Importing Libraries

In [1]:
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

from scipy.stats import norm
```

Gaussian and CLT Recap

Purchase time

The average time taken for customers to complete a purchase is 4 minutes with a standard deviation of 1 minute. Find the probability that a randomly selected customer will complete a purchase within 6 minutes? Assume Gaussian

In [2]:
```
norm.cdf(x=6,loc=4,scale=1)
```

Out[2]:
```
0.9772498680518208
```

What is the probability that the average time of the next 5 customers is less than 6 minutes?

In [5]:
```
norm.cdf(x=6,loc=4,scale=(1/np.sqrt(5)))
```

Out[5]:
```
0.9999991278917845
```

Purchase amount

The average order value on an e-commerce website is 50, with a standard deviation of 5. What is the probability that a randomly selected order will have a value exceeding 60?

In [3]:
```
1-norm.cdf(x=60,loc=50,scale=5)
```

Out[3]:
```
0.022750131948171921
```

What is the probability that the average of the next 3 orders exceeds $60?

In [6]:
```
1-norm.cdf(x=60,loc=50,scale=(5/np.sqrt(3)))
```

Out[6]:
```
0.0062660027525696515
```

Body temperature

Average body temperature has a mean of 98.6°F and a standard deviation of 0.5°F. What is the probability that a randomly chosen patient has a body temperature higher than 99.5°F?

In [4]:
```
1-norm.cdf(x=99.5,loc=98.6,scale=0.5)
```

Out[4]:
```
0.03593031911292488
```

In [ ]:
```

```

Confidence Interval using CLT

Height example

The mean height of a sample of 100 adults was found to be 65 inches, with a standard deviation of 2.5 inches.

In [12]:
```
std_error=2.5/np.sqrt(100)
std_error
```

Out[12]:
```
0.25
```

In [13]:
```
z1=norm.ppf(0.025)
z1
```

Out[13]:
```
-1.959963984540054
```

In [14]:
```
z2=norm.ppf(0.975)
z2
```

Out[14]:
```
1.959963984540054
```

In [15]:
```
x1=65+(z1*std_error)
x1
```

Out[15]:
```
64.51000900386498
```

In [16]:
```
x2=65+(z2*std_error)
x2
```

Out[16]:
```
65.48999099613502
```

95 % Confidence Interval --> [64.51,65.48]

In [17]:
```
norm.interval(0.95,loc=65,scale=std_error)
```

Out[17]:
```
(64.51000900386498, 65.48999099613502)
```

Recovery days

The sample mean recovery time of 100 patients after taking a drug was seen to be 10.5 days with a standard deviation of 2 days. Find the 95% confidence interval of the true mean.

In [19]:
```
norm.interval(0.95,loc=10.5,scale=(2/np.sqrt(100)))
```

Out[19]:
```
(10.108007203099199, 10.891992796900801)
```

In [25]:
```
std_error= 2/np.sqrt(100)
std_error
```

Out[25]:
```
0.2
```

In [26]:
```
z1=norm.ppf(0.025)
z2=norm.ppf(0.975)
```

In [27]:
```
x1=10.5+(z1*std_error)
x1
```

Out[27]:
```
10.108007203091988
```

In [28]:
```
x2=10.5+(z2*std_error)
x2
```

Out[28]:
```
10.891992796908081
```

Youtube watch hours

The mean Youtube watch time of a sample of 100 students was found to be 3.5 hours, with a standard deviation of 1 hour. Construct a 90% confidence interval for the true watch time.

In [29]:
```
norm.interval(0.90,loc=3.5,scale=(1/np.sqrt(100)))
```

Out[29]:
```
(3.335514637304852, 3.6644853626951472)
```

In [ ]:
```

```

Confidence Interval using Bootstrap

In [30]:
```
survey_1 = [35, 36, 33, 37, 34, 35]
np.mean(survey_1)
```

Out[30]:
```
35.0
```

In [31]:
```
survey_2 = [20, 37, 17, 50, 53, 33]
np.mean(survey_2)
```

Out[31]:
```
35.0
```

In [82]:
```
np.random.choice(survey_1,size=10)
```

Out[82]:
```
array([37, 37, 37, 35, 35, 33, 37, 36, 35, 36])
```

In [126...
```
bootstrapped_sample=np.random.choice(survey_1,size=6)
np.mean(bootstrapped_sample)
```

Out[126]:
```
35.666666666666664
```

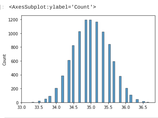
In [147...
```
bootstrapped_sample_2=np.random.choice(survey_2,size=6)
np.mean(bootstrapped_sample_2)
```

Out[147]:
```
32.166666666666664
```

In [150...
```
bootstrapped_sample_mean=[]
for i in range(10000):
    bootstrapped_sample=np.random.choice(survey_1,size=6)
    bootstrapped_sample_mean.append(np.mean(bootstrapped_sample))
```

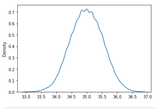
In [155...
```
sns.histplot(bootstrapped_sample_mean)
```

Out[155]:
```
<AxesSubplot:ylabel='Count'>
```



In [152...
```
sns.kdeplot(bootstrapped_sample_mean)
```

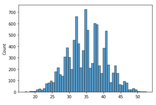
Out[152]:
```
<AxesSubplot:ylabel='Density'>
```



In [153...
```
bootstrapped_sample_mean_2=[]
for i in range(10000):
    bootstrapped_sample_2=np.random.choice(survey_2,size=6)
    bootstrapped_sample_mean_2.append(np.mean(bootstrapped_sample_2))
```

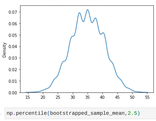
In [154...
```
sns.histplot(bootstrapped_sample_mean_2)
```

Out[154]:
```
<AxesSubplot:ylabel='Count'>
```



In [155...
```
sns.kdeplot(bootstrapped_sample_mean_2)
```

Out[155]:
```
<AxesSubplot:ylabel='Density'>
```



In [156...
```
np.percentile(bootstrapped_sample_mean,2.5)
```

Out[156]:
```
34.0
```

In [157...
```
np.percentile(bootstrapped_sample_mean,97.5)
```

Out[157]:
```
36.0
```

For Survey 1 --> 95 % CI [34.0,36.0]

In [158...
```
np.percentile(bootstrapped_sample_mean_2,2.5)
```

Out[158]:
```
24.0
```

In [159...
```
np.percentile(bootstrapped_sample_mean_2,97.5)
```

Out[159]:
```
46.0
```

For Survey 1 --> 95 % CI [24.0, 46.0]

In [ ]:
```

```

In [160...
```
df=pd.read_csv("sehwag.csv")
df
```

Out[160]:

	Runs	Mins	BF	4s	6s	SR	Pos	Dismissal	Inns	Unnamed: 9	Opposition	Ground	Start Date	Unnamed: 13
0	1	5	2	0	0	50.00	7	lbw	1	NaN	v Pakistan	Mohali	1 Apr 1999	ODI # 1427
1	19	18	24	0	1	79.16	6	caught	1	NaN	v Zimbabwe	Rajkot	14 Dec 2000	ODI # 1660
2	58	62	54	8	0	107.40	6	bowled	1	NaN	v Australia	Bengaluru	25 Mar 2001	ODI # 1696
3	7	7	0	0	0	28.57	6	caught	2	NaN	v Zimbabwe	Bulawayo	27 Jun 2001	ODI # 1730
4	11	19	16	1	0	68.75	6	not out	2	NaN	v West Indies	Bulawayo	30 Jun 2001	ODI # 1731
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
240	15	21	15	2	0	100.00	2	caught	1	NaN	v Sri Lanka	Hambantota	24 Jul 2012	ODI # 3292
241	3	6	6	0	0	50.00	2	caught	2	NaN	v Sri Lanka	Colombo (RPS)	28 Jul 2012	ODI # 3293
242	68	46	29	6	0	117.24	2	caught	2	NaN	v Sri Lanka	Colombo (RPS)	31 Jul 2012	ODI # 3294
243	4	20	11	1	0	36.36	2	bowled	1	NaN	v Pakistan	Chennai	30 Dec 2012	ODI # 3314
244	31	70	43	3	0	72.09	2	lbw	2	NaN	v Pakistan	Kolkata	3 Jan 2013	ODI # 3315

245 rows × 14 columns

In [162...
```
sample_size=6
```

In [163...
```
df["Runs"].shape
```

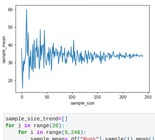
Out[163]:
```
(245,)
```

In [164...
```
df["Runs"].sample(sample_size).mean()
```

Out[164]:
```
12.333333333333334
```

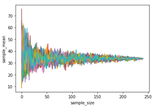
In [167...
```
sample_size_trend=[]
for i in range(5,246):
    sample_mean= df["Runs"].sample(i).mean()
    sample_size_trend.append(sample_mean)
plt.plot(sample_size_trend)
plt.xlabel("sample_size")
plt.ylabel("sample_mean")
```

Out[167]:
```
Text(0, 0.5, 'sample_mean')
```



In [168...
```
sample_size_trend=[]
for j in range(20):
    for i in range(5,246):
        sample_mean= df["Runs"].sample(i).mean()
        sample_size_trend.append(sample_mean)
    plt.plot(sample_size_trend)
    sample_size_trend=[]
plt.xlabel("sample_size")
plt.ylabel("sample_mean")
```

Out[168]: Text(0, 0.5, 'sample_mean')



In [ ]: