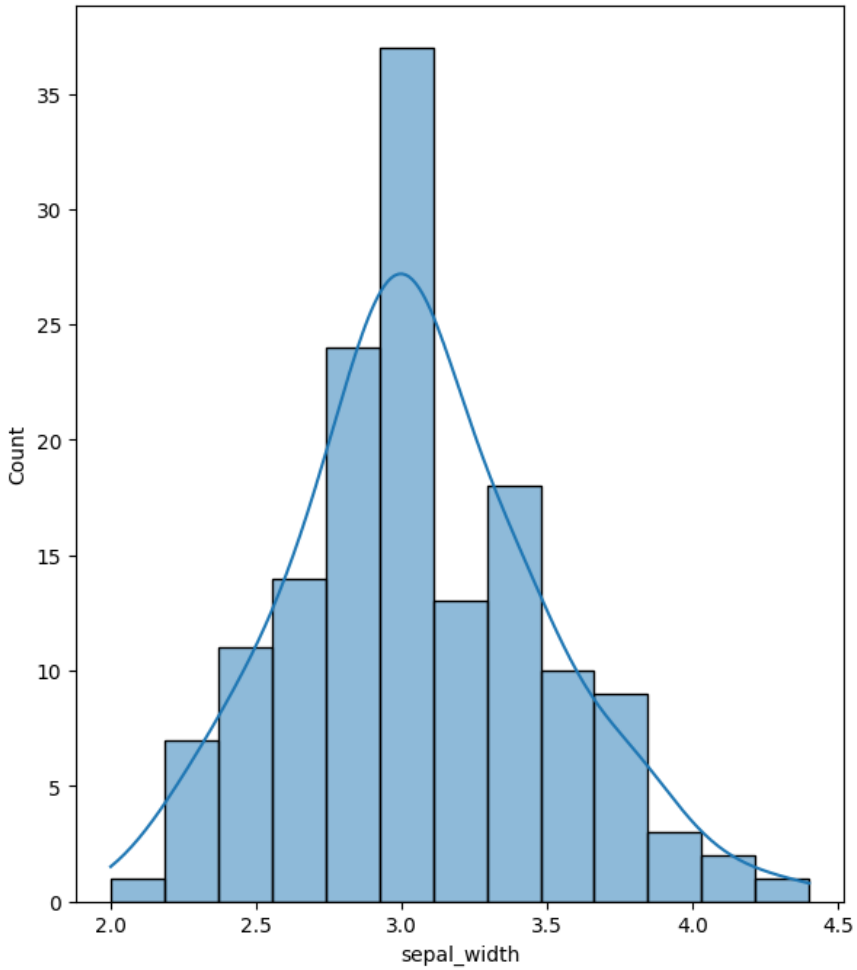


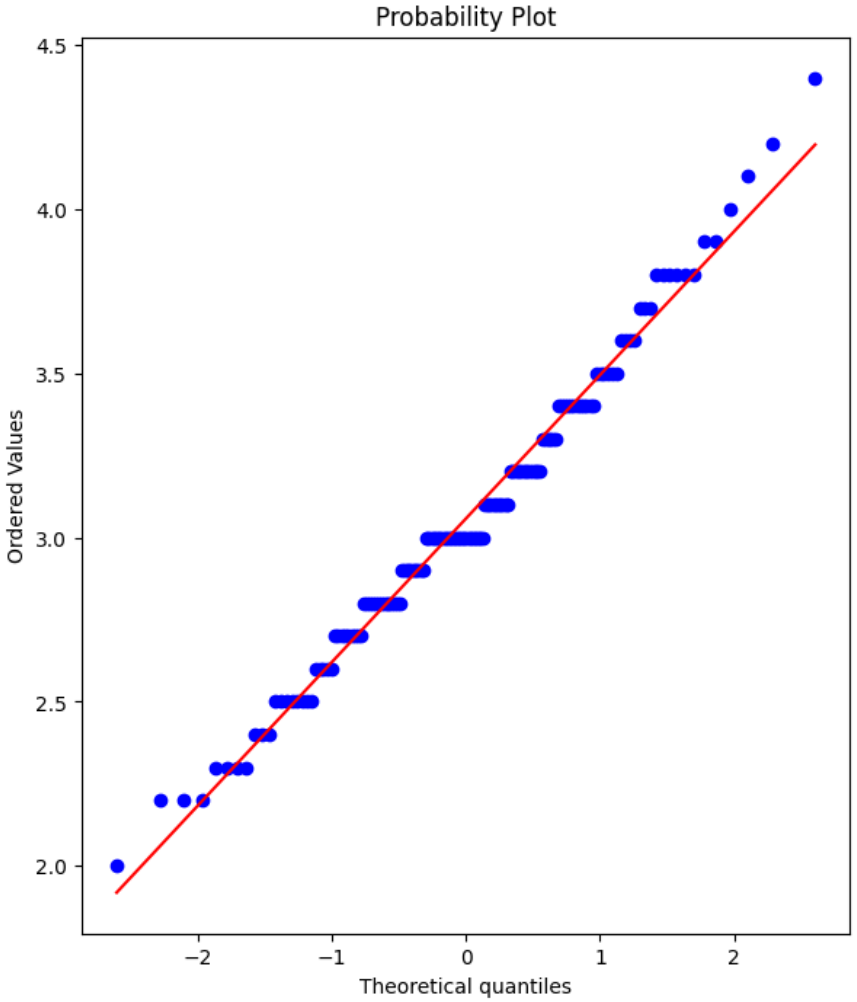


In [25]:
```
# we can even check it for the Iris dataset
```

In [26]:
```
def plot_data(sample):
    plt.figure(figsize=(15,8))
    plt.subplot(1,2,1)
    sns.histplot(sample,kde=True)
    plt.subplot(1,2,2)
    stat.probplot(sample,dist='norm',plot=pylab)
    plt.show

#s=np.random.normal(0.5,0.2,1000)
plot_data(df['sepal_width'])
```



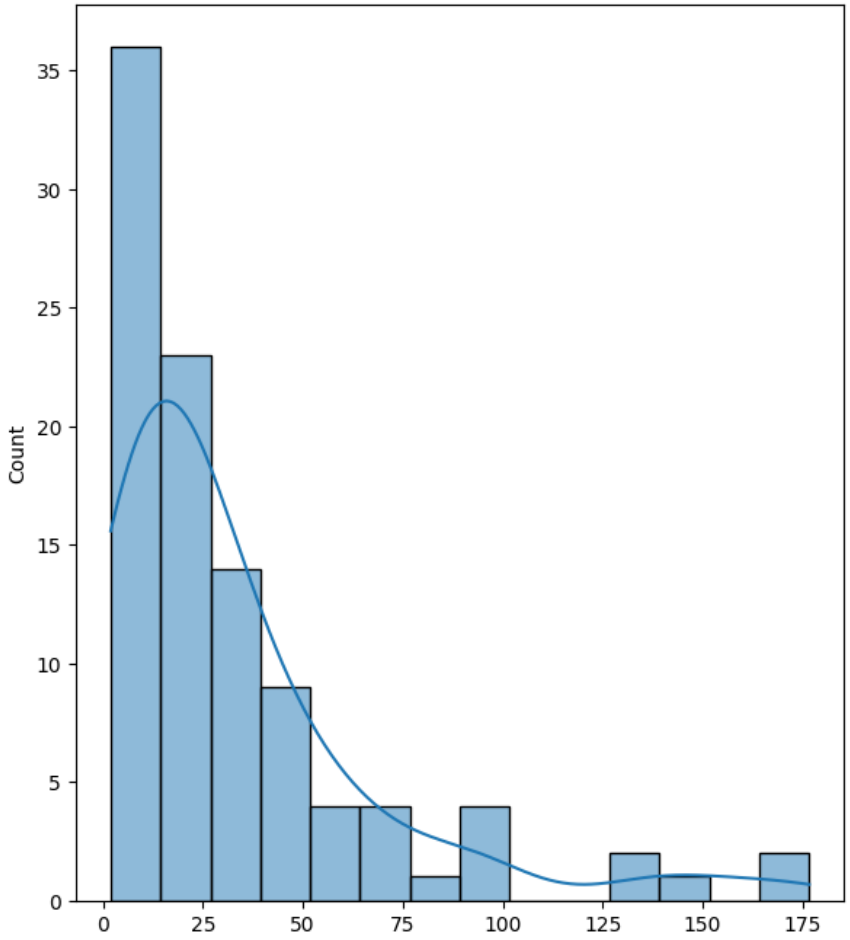


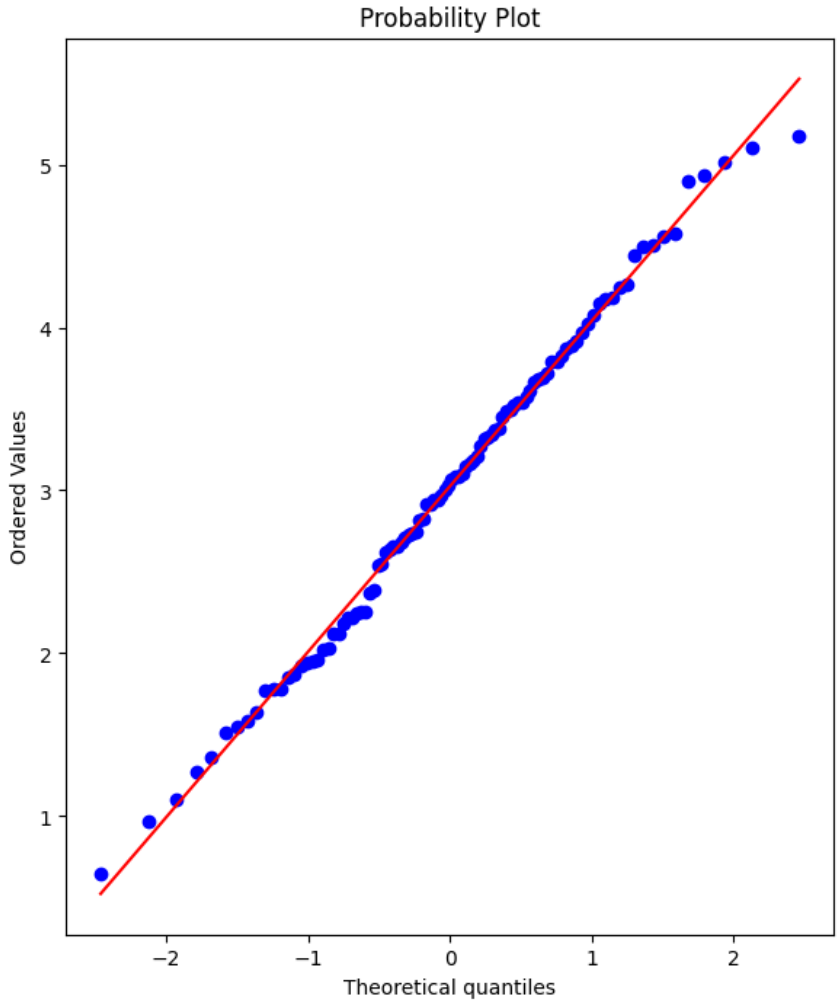
In [27]:
```
# In the same we can Check for the Lognormal Distributed data.
```

In [28]:
```
mu,sigma =3.,1
sample=np.random.lognormal(mu,sigma,100)
```

In [29]:
```
def plot_data_log(sample):
    plt.figure(figsize=(15,8))
    plt.subplot(1,2,1)
    sns.histplot(sample,kde=True)
    plt.subplot(1,2,2)
    stat.probplot(np.log(sample),dist='norm',plot=pylab)
    plt.show()

plot_data_log(sample)
```





Page No.:
Date: / /

Normal Distribution :-

{ Empirical Rule (3-Sigma Rule)

→ 68-95-99.7

→ Normal Distribution is also known as 'Gaussian Distribution'. It is a probability distribution that is symmetic about the mean, showing that data near the mean are more frequent in occurence that data far from the ~~Curve~~ Mean.

In graphical form, the normal distribution appears as a "Bell Curve".

*→ In Normal Distribution the mean is zero (0) & the Standard deviation is 1. Skew → 0 & Kurtosis → 3.

*→ They are Symmetrical.

*→ Two parameters :- Mean & Standard Deviation.

The Empirical Rule :-

1) For all ND, 68.2% of the observations will appear within plus & minus of one standard Deviation of the mean.

2) 95.4% of the observations will fall within plus & minus of second standard Deviation.

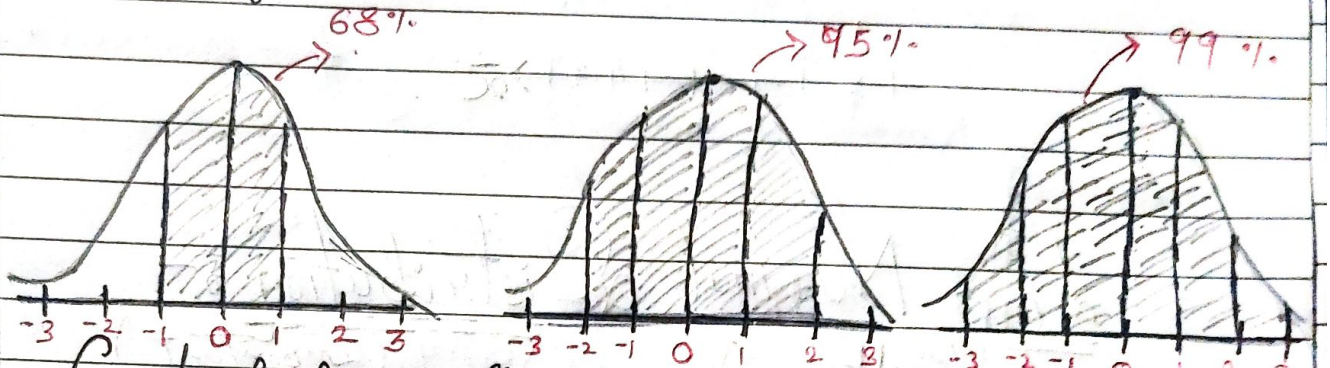
3) 99.9% of the observations will fall with +/- of third standard Deviation.

Iris Dataset → { Sepal lenght & Patel length }
↓
Gaussian Dist.

Page No.:
Date: / /

Example :- Height, Weight, Size { T-Shert }, Gym plate, technical Stock Market analysis.

* Diagram of Empirical formula :-



* Central Limit Theorem :- (Important)

{ log Normal Distribution

It State that the distribution of a Sample variable approximates a normal distribution i.e., ("bell Curve"). As the Sample Size become larger, assuming that all samples are identical in Size, and regardless of the population's actual distribution Shape.

Taking Random Sample.

→ Condition → $n \geq 30$ { Sample Size } → it can be anything. { imp }.

$x_1 = \{ x_1 \text{ ---- ---- } \}$ → take a mean of $x_1 \rightarrow \bar{x}_1$

$x_2 = \{ x_1 \text{ ---- ---- } \}$ → " $x_2 \rightarrow \bar{x}_2$

$x_m \{ \text{ ---- } \}$ → " $x_m \rightarrow \bar{x}_m$.

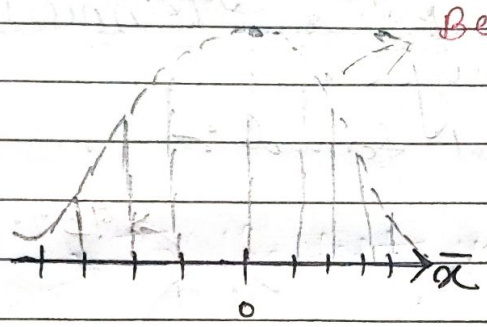
→ Now take all the Mean (Random Samples i.e $x_1, x_2 \ldots x_m$) & Convert it into histogram.

→ Then it will give us 'Normal Distribution'. Regardless whether the population Distributed. we will get a bell Curve.

PDF → Probability Density function.

Page No.:
Date: / /

Result :-



Log Normal Distribution :-

((Galton's Dist.))

(Important)

→ Right Skwed.
(property).

→ if $x \approx$ log Normal Distribution

then

$y \approx \ln(x)$.

↳ Natual log.

* We will get "y" as a Normal Distribution

OR (Reversel).

if $x \approx \exp(y)$

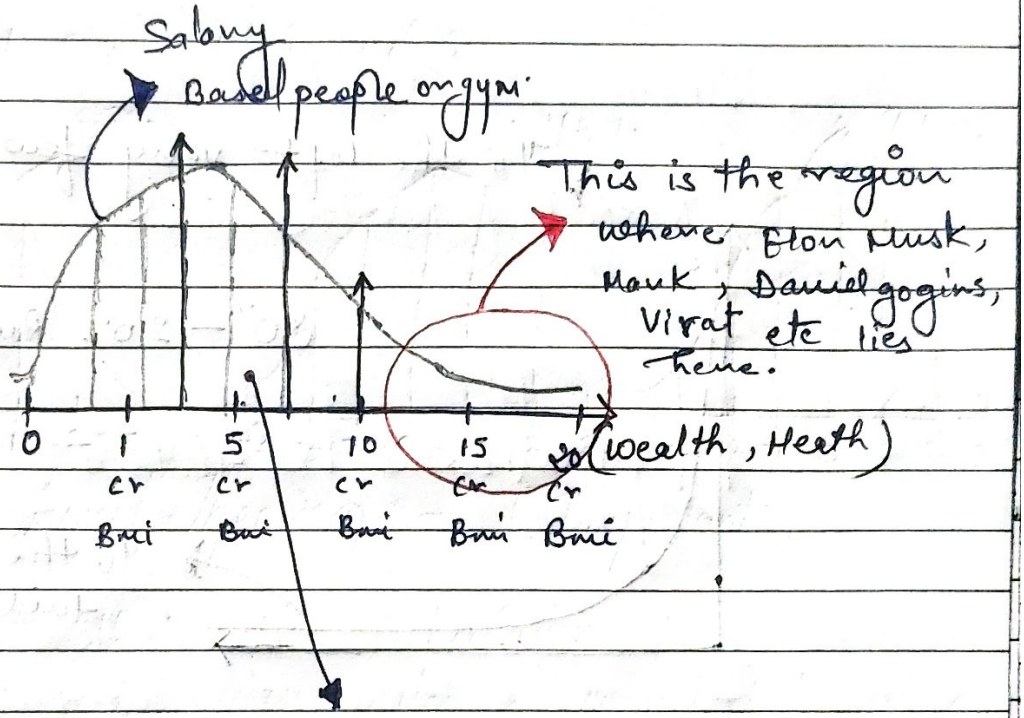
↳ exponential

then

y is Normal Distribution.

Page No.:
Date: / /

Example:- Wealth Distribution, Health Distribution etc.



Use:-

1) Machine learning.

Simple Linear Regression.

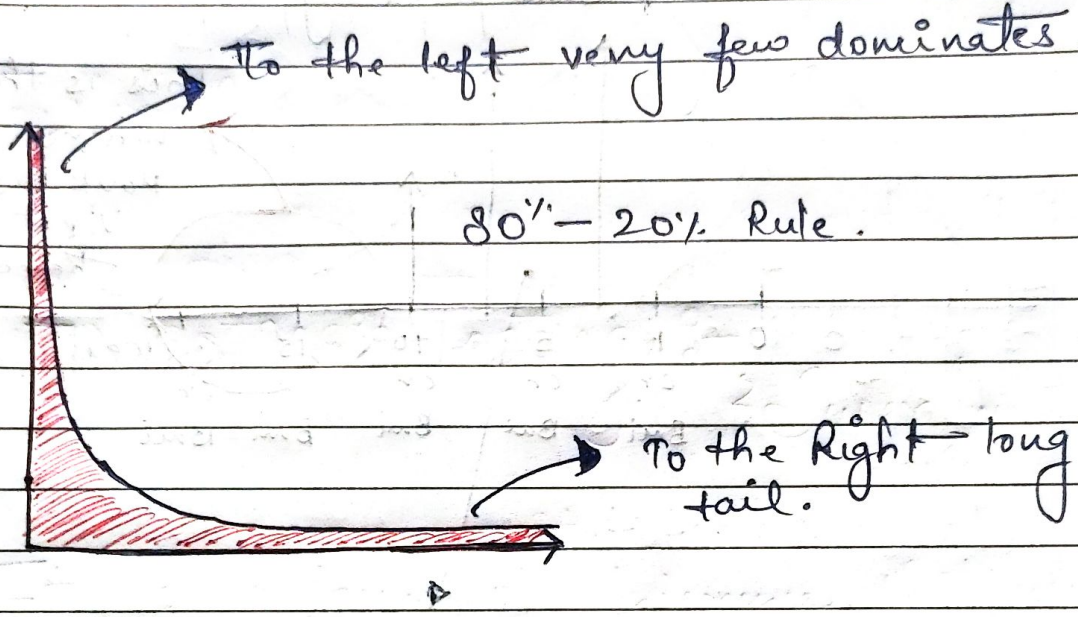
data transformation.

→ Suppose that we have a dataset which is log Normally Distributed then we will simply put (log e) in it. Then we will get a Normal Dist. Thereafter we will sent it to our Algo. (Simple linear Regression) for training purpose.

It work Efficiently on Normally Distributed Dataset.

Page No.:
Date: / /

Power law Distribution :-



∴ Power law graph :-

Example :-

1) Eg → IPL (RCB) { 20% of team is responsible for winning 80% of Match }.

2) 80% of wealth is distributed with 20% of the total population.

3) Oil Rich Nation { 80% → use to the 20% of the Nation }.

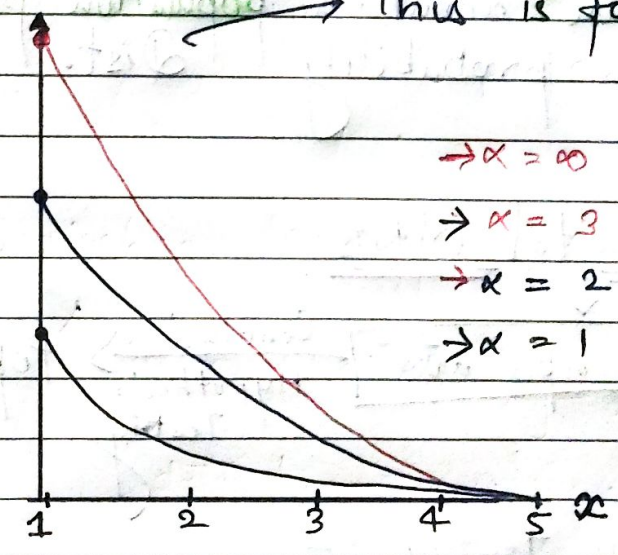
Note :- We Can Convent the Power law distributed data into Normally distributed data.

→ Conversion Can be done by using the "Box Cox" transform.

→ Pareto distribution {Non Gaussion dist}

Pareto Distribution :-

* Any distribution which follow the Power law distribution is known Pareto Distribution
* It is non-Gaussion.
* It is denoted by ~~alpha~~ Alpha : (α).



* We Can Convert it into Gaussion Dist.
* By using Box-Cox Transformation.

Eg → 20% of the Amazon product is responsible for the 80% of the Sales.

d) 20% of team is responsible in delivering 80% of the project.

:- Interview :-

Log Normal & Pareto Dist.

* Hypothesis Testing :-

{Super important topic}.

It is a form of Statistical Inference that uses data from a Sample to draw Conclusion about a population parameter or a population probability Dist.

Inferential Stats

↓

Sample data —(Conclusion / Hypothesis Testing)→ Population Data

1) Hypothesis testing :-

Coin is fair or not.

Step-1 Null Hypothesis (H_0) = Coin is fair.

Step-2 Alternate Hypothesis (H_1) = Coin is not fair.

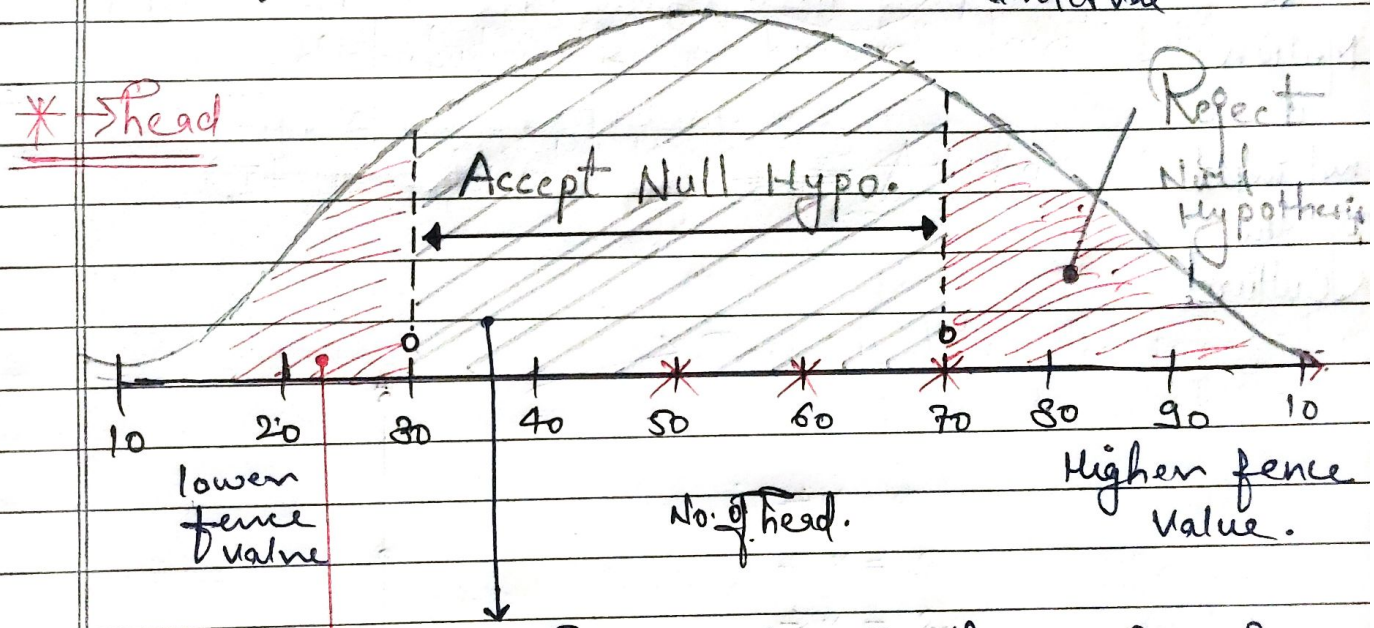
Page No.:
Date: / /

Step-3 Experiment :- Tossing Coin for 100 times.

if my no. of head is 50, 60. → fair
if my no of head is 80, 10, 20 → Not fair.

This type of test is known as "2 tail test".

If we get this, then we do/define "Confidence Interval"



if my head is lies in this region So, we call it as a "fair" {i.e., 30 to 70}.

(Significence Values)

But if my head is lies in this region, then we call it as a "Not fair" {i.e., {10, 20, 80, 90}}

→ Domain Expert will decide the "Confidence Interval"!

→ Eg. Heath Expert, Security Expert etc.

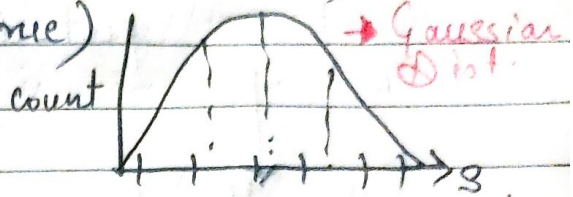
#By using the QQ-Plot we can find out weather the dataset is following the Normal Dist. or not.

Page No.:
Date: / /

→ Create a Normal Distributed.

Using Seaborn:-

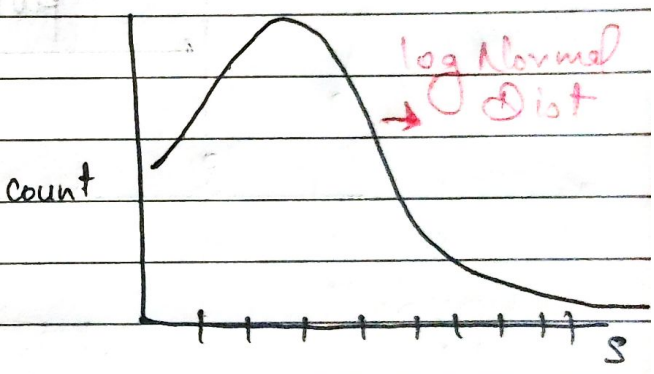
```
S = np.random.normal(0.5, 0.2, 1000)
Sns.histplot(S, kde = True)
```



→ Create a log Normal Distribution.

mean, standard Deviation

```
mu, Sigma = 3., 1
S = np.random.lognormal(mu, Sigma, 100)
Sns.histplot(S, kde = True).
```



→ Check weather a distribution is normal or not:-

→ We use Q-Q Plot to Check.

```
import → Matplotlib.pyplot, Scipy.stats & pylab
  → def plot_data(Sample):
        plt.figure(figsize = (15, 9))
        plt.subplot(1, 2, 1)
        Sns.histplot(Sample)
        plt.subplot(1, 2, 2)
        Stat.probplot(Sample, dist = 'norm',
                      plot = pylab)
        plt.show()
```

Page No.:
Date: / /

```
s = np.random.normal(0.5, 0.2, 1000)
sns.histplot(s, kde=True)
      or
plot_data(s)
```

In the same way we can create an log distributed data.

~~s = np.random.lognormal~~

We only need change in

```
→ stat.probplot(np.log(sample), dist='norm',
                plot=pylab)
```