Probability

- Probability is all about how likely an event will occur like whats the probability of getting head when tossing a coin is 0.5 or the probability of getting 3 when rolling a die is 1/6.
- Its expressed in a number b/w 0 to 1
- 0 means it will not happen, 1 means it will happen & 0.5 means it will happen half the time

Emperical Probability

- Empirical probability, also known as experimental probability, is a type of probability that is based on observations or experiments. Instead of using theoretical calculations or mathematical formulas, empirical probability is determined by actually conducting experiments or observations and recording the outcomes.
- when tossing a coin 100 times, we got head 55 times so emperical probability of getting a head will be 55/100

Theoritical Probability

- Theoretical probability, also known as classical probability, is a type of probability when each outcome in a sample space is equally likely to occur
- like when tossing a coin, possibility of both outcomes in sample space {H,T} is equal or when rolling a dice, probability of getting any number is 1/6
- formula: no of favorable outcomes/ total number of outcomes in sample space
- getting 3 when rolling a dice is : 1/6
- Emperical probability will be very close to theoritical probability when no of trials increases

Random Variable

(Its not a "variable", Its a function)

- random variable is a function which takes as input, runs a logic & provides an ouput
- input will be complete sample space for example: in tossing a coin experiment, input will be {H,T} or in rolling a dice experiment input will be {1,2,3,4,5,6}
- output will always be a real numberthat we asign to each possible outcome
- random variable will convert sample space into real number based on logic which comes from the events you want to study so that we can study them with the help of probability distributions

How do we decide the logic

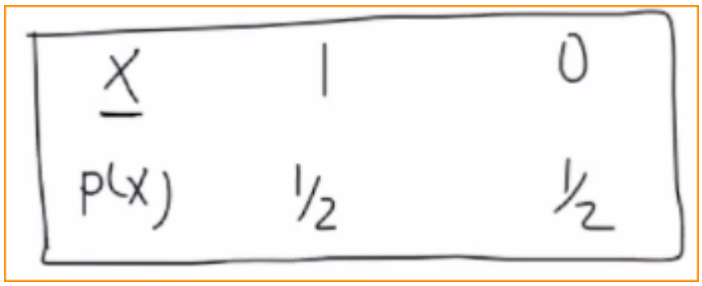
- logic comes from event --> for whatever reason we're conducting the experiment
- for example: rolling 2 dice & we want a probability of "getting a sum of 7"
- so the sample space will be sum of 1-1 number {(1,1)(1,2)(1,3)... (2,1)(2,2).. (3,1)... (6,6)}
- sample space will have 6 x 6 = 36 items --> X = {2,3,4,5,6,7,8,9... 36}
- probability will be: 1/36

There are 2 types of random variables:

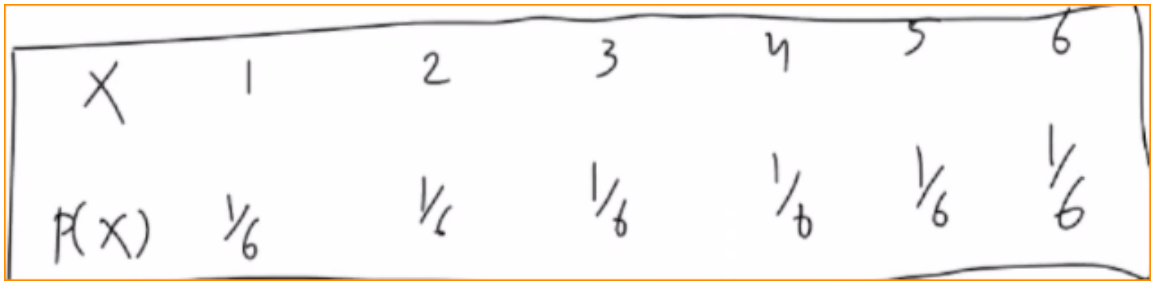
1. discrete
2. continous

Probability distribution of a random variable

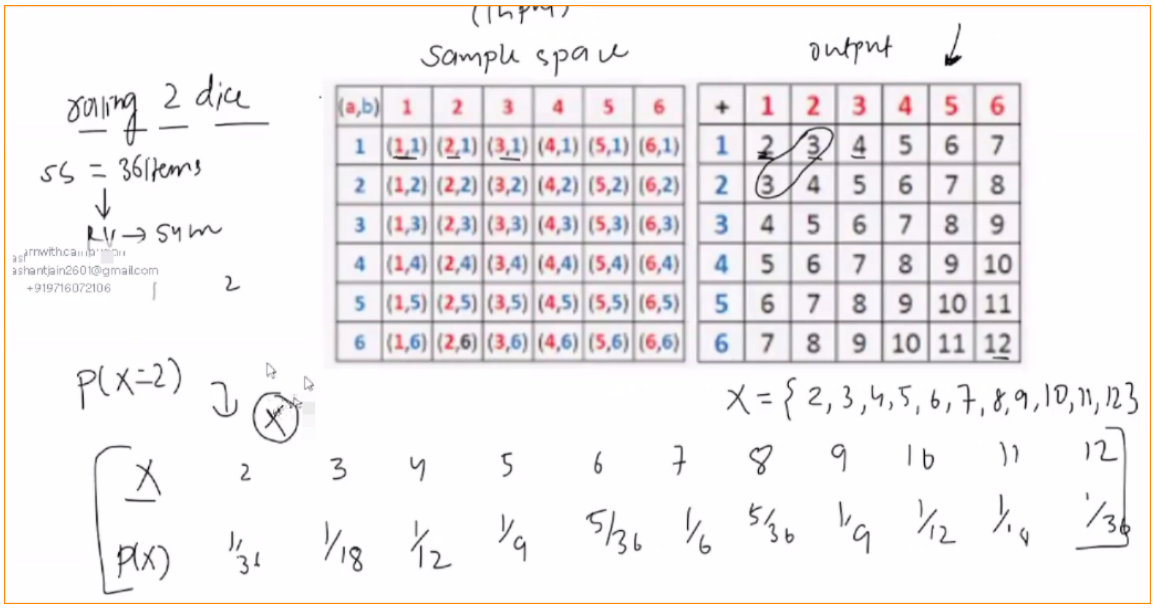
- probability distribution is a list of all the possible outcomes of a random variable along with thier corresponding probability values
- Its a table which contains all possible outcomes & its probability
- for tossing a coin experiment:



- for rolling a dice experiment:



- for rolling 2 dice experiment:



- Above is manual process & its only possible when X is discreete & possible values are less but if they are very big or continous then it will be very difficult like rolling 1000 dice altogether
- In all those cases we need to find the relationship b/w X & P(X)
- lets call P(X) as y so the formula will be y = f(X) where as soon as we pass the value of X we get the value of y which is the probability of X based on the mathematical relationship function "f"
- **This function which represents the mathematical relation b/w X & P(X) is called "probability distribution function"**

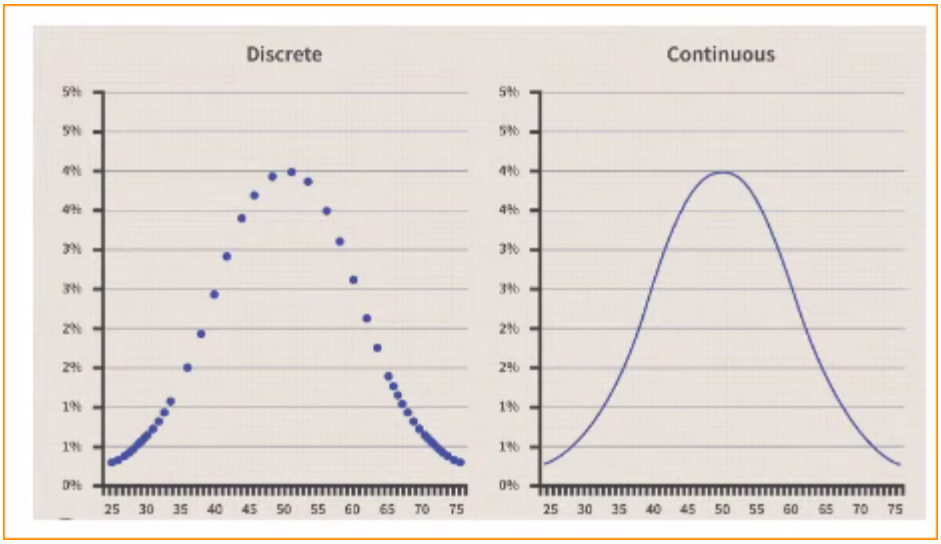
- **In case of continous values: Its called PDF (density function)**
- **In case of discrete values: Its called PMF (mass function)**

Probability Distribution

- Probability distribution shows the probability of each item from sample space to occur However, in large experiments or continuous variables, it's not feasible to list out all possible outcomes and their probabilities.
- The Probability Distribution Function (PDF) is a mathematical function that describes the likelihood of a random variable taking on particular values.
- Instead of listing out probabilities for each outcome, the PDF gives us a formula or function to calculate probabilities.
- PDF is a mathematicl function b/w X & y where X is all the possible outcomes like 1,2,3,4,5,6 & y is thier probability

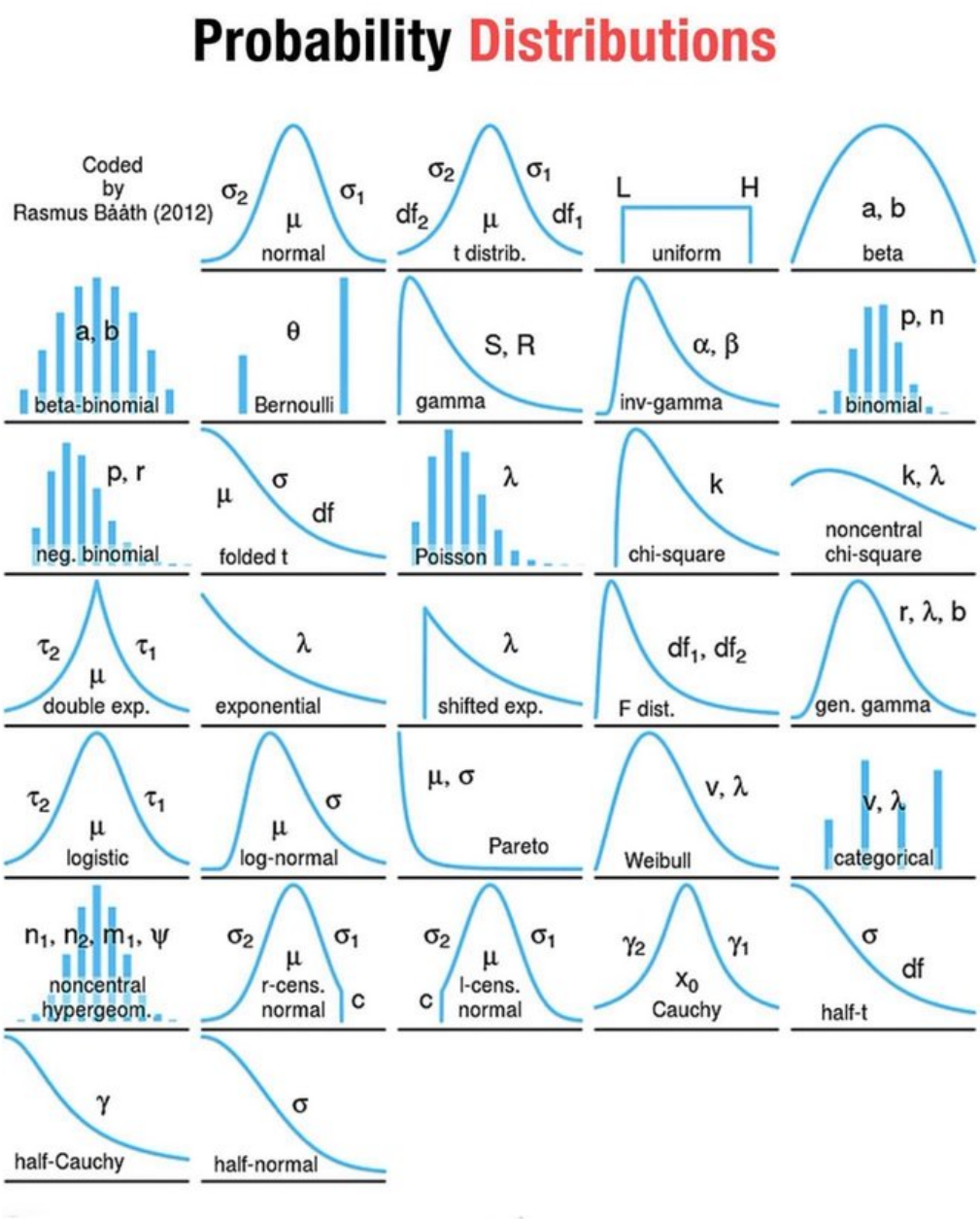
Types of PDF

1. Discrete --> only integers
2. Continous --> all numbers even with decimals



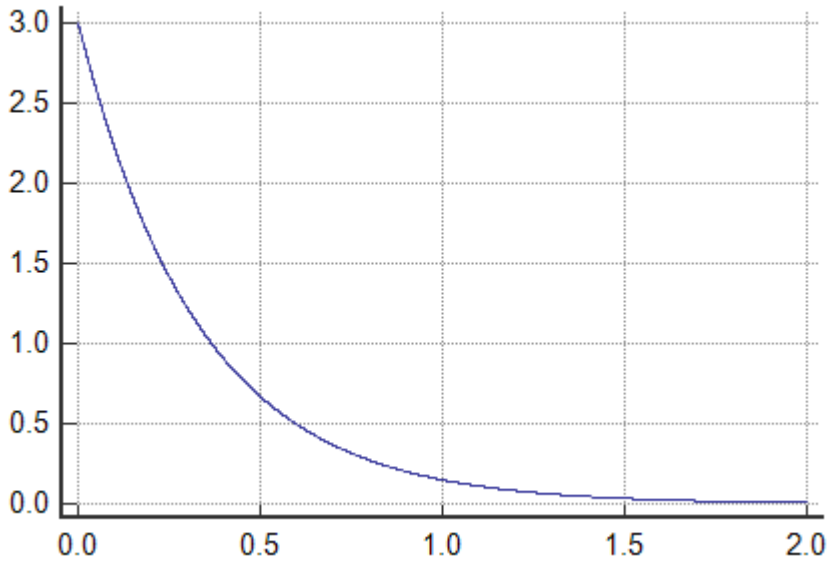
- most common pdf grapsh which exixts in nature
- It means there is a high chance that when when you plot a graph b/w random variable & its probability then most probably it wil look like any of

the below



Whats the importance?

- gives a quick idea about the shape/distribution of data
- like below graph shows the earning of most people, which clearly shows lot of people earn very less & very few earn a lot



- If distribution graph of your data falls in any of the famous distribution then you'll know a lot about your data automatically
- every probability distribution have a tuning knob "parameters" which changes thier shape, size etc

Types of Probability Distribution Functions

1. PMF (Probability Mass Function) gives probability for discrete random variables
2. PDF (Probability Density Functions) gives probability for continous random variable
3. CDF (Cumulative Distribution Function) using both

PMF (Probability Mass Function)

- probability asigned to each value must be non negative
- sum of all probability will be 1

In [26]:
```
#when rolling 1 dice

import pandas as pd
import random
```

In [27]:
```
l = []
for i in range(10000):
    l.append(random.randint(1,6))
```

In [28]:
```
len(l)
```

Out[28]:
```
10000
```

In [29]:
```
l[:5]
```

Out[29]:
```
[5, 5, 2, 1, 5]
```

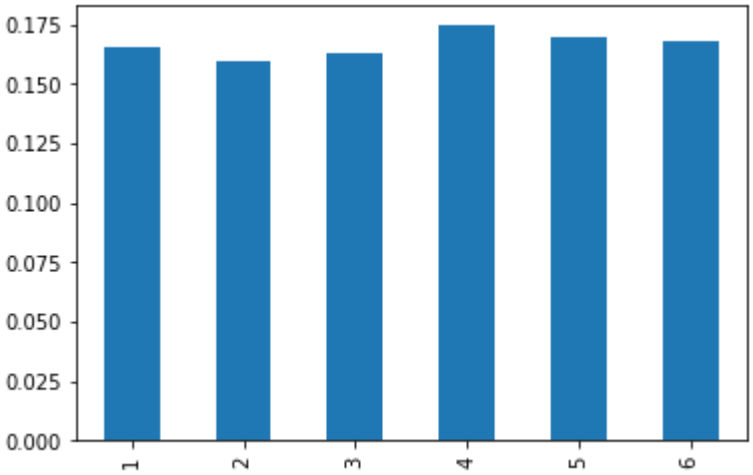
In [30]:
```
pd.Series(l).value_counts()/ pd.Series(l).value_counts().sum()
```

Out[30]:
```
4    0.1744
5    0.1698
6    0.1677
1    0.1653
3    0.1629
2    0.1599
Name: count, dtype: float64
```

In [31]:
```
s = (pd.Series(l).value_counts()/ pd.Series(l).value_counts().sum()).sort_ir
```

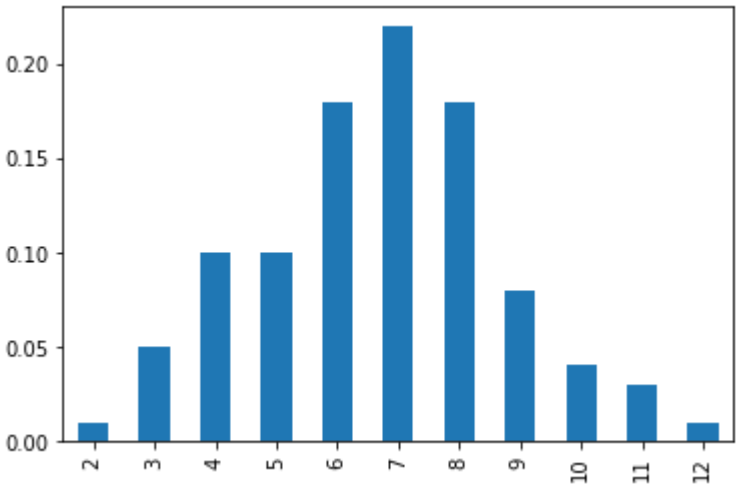
In [32]:
```
s.plot(kind='bar')
```

Out[32]:
```
<AxesSubplot:>
```



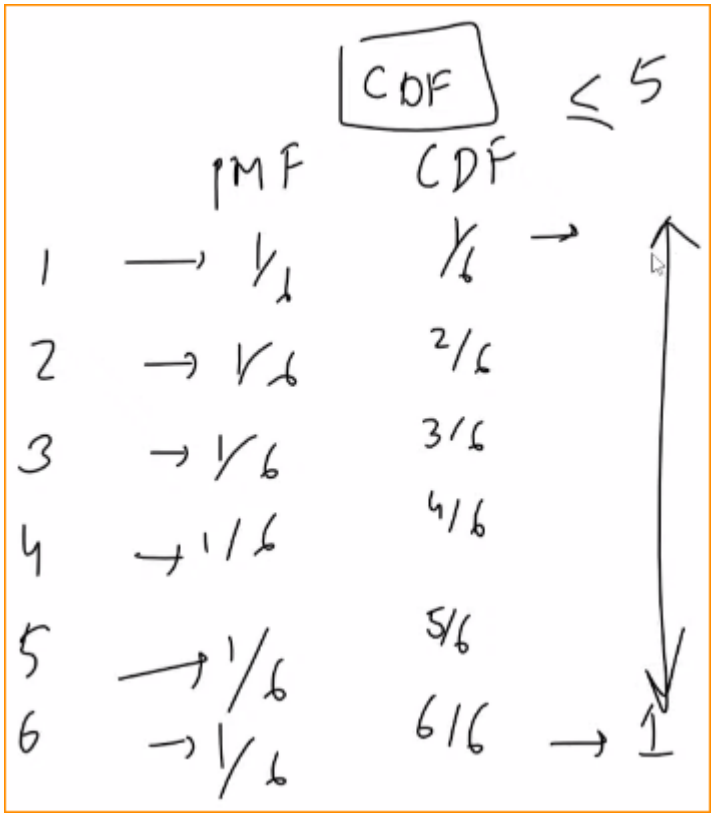
In [34]:
```
#when rolling 2 dice together

import pandas as pd
import random

l1 = []
for i in range(100):
    a1 = random.randint(1,6)
    b1 = random.randint(1,6)
    l1.append(a1+b1)

s1 = (pd.Series(l1).value_counts()/ pd.Series(l1).value_counts().sum()).sort
s1.plot(kind='bar')
```

Out[34]:
```
<AxesSubplot:>
```



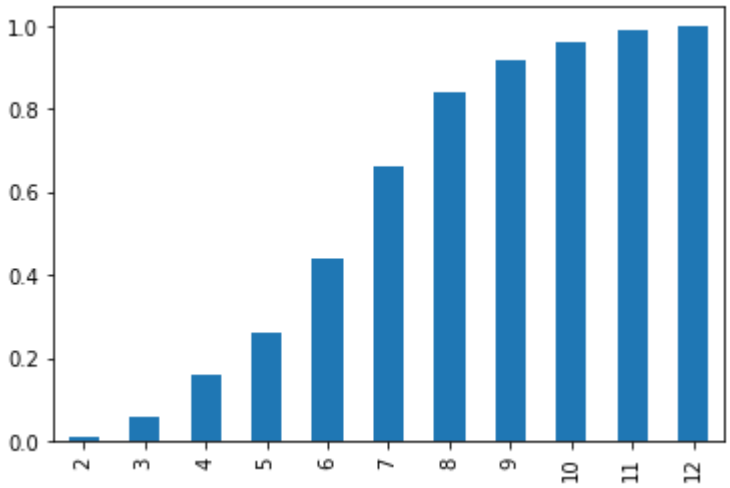
CDF of PMF

- CDF gives the probability that a random variable is less than or equal to a certain value



In [36]:
```
import numpy as np

np.cumsum(s1).plot(kind='bar')
```

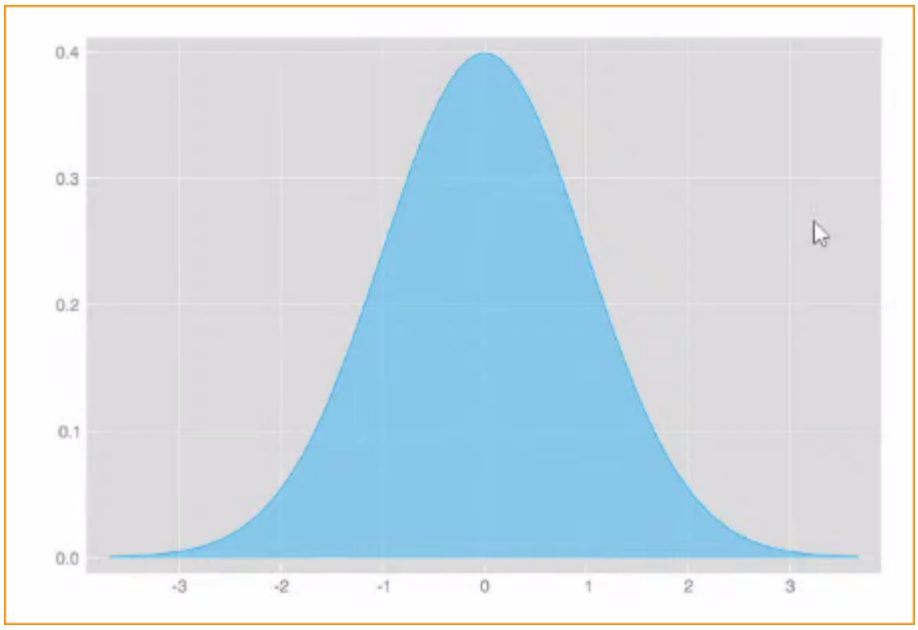
Out[36]: <AxesSubplot:>



In [38]:
```
#PMf was telling us the probability of x but CDF tells us all the probabilit
```

Probability density function

- PDF is a mathematical function that describes the probability distribution of a "continous random variable"
- whereas PMF describes probability distribution of a "discrete random variable"
- In PMF, y-axis hold probability but in PDF, Its Probability Density



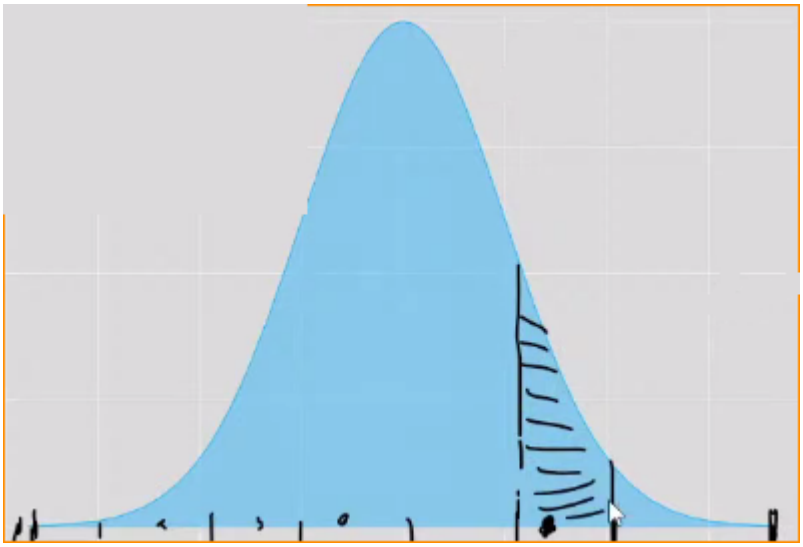
Why Probability density & not Probability?

- Since x-axis holds continous value which also means it holds infinite values

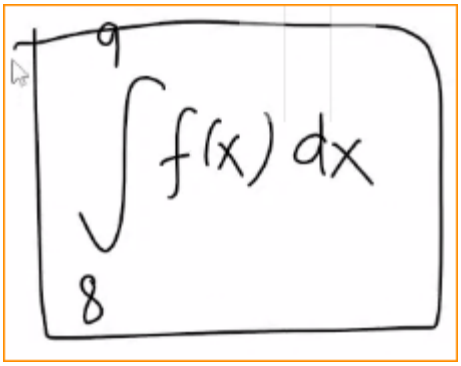
- Probability for any infinite value will be very close to 0 thats why y-axis doesnt have probability instead it has probability density

what does the area (in blue) of graph above represents?

- It encompasses the total likelihood of all conceivable outcomes.
- "Probability Density" denotes the probability that a value falls within a given range of numbers.
- In order to calculate: lets say betrween 8 & 9, so we cut that area



- calculate area.. which tells us..? if complete area is telling us probability between 0 to 10 then this area tell us probability between 8 to 9
- apply integration & find probability between 8 & 9



- if we can get the probability between 8 & 9, we can cut even smaller area to get probability between 8 to 8.01 or 8 to 8.0001
- so we can roughly predict the probability of each point
- which actually not probability of that point bu the probability between 2 points but since the difference between 2 points is very small so can consider it similar
-

How to calculate graph of PDF

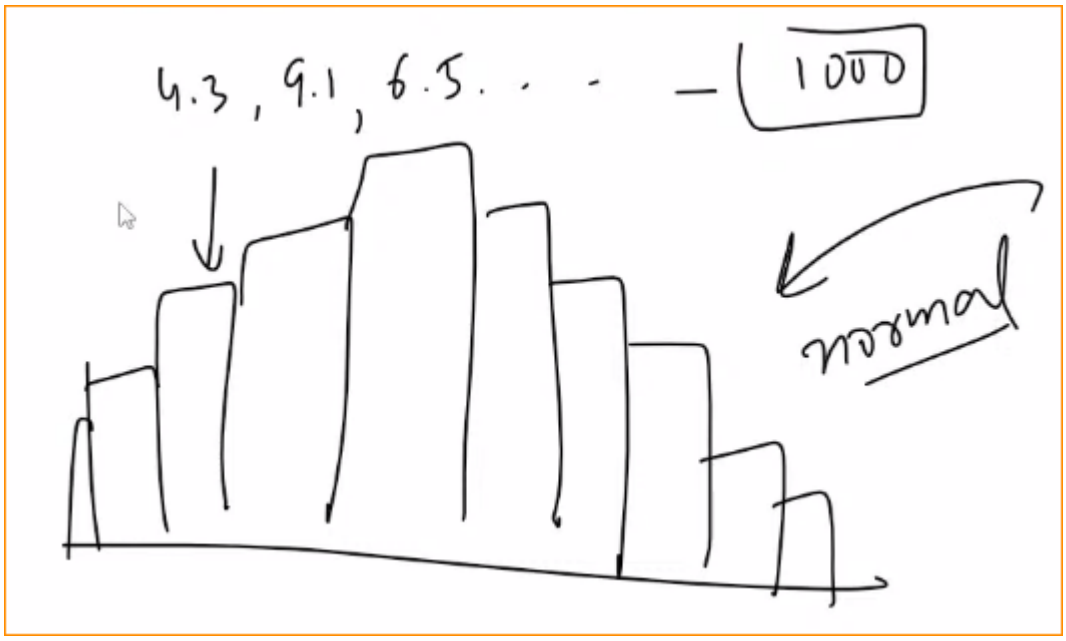
- PMF graph was easy.. we were calculating probabilities & plotting it on y-axis
- PDF doesnt have probabilities instead it has probability density
- In order to plot PDF from a given data, we have to do "Density Estimation"

Density Estimation

- Its used to estimate the PDF of a random variable
- There are 2 methods of density estimation:
    1. parametric --> This approach assumes that data data follows a specific probability distribution like normal, uniform, log-normal etc
    2. non-parametric --> This approach doesnt make any assumption instead it directly estimate from data
- commonly used techniques for density estimation are kernel density estimtion(KDE), histogram estimation, gaussian mixture model(GMM)
- choice of method depens on characteritics of data & intended use of density estimtion
- we have a data which is continous random variable.. so we have to make PDF but in order to draw PDF we need y-axis value which is probability density, so we'll use density estimation.. density estimation also have 2 techniques parametric (where we assum how data looks like) & non-parametric (where we dont assume, we estimate direclty from data)

Parametric density estimation

- In Parametric density estimation, we assume how our data looks like normal, binomial etc
- for example: we have CGPA of 1000 students & we want to plot PDF, since it continous random variable so we need density estimation, In order to get density estimation we can use parametric or non-parametric, In parametric, we plot histogram on the data & by looking at data, we'll get an idea how data looks like.. lets say it looks like "Normal Distribution"



- Once we get an idea how our data looks like, we'll follow that distribution
- In order to draw PDF of normal distribution we need mean & std which is a sample mean & sample std
- Try to estimate population mean & std
- then we can put it in normal distribution pdf equation & we'll get probability density & which we can plot

In [1]:
```
import matplotlib.pyplot as plt
import numpy as np
from numpy.random import normal
```

In [8]:
```
#we're specifically generating normally distribued data
#so we already know that our data is normally distributed
sample = normal(loc=50, scale=5, size=1000)
```

In [9]:
```
sample.mean()
```

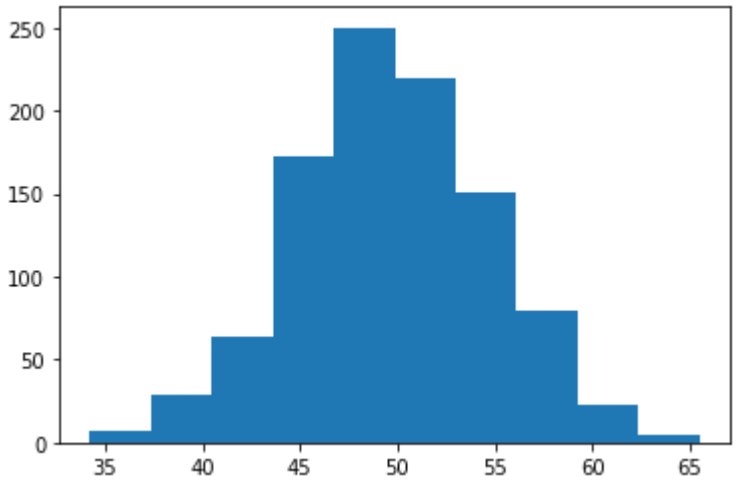
Out[9]:
```
49.73048029754684
```

In [10]:
```
sample.std()
```

Out[10]:
```
4.991364724166247
```

In [11]:
```
# mean & std is not exactly same bcoz its sample data & above is population
```

In [14]:
```
#plot histogram to understand data distribution
plt.hist(sample, bins=10)
plt.show()
```



In [15]:
```
#it looks like somewhat normal
```

In [16]:
```
#calculate sample mean & sample std
sample_mean = sample.mean()
sample_std = sample.std()
```

In [17]:
```
#fit the distribution with above parameters

from scipy.stats import norm
dist = norm(sample_mean, sample_std)
```

In [18]:
```
#generating 100 values between min & max of my sample data which will b "x"
values = np.linspace(sample.min(), sample.max(),100)
```

In [20]:
```
#sending x 1 by 1 to calculate probability density
probabilities = [dist.pdf(value) for value in values]
```

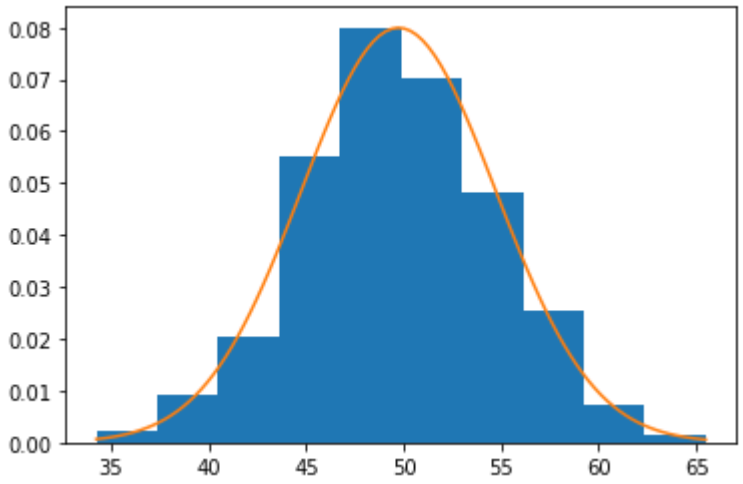
In [21]:
```
probabilities
```

Out[21]: [0.0006433354068960486,
 0.0007814083029604666,
 0.0009453248025138907,
 0.0011390598587610638,
 0.0013670188575274631,
 0.001634048442601674,
 0.0019454398731274815,
 0.0023692333374175,
 0.0027246651582902664,
 0.0032051713389759247,
 0.003755380876087626,
 0.004382472440622332,
 0.005093858311279036,
 0.005897079615585323,
 0.006799697385980267,
 0.007809165788213497,
 0.008932687979532459,
 0.010177055645681436,
 0.011548473911700441,
 0.013052374005254103,
 0.014693216752041076,
 0.01647429067633508,
 0.018397509138417985,
 0.020463211534895717,
 0.022669974086823752,
 0.025014436113550604,
 0.02749114790816079,
 0.030092446367424447,
 0.032808364363999414,
 0.03562657946620383,
 0.03853240700342757,
 0.04150884164424023,
 0.04453665060994812,
 0.04759452040900708,
 0.050659257577297,
 0.0537060423849647,
 0.05670873286962583,
 0.059640214932059656,
 0.06247279264261958,
 0.06517861141520392,
 0.06773010537125797,
 0.07010045909634362,
 0.07226407313797438,
 0.07419702204895363,
 0.07587749357796249,
 0.07728619776864298,
 0.07840673525599011,
 0.0792259149359308,
 0.0797340124072481,
 0.07992496210719605,
 0.07979647783325865,
 0.0793500983028645,
 0.0785911564816598,
 0.07752867353514603,
 0.07617518035210376,
 0.4715765844,

```
 0.07266129889827615,
 0.07054101192626375,
 0.06820915625634702,
 0.06569103929724346,
 0.06301327502007777,
 0.06020331902707742,
 0.05728900520260670 6,
 0.05429809472995139 6,
 0.05125784746088385 4,
 0.04819462455335022,
 0.04513352999584404 5,
 0.04209809717167982 4,
 0.03911002504255258,
 0.03618896690939775 5,
 0.0335237309878328,
 0.03061538737981633 5,
 0.02799079548853939,
 0.02548902286487051,
 0.02311817762314701 3,
 0.02088413390344446 8,
 0.01879065009942415,
 0.01683951603236634,
 0.01503072293432668 2,
 0.01336265010230187 9,
 0.01183226226976157 9,
 0.01043531208675524 6,
 0.00916654257646861 2,
 0.00801988501412187,
 0.00698864832257902 35,
 0.00606569676797404 3,
 0.00524361344007375 7,
 0.00451484769093516 2,
 0.00387184536021366,
 0.00330716121879116 47,
 0.00281355360097644 43,
 0.00238406166032435 8,
 0.00201206607004553 87,
 0.00169133429457814 9,
 0.00141605178588045 25,
 0.00118084061071622 8,
 0.00098076710004581 67,
 0.00081134013648845 48,
 0.00066850166950809 65,
 0.00054861097974702 54]
```

In [25]:
```
plt.hist(sample, bins=10, density=True)
plt.plot(values, probabilities)
```

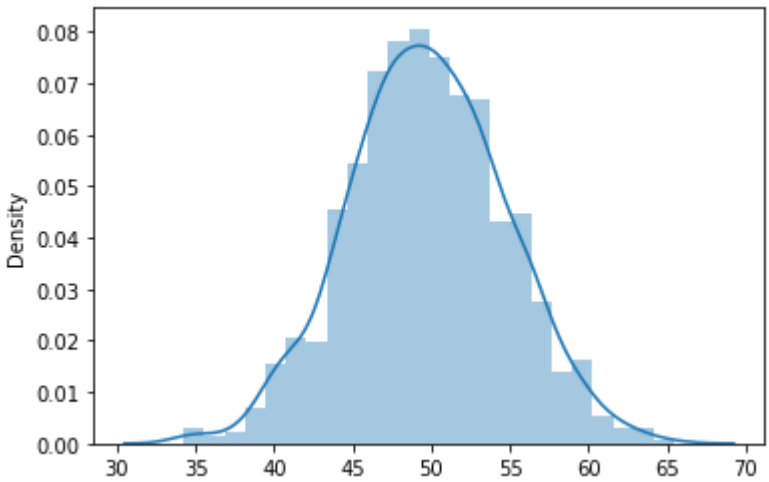
Out[25]:
```
[<matplotlib.lines.Line2D at 0x155e9138160>]
```



In [27]:
```
import seaborn as sns
sns.distplot(sample)
```

```
C:\Users\iampr\AppData\Local\Temp\ipykernel_1520\1482356190.py:2: UserWarnin
g:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(sample)
C:\Users\iampr\AppData\Local\Programs\Python\Python310\lib\site-packages\sea
born\_oldcore.py:1119: FutureWarning: use_inf_as_na option is deprecated and
will be removed in a future version. Convert inf values to NaN before operat
ing instead.
  with pd.option_context('mode.use_inf_as_na', True):
```

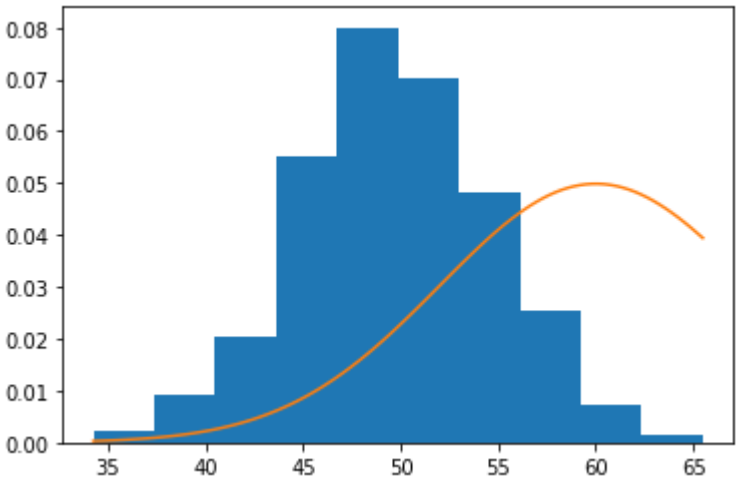
Out[27]: <AxesSubplot:ylabel='Density'>



In [28]:
```
#this method is called parametric bcoz it uses 2 parameters --> mean & std t
#if we change the value of mean & std then graph will also change its shape
```

In [38]:
```
dist1 = norm(60, 8)
values1 = np.linspace(sample.min(), sample.max(),100)
probabilities1 = [dist1.pdf(value) for value in values1]
plt.hist(sample, bins=10, density=True)
plt.plot(values1, probabilities1)
```

Out[38]:
```
[<matplotlib.lines.Line2D at 0x155ed76ca30>]
```



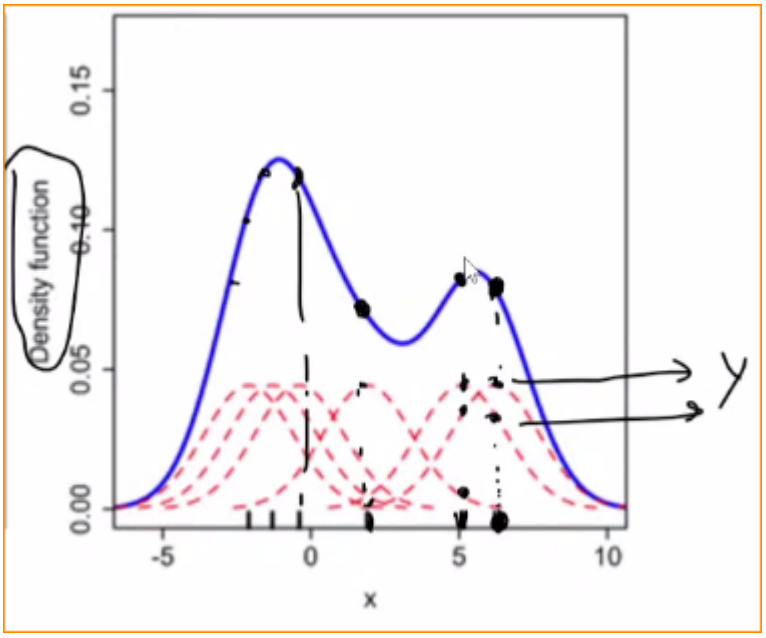
Non-Parametric density estimation

- Sometimes data is not following any distribution, so we cant use parametric
- This method involves constructing an estimate of PDF using available data points
- Unlike parametric method where we have used mean & std of data
- We use Kernel Density Estimation
- The advantage here is that it doesnt require any assumption of specific distribution which makes it easier to use for any type data
- Its computationally difficult & require more data to achieve accurate estimates

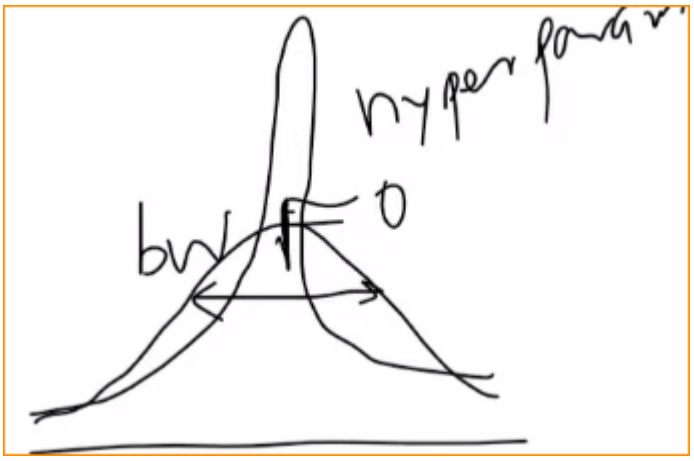
KDE

- Kernel density estimation (KDE) is a technique used in statistics and machine learning to estimate the probability density function (PDF) of a dataset.
- Imagine you have a set of data points representing the heights of people in a population. Kernel density estimation works by placing a "kernel" (probability distribution, mostly normal distribution) considering that point as middle point of normal distribution - on each data point and calculate probability

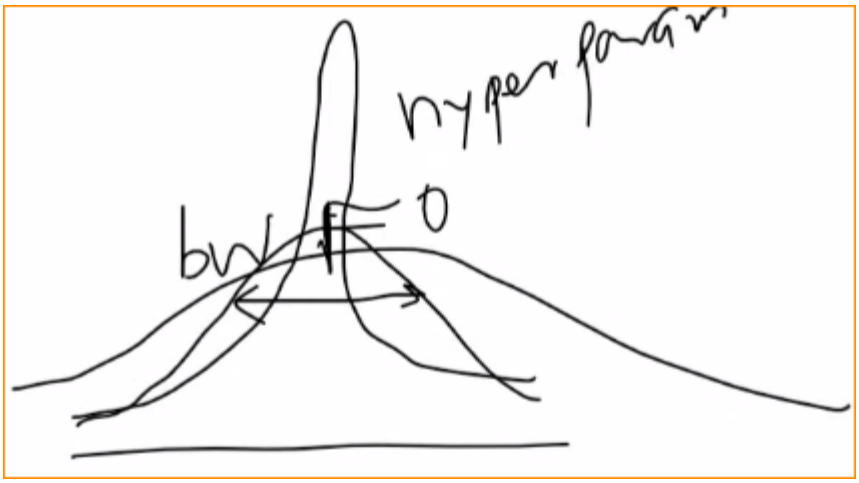
density by adding both points density on y-axis



- In Normal distribution, we need 2 parameters.. mean which is the point itself but what is "std".. how thick or thin the shape would be depends upon std !!
- In KDE, STD is Bandwidth, If we asign a very less value then it will have peakedness bcoz y will increase



- If we increase the value of bandwidth, y will reduce so shape will spread out



Why not average & why sum?

- In kernel density estimation (KDE), we don't take the average of the densities because we're interested in estimating the underlying probability density function (PDF) of the data. The goal is to create a smooth representation of how the data is distributed.
- By summing the densities of the kernels placed on each data point, we're effectively capturing the contribution of each individual data point to the overall density estimation. This allows us to model the variability and spread of the data accurately.
- Taking the average would not adequately capture the shape and spread of the data. Averaging the densities would flatten out the resulting curve and could potentially distort the true distribution of the data. Instead, by summing the densities, we preserve the relative contributions of each data point to the overall density estimation, resulting in a more accurate representation of the underlying PDF.

In [39]:
```
#generate sample data
sample1 = normal(loc=20, scale = 5, size = 300)
sample2 = normal(loc=40, scale = 5, size = 700)
sample = np.hstack((sample1,sample2))
```

In [40]:
```
sample
```

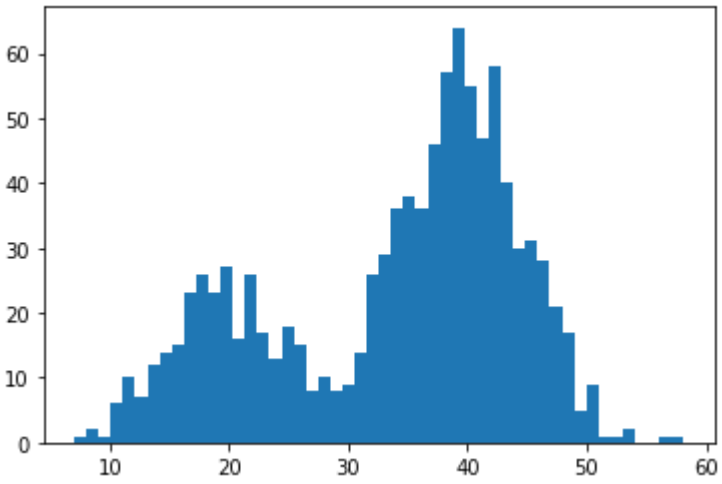
Out[40]:
```
array([20.33727281, 20.89352706, 21.60827056, 13.31104928, 22.37325594,
       23.54459755, 23.87357835, 21.56190114, 14.17268391, 21.66176693,
       26.3804438 , 22.06371532, 26.98036149, 11.31286104, 19.90970816,
       21.08383935, 19.48751647, 15.22259481, 20.28967856, 15.26507178,
       17.31935634, 12.21306623, 23.1513612 , 13.79109494, 23.21167416,
       16.82258829, 18.09357506, 16.93148692, 27.58565724, 22.02637172,
       15.5541701 , 16.92424688, 14.80117957, 18.52672087, 24.1944709 ,
       18.17334549, 22.95256406, 25.13065105, 12.39530627, 19.10185701,
       14.33498591, 21.67723115, 15.24787548, 25.52641497, 15.00052586,
       23.70783418, 12.88975539, 19.01443042, 21.44152633, 32.13113953,
       21.63008572, 17.23961727, 30.7397991 , 21.75184878, 19.65972342,
       18.91548164, 24.39350679, 32.51646309, 26.39089709, 18.15173519,
       18.91800717, 31.24607355, 15.20241959, 31.31798541, 15.48468278,
       22.14924714, 18.77787968, 28.903499  , 12.13591309, 29.50761301,
       20.09968417, 12.00233867, 23.64806844, 28.10576729, 24.53388377,
        9.51272343, 15.17659076, 25.68177463, 22.69477911, 21.20400241,
       13.60605227, 15.33095926, 21.09083022, 20.75267685, 18.89003223,
       12.28804501, 17.75059143, 25.16367016, 24.38315516, 14.43128891,
       17.36681189, 37.90140329, 19.96468945, 17.81491359, 13.85777257,
       27.19269015, 16.12583878, 17.33266836, 19.86358049, 20.61145456,
       26.70257696, 25.79693128, 15.1859409 , 18.81588375, 16.94121212,
       22.00017903, 19.41239206, 18.26895842, 23.80871711, 15.4387911 ,
       18.21989077, 21.40602266, 12.91912063, 23.12625194, 31.00227402,
       20.15335757, 16.84560688, 28.73554111, 25.84281515, 15.87780614,
       22.39736884, 17.31084441, 27.49304126, 18.47022207, 24.87237889,
       12.77688468, 24.79026173, 18.09871411, 23.49976958, 18.34273487,
       23.28035498, 21.22431164, 10.52889427, 16.68573786, 29.74651992,
        8.80574471, 24.7058542 , 21.72141993, 26.12470754, 22.27521349,
       23.64236366, 11.39659898, 26.55225571, 18.97583263, 17.87239172,
       21.45497328, 10.84762928, 24.83165997, 16.41585513, 11.15810527,
       15.02033301, 20.11366883, 22.26871483, 18.97458982, 10.75809186,
       16.86619038, 21.33378433, 11.7072891 , 21.44660553, 27.79235232,
       17.70784719, 26.29023416, 16.31678167, 22.08728022, 27.22022624,
       17.97624834, 15.19158645, 18.07247946, 23.75070863, 13.74561945,
       17.8505236 , 17.98573548, 22.70346716, 20.19215144, 17.07524492,
       13.75281246, 21.19926218, 19.71797894, 18.44056902, 15.985753  ,
       20.8434015 , 16.837499  , 17.75675906, 16.57779973, 20.08562056,
       28.14853185, 16.37729906, 19.33376109, 26.45060559, 11.77025924,
       10.14084527, 18.8138005 , 13.0135272 , 22.81629105, 18.67104499,
       13.55073894, 17.47435715, 24.78956559, 31.27022443, 18.94814755,
       25.24854786, 10.89379716, 16.47823611, 20.18319528, 29.44259255,
       16.12212456, 23.21528495, 21.76848727, 19.24697376, 16.55289145,
       19.22492379, 23.16274995, 22.58785731, 14.5960161 , 27.52328743,
       20.66030514, 28.61776899, 25.12589643, 21.23124164, 19.31571706,
       20.32980401, 20.15447919, 15.15311967, 14.77797133, 25.32598424,
       19.27616503, 22.53070145, 16.78253541, 17.78462052, 13.78194625,
        8.66020499, 18.32672349, 17.76667974, 27.5127882 , 11.99652324,
       16.09087724, 22.01137809, 26.04922421, 13.50197138, 21.7901463 ,
       16.53669066, 19.75802332, 16.25599772, 13.857563  , 16.27921374,
       17.85551637, 18.87628142, 21.97090828, 26.34020016, 23.30813218,
       24.89345225, 24.7614134 , 19.31158441, 11.27658604, 14.87809615,
       30.40406923, 19.41999064, 21.64579473, 19.90500911, 17.46476931,
       21.49888234,  7.03105336, 23.49300669, 15.24454644, 26.6793887 ,
       27.91257293, 17.19582033, 17.13449623, 19.46871671, 19.79333664,
       16.10358547, 18.16441222, 20.33879064, 27.2146532 , 22.43452392,
       .99615753, 24.85724552, 26.16734636, 28.48085478, 19.96783887,
```

20.01370395, 18.12532157, 16.76539776, 21.92114886, 19.03159513,
26.93218085, 19.83714454, 25.54845162, 22.95042933, 19.79431051,
25.00695354, 25.11145863, 10.26965377, 14.34544878, 11.59287214,
25.05949452, 19.19794841, 20.9617554 , 23.88913509, 15.51108141,
40.56608388, 50.72018043, 43.7197369 , 43.01634763, 42.0423864 ,
37.44585849, 38.8306873 , 37.78351102, 44.85427285, 42.50442261,
40.74074771, 34.26438002, 46.04483672, 32.58621561, 43.17498226,
33.75818475, 39.12632872, 34.06838126, 28.21031777, 42.57996079,
37.31678325, 44.48013022, 32.66092628, 45.7449426 , 32.18902999,
30.82384893, 46.30720621, 38.43551392, 35.73708829, 38.66286568,
42.52293521, 34.42855335, 48.81532463, 39.3627288 , 38.81442399,
37.32129617, 38.85719779, 37.83385395, 42.7490413 , 33.14751819,
40.65821318, 35.78247731, 40.20793817, 34.14068892, 41.52534339,
41.87313966, 48.78542223, 38.68203066, 38.52238685, 29.98915158,
46.37236029, 42.48339494, 53.02212115, 36.64183145, 34.14793748,
41.92644047, 49.74485583, 41.06098381, 25.95045629, 43.44432242,
31.92536075, 36.94720952, 42.07118143, 46.07200642, 45.96730695,
46.3857442 , 47.95191798, 45.87231817, 36.72458778, 50.79266442,
40.44863494, 35.47552163, 34.89856412, 36.55428022, 42.92376321,
46.00111106, 38.68517556, 42.08870423, 36.94563458, 40.70313229,
39.03143517, 34.29000563, 26.32509246, 41.56145068, 38.06473113,
42.37720497, 39.46541833, 40.8000876 , 40.77960755, 37.77290744,
44.00313618, 36.90392201, 38.2082201 , 40.76928008, 42.61585278,
39.9095006 , 36.67007271, 33.69190709, 33.25462535, 40.57849035,
40.4164014 , 41.44874444, 35.3204   , 46.76790343, 41.91958879,
38.51629136, 46.29086115, 38.12601034, 42.62722364, 41.91549905,
37.06459479, 46.2943594 , 32.31211655, 43.87125994, 41.72204339,
39.97477198, 41.50471536, 33.43438783, 46.09717745, 31.6467437 ,
36.16217315, 46.92745493, 58.13586151, 48.22020019, 40.81536149,
34.64161036, 37.22918287, 35.70434369, 46.33752706, 25.2105545 ,
40.16333289, 44.9898135 , 28.83202965, 44.00503314, 31.1943194 ,
48.80189072, 30.21999239, 43.77524767, 44.05041759, 43.32647793,
45.38602874, 33.80599647, 38.6060107 , 42.16111977, 41.96333188,
39.50887206, 38.55959654, 40.91175211, 39.41045622, 34.90519976,
40.88574907, 38.86591476, 37.92954082, 35.5124153 , 39.53028952,
39.62851397, 41.8561682 , 35.49992021, 47.7447367 , 35.99822392,
36.64238082, 44.42584368, 34.99665603, 33.6201578 , 37.73817074,
39.45960024, 31.14385656, 42.7953512 , 43.35105321, 43.17910852,
43.42038151, 45.40124187, 34.30630904, 38.26517337, 40.21560992,
43.69793847, 39.71283276, 40.22205902, 40.22516812, 31.97467089,
37.01355   , 42.45122644, 40.84231769, 35.08187929, 38.74493311,
39.4405081 , 38.58747047, 39.47871754, 39.45154604, 41.78485557,
42.35227769, 41.07952048, 45.35823862, 41.19693688, 45.0433537 ,
45.99319315, 48.21473796, 39.54430984, 34.81809926, 38.81258149,
40.11778774, 47.2337074 , 36.17617534, 38.91654676, 43.92466398,
36.60602781, 34.4926639 , 34.96513222, 38.22353762, 47.96154242,
39.51694972, 38.60784384, 35.21748246, 39.79092255, 38.87296785,
39.55476612, 33.03167073, 44.20392805, 47.13032412, 44.74140699,
41.11094977, 42.5100963 , 40.00163418, 33.80012866, 40.5298266 ,
33.61281434, 40.82824318, 33.69720611, 39.10313263, 34.51646641,
39.45203596, 37.35629386, 41.93630145, 38.19183988, 39.3892859 ,
42.67558566, 41.09552415, 47.17112562, 42.60042019, 38.95586586,
44.04248329, 46.97846758, 45.61390859, 39.62473743, 35.94548354,
40.20412708, 50.22299587, 40.2903042 , 41.22244796, 31.93634823,
34.38358125, 32.19139889, 44.05212697, 43.46755765, 43.837489 ,
37.10410812, 39.06742424, 39.99114464, 37.28324809, 45.56254526,

33.15380374, 43.51180802, 32.88097149, 32.0916217 , 45.15509355,
47.64969413, 47.39519912, 43.18556575, 39.78406722, 36.05688493,
32.46420606, 32.41295073, 39.59123733, 42.58212242, 38.48903447,
48.50074093, 45.7535747 , 45.17845106, 33.46399181, 38.74912933,
38.48876649, 44.59031465, 37.60411356, 39.5063999 , 43.8614742 ,
48.5268603 , 47.29148262, 37.81044894, 33.78445504, 35.19853563,
39.10784002, 50.1282358 , 43.16613496, 38.53405213, 31.72879149,
34.77665996, 41.37107521, 38.42545666, 33.93974475, 41.01319654,
37.09384236, 36.40866817, 43.67160791, 29.06716439, 39.74690478,
32.53255901, 40.65219677, 43.09323546, 44.60141246, 38.38851799,
37.09441442, 38.87790921, 38.51509719, 33.95457266, 47.83188399,
41.32063264, 37.62931583, 38.94806647, 38.41776114, 40.310286 ,
39.96628146, 37.33343385, 43.77700662, 36.9065766 , 47.78252335,
37.96331654, 39.70137783, 45.0928546 , 36.83577671, 42.01944487,
39.37394721, 39.28578501, 37.47623033, 37.95984891, 30.60343065,
34.67484211, 37.7296572 , 35.58942434, 36.8402414 , 43.85996484,
38.58707849, 34.00688599, 40.0580377 , 46.72787315, 38.21123697,
35.25662685, 42.07775259, 45.44491695, 46.26138514, 33.45623582,
38.11417545, 33.05071095, 37.2446726 , 33.73211997, 38.77730414,
41.09799453, 38.805092 , 38.79475398, 45.52383615, 40.43810144,
37.98926923, 42.58663739, 34.59330947, 36.84117237, 41.57609739,
40.92662675, 33.83407096, 33.29424122, 44.24359465, 42.59710858,
37.97913985, 45.86771837, 40.29508106, 40.61033967, 38.1482425 ,
36.24069177, 43.92134542, 42.04854628, 43.76687398, 42.60137123,
40.44656633, 47.80499383, 47.32139209, 41.13169634, 32.2034527 ,
56.48129613, 38.00629826, 30.86560183, 43.48438655, 42.61500809,
39.07964153, 36.22909447, 44.68760249, 40.27214476, 36.05409747,
34.59639776, 39.93668097, 41.59834224, 33.72825196, 41.5643372 ,
42.58126259, 32.65184442, 34.18963273, 32.03671827, 48.54754236,
41.38380252, 49.66613153, 36.25141871, 33.3420636 , 33.91004901,
41.39638027, 29.93355154, 31.60462041, 40.62289091, 46.52684639,
42.83191293, 35.89082672, 35.20134861, 30.83470433, 36.51686821,
46.28331177, 41.9441457 , 47.22833625, 40.69092305, 45.39032019,
43.94261278, 32.55154676, 33.70950608, 45.00426536, 45.89575745,
34.95046803, 45.86362215, 33.85268366, 40.33558807, 36.79130013,
46.91182436, 33.56200913, 30.45758894, 33.25622924, 39.58745641,
43.60744426, 37.67422114, 37.00110888, 41.91658574, 39.97174451,
33.59778364, 49.65465145, 36.48379272, 39.76423083, 37.19373412,
46.1084865 , 42.23035259, 44.37212423, 44.38045129, 41.78865924,
38.6208447 , 36.21647666, 31.80396943, 32.3463412 , 39.24151485,
35.22461573, 45.79041231, 46.06767811, 38.69677206, 38.10686407,
45.13146442, 44.61221272, 35.29487022, 41.15674939, 43.25885726,
50.28514381, 37.16109745, 43.6087213 , 53.97135778, 44.28903801,
39.41069978, 42.56724857, 34.62023169, 39.69451135, 37.21832111,
34.50098518, 41.57672068, 48.19732142, 33.47018128, 36.73138247,
48.19390811, 41.87547782, 43.92634658, 36.5105875 , 41.5841466 ,
35.2181018 , 42.07758508, 39.87808162, 39.44717777, 37.29571797,
46.25339173, 49.89187851, 30.80685679, 47.29932092, 40.60577705,
30.05417493, 38.22361765, 33.05009535, 38.19999894, 39.43891719,
42.3148982 , 40.71057257, 29.06603354, 36.15392497, 39.18312808,
42.65300439, 39.69005946, 35.55168885, 38.11617399, 44.96845623,
34.82328536, 40.99031102, 43.72768505, 36.84537453, 45.27425933,
50.96176437, 35.04567496, 41.93134088, 49.21807532, 53.2396242 ,
30.85032838, 44.8487671 , 41.72835443, 42.81278273, 37.66371013,
43.95880615, 31.73142117, 37.91458483, 36.46469953, 41.04816847,
35.56261492, 41.10728771, 41.08870355, 48.19801047, 40.1193527 ,

```
38.71623105, 36.87164868, 38.72262223, 40.31495596, 40.66420881,
43.73436077, 35.43704461, 43.5667864 , 46.16128699, 38.96573958,
39.02881243, 45.73995812, 35.53277897, 33.73979234, 37.42302765,
32.36638682, 43.54481449, 43.41654959, 46.82177565, 37.05170445,
40.19063304, 40.61996347, 47.7650155 , 43.0942194 , 34.75529174,
33.88765741, 38.48742675, 47.11902128, 41.86047358, 38.64916659,
39.16012596, 35.69207661, 41.90657287, 37.76441623, 45.03634282,
42.07675947, 40.46874695, 43.40912603, 41.33143884, 41.7892479 ,
37.35433957, 33.26213482, 36.66077724, 37.05979559, 36.4112256 ,
43.44243166, 39.63799494, 43.12758139, 42.17119366, 40.44630791,
45.12101699, 45.79246155, 35.67652054, 48.60779773, 35.54542303,
45.34980528, 38.49531956, 32.66693871, 44.68591768, 36.49374902,
38.89777123, 36.74746119, 32.49366031, 41.73494415, 37.25783225,
34.29917524, 43.38704266, 32.60651623, 36.17650753, 36.70272429,
47.9955382 , 35.85351958, 41.04891792, 39.21746955, 35.6462143 ,
39.68003759, 37.13235631, 29.76839436, 45.9382615 , 41.63696941,
39.97541395, 31.32947876, 38.66748433, 32.06198626, 32.18149597,
51.34095745, 47.81401937, 36.73498044, 33.56886814, 37.36511436,
38.68516558, 48.7667802 , 34.14656569, 42.59886985, 39.66546371,
48.50099794, 47.44475619, 38.11630136, 43.5232261 , 33.6047126 ,
33.01732081, 44.28598996, 46.04920606, 39.93670717, 39.53757936,
36.50989865, 30.35513059, 39.80639187, 37.74946303, 39.74655639,
42.73191339, 42.94893744, 32.87875268, 35.60559787, 45.12739418,
50.23313674, 44.22252405, 35.63752964, 33.23795533, 40.47367248,
41.50442278, 45.34089528, 41.93697061, 41.47118265, 42.00018698,
34.87831968, 33.02490559, 41.0716431 , 38.85396665, 46.5998578 ,
36.63961244, 40.84224215, 34.81801547, 33.02117646, 43.72575146,
40.19420538, 42.42391909, 46.94989362, 35.11205444, 43.63595903,
41.80183804, 36.17335712, 50.67532696, 42.00510943, 41.82782991,
38.77673311, 35.61904536, 44.03625544, 35.60543877, 41.76678583,
45.99987355, 36.49662574, 32.02258996, 37.35312105, 38.06496731,
40.45234077, 50.82588688, 43.57984472, 38.6438806 , 21.68290272])
```

In [42]:
```
#plot histogram
plt.hist(sample, bins=50)
plt.show()
```



In [43]:
```
#since its nt following any distribution so we have to apply non-parametric
```

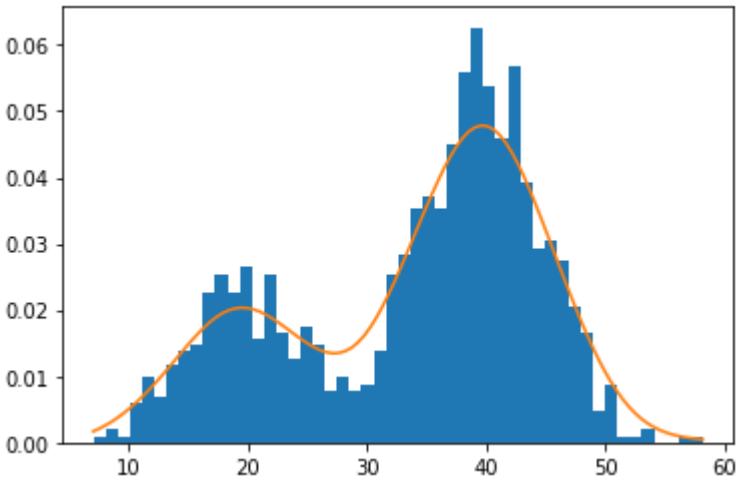
In [45]:
```
from sklearn.neighbors import KernelDensity

model = KernelDensity(bandwidth=3, kernel='gaussian')

#convert data to 2d array bcoz all ml algo works on 2d data
sample = sample.reshape(len(sample),1)

model.fit(sample)
```

Out[45]:
```
▼      KernelDensity

KernelDensity(bandwidth=3)
```

In [47]:
```
#generating 100 points between min & max value of data
values = np.linspace(sample.min(), sample.max(),100)

#converting data into 2d
values = values.reshape(len(values),1)
```

In [48]:
```
probabilities = model.score_samples(values)

#kde returns log of probability density so converting into exponents
probabilities = np.exp(probabilities)
```

In [50]:
```
plt.hist(sample, bins=50, density=True)
plt.plot(values[:], probabilities)
plt.show()
```

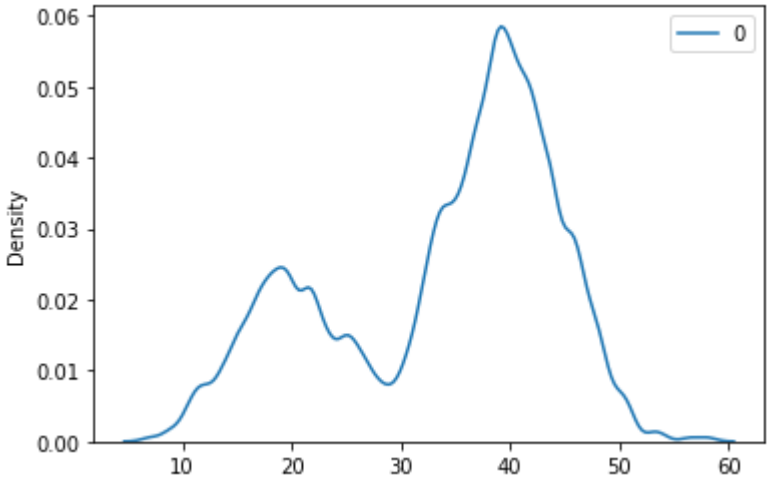


In [51]:
```
#if we decrease bandwidth then it will peakeds & when we increase it will be
#so we need to do some experiment to get right bandwidth
```

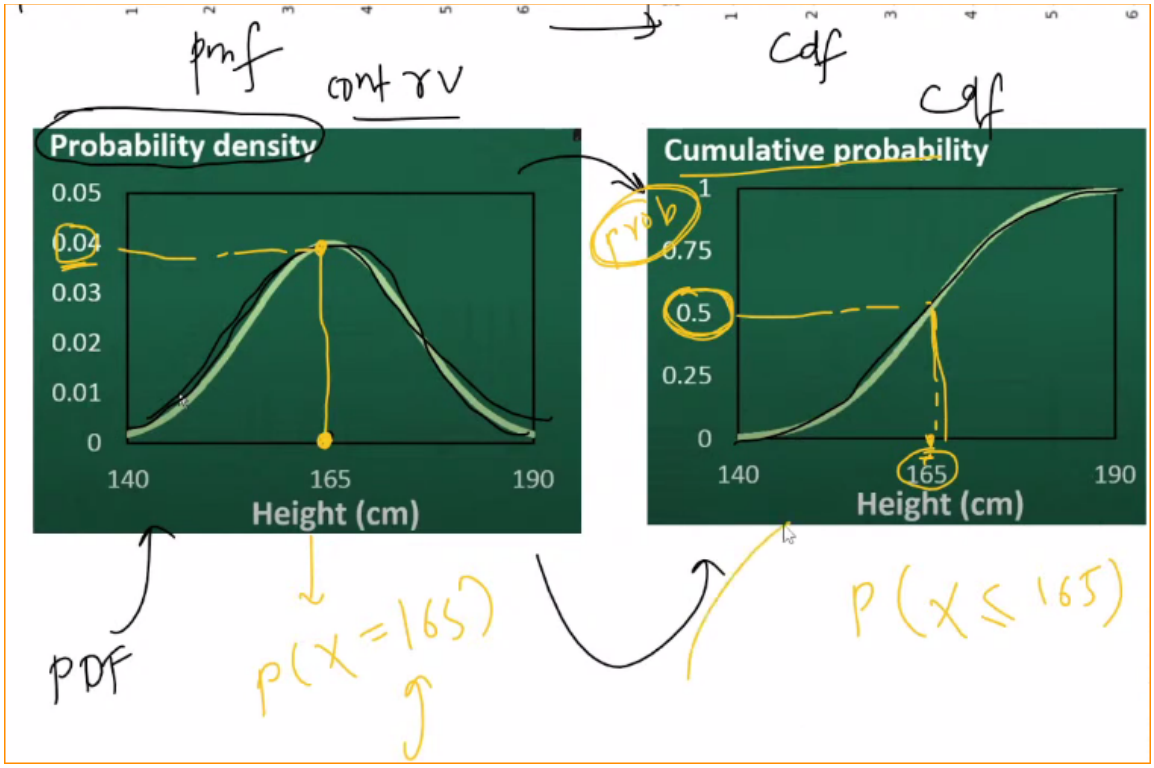
In [60]:
```
#here you can adjust bw & see the difference
sns.kdeplot(sample,bw_adjust=0.3)
```

```
C:\Users\iampr\AppData\Local\Programs\Python\Python310\lib\site-packages\seaborn\_oldcore.py:1119: FutureWarning: use_inf_as_na option is deprecated and will be removed in a future version. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
```

Out[60]: <AxesSubplot:ylabel='Density'>



CDF for PDF



In [61]:
```
#In PDF we're getting probability density exactly of that point
#In CDF, we're getting probability of values less then equal to that value
#PDF integration will give CDF & CDF differentiation will give PDF
```

- In probability theory, the relationship between the probability density function (PDF) and the cumulative distribution function (CDF) is based on integration and differentiation.
- PDF (Probability Density Function): This function gives the probability density at each point in the sample space. Integrating the PDF over a range gives the probability of observing a random variable within that range.
- CDF (Cumulative Distribution Function): This function gives the probability that a random variable takes on a value less than or equal to a certain point. It's the cumulative sum of probabilities up to that point.
- The relationship between them is:
- Integrating the PDF over a range gives you the probability (area under the curve) of observing the random variable within that range.
- Differentiating the CDF gives you the PDF.

How PDF is used on Real Data?

iris data

In [1]:
```
import seaborn as sns
```

In [2]:
```
df = sns.load_dataset('iris')
```

In [3]:
```
df.head()
```

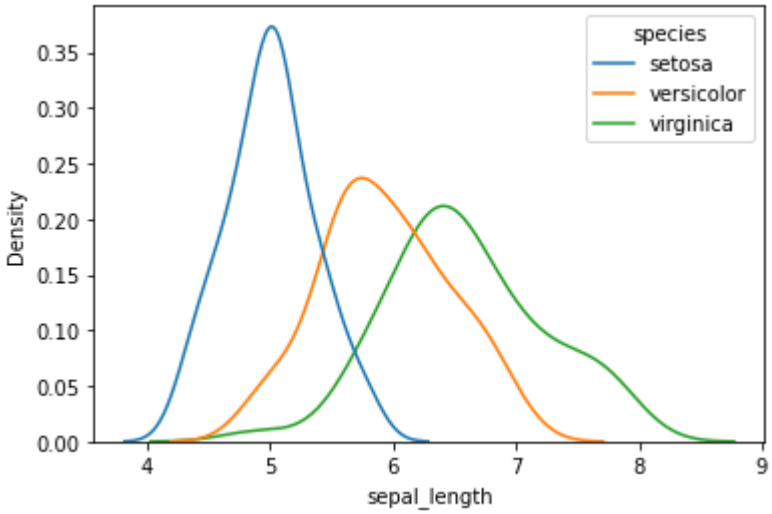
Out[3]:

	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 [9]:
```
#we have iris dataset where we have to predict flower type based on sepal &
#we do feature selection for these type of problems --> how to find those fe
#plot pdf of data we have since data is a continous random variable

sns.kdeplot(data=df, x='sepal_length', hue='species')
```

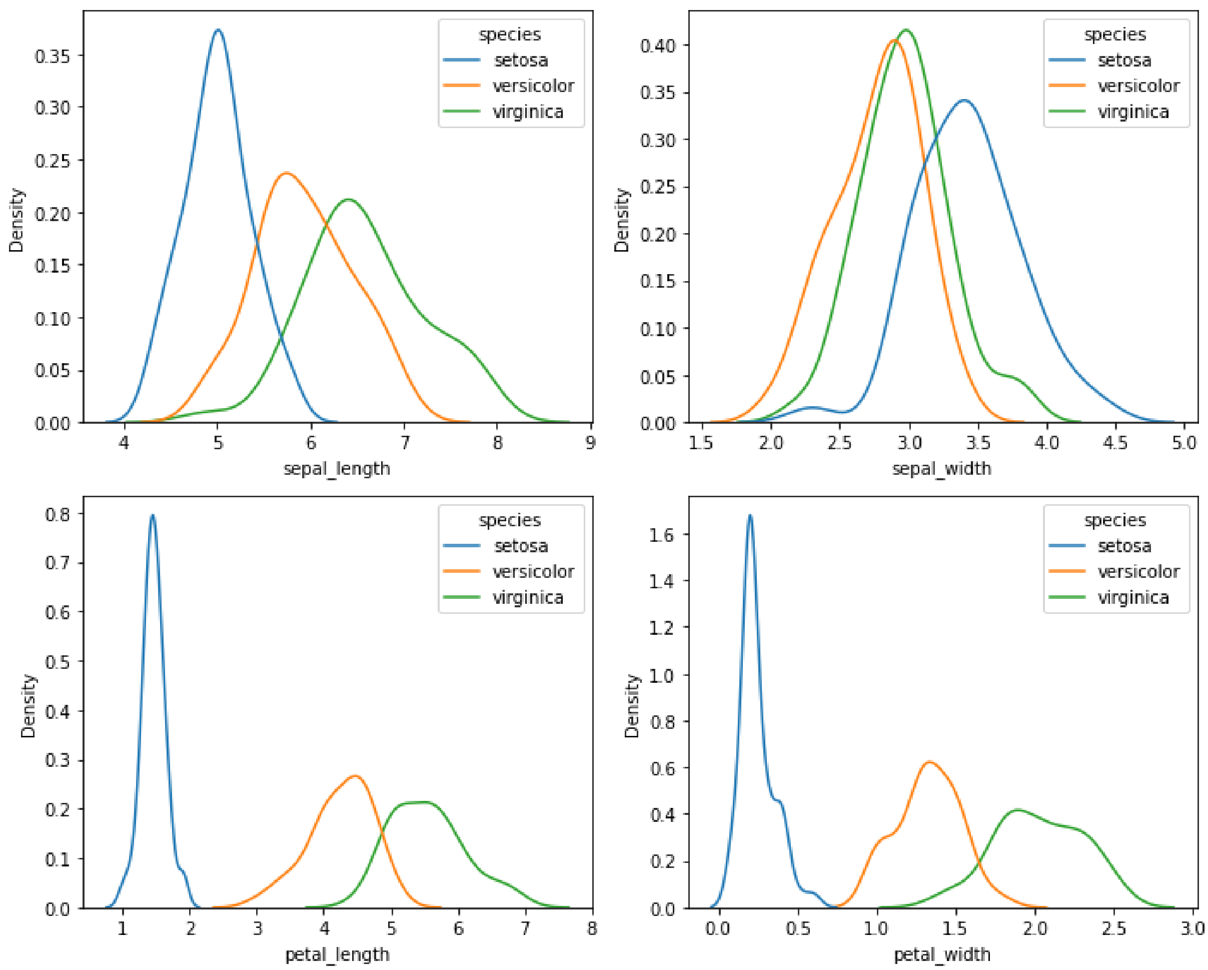
```
C:\Users\iampr\AppData\Local\Programs\Python\Python310\lib\site-packages\seaborn\_oldcore.py:1119: FutureWarning: use_inf_as_na option is deprecated and will be removed in a future version. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
```

Out[9]: <AxesSubplot:xlabel='sepal_length', ylabel='Density'>



In [12]:
```
import seaborn as sns
import pandas as pd
import matplotlib.pyplot as plt


# Create a figure with subplots
fig, axes = plt.subplots(2, 2, figsize=(10, 8))

# Iterate over each feature and plot its KDE plot
for i, feature in enumerate(['sepal_length', 'sepal_width', 'petal_length',
    sns.kdeplot(data=df, x=feature, hue='species', ax=axes[i//2, i%2])

plt.tight_layout()
plt.show()
```

```
C:\Users\iampr\AppData\Local\Programs\Python\Python310\lib\site-packages\sea
born\_oldcore.py:1119: FutureWarning: use_inf_as_na option is deprecated and
will be removed in a future version. Convert inf values to NaN before operat
ing instead.
  with pd.option_context('mode.use_inf_as_na', True):
C:\Users\iampr\AppData\Local\Programs\Python\Python310\lib\site-packages\sea
born\_oldcore.py:1119: FutureWarning: use_inf_as_na option is deprecated and
will be removed in a future version. Convert inf values to NaN before operat
ing instead.
  with pd.option_context('mode.use_inf_as_na', True):
C:\Users\iampr\AppData\Local\Programs\Python\Python310\lib\site-packages\sea
born\_oldcore.py:1119: FutureWarning: use_inf_as_na option is deprecated and
will be removed in a future version. Convert inf values to NaN before operat
ing instead.
  with pd.option_context('mode.use_inf_as_na', True):
C:\Users\iampr\AppData\Local\Programs\Python\Python310\lib\site-packages\sea
born\_oldcore.py:1119: FutureWarning: use_inf_as_na option is deprecated and
will be removed in a future version. Convert inf values to NaN before operat
ing instead.
  with pd.option_context('mode.use_inf_as_na', True):
```



In [14]:
```
#how to decice which column is useful?
#our goal is to determin species of a flower from the given data
#since petal length & petal width is able to easily differentiate all 3 spec
#if petal length < 2.1 then its setosa else it will be versicolor or virgini
#if petal length is < 5 & > 2.3 then more chances that it is versicolor else
#in sepal length & sepal width, we'll have issue to differentiate, ml algori
```

titanic

In [15]:
```
titanic = pd.read_csv('https://raw.githubusercontent.com/datasciencedojo/dat
```

In [16]:
```
titanic.head()
```

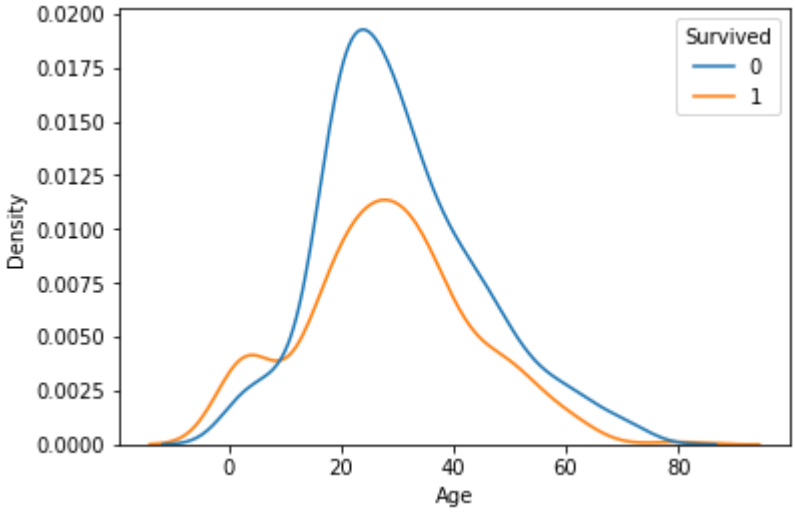
Out[16]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Tic
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	21
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/ 3101
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373

In [18]:
```
sns.kdeplot(data=titanic, x='Age', hue='Survived')
```

```
C:\Users\iampr\AppData\Local\Programs\Python\Python310\lib\site-packages\sea
born\_oldcore.py:1119: FutureWarning: use_inf_as_na option is deprecated and
will be removed in a future version. Convert inf values to NaN before operat
ing instead.
  with pd.option_context('mode.use_inf_as_na', True):
```

Out[18]:
```
<AxesSubplot:xlabel='Age', ylabel='Density'>
```

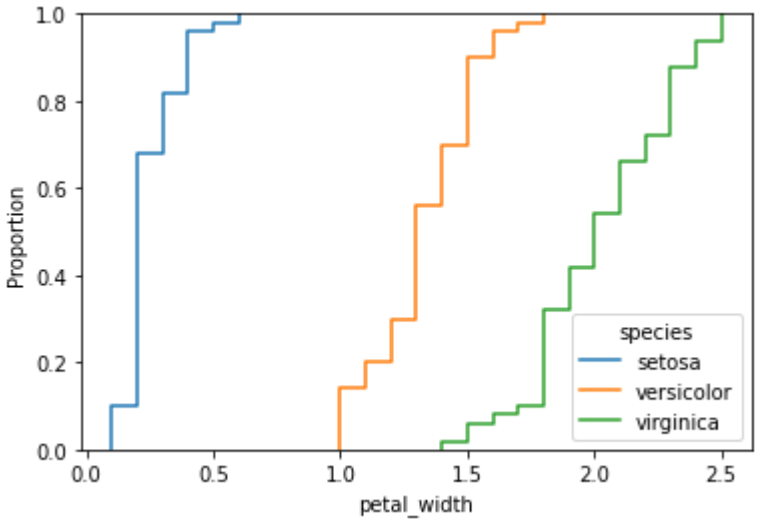


In [23]:
```
#blue represent died & orance is survived
#from 0 to 8, survival probability denisty is more then dying,after 8 dying
#even if you're elder till 80
```

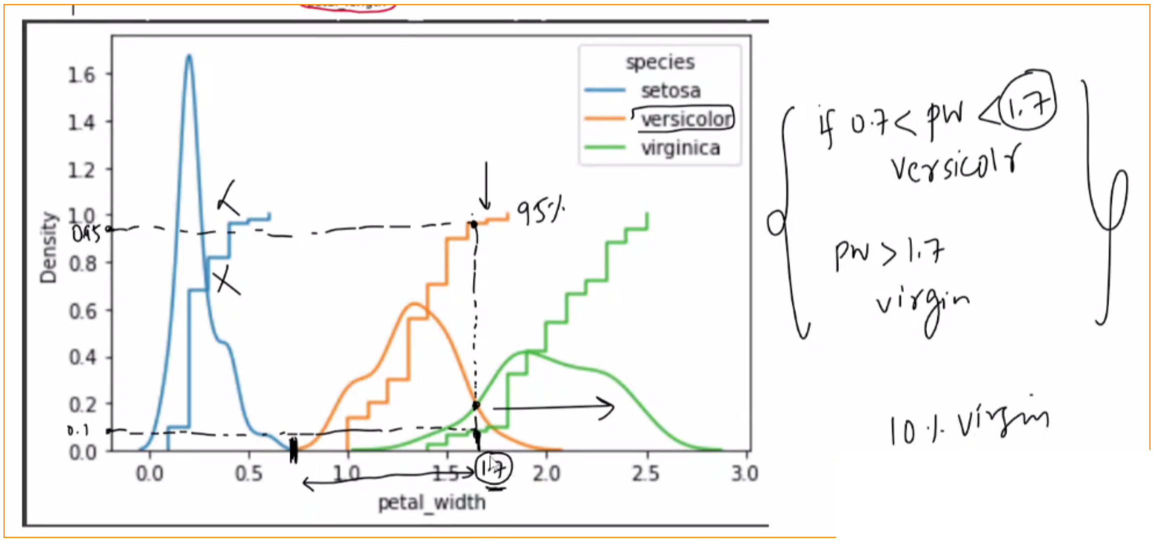
In [25]:
```
sns.ecdfplot(data=df,x='petal_width',hue='species')
```

```
C:\Users\iampr\AppData\Local\Programs\Python\Python310\lib\site-packages\seaborn\_oldcore.py:1119: FutureWarning: use_inf_as_na option is deprecated and will be removed in a future version. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
```

Out[25]: <AxesSubplot:xlabel='petal_width', ylabel='Proportion'>



How CDF helps?



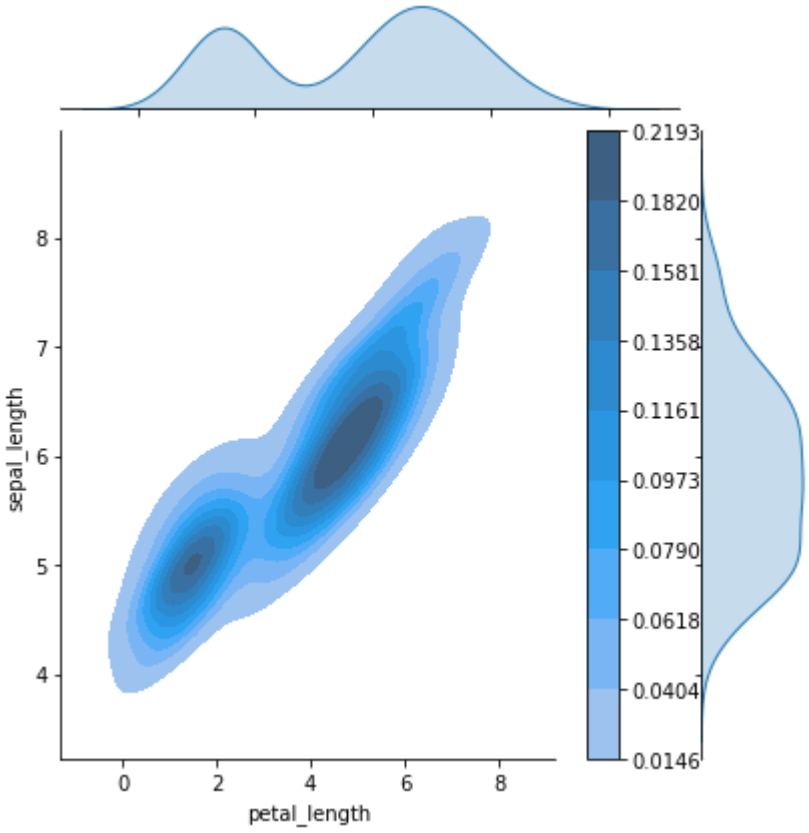

```
In [29]: #lets say we have made a rule using pdf that > 0.7 & < 1.7 is versicolor
         #To verify we'll use CDF, Now CDF telling us 95% flowers of this range are v
         #this means I'll be correct 95% times so we can quantify using CDF
```

2D density plot

```
In [30]: sns.jointplot(data=df, x='petal_length', y='sepal_length', kind='kde', fill=
```

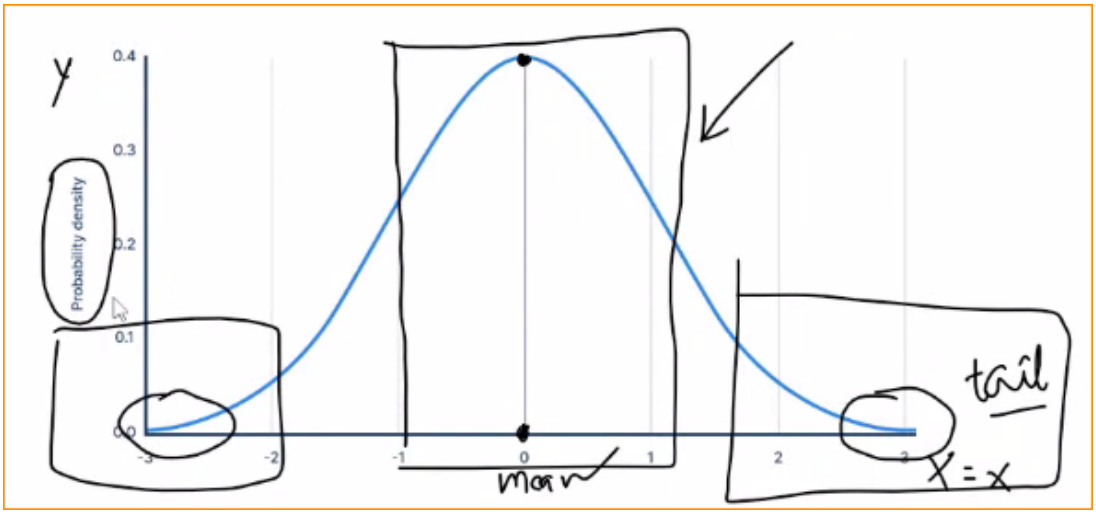
```
C:\Users\iampr\AppData\Local\Programs\Python\Python310\lib\site-packages\sea
born\_oldcore.py:1119: FutureWarning: use_inf_as_na option is deprecated and
will be removed in a future version. Convert inf values to NaN before operat
ing instead.
  with pd.option_context('mode.use_inf_as_na', True):
C:\Users\iampr\AppData\Local\Programs\Python\Python310\lib\site-packages\sea
born\_oldcore.py:1119: FutureWarning: use_inf_as_na option is deprecated and
will be removed in a future version. Convert inf values to NaN before operat
ing instead.
  with pd.option_context('mode.use_inf_as_na', True):
C:\Users\iampr\AppData\Local\Programs\Python\Python310\lib\site-packages\sea
born\_oldcore.py:1119: FutureWarning: use_inf_as_na option is deprecated and
will be removed in a future version. Convert inf values to NaN before operat
ing instead.
  with pd.option_context('mode.use_inf_as_na', True):
C:\Users\iampr\AppData\Local\Programs\Python\Python310\lib\site-packages\sea
born\_oldcore.py:1119: FutureWarning: use_inf_as_na option is deprecated and
will be removed in a future version. Convert inf values to NaN before operat
ing instead.
  with pd.option_context('mode.use_inf_as_na', True):
```

```
Out[30]: <seaborn.axisgrid.JointGrid at 0x1fec6af6b60>
```



Normal Distribution

- Its also known as "Bell Curve" or "Gaussian Distribution"
- Its a PDF of continous random variable



- Most of data points lie around the center & very few lie towards the side
- tail never touches x-axis --> they touch on infinity
- we need mean & std to plot pdf of continous data
- mean represents ceneter of the data & std represents spread of the data

Why its so important?

- Normal distribution is very common in nature, many natural phenomenon follows normal distribution such as heights of people, weights of objects, IQ score of population
- It has been researched a lot so we know a lot about normal distribution already thats we always try our data to follow normal distribution

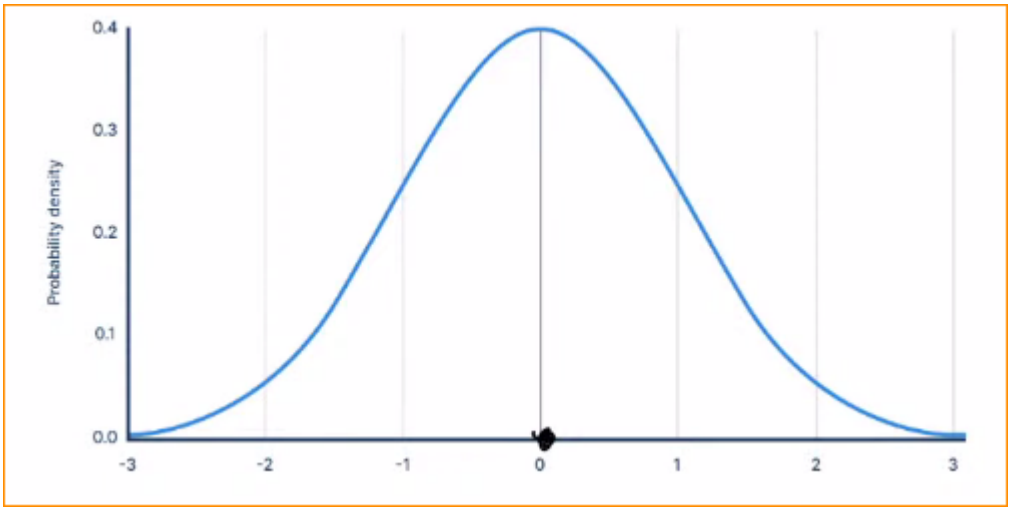
Whats is the impact of mean & std on pdf graph

In [32]:
```
# https://samp-suman-normal-dist-visualize-app-lkntug.streamlit.app
```

Standard Normal Variate

- A normal distribution with mean 0 & std 1
- Its denoted by Z, also called z-score
- Standardizing variables to z-scores transforms the original data distribution into a standard normal distribution with a mean of 0 and a standard deviation of 1. This normalization process allows for easier comparison of variables that originally had different scales or units.
- Z-scores help identify outliers by flagging data points that fall far from the mean of the distribution. Observations with z-scores beyond a certain threshold (e.g., ±3 standard deviations) are considered potential outliers, facilitating outlier detection and data cleaning processes.
- In machine learning algorithms, standardizing features to z-scores is a common preprocessing step. Standardization ensures that features contribute equally to model training and prevents features with larger scales

from dominating the learning process.



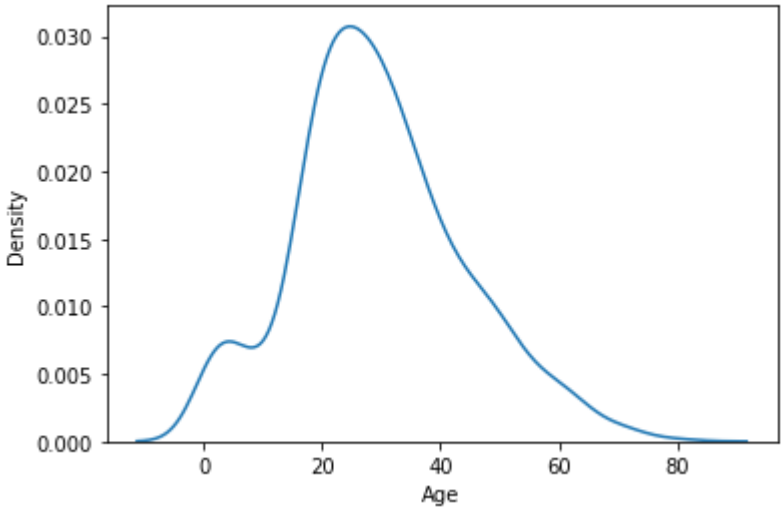
How to transform normal distribution to standard normal variate?

z-score = (datapoint - mean of data)/std

In [36]:
```
sns.kdeplot(titanic['Age'])
```

```
C:\Users\iampr\AppData\Local\Programs\Python\Python310\lib\site-packages\seaborn\_oldcore.py:1119: FutureWarning: use_inf_as_na option is deprecated and will be removed in a future version. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
```

Out[36]: <AxesSubplot:xlabel='Age', ylabel='Density'>



In [37]:
```
titanic['Age'].mean()
```

Out[37]: 29.69911764705882

In [38]:
```
titanic['Age'].std()
```

Out[38]: 14.526497332334042

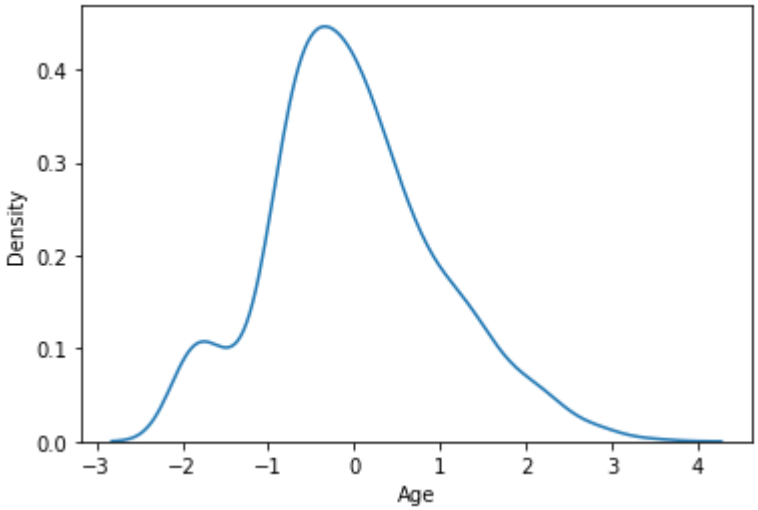
In [39]:
```
#we need to convert (29,14) into (0,1)
```

In [42]:
```
x = (titanic['Age'] - titanic['Age'].mean())/titanic['Age'].std()
```

In [43]:
```
sns.kdeplot(x)
```

```
C:\Users\iampr\AppData\Local\Programs\Python\Python310\lib\site-packages\seaborn\_oldcore.py:1119: FutureWarning: use_inf_as_na option is deprecated and will be removed in a future version. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
```

Out[43]: <AxesSubplot:xlabel='Age', ylabel='Density'>



In [44]:
```
#graph is similar but its transformed, now mean is 0 & std is 1
```

In [46]:
```
#mean is very close to 0
x.mean()
```

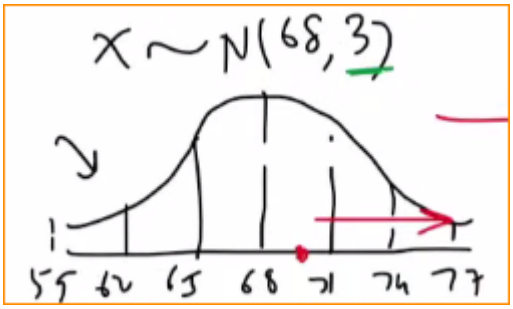
Out[46]: 2.338621049070358e-16

In [48]:
```
#same as std
x.std()
```

Out[48]: 1.0

Whats the benefit?

In [49]:
```
#suppose the height of adult males in a certain population follow a normal d
#whats the probability that a randomly seected adult male from this populati
```



In [50]:
```
#find zscore
z = (72-68)/3
print(z)
```

```
1.3333333333333333
```

In [51]:
```
#now we can find the zscore in z-table
```

z	.00	.01	.02	.03	.04	.05	.06	.07	.08	.09
−3.4	.0003	.0003	.0003	.0003	.0003	.0003	.0003	.0003	.0003	.0002
−3.3	.0005	.0005	.0005	.0004	.0004	.0004	.0004	.0004	.0004	.0003
−3.2	.0007	.0007	.0006	.0006	.0006	.0006	.0006	.0005	.0005	.0005
−3.1	.0010	.0009	.0009	.0009	.0008	.0008	.0008	.0008	.0007	.0007
−3.0	.0013	.0013	.0013	.0012	.0012	.0011	.0011	.0011	.0010	.0010
−2.9	.0019	.0018	.0018	.0017	.0016	.0016	.0015	.0015	.0014	.0014
−2.8	.0026	.0025	.0024	.0023	.0023	.0022	.0021	.0021	.0020	.0019
−2.7	.0035	.0034	.0033	.0032	.0031	.0030	.0029	.0028	.0027	.0026
−2.6	.0047	.0045	.0044	.0043	.0041	.0040	.0039	.0038	.0037	.0036
−2.5	.0062	.0060	.0059	.0057	.0055	.0054	.0052	.0051	.0049	.0048
−2.4	.0082	.0080	.0078	.0075	.0073	.0071	.0069	.0068	.0066	.0064
−2.3	.0107	.0104	.0102	.0099	.0096	.0094	.0091	.0089	.0087	.0084
−2.2	.0139	.0136	.0132	.0129	.0125	.0122	.0119	.0116	.0113	.0110
−2.1	.0179	.0174	.0170	.0166	.0162	.0158	.0154	.0150	.0146	.0143
−2.0	.0228	.0222	.0217	.0212	.0207	.0202	.0197	.0192	.0188	.0183
−1.9	.0287	.0281	.0274	.0268	.0262	.0256	.0250	.0244	.0239	.0233
−1.8	.0359	.0351	.0344	.0336	.0329	.0322	.0314	.0307	.0301	.0294
−1.7	.0446	.0436	.0427	.0418	.0409	.0401	.0392	.0384	.0375	.0367
−1.6	.0548	.0537	.0526	.0516	.0505	.0495	.0485	.0475	.0465	.0455
−1.5	.0668	.0655	.0643	.0630	.0618	.0606	.0594	.0582	.0571	.0559
−1.4	.0808	.0793	.0778	.0764	.0749	.0735	.0721	.0708	.0694	.0681
−1.3	.0968	.0951	.0934	.0918	.0901	.0885	.0869	.0853	.0838	.0823
−1.2	.1151	.1131	.1112	.1093	.1075	.1056	.1038	.1020	.1003	.0985
−1.1	.1357	.1335	.1314	.1292	.1271	.1251	.1230	.1210	.1190	.1170
−1.0	.1587	.1562	.1539	.1515	.1492	.1469	.1446	.1423	.1401	.1379
−0.9	.1841	.1814	.1788	.1762	.1736	.1711	.1685	.1660	.1635	.1611
−0.8	.2119	.2090	.2061	.2033	.2005	.1977	.1949	.1922	.1894	.1867
−0.7	.2420	.2389	.2358	.2327	.2296	.2266	.2236	.2206	.2177	.2148
−0.6	.2743	.2709	.2676	.2643	.2611	.2578	.2546	.2514	.2483	.2451
−0.5	.3085	.3050	.3015	.2981	.2946	.2912	.2877	.2843	.2810	.2776
−0.4	.3446	.3409	.3372	.3336	.3300	.3264	.3228	.3192	.3156	.3121
−0.3	.3821	.3783	.3745	.3707	.3669	.3632	.3594	.3557	.3520	.3483
−0.2	.4207	.4168	.4129	.4090	.4052	.4013	.3974	.3936	.3897	.3859
−0.1	.4602	.4562	.4522	.4483	.4443	.4404	.4364	.4325	.4286	.4247
−0.0	.5000	.4960	.4920	.4880	.4840	.4801	.4761	.4721	.4681	.4641

z	.00	.01	.02	.03	.04	.05	.06	.07	.08	.09
0.0	.5000	.5040	.5080	.5120	.5160	.5199	.5239	.5279	.5319	.5359
0.1	.5398	.5438	.5478	.5517	.5557	.5596	.5636	.5675	.5714	.5753
0.2	.5793	.5832	.5871	.5910	.5948	.5987	.6026	.6064	.6103	.6141
0.3	.6179	.6217	.6255	.6293	.6331	.6368	.6406	.6443	.6480	.6517
0.4	.6554	.6591	.6628	.6664	.6700	.6736	.6772	.6808	.6844	.6879
0.5	.6915	.6950	.6985	.7019	.7054	.7088	.7123	.7157	.7190	.7224
0.6	.7257	.7291	.7324	.7357	.7389	.7422	.7454	.7486	.7517	.7549
0.7	.7580	.7611	.7642	.7673	.7704	.7734	.7764	.7794	.7823	.7852
0.8	.7881	.7910	.7939	.7967	.7995	.8023	.8051	.8078	.8106	.8133
0.9	.8159	.8186	.8212	.8238	.8264	.8289	.8315	.8340	.8365	.8389
1.0	.8413	.8438	.8461	.8485	.8508	.8531	.8554	.8577	.8599	.8621
1.1	.8643	.8665	.8686	.8708	.8729	.8749	.8770	.8790	.8810	.8830
1.2	.8849	.8869	.8888	.8907	.8925	.8944	.8962	.8980	.8997	.9015
1.3	.9032	.9049	.9066	.9082	.9099	.9115	.9131	.9147	.9162	.9177
1.4	.9192	.9207	.9222	.9236	.9251	.9265	.9279	.9292	.9306	.9319
1.5	.9332	.9345	.9357	.9370	.9382	.9394	.9406	.9418	.9429	.9441
1.6	.9452	.9463	.9474	.9484	.9495	.9505	.9515	.9525	.9535	.9545
1.7	.9554	.9564	.9573	.9582	.9591	.9599	.9608	.9616	.9625	.9633
1.8	.9641	.9649	.9656	.9664	.9671	.9678	.9686	.9693	.9699	.9706
1.9	.9713	.9719	.9726	.9732	.9738	.9744	.9750	.9756	.9761	.9767
2.0	.9772	.9778	.9783	.9788	.9793	.9798	.9803	.9808	.9812	.9817
2.1	.9821	.9826	.9830	.9834	.9838	.9842	.9846	.9850	.9854	.9857
2.2	.9861	.9864	.9868	.9871	.9875	.9878	.9881	.9884	.9887	.9890
2.3	.9893	.9896	.9898	.9901	.9904	.9906	.9909	.9911	.9913	.9916
2.4	.9918	.9920	.9922	.9925	.9927	.9929	.9931	.9932	.9934	.9936
2.5	.9938	.9940	.9941	.9943	.9945	.9946	.9948	.9949	.9951	.9952
2.6	.9953	.9955	.9956	.9957	.9959	.9960	.9961	.9962	.9963	.9964
2.7	.9965	.9966	.9967	.9968	.9969	.9970	.9971	.9972	.9973	.9974
2.8	.9974	.9975	.9976	.9977	.9977	.9978	.9979	.9979	.9980	.9981
2.9	.9981	.9982	.9982	.9983	.9984	.9984	.9985	.9985	.9986	.9986
3.0	.9987	.9987	.9987	.9988	.9988	.9989	.9989	.9989	.9990	.9990
3.1	.9990	.9991	.9991	.9991	.9992	.9992	.9992	.9992	.9993	.9993
3.2	.9993	.9993	.9994	.9994	.9994	.9994	.9994	.9995	.9995	.9995
3.3	.9995	.9995	.9995	.9996	.9996	.9996	.9996	.9996	.9996	.9997
3.4	.9997	.9997	.9997	.9997	.9997	.9997	.9997	.9997	.9997	.9998

In [55]:
```
#find 1st two in column & check next two in rows
#1.3 in column & 0.03 in rows --> 0.9082 --> 90.8% is the probability of adu
#so in order to find the probability that a randomly seected adult male from
#(100 - 90.82) = 9.02% is the probabilty
```

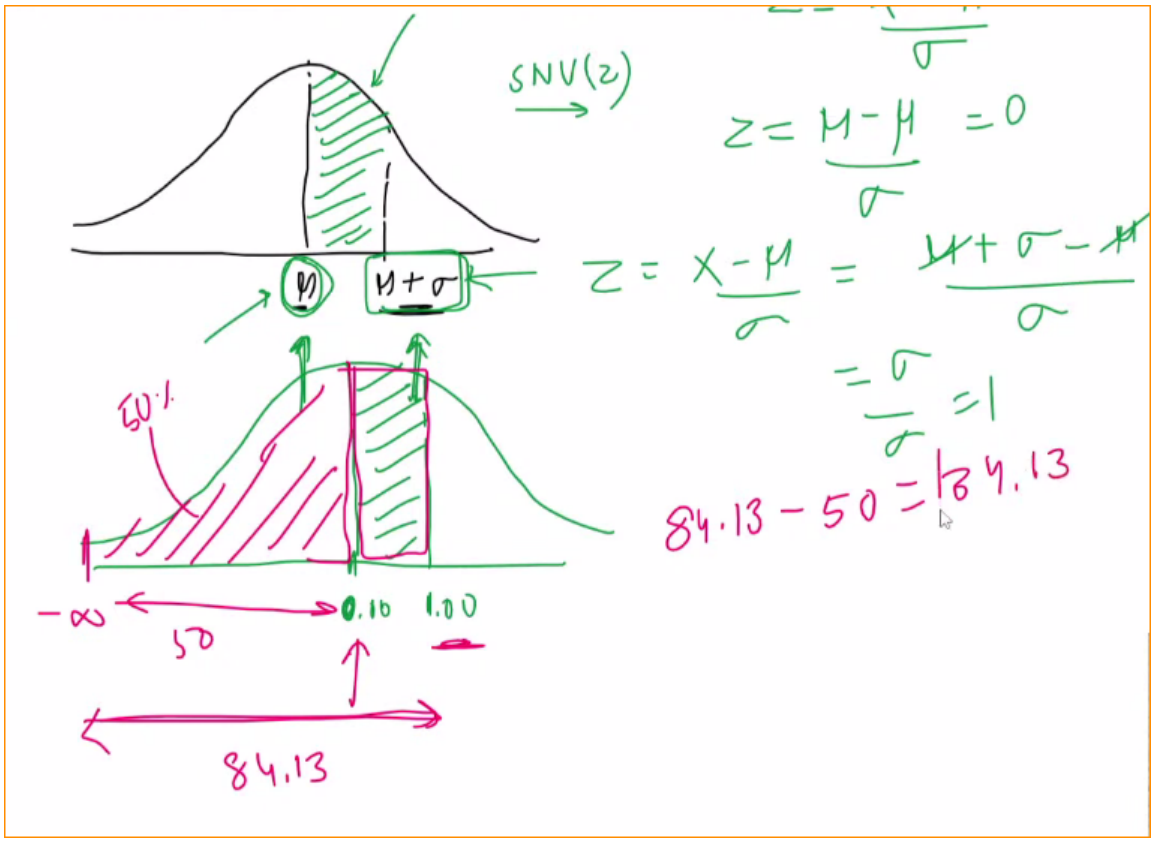
In [56]:
```
#so any normal distribution & convert to standrd normal variate
#then you can answer any probability question
```

- Suppose the scores on a standardized test follow a normal distribution with a mean of 500 and a standard deviation of 100. We want to find the probability that a randomly selected test-taker scores above 600.

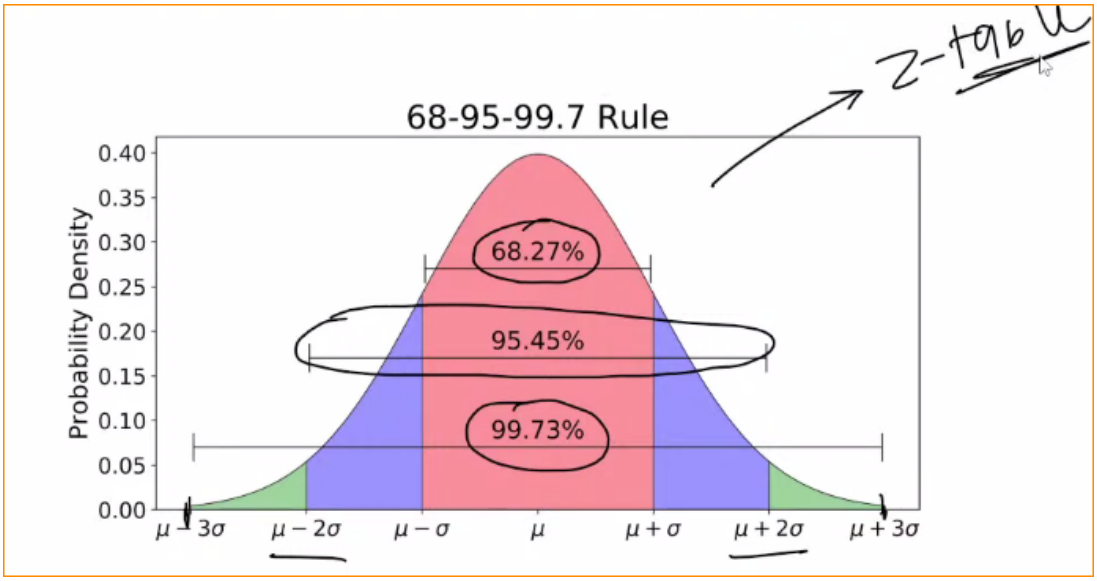
- First, we calculate the z-score: $$z = \frac{600 - 500}{100} = 1$$

- Next, we consult the standard normal distribution table (or use statistical software) to find the probability associated with a z-score of 1. Since the standard normal distribution is symmetrical, we can find the probability associated with z=1 and then subtract it from 1 to find the probability of being above z=1.
- rom the standard normal distribution table, we find that the probability associated with z=1 is approximately 0.8413. Therefore, the probability of scoring above 600 is: $1 - 0.8413 = 0.1587$
- So, there is approximately a 15.87% chance that a randomly selected test-taker will score above 600 on this standardized test.

for a normal distribution X, what % of population lie between mean & 1std, 2std & 3std?

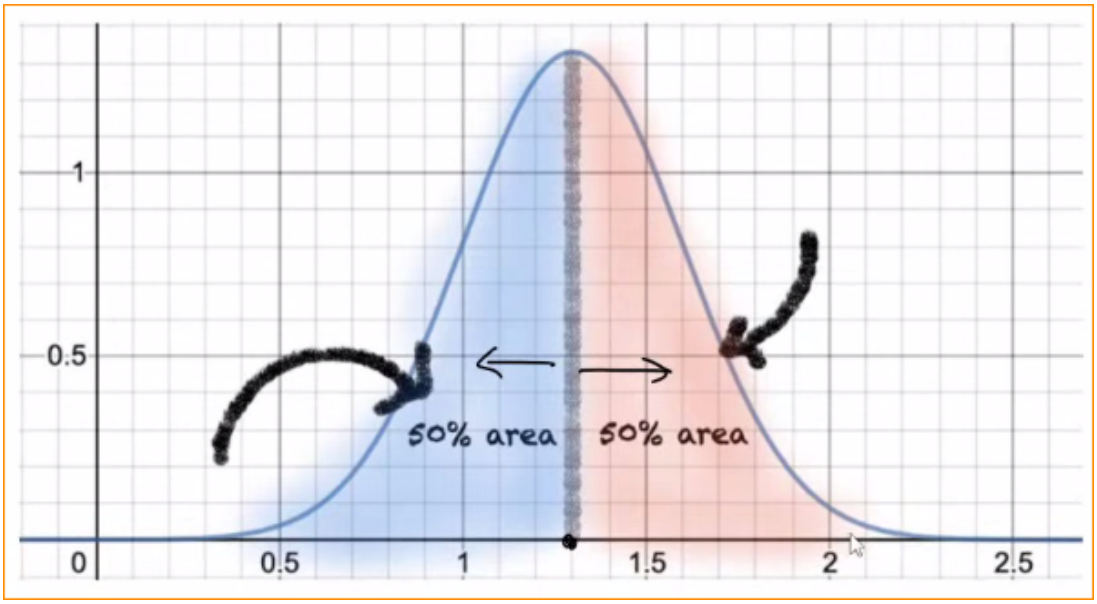


- this is also the emperical rule of normal distribution, which states thats



properties of normal distribution

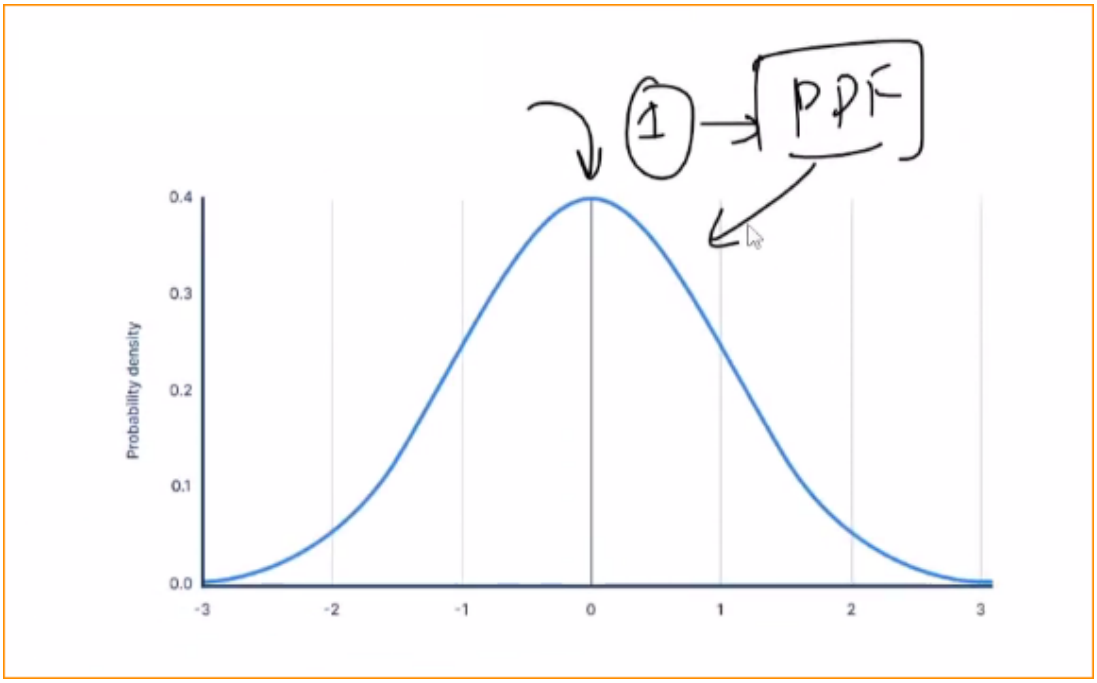
1. symmetricity - norma distribution is symmetric about its mean



2. measure of central tendancy are equal -- mean, median, mode
3. emperical rule - 68% data lies within 1 std, 95% data lies within 2 std, 99% data lies within 3std

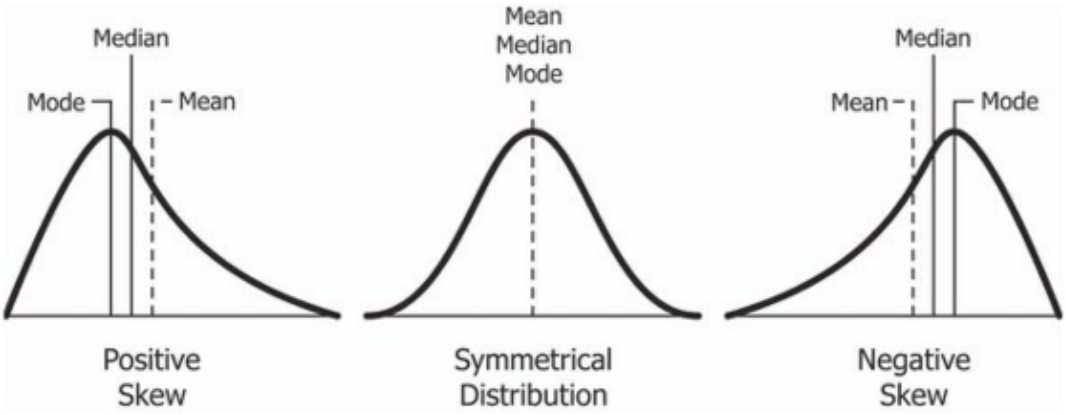
- dawn bradman example: https://www.linkedin.com/pulse/decoding-sporting-genius-sir-donald-bradman-marvel-taneja/

4. area under the curve is 1



Skewness

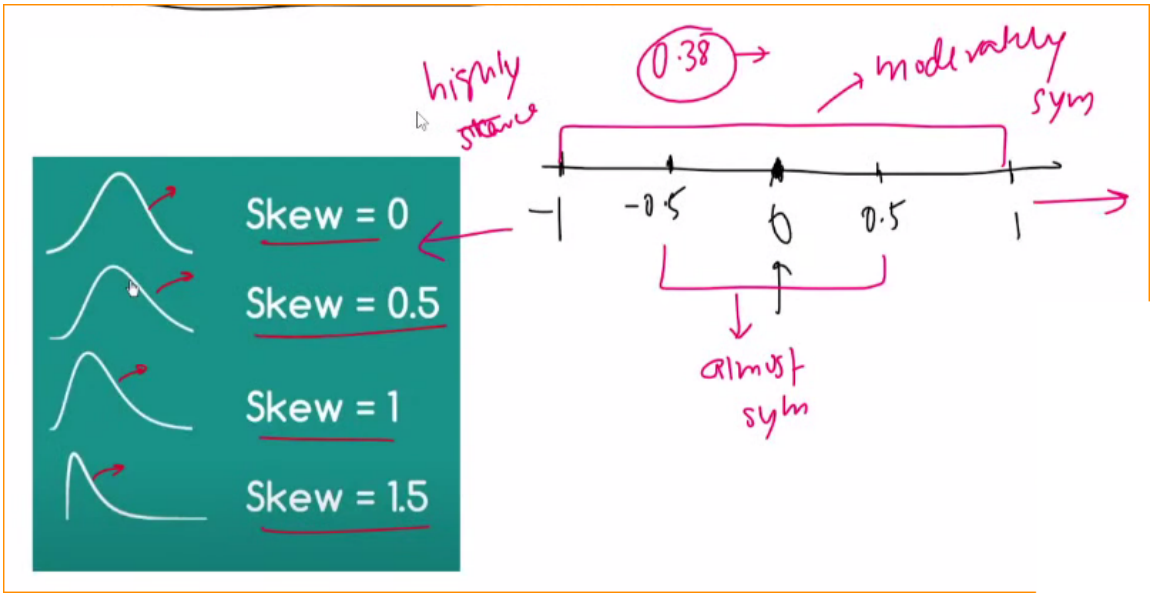
- While Normal distribution shows even distribution as per the property.. "Skewness" means data is not normally distributed, its not symmetrical
- Its a measure of asymmetry of a probability distribution
- Skewness describes the degree to which data diviates from normnal distribution
- In nomal or symmetric distribution --> mean, median, mode are equal but in skewed distribution, they are not equal.
- Distribution tend to have longer tail on either side
- Skewness can be positive, negative or zero
- Positive skewness means longer tail on right side, negative skewness means longer tail on left side and "zero skewness" means perfectly symmetric distribution
- greater the skew, greaer the distance between mean, median & mode



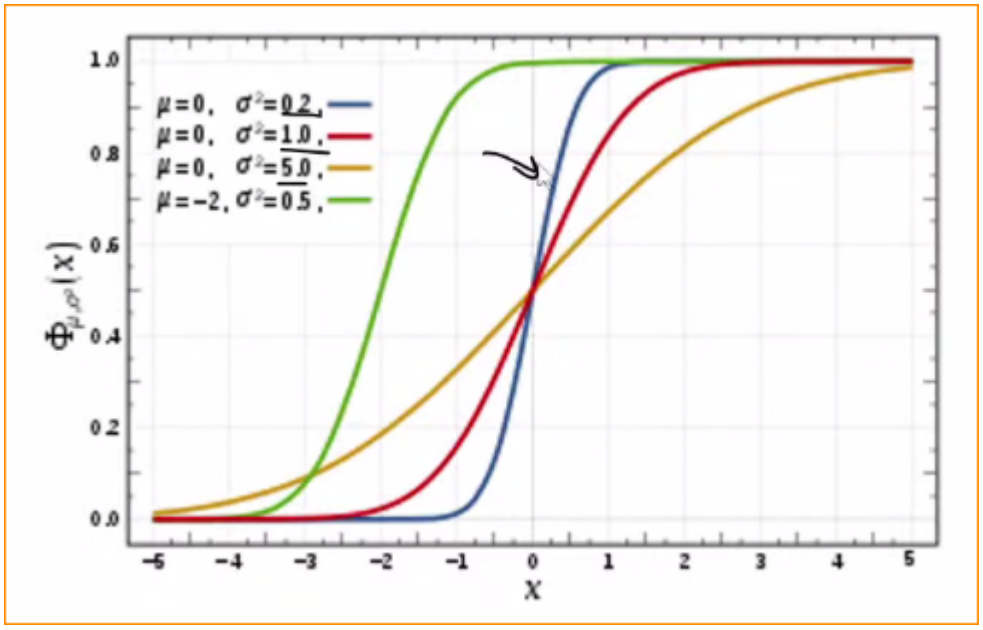
In [59]:
```
#its very close to 0

titanic['Age'].skew()
```

Out[59]: 0.38910778230082704

How to interpret skewness?



CDF of Normal Distribution



In [60]:
```
#integration of pdf is cdf --> diffrentiation of cdf is pdf
```

Normal Distribution use in Machine Learning

- Outlier detection
- Assumptions on data for ML algorithms -> Linear Regression and GMM
- Hypothesis Testing
- Central Limit Theorem

In [61]:
```
#since age is following normal distribution & as per emperical rule we know
#so they will be outliars

titanic['Age'].mean() + 3 * (titanic['Age'].std())
```

Out[61]: 73.27860964406094

In [62]:
```
titanic[titanic['Age'] > titanic['Age'].mean() + 3 * (titanic['Age'].std())]
```

Out[62]:

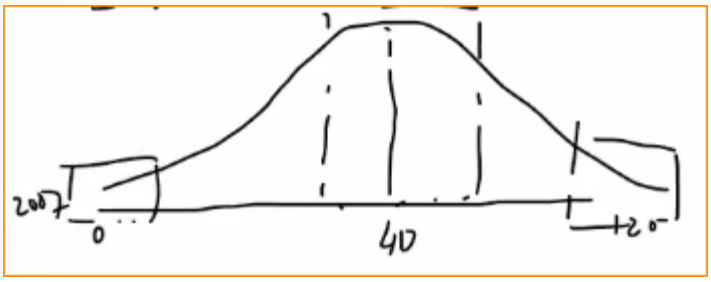
	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Tick
630	631	1	1	Barkworth, Mr. Algernon Henry Wilson	male	80.0	0	0	270
851	852	0	3	Svensson, Mr. Johan	male	74.0	0	0	3470

In [63]:
```
#above 2 can be considedred outliars
```

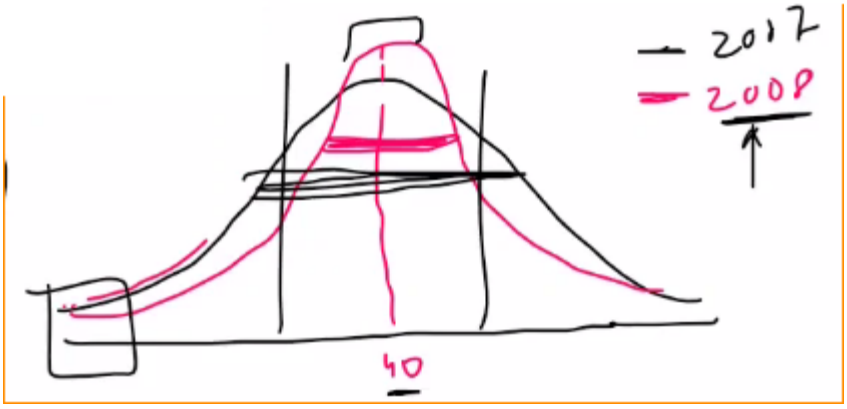
Kurtosis

- There are 4 moments in statistics
  1. Mean
  2. Std
  3. Skewness
  4. Kurtosis

- Kurtosis is the 4th moment in statistics
- Its a measure of taildness in probability distribution
- For example "Cricket"
- Sachin played 100 matches in 2007 & made runs with an average of 40
- Sachin played 100 matches in 2008 & made runs with an average of 40
- So as per mean, bth graph will look similar bcoz of same mean

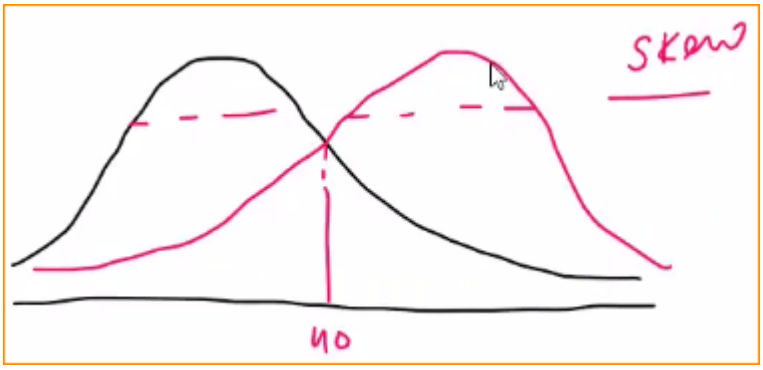


- Lets draw PDF of runs for both year:

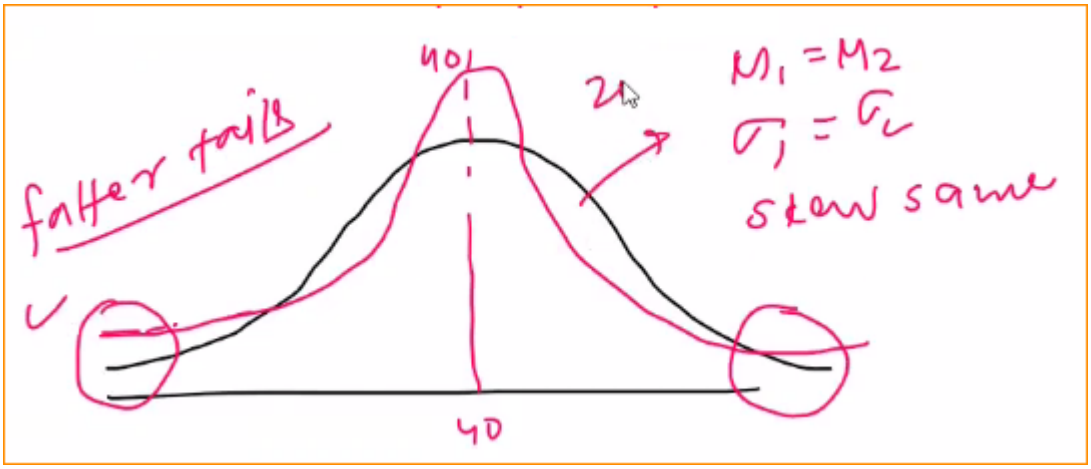


- Now we can diffrentiate on the basis of spread of data which says in 2008 sachin made more runs consistently vs 2007

- Lets a third perspective of data with skewness with same mean & spread:



- this says sachin mostly made more runs in 2008 vs 2007
- Now here's the 4th perspective where mean, std & skewness all are same but it has got fatter tails:



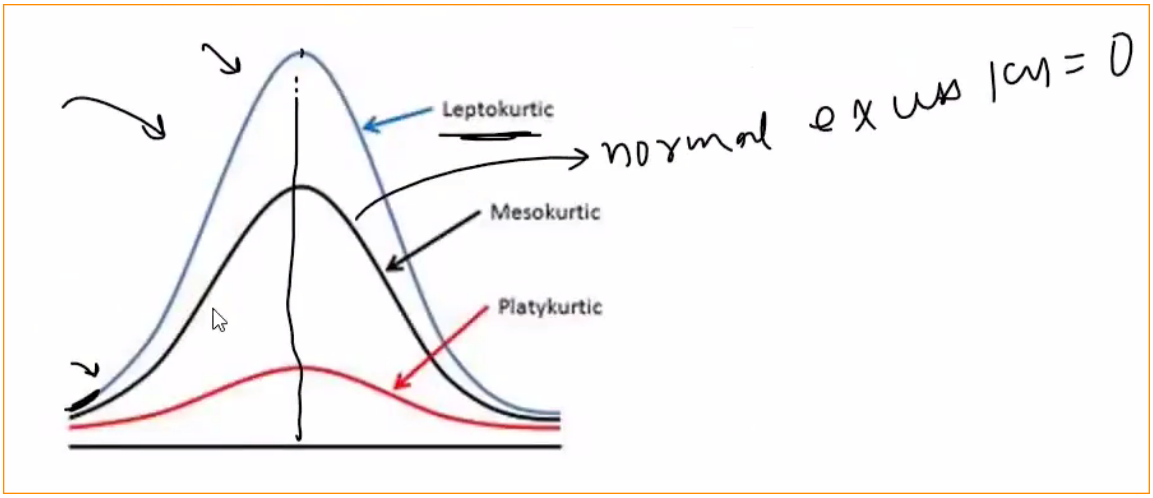
- which says there are high outliars.. means sachin out at 0 & made runs more then 100 also
- This fatness of tails which is outliars is called "kurtosis"
- Higher kurtosis indicates more extreme values in the tails compared to a normal distribution, but these extreme values could be either outliers or simply a higher concentration of data points in those regions.

Use of Kurtosis

- Kurtosis provides insight into the shape of the distribution of returns for a mutual fund. Funds with higher kurtosis tend to have fatter tails, indicating a higher likelihood of extreme returns (both positive and negative). Investors often consider kurtosis as part of their risk assessment because it helps them understand the potential for large deviations from the mean return.

Excess kurtosis

- Excess kurtosis, also known simply as kurtosis, refers to the measure of the relative peakedness or thickness of the tails of a probability distribution compared to the normal distribution. It is calculated by subtracting 3 from the sample kurtosis coefficient.
- There are 3 types:
    1. Leptokurtic - when exces kurtosis is positive means above/fatter then normal distribution tails
    2. Platykurtic - when exces kurtosis is negative means below/thinner then normal distribution tails
    3. Mesokurtic - when exess kurtosis is 0 means normal distribution



Non-Gaussian (non-normal) distribution

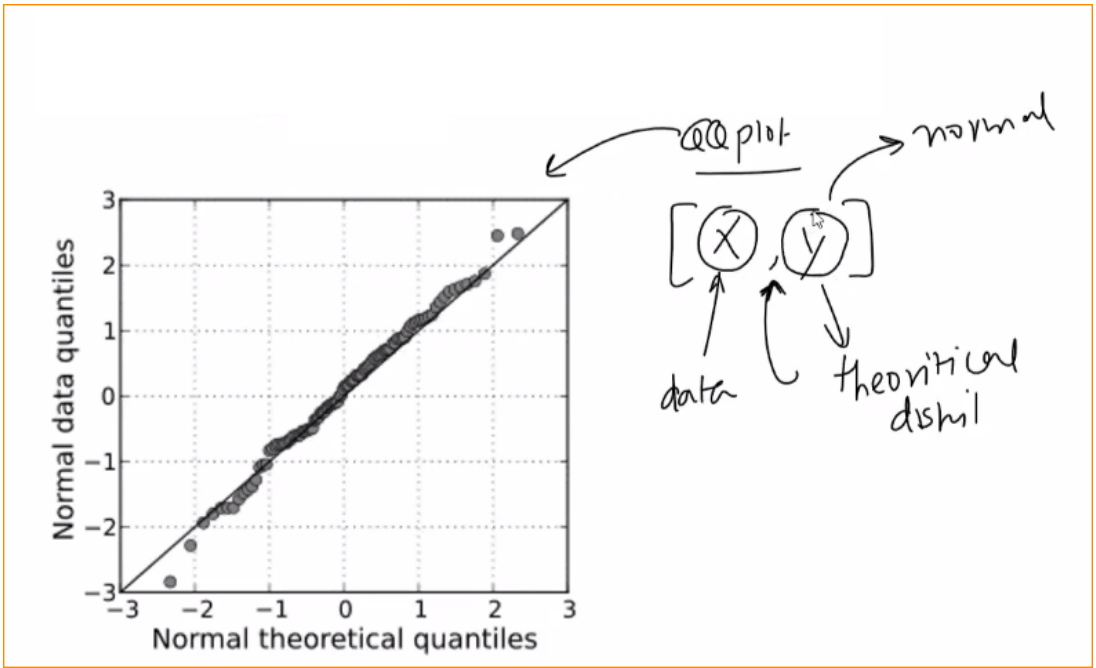
- A non-Gaussian or non-normal distribution refers to a probability distribution that does not follow the shape of a Gaussian (bell-shaped) curve. In a Gaussian distribution, also known as a normal distribution, data is symmetrically distributed around the mean, with most values clustering near the center and tapering off towards the tails according to a specific pattern dictated by the standard deviation.
- There are 2 types:
    1. Continous non-normal distribution
    2. Discrete non-normal distribution

How to find a given is Normal or Not?

1. Visual Inspection - Plot a graph (histogram or density plot) & check
2. QQ plot
3. Statistical Test

QQ Plot

- qq-plot is used to asess whether a dataset follows normal distribution/any theoritical distribution or not
- It compares the quantiles of data against quantiles of theoritical distribution typically on scatter plot



- You take a Normally distributed data (Y) & the data (X) which you want to compare
- Sort both data & calculate percentile values
- Now plot each quantiles of both data
- If it forms a line then it means your data is normally distributed

In [20]:
```
import numpy as np
import pandas as pd
import seaborn as sns
```

In [22]:
```
df = sns.load_dataset('iris')
```

In [23]:
```
df.head()
```

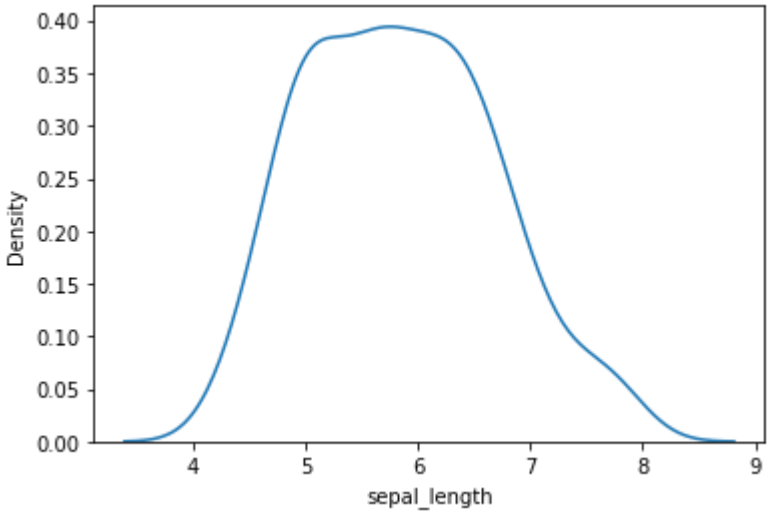
Out[23]:

	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 [24]:
```
sns.kdeplot(df['sepal_length'])
```

```
C:\Users\iampr\AppData\Local\Programs\Python\Python310\lib\site-packages\seaborn\_oldcore.py:1119: FutureWarning: use_inf_as_na option is deprecated and will be removed in a future version. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
```

Out[24]: <AxesSubplot:xlabel='sepal_length', ylabel='Density'>



In [25]:
```
#as per kde plot it looks like normally distributed
#to validate our assumtption we can draw qq plot
```

In [26]:
```
temp = sorted(df['sepal_length'])
```

In [28]:
```
y_quant = []

#calculating 100 percentiles of sepal length & appended in y_quant
for i in range(1,101):
    y_quant.append(np.percentile(temp,i))
```

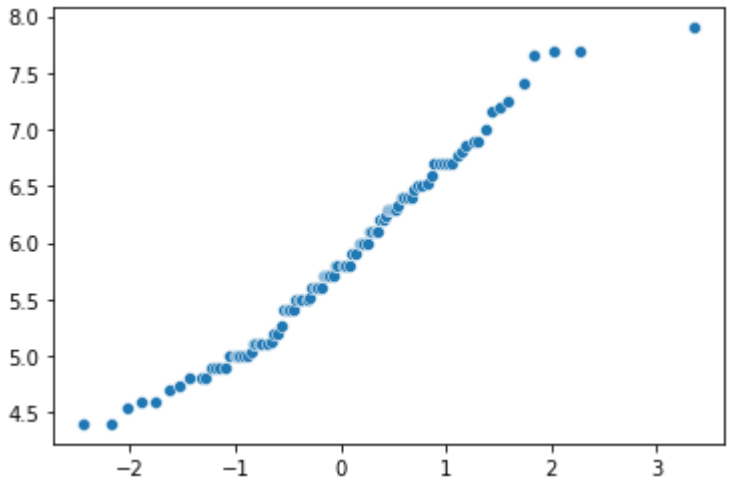
In [30]:
```
#now generating a normally distributed data of mean 0 & std 1
samples = np.random.normal(loc=0, scale = 1, size = 1000)
```

In [31]:
```
x_quant = []

#calculating 100 percentiles of generated data & appended in x_quant
```

```
for i in range(1,101):
    x_quant.append(np.percentile(samples,i))
```

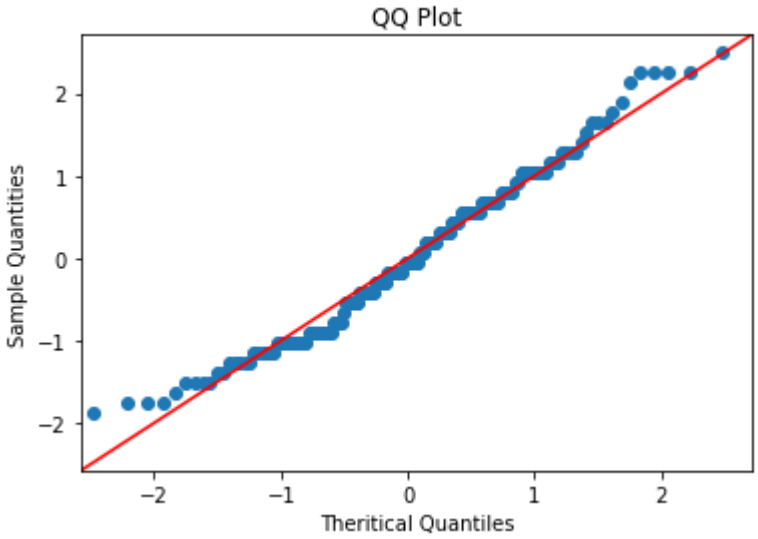
In [33]:
```
#now plot & check whether they form a line
#It looks like normally distrubuted

sns.scatterplot(x = x_quant, y = y_quant)
```

Out[33]: <AxesSubplot:>



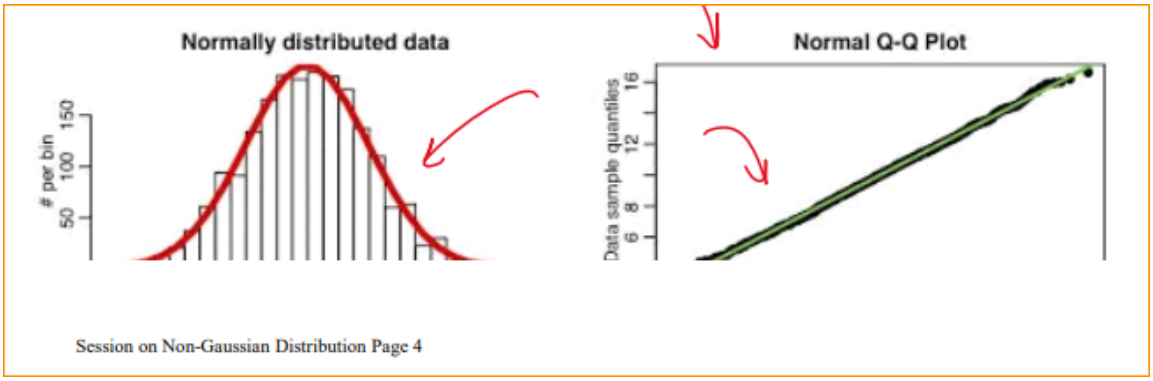
In [34]:
```
#we can do the same as above using statsmodel library
import statsmodels.api as sm
import matplotlib.pyplot as plt
```

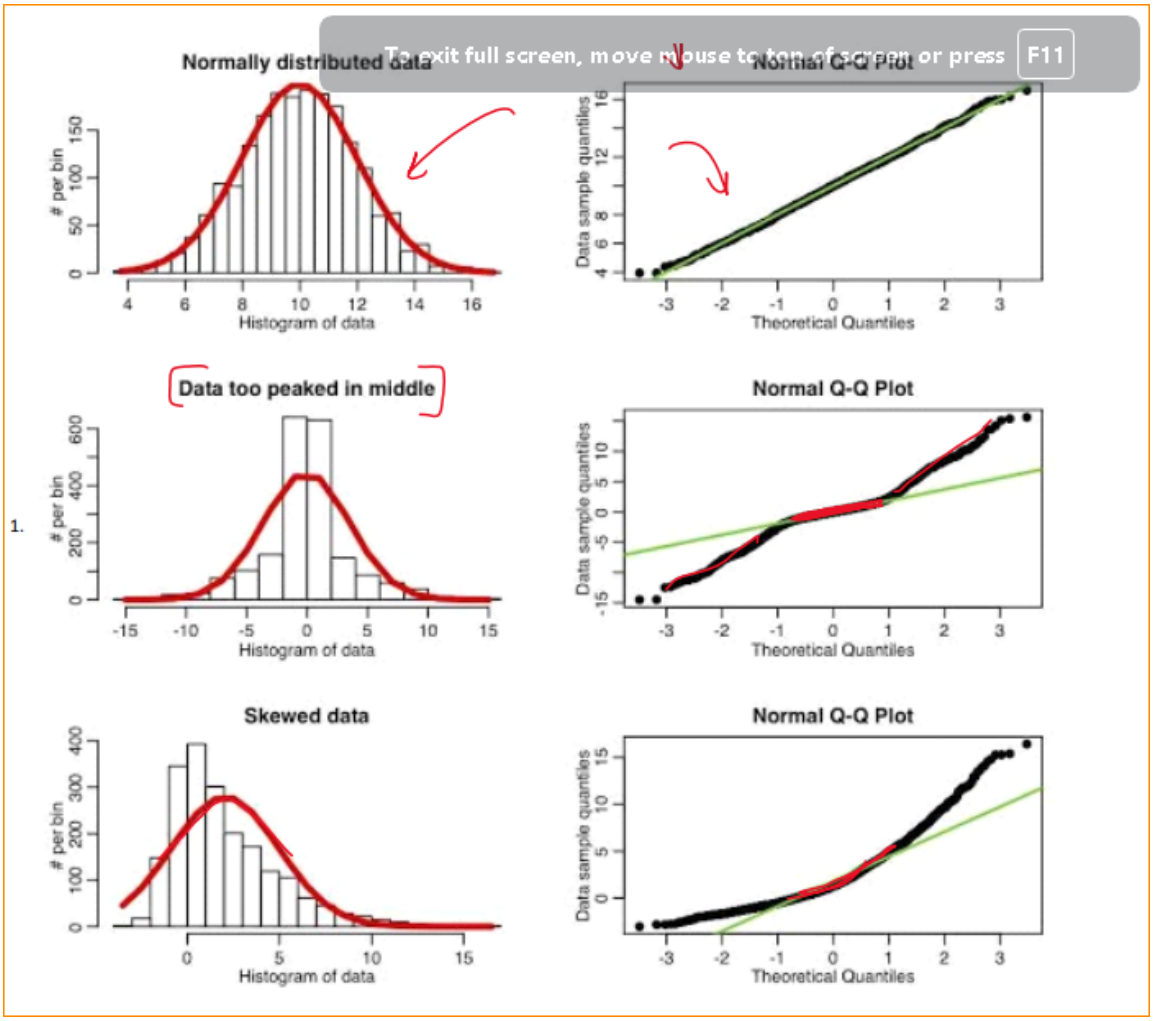
In [35]:
```
fig = sm.qqplot(df['sepal_length'], line = '45', fit = 'True')

plt.title('QQ Plot')
plt.xlabel('Theritical Quantiles')
plt.ylabel('Sample Quantities')
plt.show()
```



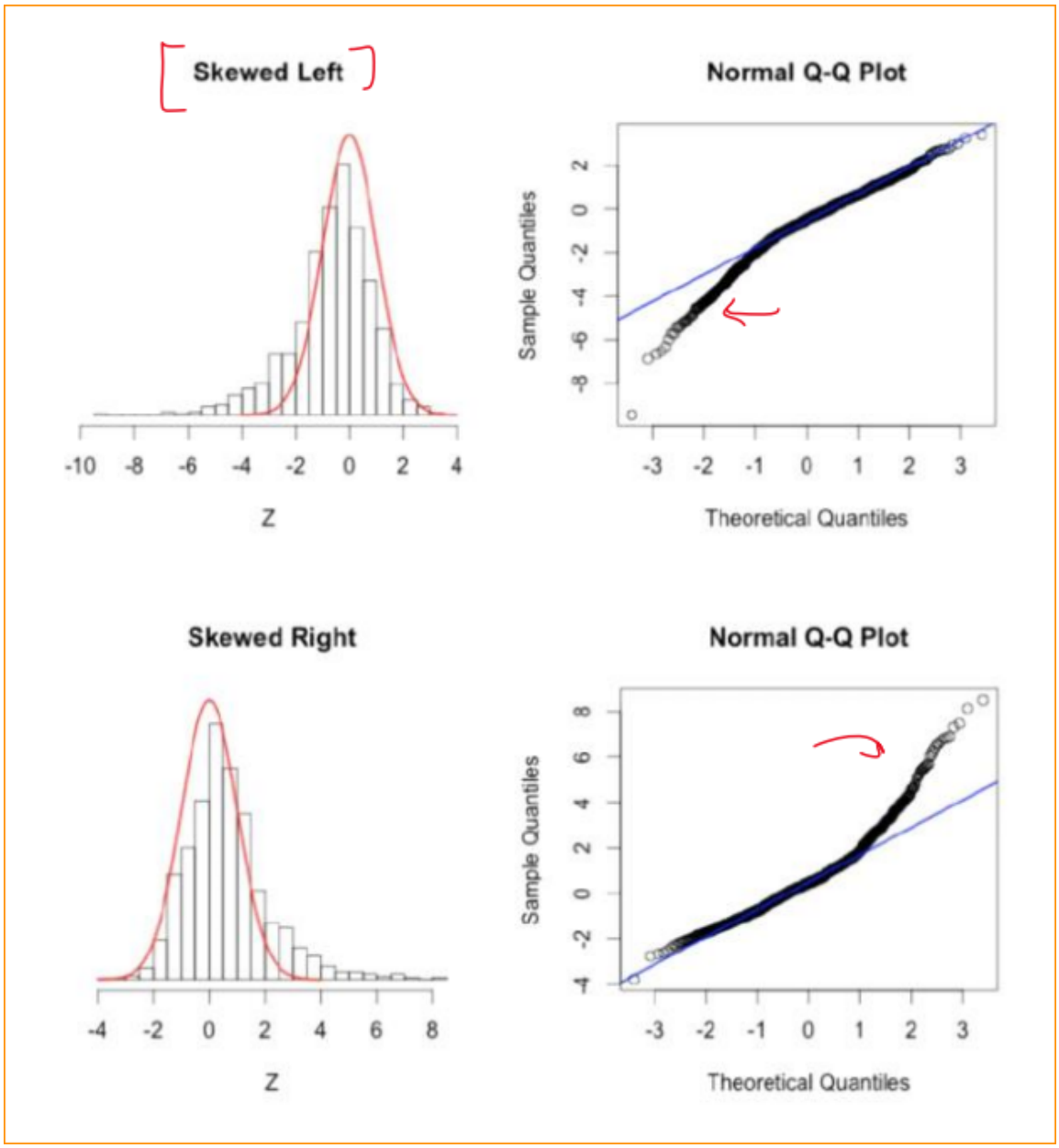
In [36]:
```
#this looks like that data is completly normal, its kind of normal
```

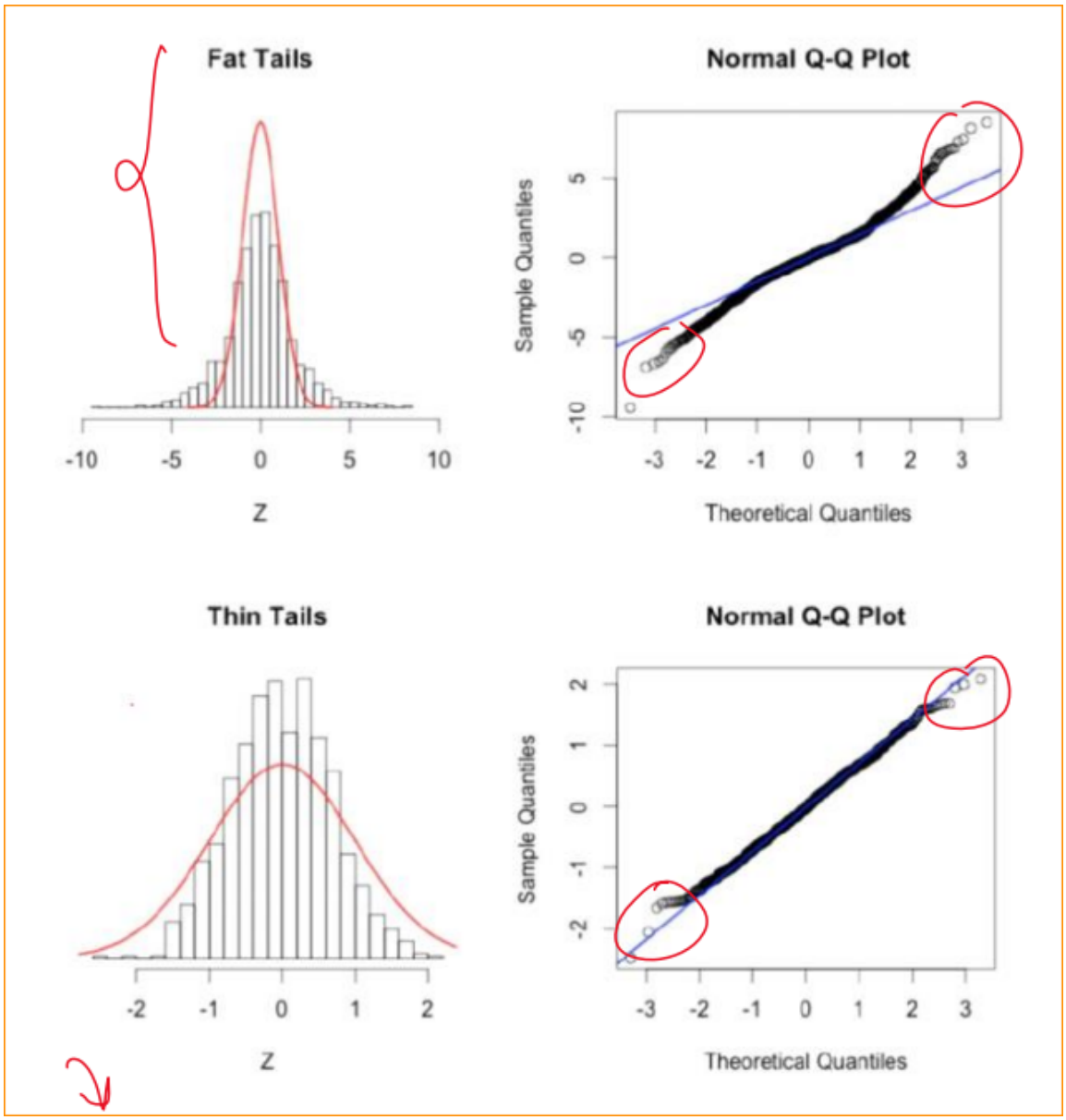
How to interpret QQ-Plot?





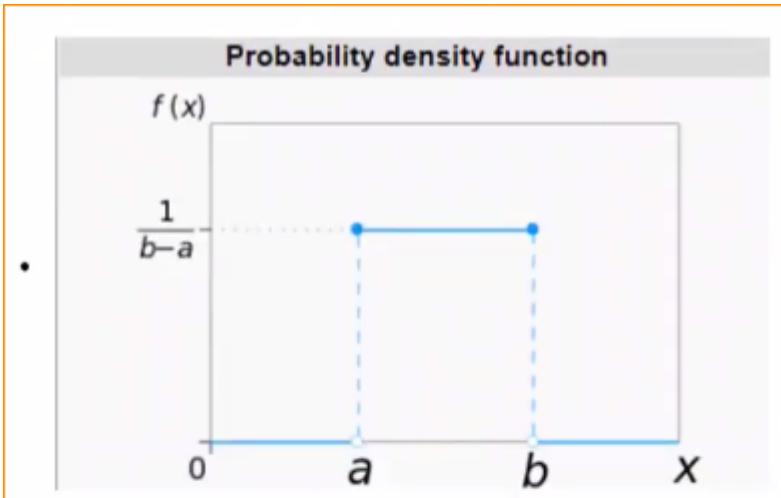
1.

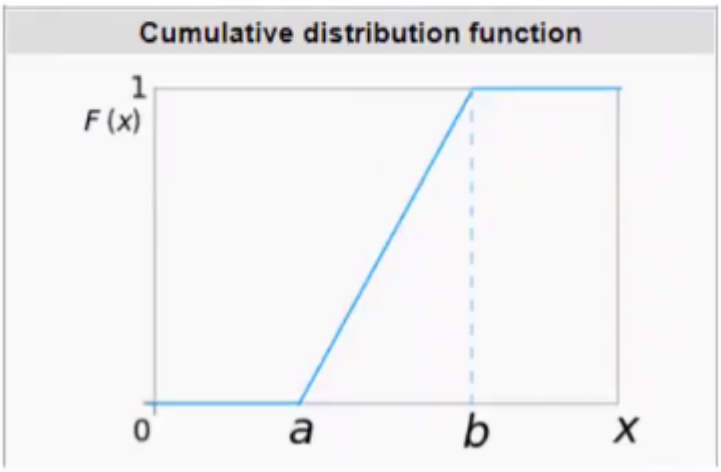




Uniform Distribution

- In a uniform distribution, all the possible outcomes have an equal chance of occurring. It's like each possibility is equally likely, just like each face of the dice has the same chance of showing up when you roll it





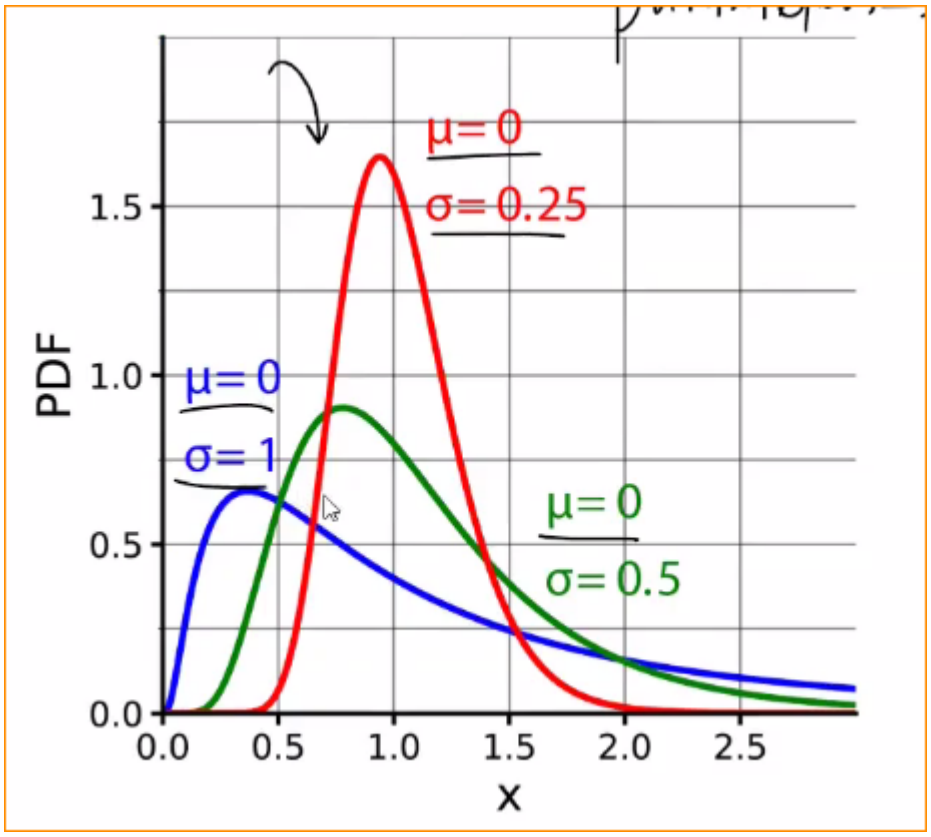
Aplication of uniform distribution in machine learning

- Random Initialization: Uniform distribution is commonly used in random initialization in machine learning and deep learning. When initializing parameters, such as weights in neural networks, it's often desirable to start with random values to prevent symmetry and encourage learning.
- Sampling: It can also be used for sampling, for ex: if you have a dataset with equal no of samples from each class you can use uniform distribution to randomly select a subset of data ensuring that each member of a population has an equal chance of being selected for inclusion in the sample.

- Data augementattion: In some cases where you need to artificially generate data which is similar to specified range of original data
- Hyper parameter tuning: In hyperparameter tuning, you're searching for the best settings for parameters that control how a machine learning model learns. For instance, imagine you're training a neural network for image recognition. Hyperparameters like learning rate, batch size, and number of layers significantly affect the model's performance. To find the best combination, you could use a method like random search. Here, you define a range for each hyperparameter. For instance, the learning rate could be between 0.001 and 0.1. Instead of trying every possible combination, you sample values from each range randomly. A uniform distribution ensures fair sampling across the entire range. So, in essence, by defining a uniform distribution for each hyperparameter, you're randomly exploring different settings to find the optimal combination for your machine learning model, just like trying different ingredients to perfect a recipe.

Log Normal Distribution

- The log-normal distribution is a probability distribution of a random variable whose logarithm follows a normal distribution. In simpler terms, if you take the logarithm of the data points in a log-normal distribution, the resulting values follow a normal (Gaussian) distribution.
- There are 2 parameters same as normal distribution --> mean & std



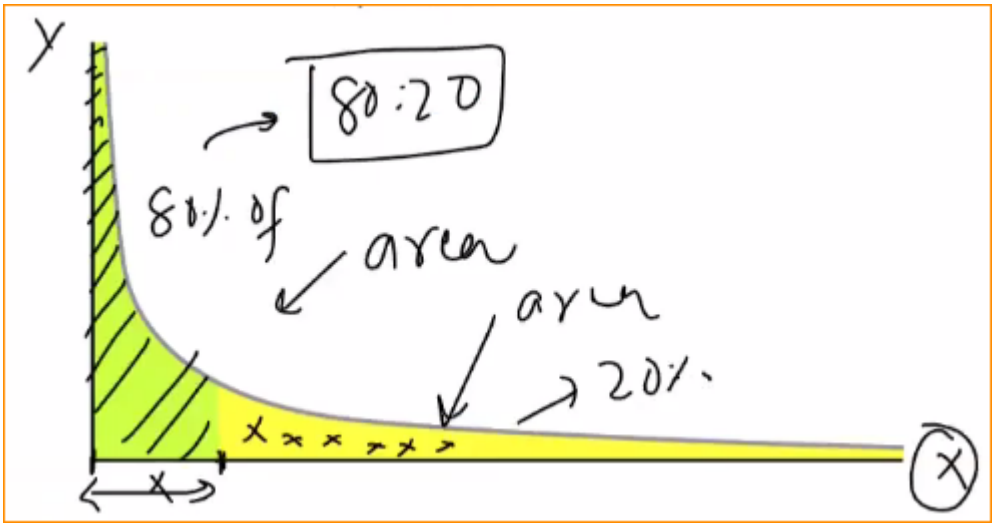
- Ex: most of the comments on social media platforms.. (most people post short comments & very few post long comments), time spent on reading blog, length of ches games, income of 97-99% population with evidence

Pareto Distribution

Power Law

- The power law, also known as the "scaling law," is a mathematical relationship between two quantities where one quantity varies as a power of another. In other words, one variable is proportional to another raised to a constant exponent. $y = kx^{\alpha}$
- where k is a constant of proportionality and, α is the exponent that determines the rate at which y changes with respect to x. The exponent α is typically a real number.
- The power law follows 80-20 rule, also known as the Pareto principle, where roughly 80% of the effects come from 20% of the causes like 80% of total

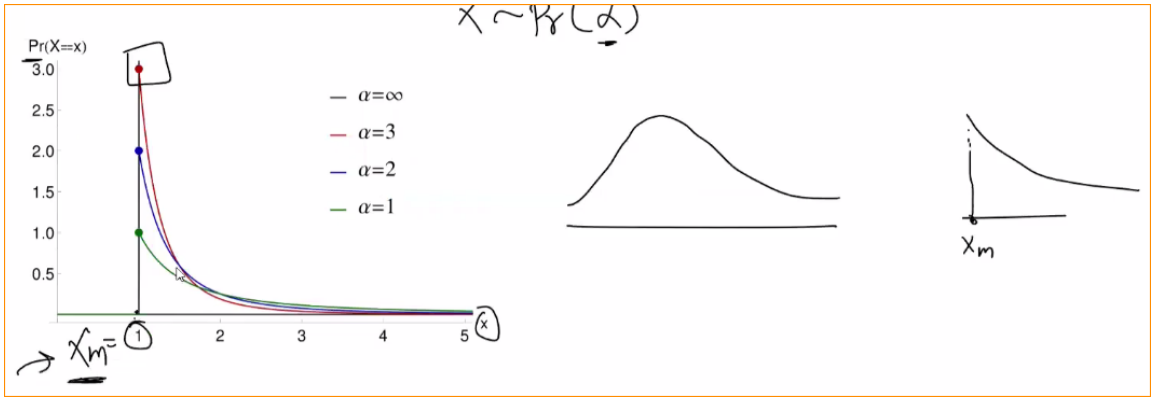
wealth is held by 20% of people



- 80-20 rule is not always applicable.. Its applicable only when value of alpha is 1.6 else 80-20 rule varies

Pareto Distribution

- Pareto distribution is based in a mathematical law called "power law"

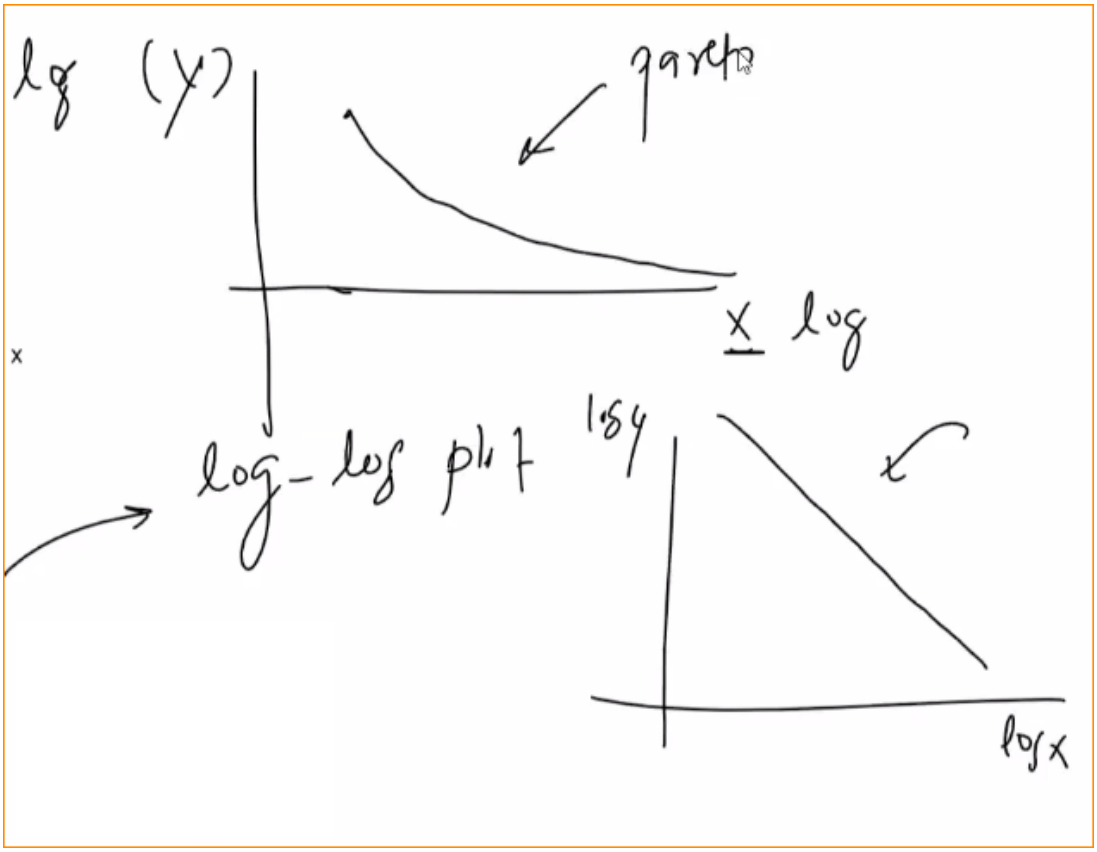


- If alpha will be higher then peak on y axis will be higher & tail will be thinner bcoz all the wealth is distributed
- If alpha is lower then peak on y axis will be lower & tail on x-axis will be fatter bcoz all the wealth is not distributed on y-axis yet
- If alpha will be infinity then it will have a straight line on y axis

How to detect if a distribution is Pareto distribution?

- log-log plot
- plot a graph of log(x) & log(y)

- if it looks like a straight line then its a pareto distribution



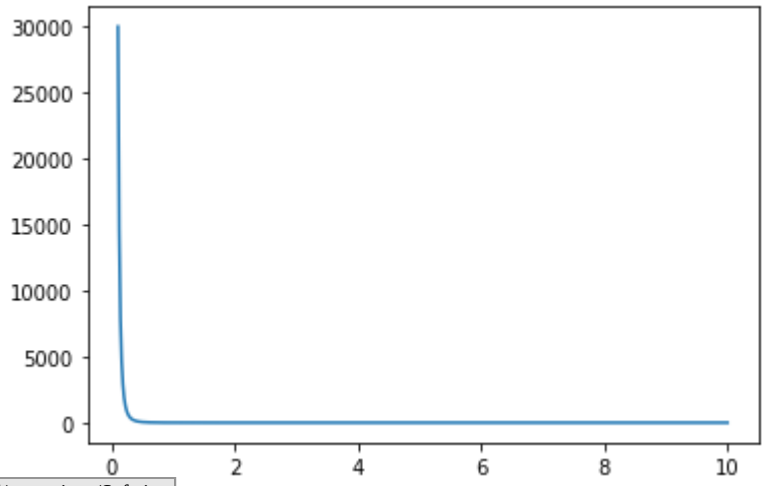
In [38]:
```
import numpy as np
import matplotlib.pyplot as plt
```

In [39]:
```
alpha = 3
xm = 1
```

In [40]:
```
x = np.linspace(0.1, 10, 1000)
```

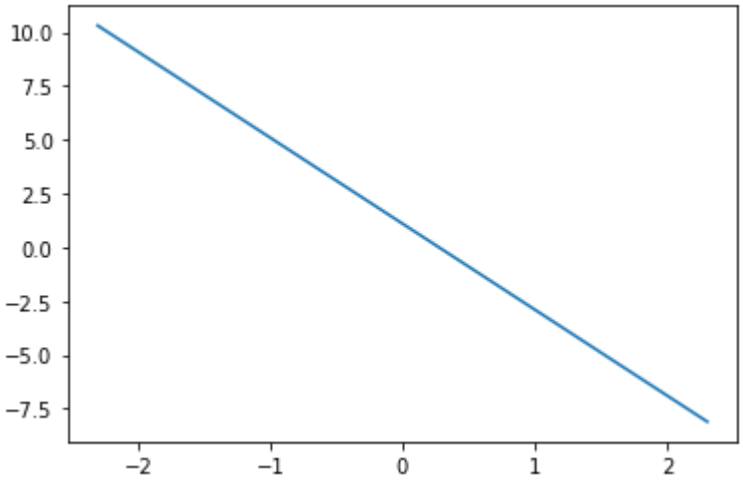
In [42]:
```
y = alpha * (xm**alpha)/(x**(alpha+1))
plt.plot(x,y)
```

Out[42]: [<matplotlib.lines.Line2D at 0x18afef89e10>]



In [43]:
```
plt.plot(np.log(x), np.log(y))
```

Out[43]:
```
[<matplotlib.lines.Line2D at 0x18afeff9000>]
```



Plot QQ plot to idneitfy if a given distribution is pareto

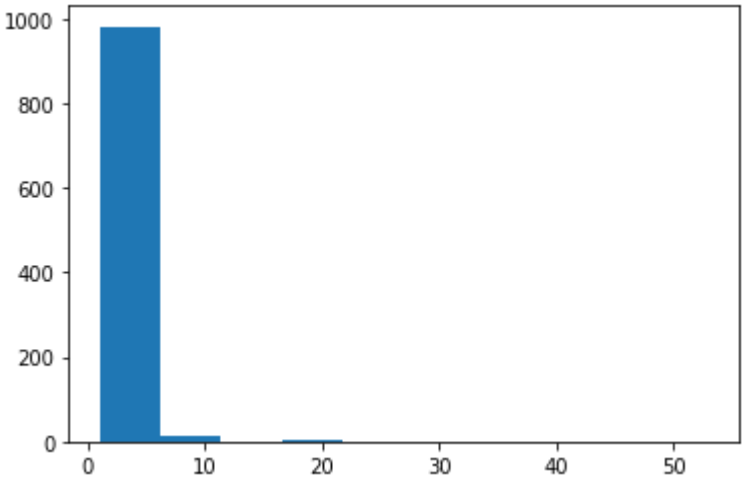
In [45]:
```
import numpy as np
import scipy.stats as stats
import statsmodels.api as sm
```

In [46]:
```
alpha = 2
xm = 1
```

In [47]:
```
x = stats.pareto.rvs(b=alpha, scale=xm, size=1000)
```

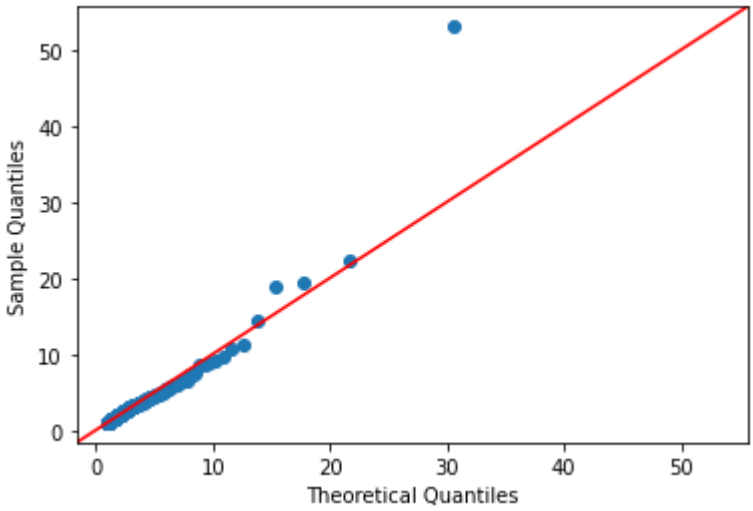
In [48]:
```
plt.hist(x)
```

Out[48]:
```
(array([981.,  14.,   1.,   2.,   1.,   0.,   0.,   0.,   0.,   1.]),
 array([ 1.00034545,  6.20122812, 11.40211079, 16.60299347, 21.80387614,
        27.00475881, 32.20564148, 37.40652416, 42.60740683, 47.8082895 ,
        53.00917217]),
 <BarContainer object of 10 artists>)
```



In [49]:
```
params = stats.pareto.fit(x, floc=0)
dist = stats.pareto(b=params[0], scale = params[2])
```

In [50]:
```
fig = sm.qqplot(x, dist=dist, line='45')
plt.show()
```



Transformations: (How to convert distribution to Normal Distributions)

- Log Transform (which transforms log-normal distribution into normal distribution)
- Boxcox Transform (whch transforms other distributions into normal distribution)

Log Transform

- just take log of the column to transform data
- Not applicable on -ve values
- apply on right skewed data bcoz after applying log transformation it will shift data to center bcoz it will bring larger values to same scale
- It improves linear model performance

Reciprocal transform (1/x) -- Square transform (x^2) -- Square root transform ($\sqrt{x}$)

- Reciprocal Transform: In this transform smaller values wll become bigger & bigger values become smaller
- e Transform: Its used for left skewed data.

- Square root transform: Its useful for stabilizing variance and reducing skewness, particularly when the original data is positively skewed.

Generally we apply all transformation & experiment which is performing better

Above all al scikit learn function transformer class methods

In [51]:
```
import pandas as pd
import numpy as np

import scipy.stats as stats

import matplotlib.pyplot as plt
import seaborn as sns

from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
from sklearn.model_selection import cross_val_score

from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier

from sklearn.preprocessing import FunctionTransformer
from sklearn.compose import ColumnTransformer
```

In [56]:
```
df = pd.read_csv('train.csv',usecols=['Age','Fare','Survived'])
```

In [57]:
```
df.head()
```

Out[57]:

	Survived	Age	Fare
0	0	22.0	7.2500
1	1	38.0	71.2833
2	1	26.0	7.9250
3	1	35.0	53.1000
4	0	35.0	8.0500

In [58]:
```
df.isnull().sum()
```

Out[58]:
```
Survived      0
Age         177
Fare          0
dtype: int64
```

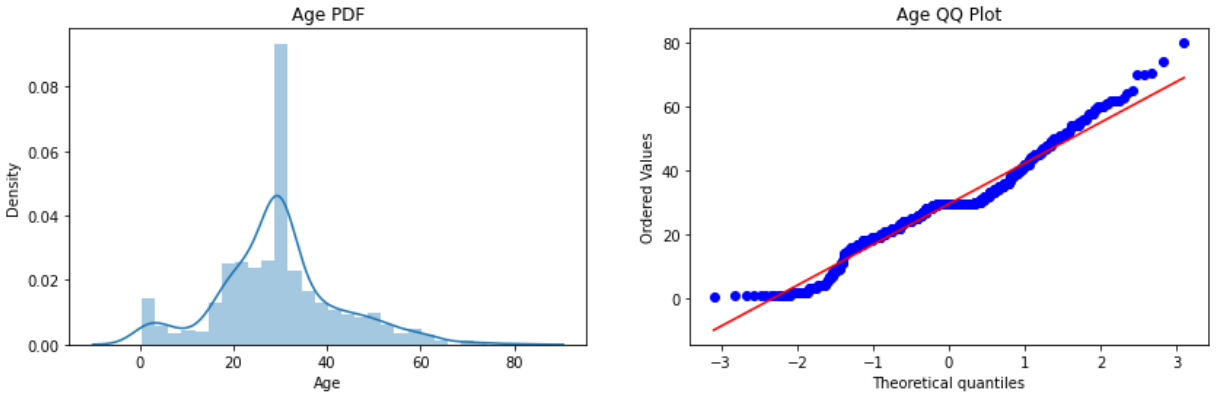
In [59]:
```
df['Age'].fillna(df['Age'].mean(),inplace=True)
```

In [60]:
```
X = df.iloc[:,1:3]
y = df.iloc[:,0]
```

In [61]:
```
X_train, X_test, y_train, y_test = train_test_split(X,y,test_size=0.2,random
```

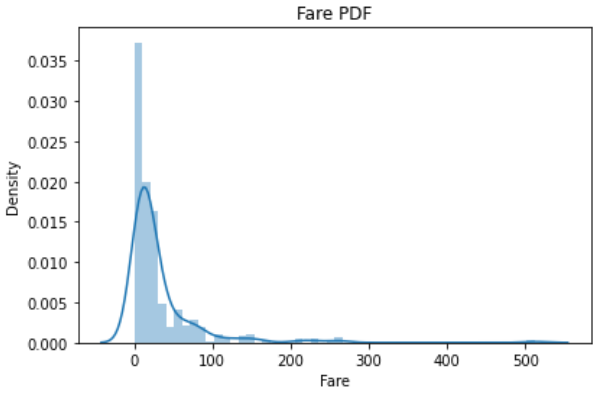
In [62]:
```
plt.figure(figsize=(14,4))
plt.subplot(121)
sns.distplot(X_train['Age'])
plt.title('Age PDF')

plt.subplot(122)
stats.probplot(X_train['Age'], dist="norm", plot=plt)
plt.title('Age QQ Plot')

plt.show()
```

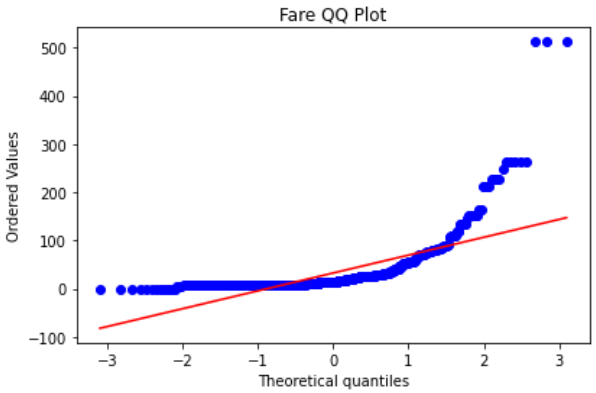
```
C:\Users\iampr\AppData\Local\Temp\ipykernel_3260\2888751792.py:3: UserWarnin
g:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(X_train['Age'])
C:\Users\iampr\AppData\Local\Programs\Python\Python310\lib\site-packages\sea
born\_oldcore.py:1119: FutureWarning: use_inf_as_na option is deprecated and
will be removed in a future version. Convert inf values to NaN before operat
ing instead.
  with pd.option_context('mode.use_inf_as_na', True):
```



In [64]:
```
plt.figure(figsize=(14,4))
plt.subplot(121)
sns.distplot(X_train['Fare'])
plt.title('Fare PDF')

plt.subplot(122)
stats.probplot(X_train['Fare'], dist="norm", plot=plt)
plt.title('Fare QQ Plot')

plt.show()
```

```
C:\Users\iampr\AppData\Local\Temp\ipykernel_3260\474975069.py:3: UserWarnin
g:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(X_train['Fare'])
C:\Users\iampr\AppData\Local\Programs\Python\Python310\lib\site-packages\sea
born\_oldcore.py:1119: FutureWarning: use_inf_as_na option is deprecated and
will be removed in a future version. Convert inf values to NaN before operat
ing instead.
  with pd.option_context('mode.use_inf_as_na', True):
```





In [65]:
```
clf = LogisticRegression()
clf2 = DecisionTreeClassifier()
```

In [66]:
```
clf.fit(X_train,y_train)
clf2.fit(X_train,y_train)

y_pred = clf.predict(X_test)
y_pred1 = clf2.predict(X_test)

print("Accuracy LR",accuracy_score(y_test,y_pred))
print("Accuracy DT",accuracy_score(y_test,y_pred1))
```

```
Accuracy LR 0.6480446927374302
Accuracy DT 0.6480446927374302
```

In [67]:
```
trf = FunctionTransformer(func=np.log1p)
```

In [68]:
```
X_train_transformed = trf.fit_transform(X_train)
X_test_transformed = trf.transform(X_test)
```

In [69]:
```
clf = LogisticRegression()
clf2 = DecisionTreeClassifier()

clf.fit(X_train_transformed,y_train)
clf2.fit(X_train_transformed,y_train)
```

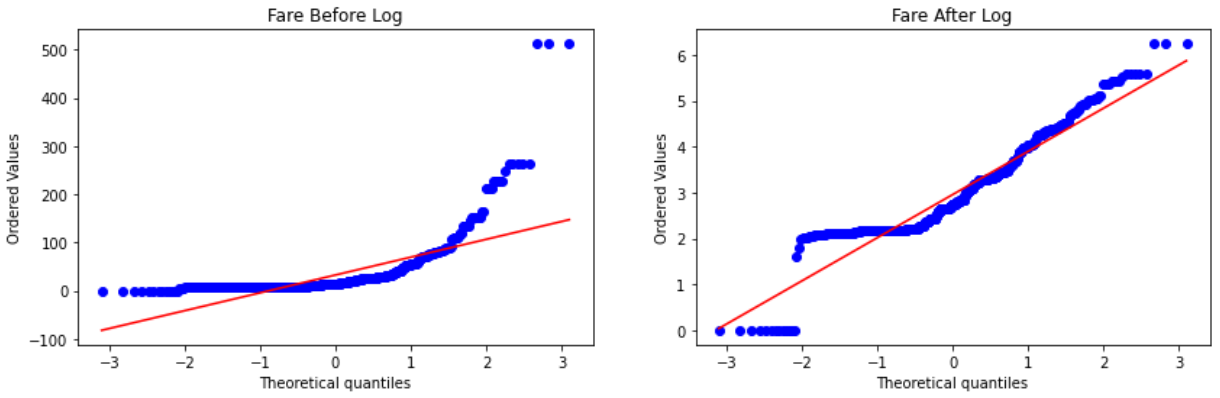
```
y_pred = clf.predict(X_test_transformed)
y_pred1 = clf2.predict(X_test_transformed)

print("Accuracy LR",accuracy_score(y_test,y_pred))
print("Accuracy DT",accuracy_score(y_test,y_pred1))
```

```
Accuracy LR 0.6815642458100558
Accuracy DT 0.6983240223463687
```

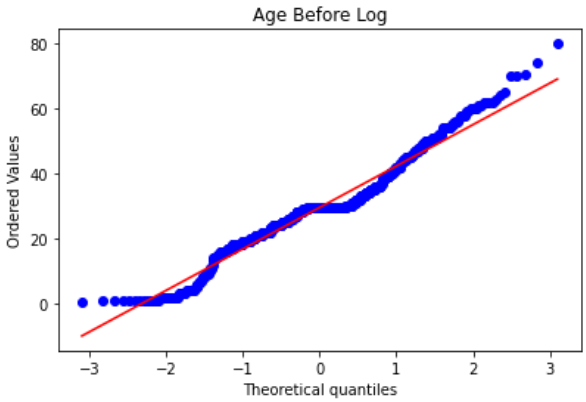
In [70]:
```
X_transformed = trf.fit_transform(X)

clf = LogisticRegression()
clf2 = DecisionTreeClassifier()

print("LR",np.mean(cross_val_score(clf,X_transformed,y,scoring='accuracy',cv
print("DT",np.mean(cross_val_score(clf2,X_transformed,y,scoring='accuracy',c
```

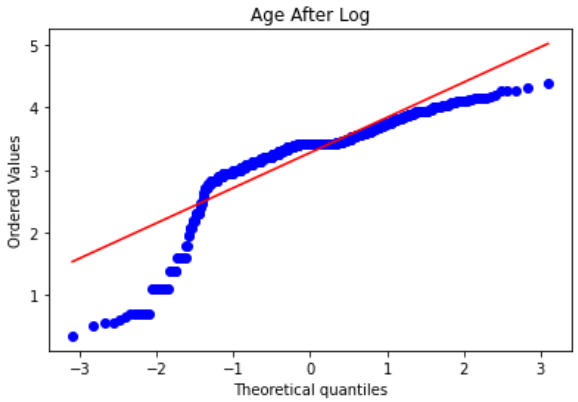
```
LR 0.678027465667915
DT 0.6588389513108613
```

In [71]:
```
plt.figure(figsize=(14,4))

plt.subplot(121)
stats.probplot(X_train['Fare'], dist="norm", plot=plt)
plt.title('Fare Before Log')

plt.subplot(122)
stats.probplot(X_train_transformed['Fare'], dist="norm", plot=plt)
plt.title('Fare After Log')

plt.show()
```



In [72]:
```
plt.figure(figsize=(14,4))

plt.subplot(121)
stats.probplot(X_train['Age'], dist="norm", plot=plt)
plt.title('Age Before Log')

plt.subplot(122)
stats.probplot(X_train_transformed['Age'], dist="norm", plot=plt)
plt.title('Age After Log')

plt.show()
```





In [73]:
```
trf2 = ColumnTransformer([('log',FunctionTransformer(np.log1p),['Fare'])],re

X_train_transformed2 = trf2.fit_transform(X_train)
X_test_transformed2 = trf2.transform(X_test)
```

In [74]:
```
clf = LogisticRegression()
clf2 = DecisionTreeClassifier()

clf.fit(X_train_transformed2,y_train)
clf2.fit(X_train_transformed2,y_train)

y_pred = clf.predict(X_test_transformed2)
y_pred2 = clf2.predict(X_test_transformed2)

print("Accuracy LR",accuracy_score(y_test,y_pred))
print("Accuracy DT",accuracy_score(y_test,y_pred2))
```

```
Accuracy LR 0.6703910614525139
Accuracy DT 0.659217877094972
```

In [75]:
```
X_transformed2 = trf2.fit_transform(X)

clf = LogisticRegression()
clf2 = DecisionTreeClassifier()

print("LR",np.mean(cross_val_score(clf,X_transformed2,y,scoring='accuracy',c
print("DT",np.mean(cross_val_score(clf2,X_transformed2,y,scoring='accuracy',
```

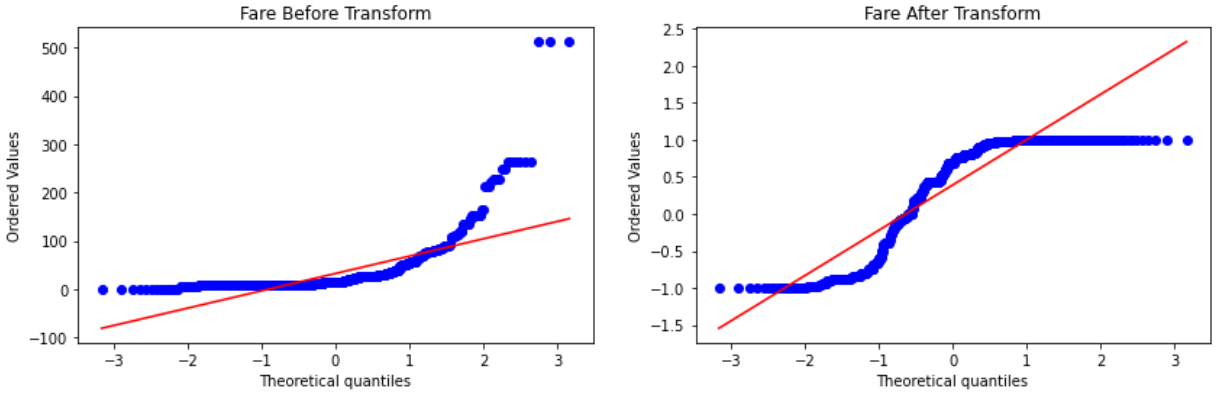
```
LR 0.6712609238451936
DT 0.6588264669163546
```

In [76]:
```
def apply_transform(transform):
    X = df.iloc[:,1:3]
    y = df.iloc[:,0]

    trf = ColumnTransformer([('log',FunctionTransformer(transform),['Fare'])

    X_trans = trf.fit_transform(X)

    clf = LogisticRegression()

    print("Accuracy",np.mean(cross_val_score(clf,X_trans,y,scoring='accuracy
```

```
    plt.figure(figsize=(14,4))

    plt.subplot(121)
    stats.probplot(X['Fare'], dist="norm", plot=plt)
    plt.title('Fare Before Transform')

    plt.subplot(122)
    stats.probplot(X_trans[:,0], dist="norm", plot=plt)
    plt.title('Fare After Transform')

    plt.show()
```

In [77]:
```
apply_transform(np.sin)
```

```
Accuracy 0.6195131086142323
```



Power Transformer Class

1. Box-Cox Transform
2. Yu-Johnson Transform (also works on negative values)

In [78]:
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

import scipy.stats as stats
```

In [79]:
```
from sklearn.model_selection import train_test_split
from sklearn.model_selection import cross_val_score

from sklearn.linear_model import LinearRegression

from sklearn.metrics import r2_score

from sklearn.preprocessing import PowerTransformer
```

In [80]:
```
df = pd.read_csv('concrete_data.csv')
```

In [81]:
```
df.head()
```

Out[81]:

	Cement	Blast Furnace Slag	Fly Ash	Water	Superplasticizer	Coarse Aggregate	Fine Aggregate	Ag
0	540.0	0.0	0.0	162.0	2.5	1040.0	676.0	2
1	540.0	0.0	0.0	162.0	2.5	1055.0	676.0	2
2	332.5	142.5	0.0	228.0	0.0	932.0	594.0	27
3	332.5	142.5	0.0	228.0	0.0	932.0	594.0	36
4	198.6	132.4	0.0	192.0	0.0	978.4	825.5	36

In [82]:
```
df.shape
```

Out[82]:
```
(1030, 9)
```

In [83]:
```
df.isnull().sum()
```

Out[83]:
```
Cement                0
Blast Furnace Slag    0
Fly Ash               0
Water                 0
Superplasticizer      0
Coarse Aggregate      0
Fine Aggregate        0
Age                   0
Strength              0
dtype: int64
```

In [84]:
```
df.describe()
```

Out[84]:

	Cement	Blast Furnace Slag	Fly Ash	Water	Superplasticizer	
count	1030.000000	1030.000000	1030.000000	1030.000000	1030.000000	1
mean	281.167864	73.895825	54.188350	181.567282	6.204660	
std	104.506364	86.279342	63.997004	21.354219	5.973841	
min	102.000000	0.000000	0.000000	121.800000	0.000000	
25%	192.375000	0.000000	0.000000	164.900000	0.000000	
50%	272.900000	22.000000	0.000000	185.000000	6.400000	
75%	350.000000	142.950000	118.300000	192.000000	10.200000	1
max	540.000000	359.400000	200.100000	247.000000	32.200000	1

In [85]:
```
X = df.drop(columns=['Strength'])
y = df.iloc[:,-1]
```

In [86]:
```
X_train, X_test, y_train, y_test = train_test_split(X,y,test_size=0.2,random
```

In [87]:
```
# Applying Regression without any transformation
lr = LinearRegression()

lr.fit(X_train,y_train)

y_pred = lr.predict(X_test)

r2_score(y_test,y_pred)
```

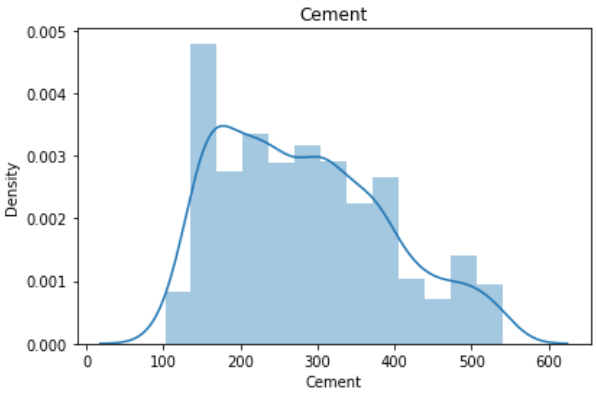
Out[87]:
```
0.6275531792314846
```

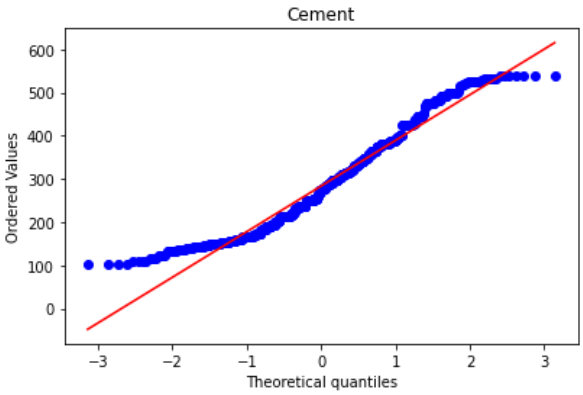
In [88]:
```
# Cross checking with cross val score
lr = LinearRegression()
np.mean(cross_val_score(lr,X,y,scoring='r2'))
```

Out[88]:
```
0.4609940491662864
```

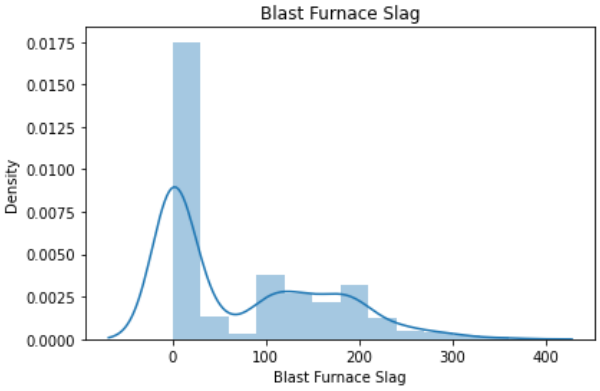
In [89]:
```
# Plotting the distplots without any transformation

for col in X_train.columns:
    plt.figure(figsize=(14,4))
    plt.subplot(121)
    sns.distplot(X_train[col])
    plt.title(col)

    plt.subplot(122)
    stats.probplot(X_train[col], dist="norm", plot=plt)
    plt.title(col)

    plt.show()
```

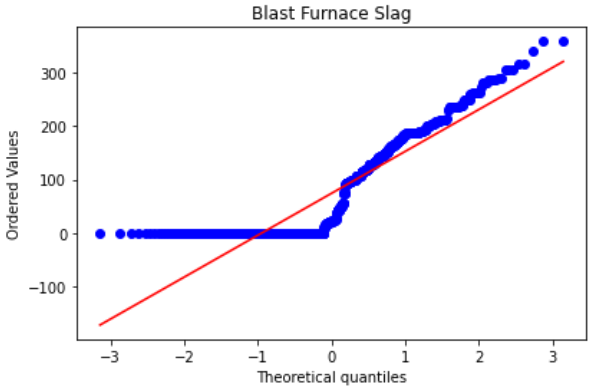
```
C:\Users\iampr\AppData\Local\Temp\ipykernel_3260\1268627929.py:6: UserWarnin
g:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(X_train[col])
C:\Users\iampr\AppData\Local\Programs\Python\Python310\lib\site-packages\sea
born\_oldcore.py:1119: FutureWarning: use_inf_as_na option is deprecated and
will be removed in a future version. Convert inf values to NaN before operat
ing instead.
  with pd.option_context('mode.use_inf_as_na', True):
```



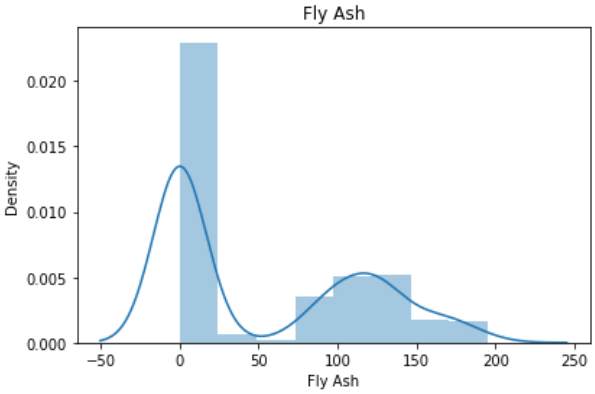


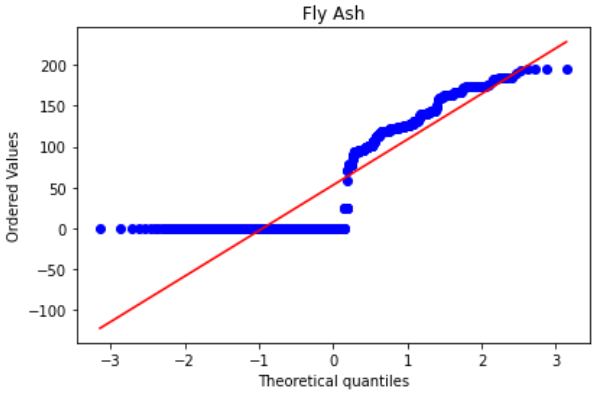

```
C:\Users\iampr\AppData\Local\Temp\ipykernel_3260\1268627929.py:6: UserWarnin
g:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(X_train[col])
C:\Users\iampr\AppData\Local\Programs\Python\Python310\lib\site-packages\sea
born\_oldcore.py:1119: FutureWarning: use_inf_as_na option is deprecated and
will be removed in a future version. Convert inf values to NaN before operat
ing instead.
  with pd.option_context('mode.use_inf_as_na', True):
```



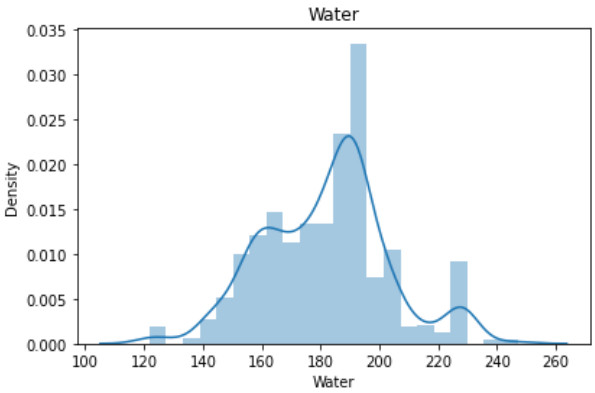


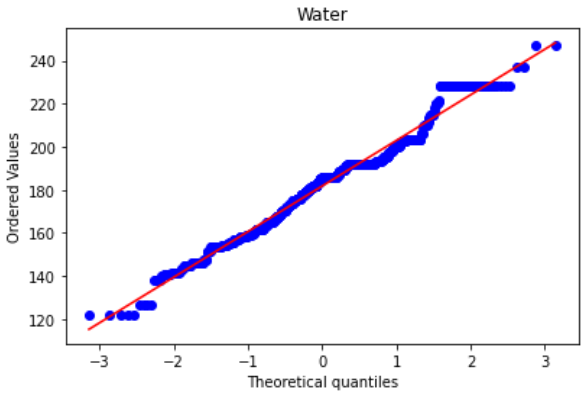
```
C:\Users\iampr\AppData\Local\Temp\ipykernel_3260\1268627929.py:6: UserWarnin
g:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(X_train[col])
C:\Users\iampr\AppData\Local\Programs\Python\Python310\lib\site-packages\sea
born\_oldcore.py:1119: FutureWarning: use_inf_as_na option is deprecated and
will be removed in a future version. Convert inf values to NaN before operat
ing instead.
  with pd.option_context('mode.use_inf_as_na', True):
```



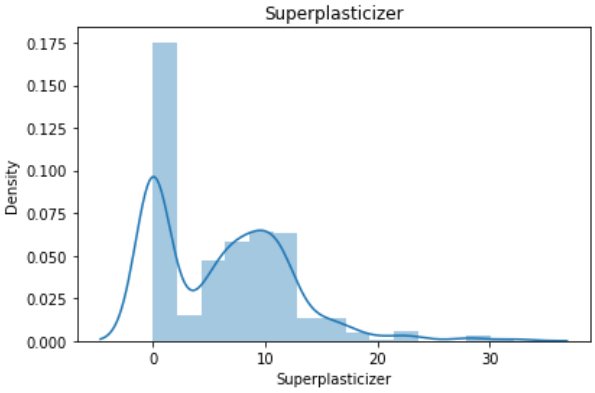


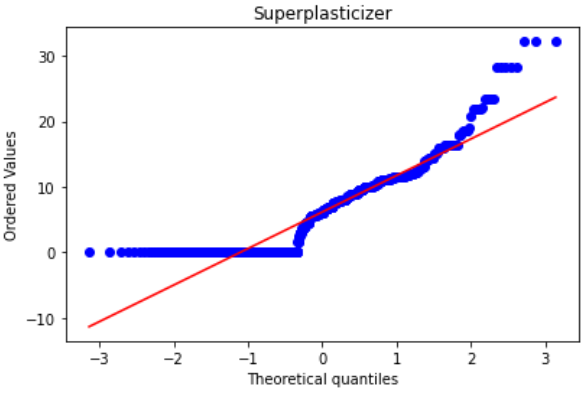

```
C:\Users\iampr\AppData\Local\Temp\ipykernel_3260\1268627929.py:6: UserWarnin
g:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(X_train[col])
C:\Users\iampr\AppData\Local\Programs\Python\Python310\lib\site-packages\sea
born\_oldcore.py:1119: FutureWarning: use_inf_as_na option is deprecated and
will be removed in a future version. Convert inf values to NaN before operat
ing instead.
  with pd.option_context('mode.use_inf_as_na', True):
```



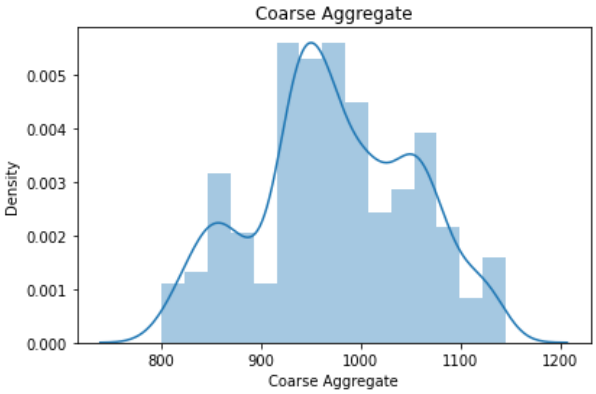


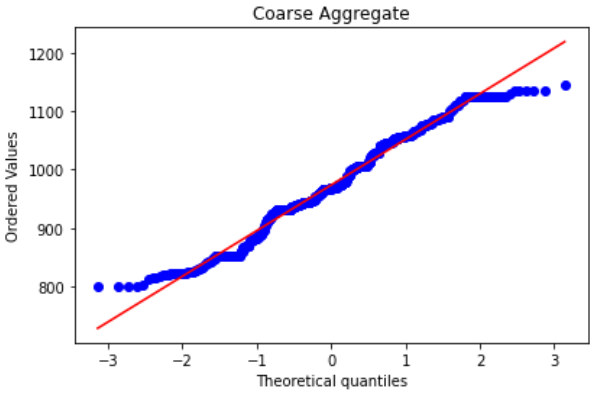

```
C:\Users\iampr\AppData\Local\Temp\ipykernel_3260\1268627929.py:6: UserWarnin
g:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(X_train[col])
C:\Users\iampr\AppData\Local\Programs\Python\Python310\lib\site-packages\sea
born\_oldcore.py:1119: FutureWarning: use_inf_as_na option is deprecated and
will be removed in a future version. Convert inf values to NaN before operat
ing instead.
  with pd.option_context('mode.use_inf_as_na', True):
```



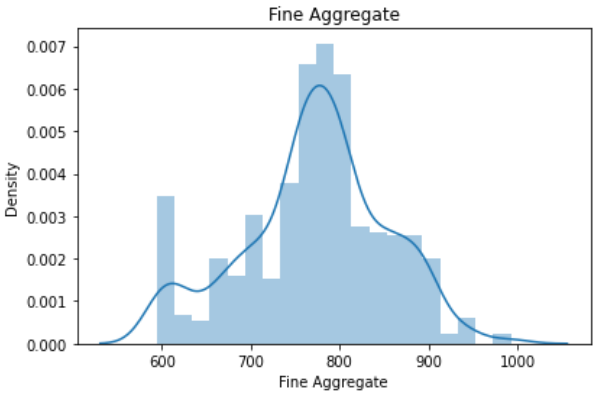


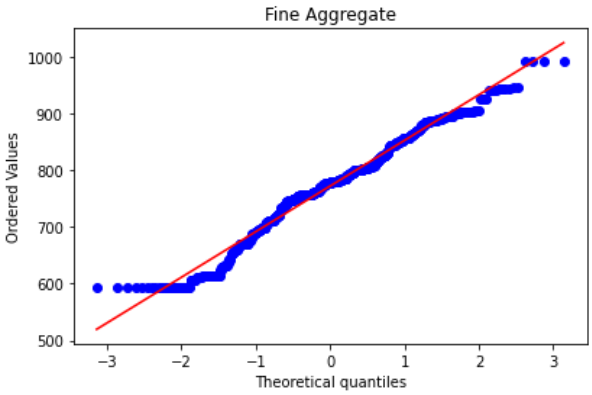
```
C:\Users\iampr\AppData\Local\Temp\ipykernel_3260\1268627929.py:6: UserWarnin
g:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(X_train[col])
C:\Users\iampr\AppData\Local\Programs\Python\Python310\lib\site-packages\sea
born\_oldcore.py:1119: FutureWarning: use_inf_as_na option is deprecated and
will be removed in a future version. Convert inf values to NaN before operat
ing instead.
  with pd.option_context('mode.use_inf_as_na', True):
```



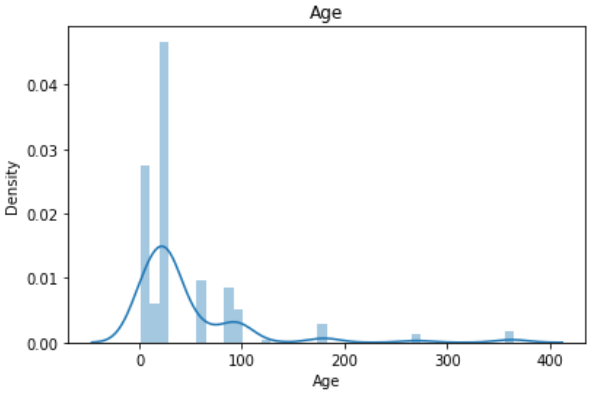


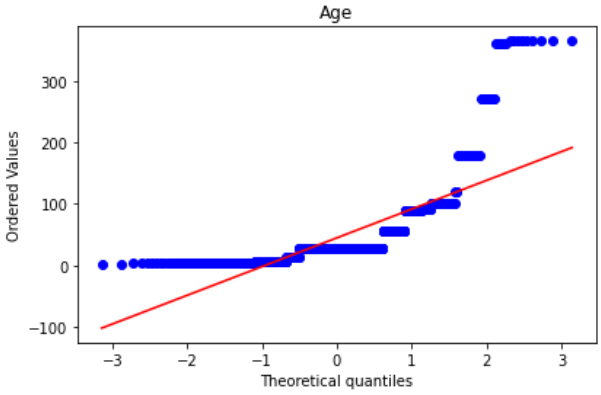

```
C:\Users\iampr\AppData\Local\Temp\ipykernel_3260\1268627929.py:6: UserWarnin
g:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(X_train[col])
C:\Users\iampr\AppData\Local\Programs\Python\Python310\lib\site-packages\sea
born\_oldcore.py:1119: FutureWarning: use_inf_as_na option is deprecated and
will be removed in a future version. Convert inf values to NaN before operat
ing instead.
  with pd.option_context('mode.use_inf_as_na', True):
```






```
C:\Users\iampr\AppData\Local\Temp\ipykernel_3260\1268627929.py:6: UserWarnin
g:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(X_train[col])
C:\Users\iampr\AppData\Local\Programs\Python\Python310\lib\site-packages\sea
born\_oldcore.py:1119: FutureWarning: use_inf_as_na option is deprecated and
will be removed in a future version. Convert inf values to NaN before operat
ing instead.
  with pd.option_context('mode.use_inf_as_na', True):
```





In [90]:
```
# Applying Box-Cox Transform

pt = PowerTransformer(method='box-cox')

X_train_transformed = pt.fit_transform(X_train+0.000001)
X_test_transformed = pt.transform(X_test+0.000001)

pd.DataFrame({'cols':X_train.columns,'box_cox_lambdas':pt.lambdas_})
```

Out[90]:

	cols	box_cox_lambdas
0	Cement	0.177025
1	Blast Furnace Slag	0.025093
2	Fly Ash	-0.038970
3	Water	0.772682
4	Superplasticizer	0.098811
5	Coarse Aggregate	1.129813
6	Fine Aggregate	1.782019
7	Age	0.066631

In [91]:
```
# Applying linear regression on transformed data

lr = LinearRegression()
lr.fit(X_train_transformed,y_train)

y_pred2 = lr.predict(X_test_transformed)

r2_score(y_test,y_pred2)
```

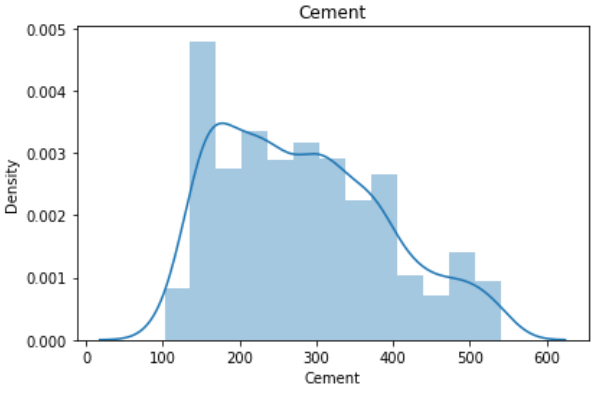
Out[91]:
```
0.8047825006181187
```

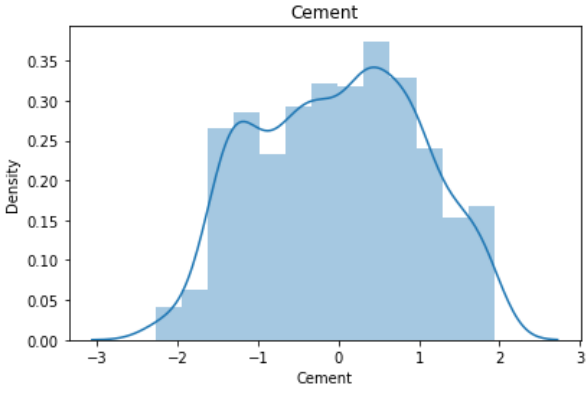
In [92]:
```
# Using cross val score

pt = PowerTransformer(method='box-cox')
X_transformed = pt.fit_transform(X+0.0000001)

lr = LinearRegression()
np.mean(cross_val_score(lr,X_transformed,y,scoring='r2'))
```

Out[92]:
```
0.6658537942219863
```

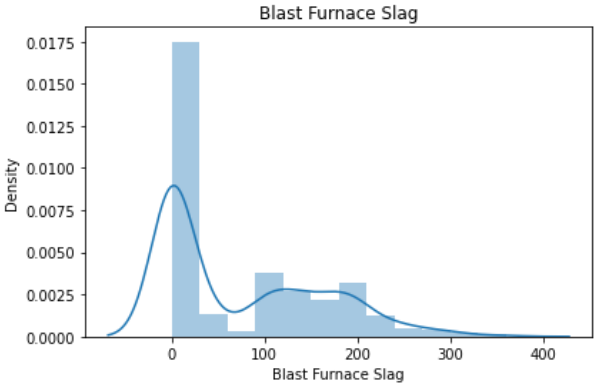
In [93]:
```
# Before and after comparision for Box-Cox Plot
X_train_transformed = pd.DataFrame(X_train_transformed,columns=X_train.colum

for col in X_train_transformed.columns:
    plt.figure(figsize=(14,4))
    plt.subplot(121)
    sns.distplot(X_train[col])
    plt.title(col)

    plt.subplot(122)
    sns.distplot(X_train_transformed[col])
    plt.title(col)

    plt.show()
```

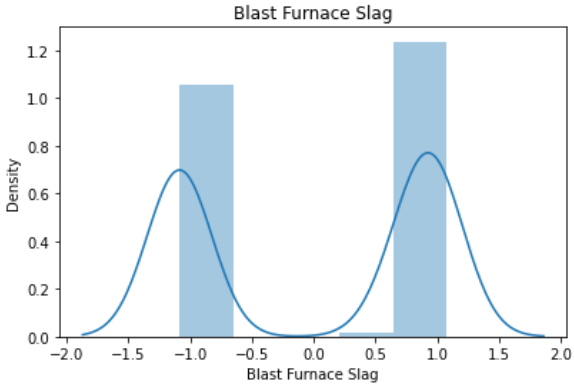
```
C:\Users\iampr\AppData\Local\Temp\ipykernel_3260\1031937615.py:7: UserWarnin
g:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(X_train[col])
C:\Users\iampr\AppData\Local\Programs\Python\Python310\lib\site-packages\sea
born\_oldcore.py:1119: FutureWarning: use_inf_as_na option is deprecated and
will be removed in a future version. Convert inf values to NaN before operat
ing instead.
  with pd.option_context('mode.use_inf_as_na', True):
C:\Users\iampr\AppData\Local\Temp\ipykernel_3260\1031937615.py:11: UserWarni
ng:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(X_train_transformed[col])
C:\Users\iampr\AppData\Local\Programs\Python\Python310\lib\site-packages\sea
born\_oldcore.py:1119: FutureWarning: use_inf_as_na option is deprecated and
will be removed in a future version. Convert inf values to NaN before operat
ing instead.
  with pd.option_context('mode.use_inf_as_na', True):
```



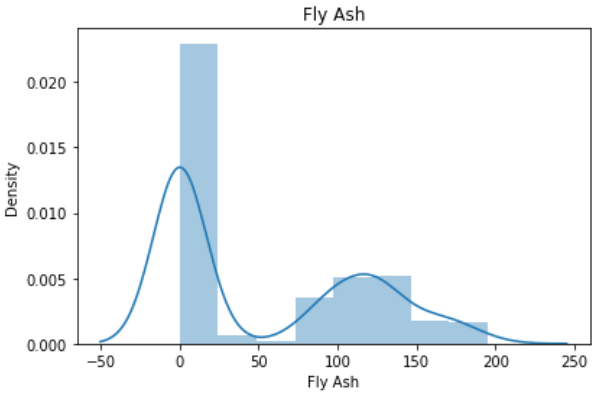


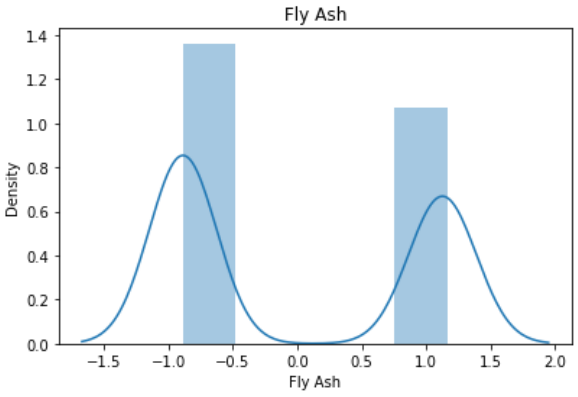
```
C:\Users\iampr\AppData\Local\Temp\ipykernel_3260\1031937615.py:7: UserWarnin
g:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(X_train[col])
C:\Users\iampr\AppData\Local\Programs\Python\Python310\lib\site-packages\sea
born\_oldcore.py:1119: FutureWarning: use_inf_as_na option is deprecated and
will be removed in a future version. Convert inf values to NaN before operat
ing instead.
  with pd.option_context('mode.use_inf_as_na', True):
C:\Users\iampr\AppData\Local\Temp\ipykernel_3260\1031937615.py:11: UserWarni
ng:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(X_train_transformed[col])
C:\Users\iampr\AppData\Local\Programs\Python\Python310\lib\site-packages\sea
born\_oldcore.py:1119: FutureWarning: use_inf_as_na option is deprecated and
will be removed in a future version. Convert inf values to NaN before operat
ing instead.
  with pd.option_context('mode.use_inf_as_na', True):
```



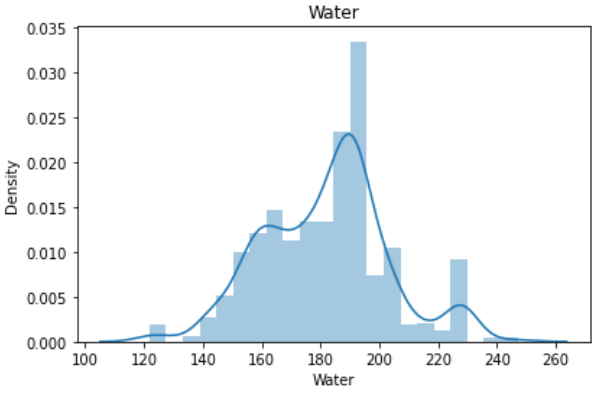


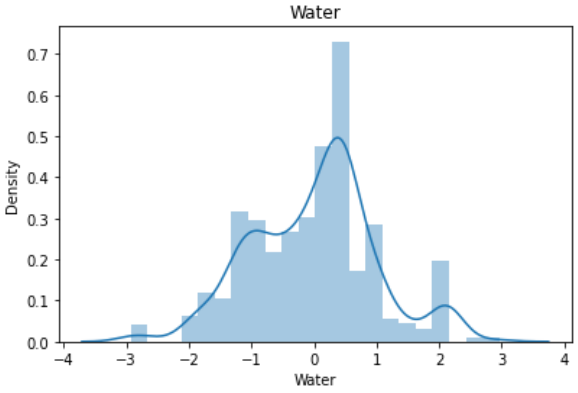
```
C:\Users\iampr\AppData\Local\Temp\ipykernel_3260\1031937615.py:7: UserWarnin
g:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(X_train[col])
C:\Users\iampr\AppData\Local\Programs\Python\Python310\lib\site-packages\sea
born\_oldcore.py:1119: FutureWarning: use_inf_as_na option is deprecated and
will be removed in a future version. Convert inf values to NaN before operat
ing instead.
  with pd.option_context('mode.use_inf_as_na', True):
C:\Users\iampr\AppData\Local\Temp\ipykernel_3260\1031937615.py:11: UserWarni
ng:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(X_train_transformed[col])
C:\Users\iampr\AppData\Local\Programs\Python\Python310\lib\site-packages\sea
born\_oldcore.py:1119: FutureWarning: use_inf_as_na option is deprecated and
will be removed in a future version. Convert inf values to NaN before operat
ing instead.
  with pd.option_context('mode.use_inf_as_na', True):
```



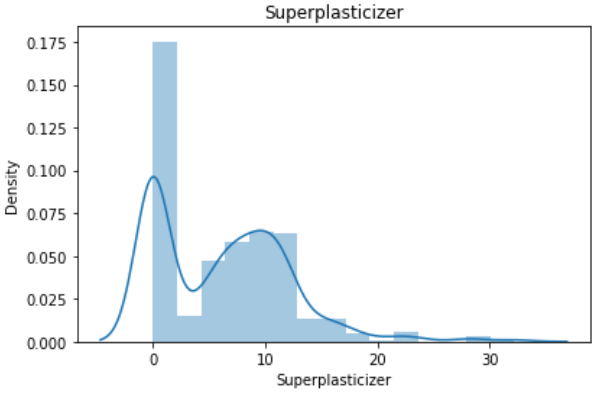


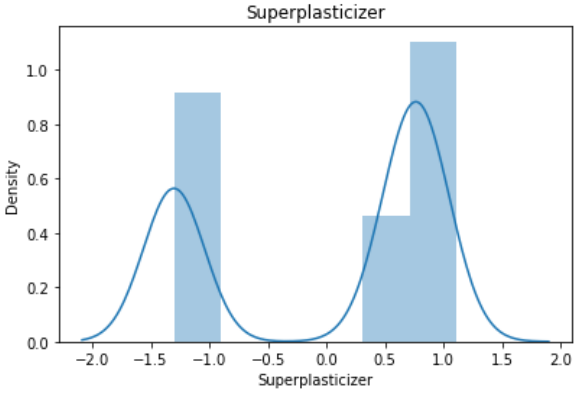
```
C:\Users\iampr\AppData\Local\Temp\ipykernel_3260\1031937615.py:7: UserWarnin
g:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(X_train[col])
C:\Users\iampr\AppData\Local\Programs\Python\Python310\lib\site-packages\sea
born\_oldcore.py:1119: FutureWarning: use_inf_as_na option is deprecated and
will be removed in a future version. Convert inf values to NaN before operat
ing instead.
  with pd.option_context('mode.use_inf_as_na', True):
C:\Users\iampr\AppData\Local\Temp\ipykernel_3260\1031937615.py:11: UserWarni
ng:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(X_train_transformed[col])
C:\Users\iampr\AppData\Local\Programs\Python\Python310\lib\site-packages\sea
born\_oldcore.py:1119: FutureWarning: use_inf_as_na option is deprecated and
will be removed in a future version. Convert inf values to NaN before operat
ing instead.
  with pd.option_context('mode.use_inf_as_na', True):
```



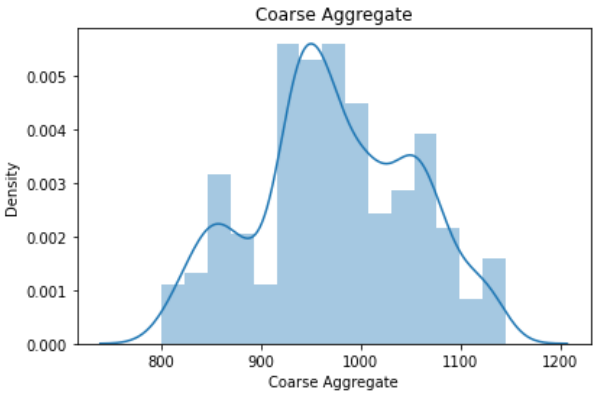


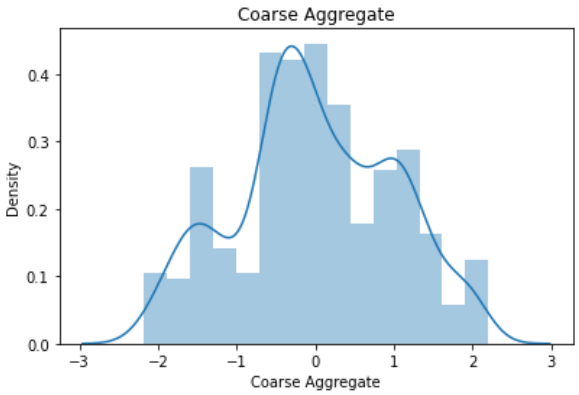
```
C:\Users\iampr\AppData\Local\Temp\ipykernel_3260\1031937615.py:7: UserWarnin
g:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(X_train[col])
C:\Users\iampr\AppData\Local\Programs\Python\Python310\lib\site-packages\sea
born\_oldcore.py:1119: FutureWarning: use_inf_as_na option is deprecated and
will be removed in a future version. Convert inf values to NaN before operat
ing instead.
  with pd.option_context('mode.use_inf_as_na', True):
C:\Users\iampr\AppData\Local\Temp\ipykernel_3260\1031937615.py:11: UserWarni
ng:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(X_train_transformed[col])
C:\Users\iampr\AppData\Local\Programs\Python\Python310\lib\site-packages\sea
born\_oldcore.py:1119: FutureWarning: use_inf_as_na option is deprecated and
will be removed in a future version. Convert inf values to NaN before operat
ing instead.
  with pd.option_context('mode.use_inf_as_na', True):
```



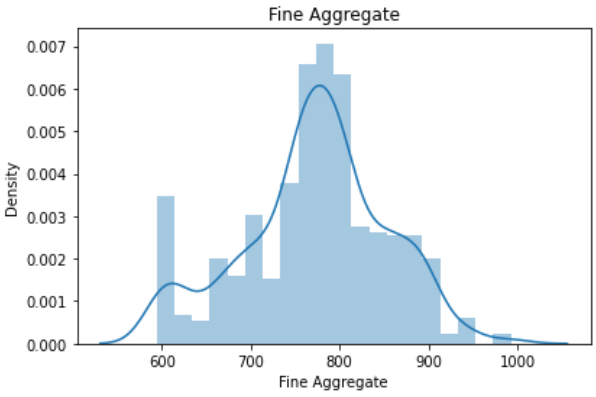


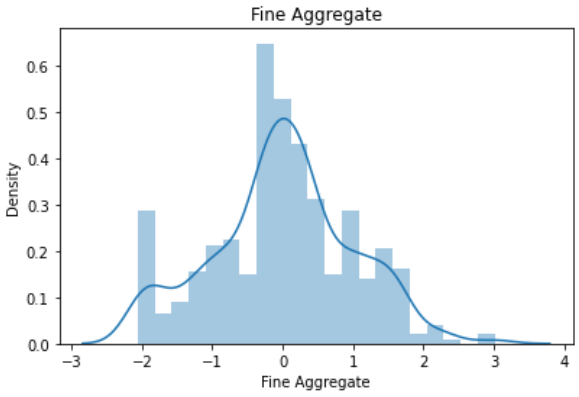
```
C:\Users\iampr\AppData\Local\Temp\ipykernel_3260\1031937615.py:7: UserWarnin
g:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(X_train[col])
C:\Users\iampr\AppData\Local\Programs\Python\Python310\lib\site-packages\sea
born\_oldcore.py:1119: FutureWarning: use_inf_as_na option is deprecated and
will be removed in a future version. Convert inf values to NaN before operat
ing instead.
  with pd.option_context('mode.use_inf_as_na', True):
C:\Users\iampr\AppData\Local\Temp\ipykernel_3260\1031937615.py:11: UserWarni
ng:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(X_train_transformed[col])
C:\Users\iampr\AppData\Local\Programs\Python\Python310\lib\site-packages\sea
born\_oldcore.py:1119: FutureWarning: use_inf_as_na option is deprecated and
will be removed in a future version. Convert inf values to NaN before operat
ing instead.
  with pd.option_context('mode.use_inf_as_na', True):
```



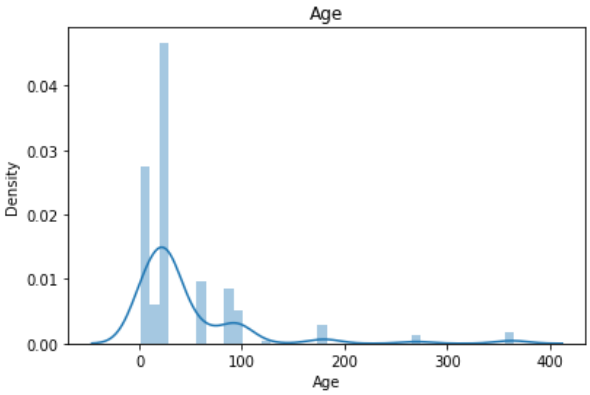


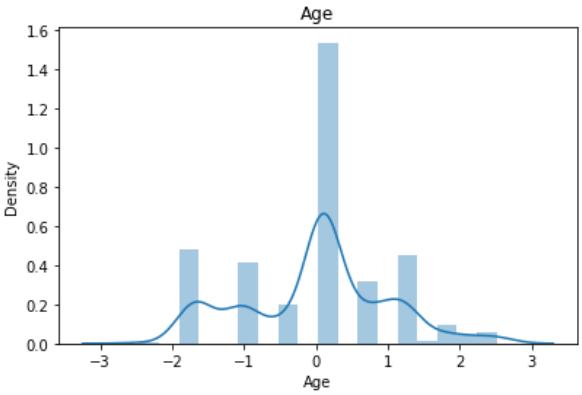
```
C:\Users\iampr\AppData\Local\Temp\ipykernel_3260\1031937615.py:7: UserWarnin
g:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(X_train[col])
C:\Users\iampr\AppData\Local\Programs\Python\Python310\lib\site-packages\sea
born\_oldcore.py:1119: FutureWarning: use_inf_as_na option is deprecated and
will be removed in a future version. Convert inf values to NaN before operat
ing instead.
  with pd.option_context('mode.use_inf_as_na', True):
C:\Users\iampr\AppData\Local\Temp\ipykernel_3260\1031937615.py:11: UserWarni
ng:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(X_train_transformed[col])
C:\Users\iampr\AppData\Local\Programs\Python\Python310\lib\site-packages\sea
born\_oldcore.py:1119: FutureWarning: use_inf_as_na option is deprecated and
will be removed in a future version. Convert inf values to NaN before operat
ing instead.
  with pd.option_context('mode.use_inf_as_na', True):
```





```
C:\Users\iampr\AppData\Local\Temp\ipykernel_3260\1031937615.py:7: UserWarnin
g:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(X_train[col])
C:\Users\iampr\AppData\Local\Programs\Python\Python310\lib\site-packages\sea
born\_oldcore.py:1119: FutureWarning: use_inf_as_na option is deprecated and
will be removed in a future version. Convert inf values to NaN before operat
ing instead.
  with pd.option_context('mode.use_inf_as_na', True):
C:\Users\iampr\AppData\Local\Temp\ipykernel_3260\1031937615.py:11: UserWarni
ng:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(X_train_transformed[col])
C:\Users\iampr\AppData\Local\Programs\Python\Python310\lib\site-packages\sea
born\_oldcore.py:1119: FutureWarning: use_inf_as_na option is deprecated and
will be removed in a future version. Convert inf values to NaN before operat
ing instead.
  with pd.option_context('mode.use_inf_as_na', True):
```





In [94]:
```
# Apply Yeo-Johnson transform

pt1 = PowerTransformer()

X_train_transformed2 = pt1.fit_transform(X_train)
X_test_transformed2 = pt1.transform(X_test)

lr = LinearRegression()
lr.fit(X_train_transformed2,y_train)
```

```
y_pred3 = lr.predict(X_test_transformed2)

print(r2_score(y_test,y_pred3))

pd.DataFrame({'cols':X_train.columns,'Yeo_Johnson_lambdas':pt1.lambdas_})
```

```
0.8161906513354853
```

Out[94]:

	cols	Yeo_Johnson_lambdas
0	Cement	0.174348
1	Blast Furnace Slag	0.015715
2	Fly Ash	-0.161447
3	Water	0.771307
4	Superplasticizer	0.253935
5	Coarse Aggregate	1.130050
6	Fine Aggregate	1.783100
7	Age	0.019885

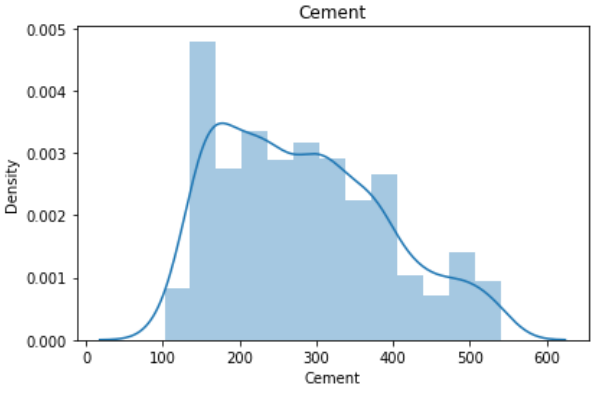
In [95]:
```
# applying cross val score

pt = PowerTransformer()
X_transformed2 = pt.fit_transform(X)

lr = LinearRegression()
np.mean(cross_val_score(lr,X_transformed2,y,scoring='r2'))
```

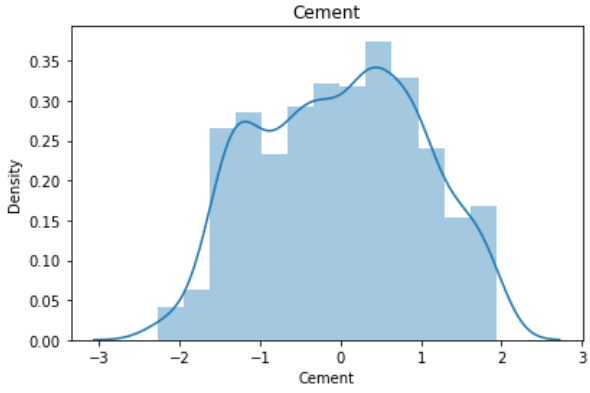
Out[95]:
```
0.6834625141500865
```

In [96]:
```
X_train_transformed2 = pd.DataFrame(X_train_transformed2,columns=X_train.col
```

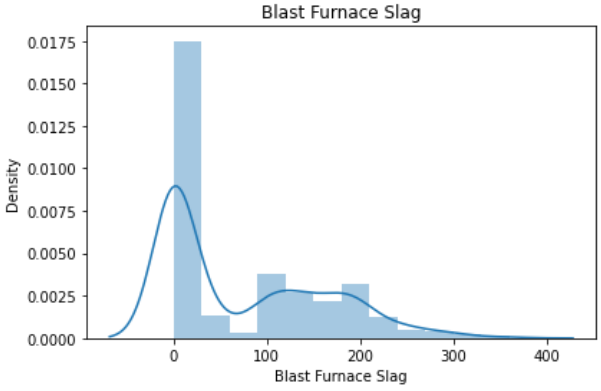
In [97]:
```
# Before and after comparision for Yeo-Johnson

for col in X_train_transformed2.columns:
    plt.figure(figsize=(14,4))
    plt.subplot(121)
    sns.distplot(X_train[col])
    plt.title(col)

    plt.subplot(122)
    sns.distplot(X_train_transformed2[col])
    plt.title(col)

    plt.show()
```

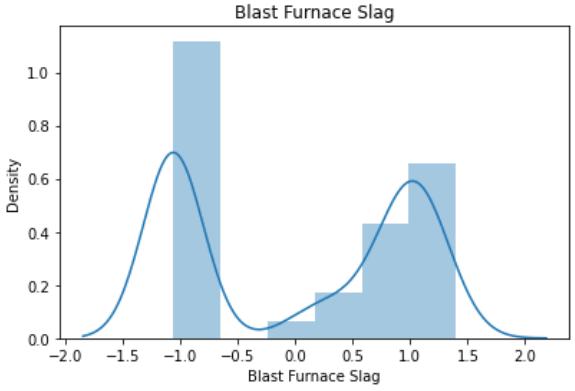
```
C:\Users\iampr\AppData\Local\Temp\ipykernel_3260\1516575210.py:6: UserWarnin
g:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(X_train[col])
C:\Users\iampr\AppData\Local\Programs\Python\Python310\lib\site-packages\sea
born\_oldcore.py:1119: FutureWarning: use_inf_as_na option is deprecated and
will be removed in a future version. Convert inf values to NaN before operat
ing instead.
  with pd.option_context('mode.use_inf_as_na', True):
C:\Users\iampr\AppData\Local\Temp\ipykernel_3260\1516575210.py:10: UserWarni
ng:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(X_train_transformed2[col])
C:\Users\iampr\AppData\Local\Programs\Python\Python310\lib\site-packages\sea
born\_oldcore.py:1119: FutureWarning: use_inf_as_na option is deprecated and
will be removed in a future version. Convert inf values to NaN before operat
ing instead.
  with pd.option_context('mode.use_inf_as_na', True):
```



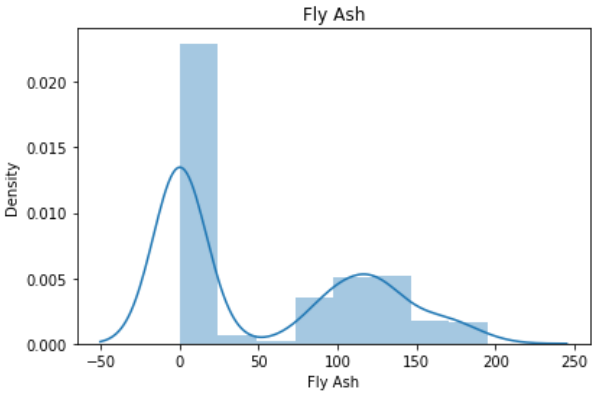


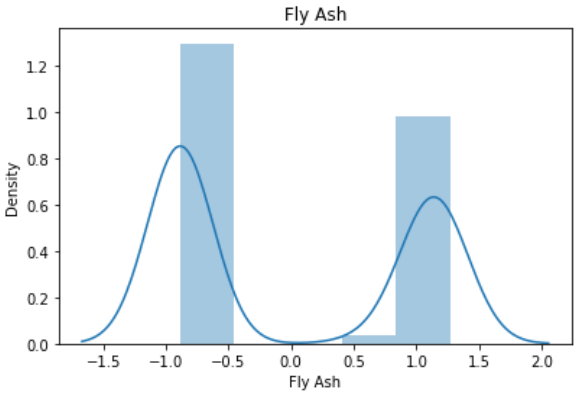
```
C:\Users\iampr\AppData\Local\Temp\ipykernel_3260\1516575210.py:6: UserWarnin
g:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(X_train[col])
C:\Users\iampr\AppData\Local\Programs\Python\Python310\lib\site-packages\sea
born\_oldcore.py:1119: FutureWarning: use_inf_as_na option is deprecated and
will be removed in a future version. Convert inf values to NaN before operat
ing instead.
  with pd.option_context('mode.use_inf_as_na', True):
C:\Users\iampr\AppData\Local\Temp\ipykernel_3260\1516575210.py:10: UserWarni
ng:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(X_train_transformed2[col])
C:\Users\iampr\AppData\Local\Programs\Python\Python310\lib\site-packages\sea
born\_oldcore.py:1119: FutureWarning: use_inf_as_na option is deprecated and
will be removed in a future version. Convert inf values to NaN before operat
ing instead.
  with pd.option_context('mode.use_inf_as_na', True):
```



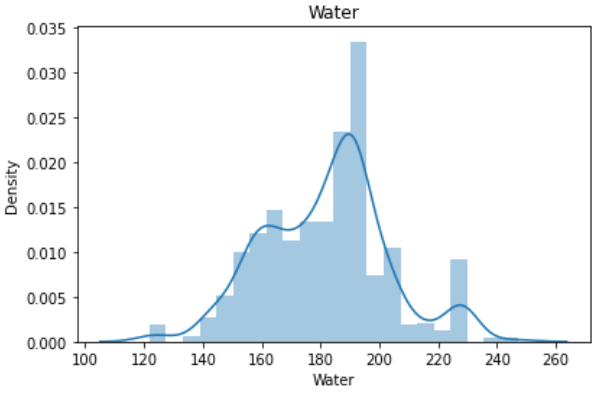


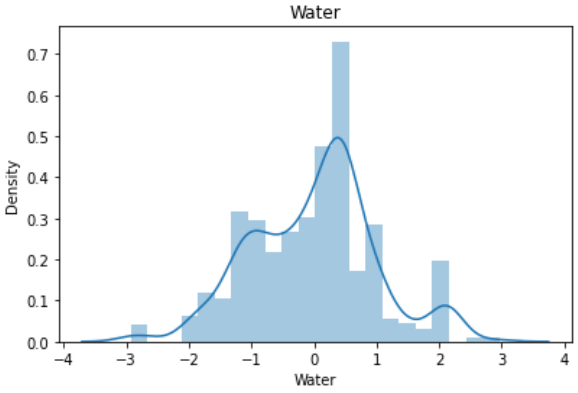
```
C:\Users\iampr\AppData\Local\Temp\ipykernel_3260\1516575210.py:6: UserWarnin
g:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(X_train[col])
C:\Users\iampr\AppData\Local\Programs\Python\Python310\lib\site-packages\sea
born\_oldcore.py:1119: FutureWarning: use_inf_as_na option is deprecated and
will be removed in a future version. Convert inf values to NaN before operat
ing instead.
  with pd.option_context('mode.use_inf_as_na', True):
C:\Users\iampr\AppData\Local\Temp\ipykernel_3260\1516575210.py:10: UserWarni
ng:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(X_train_transformed2[col])
C:\Users\iampr\AppData\Local\Programs\Python\Python310\lib\site-packages\sea
born\_oldcore.py:1119: FutureWarning: use_inf_as_na option is deprecated and
will be removed in a future version. Convert inf values to NaN before operat
ing instead.
  with pd.option_context('mode.use_inf_as_na', True):
```



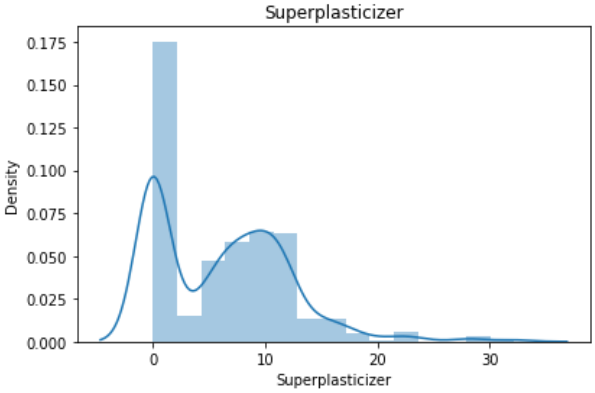


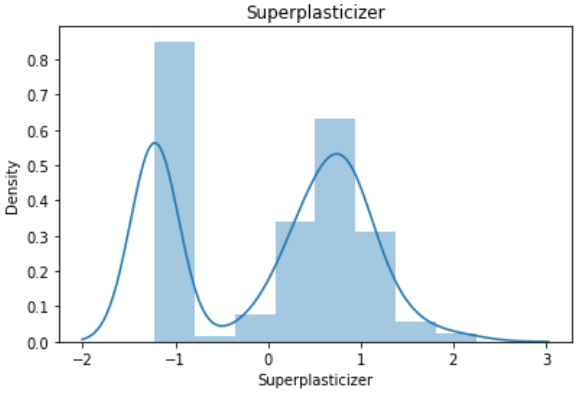
```
C:\Users\iampr\AppData\Local\Temp\ipykernel_3260\1516575210.py:6: UserWarnin
g:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(X_train[col])
C:\Users\iampr\AppData\Local\Programs\Python\Python310\lib\site-packages\sea
born\_oldcore.py:1119: FutureWarning: use_inf_as_na option is deprecated and
will be removed in a future version. Convert inf values to NaN before operat
ing instead.
  with pd.option_context('mode.use_inf_as_na', True):
C:\Users\iampr\AppData\Local\Temp\ipykernel_3260\1516575210.py:10: UserWarni
ng:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(X_train_transformed2[col])
C:\Users\iampr\AppData\Local\Programs\Python\Python310\lib\site-packages\sea
born\_oldcore.py:1119: FutureWarning: use_inf_as_na option is deprecated and
will be removed in a future version. Convert inf values to NaN before operat
ing instead.
  with pd.option_context('mode.use_inf_as_na', True):
```



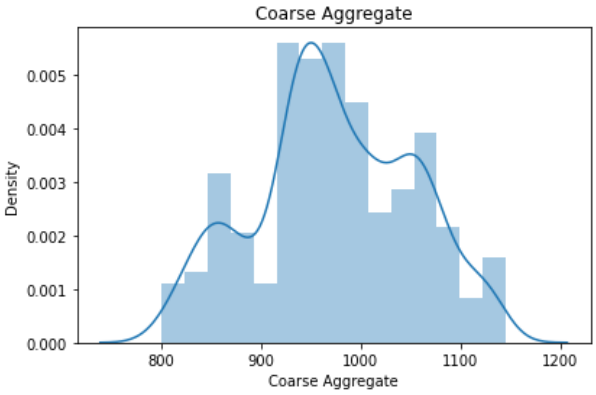


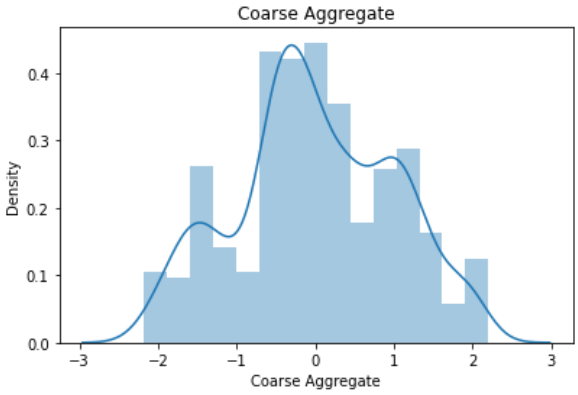
```
C:\Users\iampr\AppData\Local\Temp\ipykernel_3260\1516575210.py:6: UserWarnin
g:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(X_train[col])
C:\Users\iampr\AppData\Local\Programs\Python\Python310\lib\site-packages\sea
born\_oldcore.py:1119: FutureWarning: use_inf_as_na option is deprecated and
will be removed in a future version. Convert inf values to NaN before operat
ing instead.
  with pd.option_context('mode.use_inf_as_na', True):
C:\Users\iampr\AppData\Local\Temp\ipykernel_3260\1516575210.py:10: UserWarni
ng:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(X_train_transformed2[col])
C:\Users\iampr\AppData\Local\Programs\Python\Python310\lib\site-packages\sea
born\_oldcore.py:1119: FutureWarning: use_inf_as_na option is deprecated and
will be removed in a future version. Convert inf values to NaN before operat
ing instead.
  with pd.option_context('mode.use_inf_as_na', True):
```



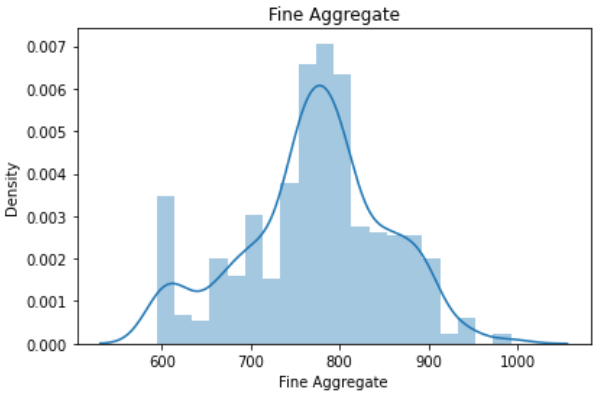


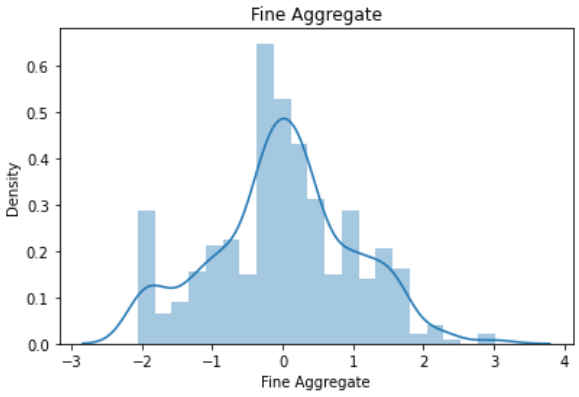
```
C:\Users\iampr\AppData\Local\Temp\ipykernel_3260\1516575210.py:6: UserWarnin
g:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(X_train[col])
C:\Users\iampr\AppData\Local\Programs\Python\Python310\lib\site-packages\sea
born\_oldcore.py:1119: FutureWarning: use_inf_as_na option is deprecated and
will be removed in a future version. Convert inf values to NaN before operat
ing instead.
  with pd.option_context('mode.use_inf_as_na', True):
C:\Users\iampr\AppData\Local\Temp\ipykernel_3260\1516575210.py:10: UserWarni
ng:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(X_train_transformed2[col])
C:\Users\iampr\AppData\Local\Programs\Python\Python310\lib\site-packages\sea
born\_oldcore.py:1119: FutureWarning: use_inf_as_na option is deprecated and
will be removed in a future version. Convert inf values to NaN before operat
ing instead.
  with pd.option_context('mode.use_inf_as_na', True):
```



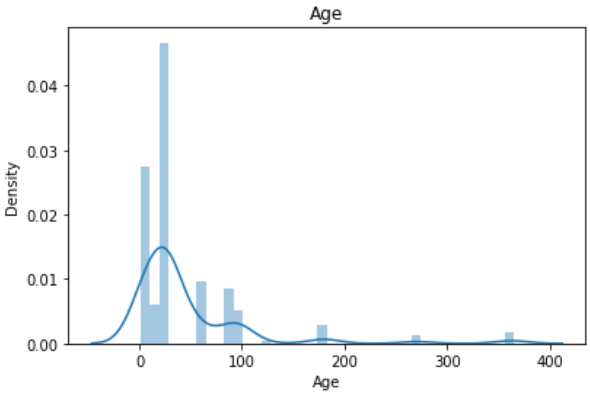


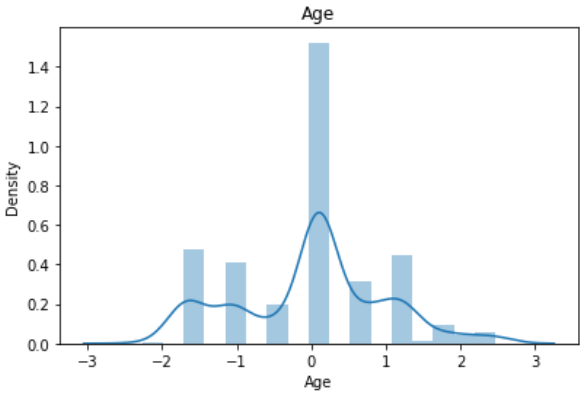
```
C:\Users\iampr\AppData\Local\Temp\ipykernel_3260\1516575210.py:6: UserWarnin
g:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(X_train[col])
C:\Users\iampr\AppData\Local\Programs\Python\Python310\lib\site-packages\sea
born\_oldcore.py:1119: FutureWarning: use_inf_as_na option is deprecated and
will be removed in a future version. Convert inf values to NaN before operat
ing instead.
  with pd.option_context('mode.use_inf_as_na', True):
C:\Users\iampr\AppData\Local\Temp\ipykernel_3260\1516575210.py:10: UserWarni
ng:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(X_train_transformed2[col])
C:\Users\iampr\AppData\Local\Programs\Python\Python310\lib\site-packages\sea
born\_oldcore.py:1119: FutureWarning: use_inf_as_na option is deprecated and
will be removed in a future version. Convert inf values to NaN before operat
ing instead.
  with pd.option_context('mode.use_inf_as_na', True):
```





```
C:\Users\iampr\AppData\Local\Temp\ipykernel_3260\1516575210.py:6: UserWarnin
g:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(X_train[col])
C:\Users\iampr\AppData\Local\Programs\Python\Python310\lib\site-packages\sea
born\_oldcore.py:1119: FutureWarning: use_inf_as_na option is deprecated and
will be removed in a future version. Convert inf values to NaN before operat
ing instead.
  with pd.option_context('mode.use_inf_as_na', True):
C:\Users\iampr\AppData\Local\Temp\ipykernel_3260\1516575210.py:10: UserWarni
ng:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(X_train_transformed2[col])
C:\Users\iampr\AppData\Local\Programs\Python\Python310\lib\site-packages\sea
born\_oldcore.py:1119: FutureWarning: use_inf_as_na option is deprecated and
will be removed in a future version. Convert inf values to NaN before operat
ing instead.
  with pd.option_context('mode.use_inf_as_na', True):
```





In [100…

```
# Side by side Lambdas
df = pd.DataFrame({'cols':X_train.columns,'box_cox_lambdas':pt.lambdas_,'Yeo
df['diff'] = df['box_cox_lambdas'] - df['Yeo_Johnson_lambdas']
df
```

Out[100…

	cols	box_cox_lambdas	Yeo_Johnson_lambdas	diff
0	Cement	0.169544	0.174348	-0.004804
1	Blast Furnace Slag	0.016633	0.015715	0.000918
2	Fly Ash	-0.136480	-0.161447	0.024967
3	Water	0.808438	0.771307	0.037131
4	Superplasticizer	0.264160	0.253935	0.010225
5	Coarse Aggregate	1.129395	1.130050	-0.000655
6	Fine Aggregate	1.830763	1.783100	0.047664
7	Age	0.001771	0.019885	-0.018114

In [ ]: