Content

Central Limit Theorem

The central limit theorem relies on the concept of a sampling distribution, which is the probability distribution of a statistic for a large number of samples taken from a population.

> **Let's recall the sampling distribution**

Draw random samples from a population, calculate means for each sample, and repeat. The collection of these sample means forms a sampling distribution.

Imagine a scenario:

```
Imagine you have a big jar filled with jellybeans.

Each jellybean represents a piece of data in your population.
The color and weight of the jellybeans can be different, representing the diversity in yo

Now, grab a handful of jellybeans from the jar and calculate the average weight of those 

Put those jellybeans back, shake the jar, and grab another handful, calculate the average
```

According to the Central Limit Theorem:

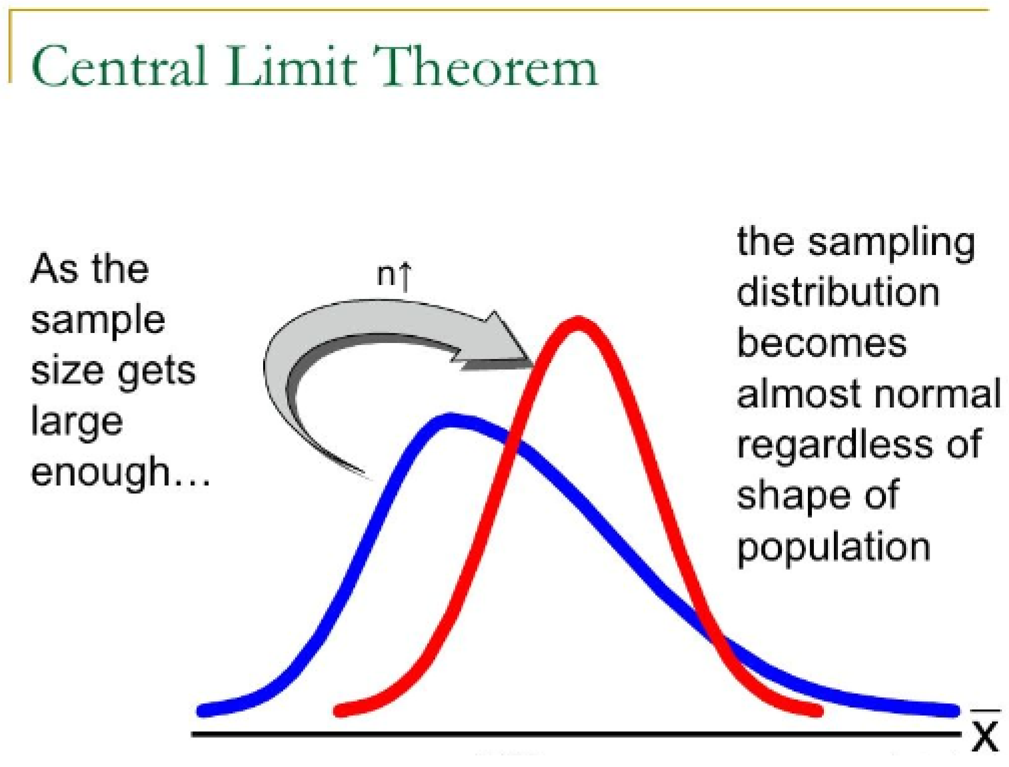
1. **No matter how the weight of the jellybeans are distributed originally**, as you take more and more samples and calculate the average each time, the distribution of those averages will start to look like a bell curve.
2. **The more handfuls of jellybeans you take, the closer the distribution gets to a perfect bell curve**, even if the original distribution of jellybeans was not bell-shaped at all.

This is powerful because it allows us to make certain assumptions and predictions about the averages of large samples, even if they don't know much about the original population.

So,

The central limit states that "the mean of a random sample will resemble even closer to the population mean as the sample size increases and it will approximate a normal distribution regardless of the shape of the population distribution"

- Means, if you take sufficiently large samples from a population, the samples' means **will be normally distributed**, even if the population isn't normally distributed.


Central Limit Theorem
As the sample size gets large enough…
n↑
the sampling distribution becomes almost normal regardless of shape of population
$\overline{X}$

Similarlly,

- If we have a sufficiently large number of independent and identically distributed (i.i.d.) random variables and sum them up, the distribution of the sum will tend to be approximately normal, regardless of the shape of the original distribution.

This is called **CLT for sums of random variables**

In summary both versions essentially state that as you **sum or average a sufficiently large number of i.i.d. random variables**, the resulting distribution tends to approach normality.

The "30" rule of thumb is often mentioned as a guideline for the classical CLT, but the actual requirement may vary based on the characteristics of the population distribution.

Sample size (n) plays a vital role in the context of the Central Limit Theorem (CLT) and its impact can be summarized as follows:

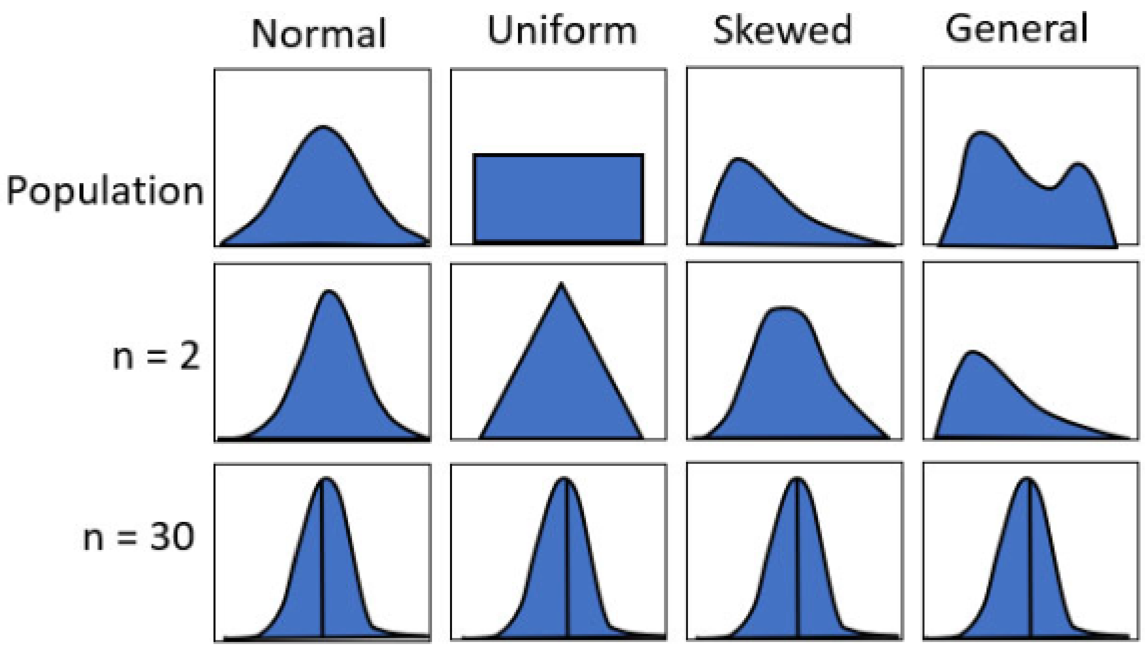
Sample Size and Normality:

A **larger sample size leads** to a sampling distribution that closely resembles a **normal distribution**.

- The CLT tends to work well when the **sample size** (n) **is sufficiently large**, typically considered as $n \geq 30$. However, it is not a strict rule but a rough guideline.

  For **moderately skewed distributions, even smaller sample sizes can sometimes be sufficient**.

- For Small n $(n < 30)$ the CLT can still be useful, especially if the population distribution is close to normal. However, for smaller sample sizes, the normality assumption becomes more critical.



Here,

- We started by taking 2 samples **(n=2)** from the population, calculated their mean, and repeated this many times to form a distribution. This distribution tends to look more like a normal distribution.
- When we **increased the sample size, the distribution resembled a normal distribution even more closely**.

Sample Size and Standard Deviations:

As you can see in the image above, sample size also affects the spread of the sampling distribution.

- A **smaller n (n = 2), will have a high standard deviation** because sample means are less precise estimates of the population mean, resulting in more spread.
- A **larger n (n >= 30) will have a low standard deviation** since sample means become more precise estimates of the population mean, leading to less spread.

Conclusion

The **sample size not only influences how closely the sampling distribution approximates a normal curve but also impacts the spread or precision of sample means**.

As the sample size increases, the CLT becomes more applicable, and the standard deviation decreases, making the estimates of population parameters more reliable.

If we summarize the CLT,

Conditions of the CLT will be

To apply the central limit theorem, the following conditions must be met:

1. **Randomization**:
    - Data should be randomly sampled, ensuring every population member has an equal chance of being included.
2. **Independence**:
    - Each sample value should be independent, with one event's occurrence not affecting another.
    - Commonly met in probability sampling methods, which independently select observations.
3. **Large Sample Condition**:
    - A sample size of 30 or more is generally considered "sufficiently large."
    - This threshold can vary slightly based on the population distribution's shape.

These conditions ensure the applicability of the central limit theorem.

Application of CLT on real life dataset

Let's apply the central limit theorem to real distribution to see if the distribution of the sample means tends to follow normal distribution or not.

We'll take the height dataset on which we have been working from few lectures.

As we know it is already normally distributed, so let's take the sample means and see if they follow normal distribution.

```
!wget --no-check-certificate https://drive.google.com/uc?id=1Mrt008vkE4nVb1zE4f06
```

```
--2024-01-18 10:05:52--  https://drive.google.com/uc?id=1Mrt008vkE4nVb1zE4f06
Resolving drive.google.com (drive.google.com)... 74.125.31.139, 74.125.31.102
Connecting to drive.google.com (drive.google.com)|74.125.31.139|:443... conne
HTTP request sent, awaiting response... 303 See Other
Location: https://drive.usercontent.google.com/download?id=1Mrt008vkE4nVb1zE4
--2024-01-18 10:05:52--  https://drive.usercontent.google.com/download?id=1Mr
Resolving drive.usercontent.google.com (drive.usercontent.google.com)... 108.
Connecting to drive.usercontent.google.com (drive.usercontent.google.com)|108
HTTP request sent, awaiting response... 200 OK
Length: 428120 (418K) [application/octet-stream]
Saving to: ‘weight-height.csv’

weight-height.csv   100%[===================>] 418.09K  --.-KB/s    in 0.005s

2024-01-18 10:05:52 (86.3 MB/s) - ‘weight-height.csv’ saved [428120/428120]
```

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from scipy.stats import norm
```

```
df_hw = pd.read_csv('weight-height.csv')
df_hw.head()
```

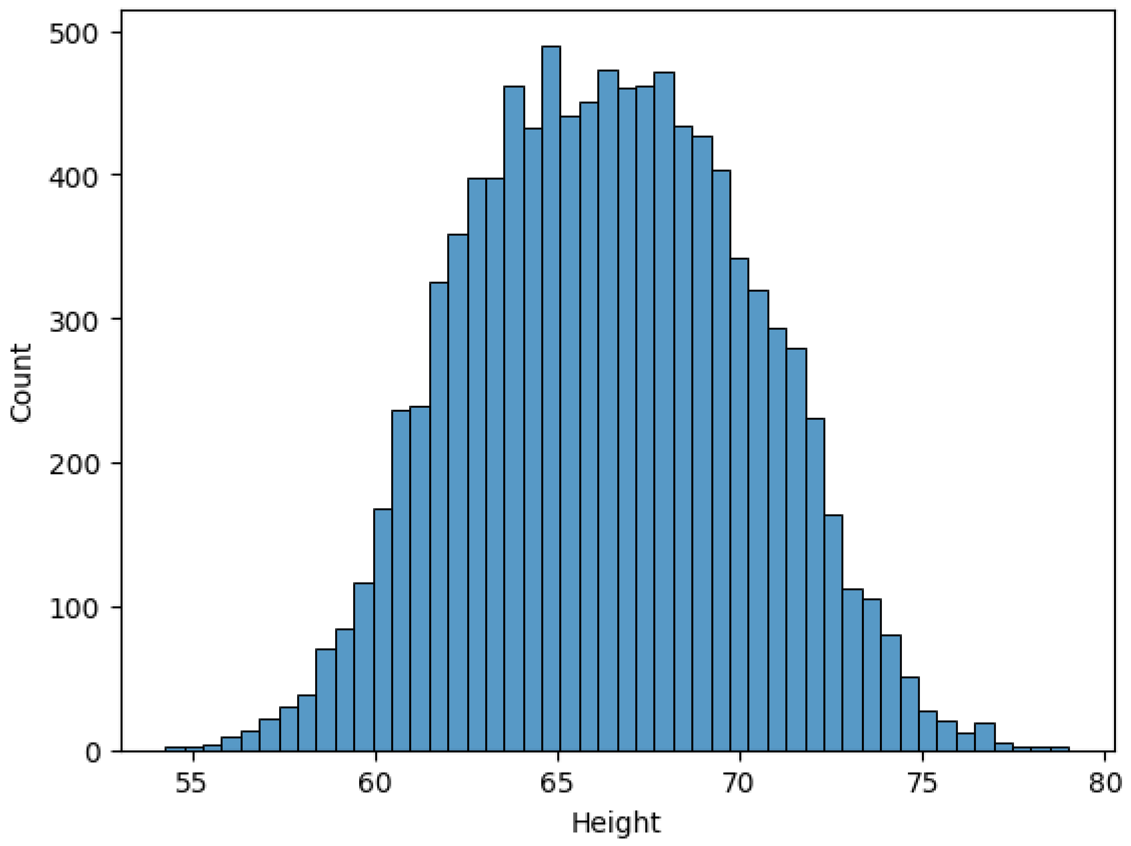
	Gender	Height	Weight
0	Male	73.847017	241.893563
1	Male	68.781904	162.310473
2	Male	74.110105	212.740856
3	Male	71.730978	220.042470
4	Male	69.881796	206.349801

We are going to work on height column so let's store it in a different dataframe.

```
df_height = df_hw["Height"]
```

```
sns.histplot(df_height)
```

```
<Axes: xlabel='Height', ylabel='Count'>
```



```
# mean of the entire population
mu = df_height.mean()
mu
```

```
66.36755975482124
```

```
sigma = df_height.std()
sigma
```

```
3.8475281207732293
```

We will now randomly select five samples and determine the average height of these samples

Sample size = 5

```
df_height.sample(5)
```

```
6370    60.568529
944     70.811262
5038    63.407290
3876    69.322317
```

```
3754    72.442899
Name: Height, dtype: float64
```

```
np.mean(df_height.sample(5))
```

```
65.30073828576094
```

Observation

- We can notice that on running the above code, it is generating 5 different samples every time and the sample mean is also changing with that.

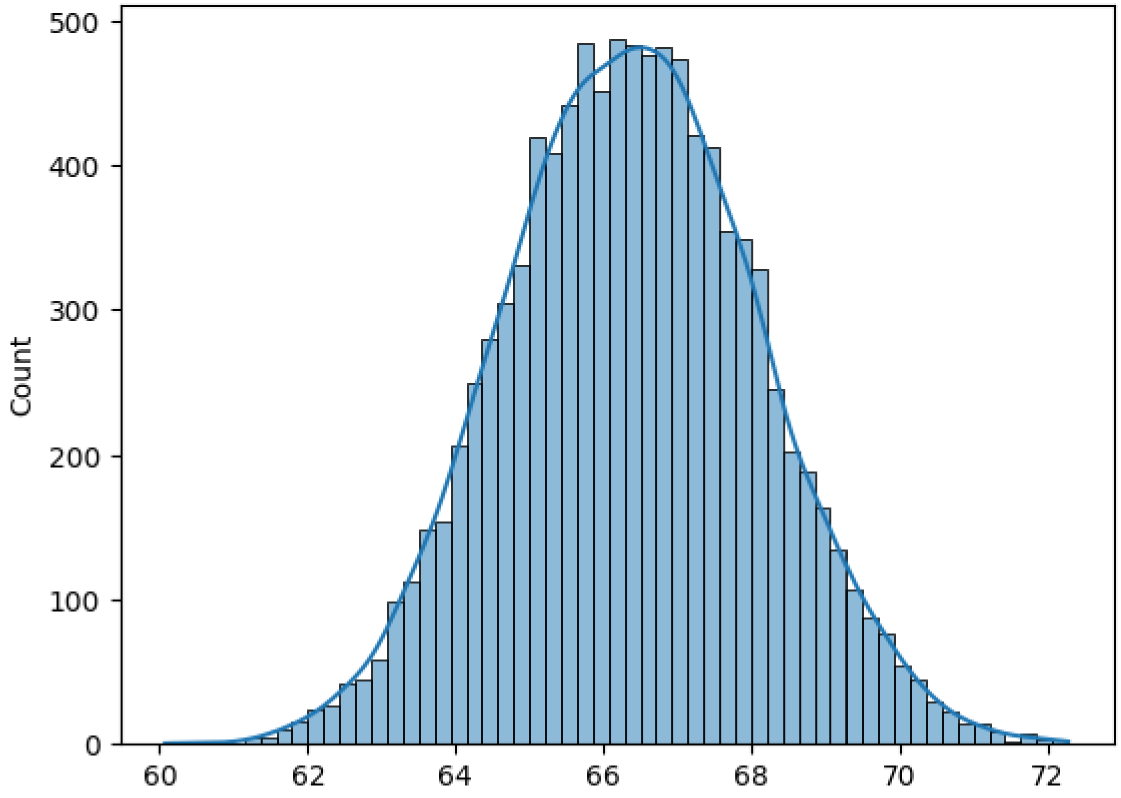
Let's repeat this process 10,000 times so we will get sample means of 10,000 unique samples (size = 5).

We will plot the distributions of these 10,000 sample means to see if they follow the normal distribution.

```
sample_5 = [np.mean(df_height.sample(5)) for i in range(10000) ]
```

```
sns.histplot(sample_5, kde=True)
```

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



```
np.mean(sample_5)
```

```
66.37463912527758
```

```
np.std(sample_5)
```

```
1.7112978334301363
```

Observation

- We can conclude that the distribution of those 10000 samples means is normally distributed and most of the values lies between 62 and 72.
- There might be some cases where the samples contain only short peoples that is why we can see some values between 60 and 62
- Similarly, there might be some cases where the samples contain only tall people that is why we can see some values between 70 and 72.

Sample size = 20

> **Q1. What would happen If we increase the size of our sample?**

We studied earlier in the lecture that as we increase the size of the sample, the spread of data will be less.

- This means, as we increase the size of the sample, the sample mean will come closer and closer to the population mean

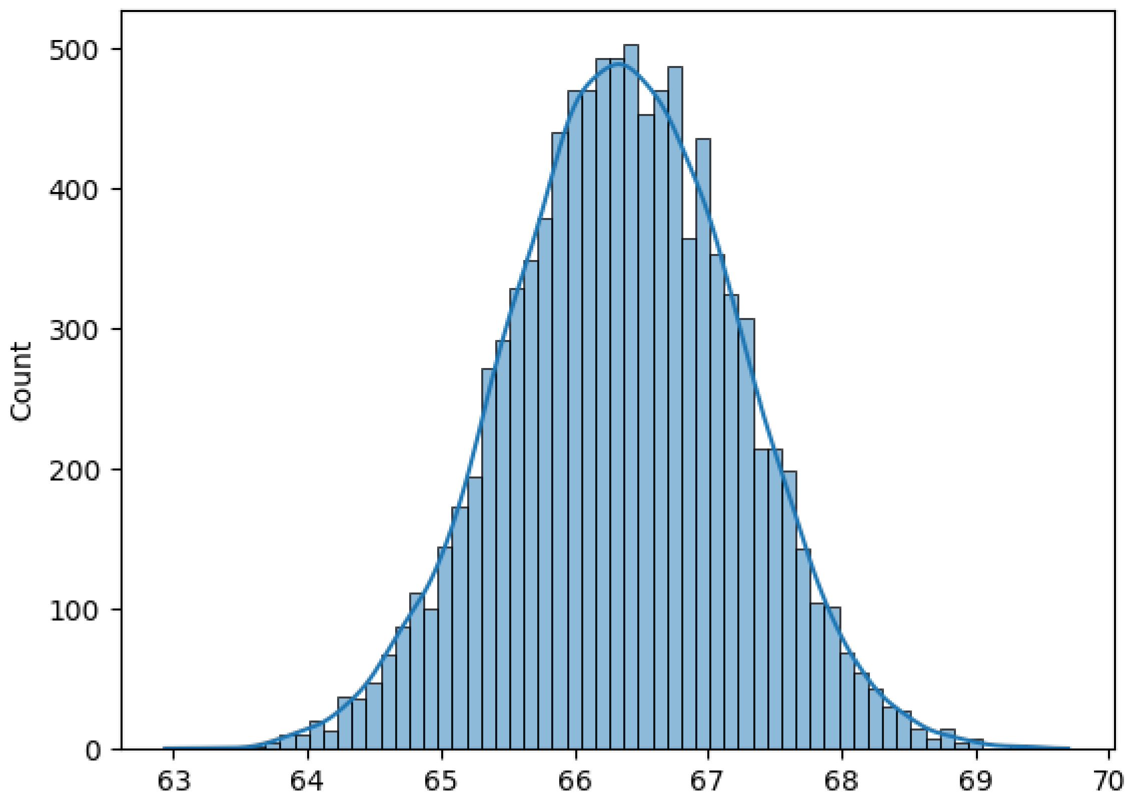
Let's try this out.

- Let's increase the sample size to 20.
- We will again perform 10,00 iterations and plot the distributions of the sample means

```
sample_20 = [np.mean(df_height.sample(20)) for i in range(10000) ]
```

```
sns.histplot(sample_20, kde=True)
```

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



```
np.mean(sample_20)
```

```
66.35931508817528
```

```
np.std(sample_20)
```

```
0.86147661644173
```

Observation

- We can clearly see that as **we increase the number of samples from 5 to 20, the sample means come closer to the actual mean and the standard deviation becomes less**.
- Previously the majority of the values were between 62 and 72. Now the spread of the data has decreased and values lie between 64 and 69

So we can assert that by increasing the size of the sample, the variability or SD of the sample distributions decreases and the sample mean tends to be much closer to the population mean.

Comparison of Statistics

Let's compare the statistics of population data and sample data to observe some patterns

```
# population mean
mu = df_height.mean()

# population SD
sigma = df_height.std()

# mean of sample distributions having sample size = 5
mu_5 = np.mean(sample_5)

# SD of sample distributions having sample size = 5
sigma_5 = np.std(sample_5)

# mean of sample distributions having sample size = 20
mu_20 = np.mean(sample_20)

# SD of sample distributions having sample size = 20
sigma_20 = np.std(sample_20)
```

```
print(mu, mu_5, mu_20)
print(sigma, sigma_5, sigma_20)
```

```
66.36755975482124 66.37463912527758 66.35931508817528
3.8475281207732293 1.7112978334301363 0.86147661644173
```

Observation

Here,

Population Statistics:

- μ = population mean
- σ = population standard deviation

Sample Statistics:

- μ_5 = mean of sample means (from samples of size 5)
- σ_5 = standard deviation of the sample means (from samples of size 5)
- μ_{20} = mean of sample means (from samples of size 20)
- σ_{20} = standard deviation of the sample means (from samples of size 20)

1. We can clearly observe that,
    - As we increase the sample size, the SD of sample means decreases.
    - The **SD of sampling distribution** ($\sigma_{\bar{x}}$) **is less than the population SD** (σ).

$$\sigma > \sigma_5 > \sigma_{20}$$

This aligns with the CLT, which states that the standard deviation of the sampling distribution ($\sigma_{\bar{x}}$) is the standard deviation of the population (σ) divided by the square root of the sample size.

- $\sigma_{\bar{x}} = \frac{\sigma}{\sqrt{n}}$

  We have already studied it, it is known as **Standard Error**. It indicates that how far my sample mean is from the actual mean.

By looking into the means we observe that,

2. the mean of the sampling distribution is equal to the mean of the population
   - $\mu_{\bar{x}} = \mu$

Summarizing CLT

> **Q1. How can we mathematically represent it?**

We can describe the sampling distribution of the mean using this notation:

$$\bar{X} \backsim N(\mu\ ,\ \frac{\sigma}{\sqrt{n}})$$

Where:

- $\bar{X}$ is the sampling distribution of the sample means
- $\backsim$ means “follows the distribution”
- N is the normal distribution
- μ is the mean of the population
- σ is the standard deviation of the population
- n is the sample size

Now, let's solve some examples

Example 1:

```
Systolic blood pressure of a group of people is known to have an average of 122 mmHg
and a standard deviation of 10 mmHg.
Calculate the probability that the average blood pressure of 16 people will be greater th
```

Given,
For the entire population

- μ = 122
- σ = 10

We need to calculate the probability that average BP of 16 people will be > 125

- Sample size n = 16
- and by CLT, the $\bar{X}$ = μ = 122
- The standard deviation will be

```
# SE = σ/sqrt(n)

sigma = 10/np.sqrt(16)
sigma
```

```
    2.5
```

So, the probability of finding people having greater than 125 average BP will be
we know,

$$P[X > 125] = 1 - P[X < 125]$$

> **How can we find P[X < 125]?**

by calculating "norm.cdf(zscore)"

```
# zscore = (X - mu)/σ

z_score = (125 - 122)/sigma

z_score
```

```
    1.2
```

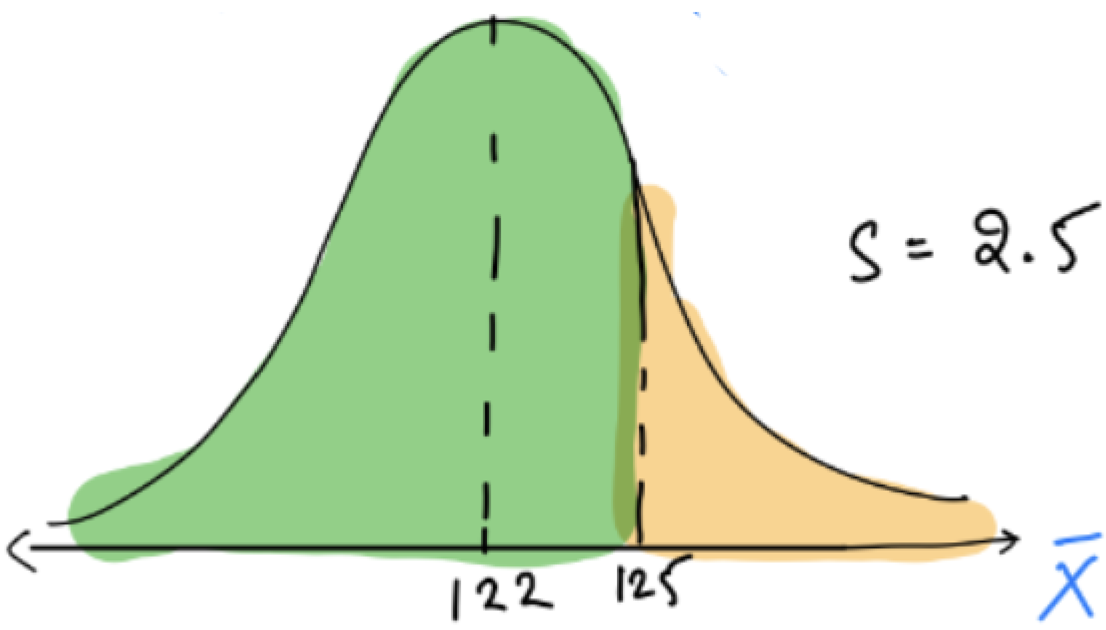
```
# P[X>125]=1-P[X<125]

probability = 1 - norm.cdf(z_score)

probability
```

```
    0.11506967022170822
```

The probability that the average blood pressure of 16 people will be greater than 125 mmHg is 0.115



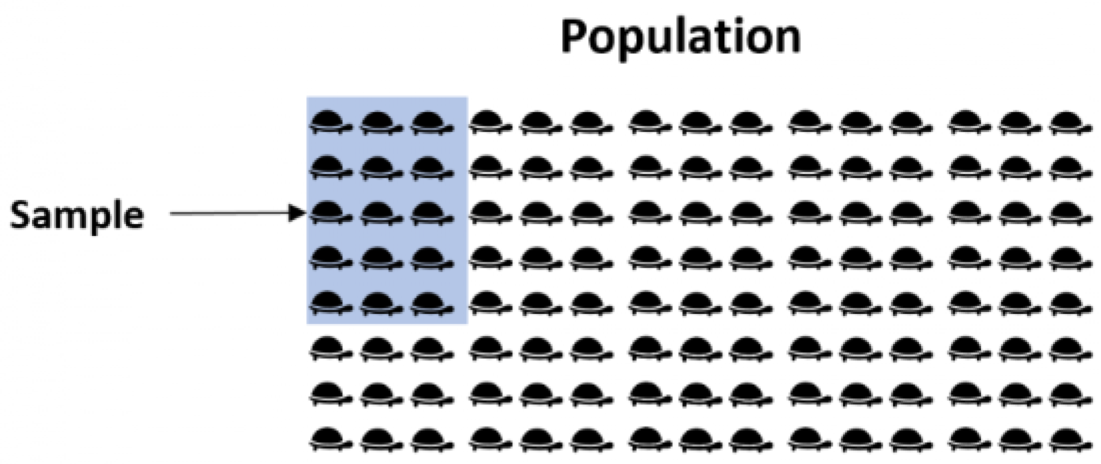
Now, let's jump into the next important concept in statistics, Confidence Interval.

Confidence Interval

We solved an example of turtles in our previous lectures to discuss point estimates. Let's recall it.

Example:

We wanted to estimate the mean weight of a certain species of turtle in Florida by taking a single random sample (S) of 50 turtles and using the sample mean to estimate the true population mean.



But as we know there is one problem here.

Problem:

- The mean weight of turtles in the sample is not guaranteed to exactly match the mean weight of turtles in the whole population.

Solution:

In order to capture this uncertainty, we will say that the population mean will lie in the **range of the sample mean (point estimates) +/— some errors here and there**.

So, whatever sample mean I will get from the sample, I will try to provide a range and the population mean will lie within that range.

This interval or range is called **Confidence Interval**.

In summary,

- A confidence interval **is the mean of your estimate plus and minus the variation in that estimate**.
    - This is the range of values you expect your estimate to fall within a certain level of confidence.

> **Q. What is confidence?**

- If you construct a confidence interval with a 95% confidence level, you are confident that 95 out of 100 times the estimate will fall between the upper and lower values specified by the confidence interval.
- x_1 **is the lower bound and** x_2 **is the upper bound**.

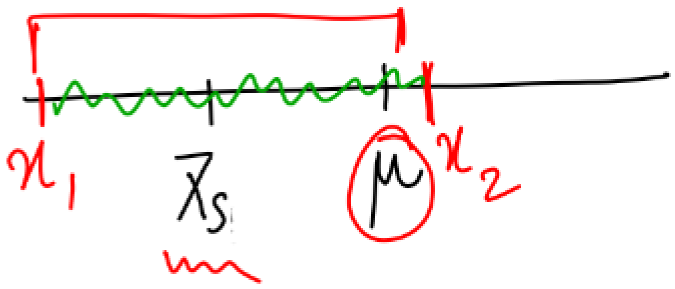
Q. How to calculate the confidence interval?

There are 2 ways to find confidence interval

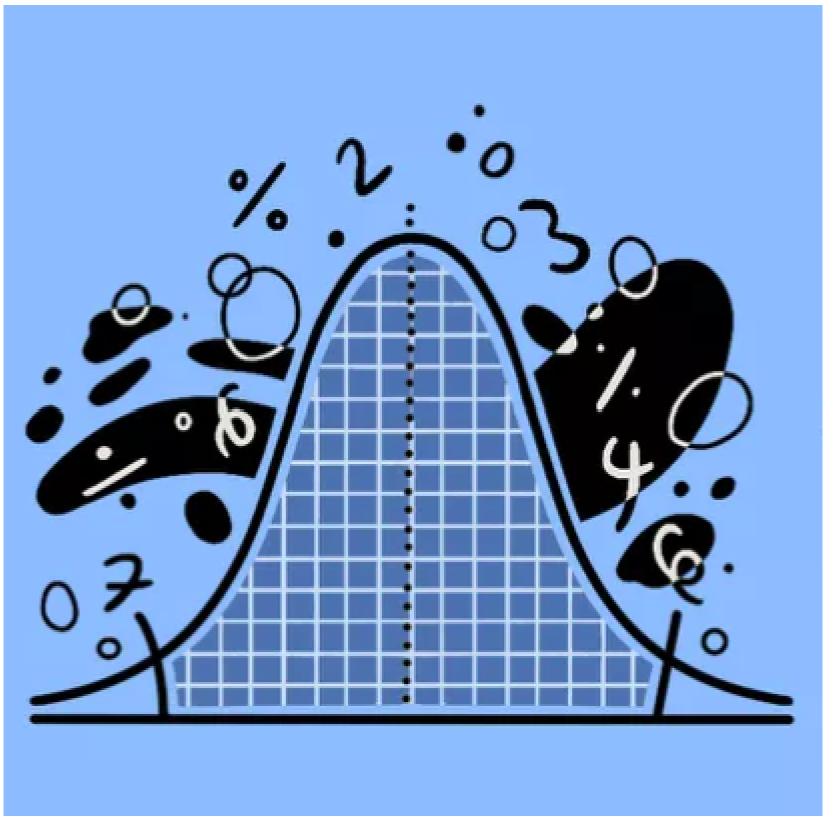
1. **Using CLT**
    - This is for mean values. If we have mean as a statistic then we will calculate CI using the central limit theorem (CLT)
2. **Using Bootstrapping**
    - If you have statistics other than mean like median then we are not allowed to use CLT. We can use bootstrap method to calculate confidence intervals in those scenarios.

Let's calculate the confidence interval using CLT first,

- Using CLT, we will get these values and range between these values will be our confidence interval.



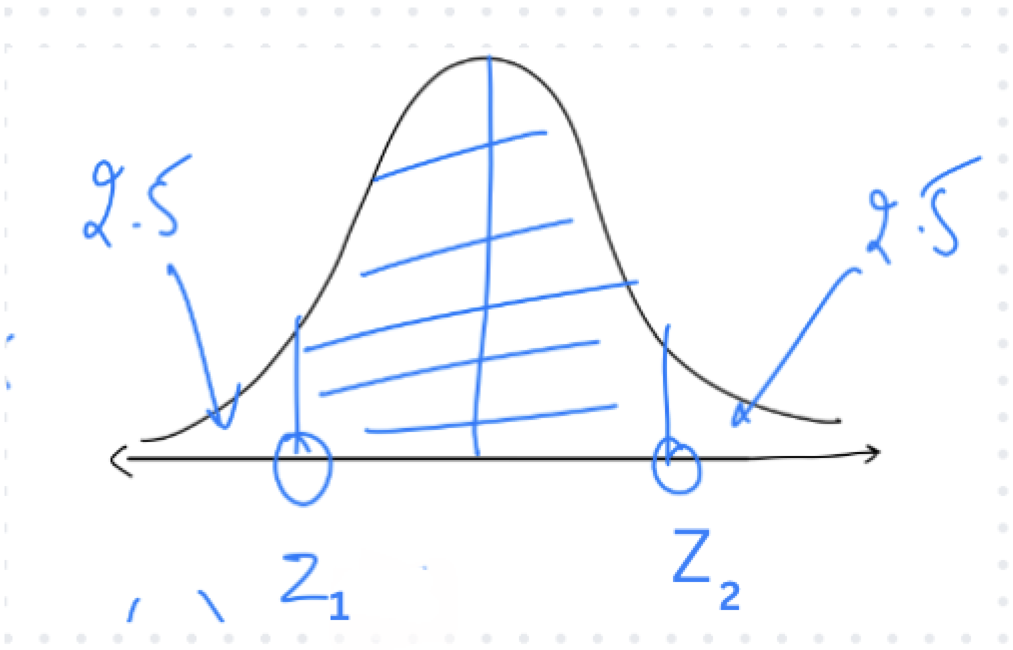
Here $\bar{X}_S$ is the mean of sample (S).



Compute a 95% Confidence Interval

To get a 95% confidence interval, we need to find the area that covers 95% of the data

- Let's say we have 2 points around mean one on the left and one on the right. These points will be upper bound and lower bound.
    - We can calculate the data points for these z scores (z1 and z2) which will be the confidence interval.
- We know that between z1 and z2, 95% of the population lies. So, on the **left hand side of z1 there is 2.5% population and on the right side of z2 there is 2.5% population**



So, to find the value below this percentage of data falls, we will use the PPF.

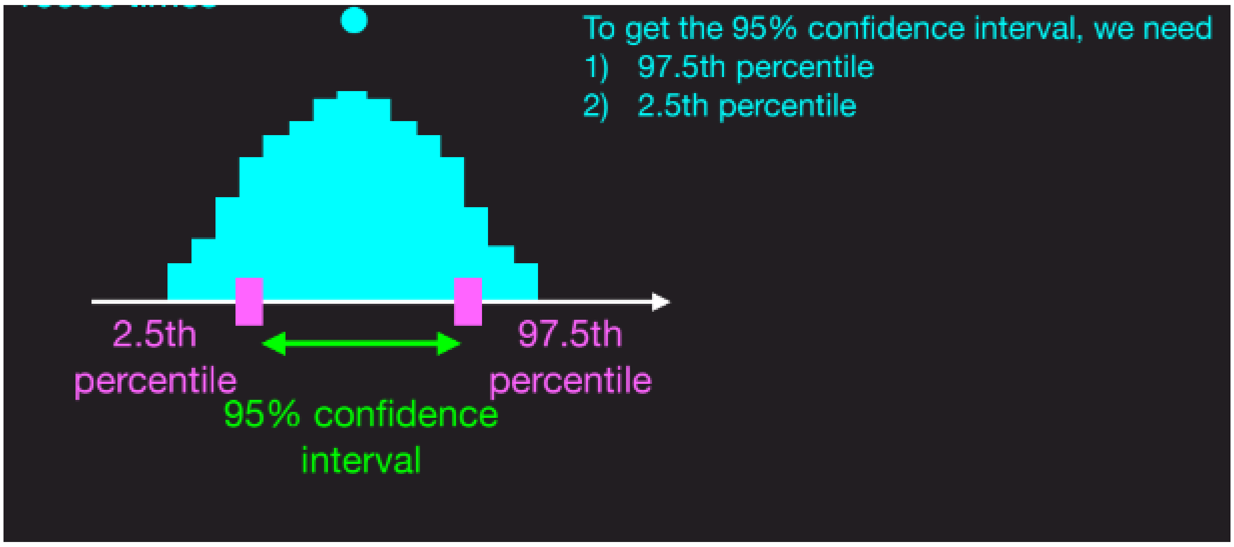
> **Q1. How do we find z1 and z2**

- $z1 = norm.ppf(0.025)$ **as we have 2.5% data till z1**

Similarly to calculate z2 will use

- $z2 = norm.ppf(1 - 0.025)$ **as we have 2.5% data remaining after z2**

We can also represent the z1 points as 2.5th percentile and z2 as 97.5th percentile (1-0.025)



```
# z1 will be
z1 = norm.ppf(0.025)
z1
```

```
-1.9599639845400545
```

```
# z2 will be
z2 = norm.ppf(1 - 0.025)  # we can also use norm.ppf(0.975)
z2
```

```
1.959963984540054
```

> Q. What is the formula for the z score?

$$Z = \frac{X - \mu}{\sigma}$$

So,

$$X = \mu + (Z * \sigma)$$

Where

- X is individual data point
- μ is population mean
- σ is population standard deviation

Note:

From this, we will get data points associated with the corresponding z score, which is Z in this case.

Here we are dealing with Samples so,

- We will use $\bar{X}$ which is sample mean.

- In the case of sample we consider standard error so $\sigma = \frac{\sigma}{\sqrt{n}}$

The Z-score is a measure of **how many standard deviations a data point (in this case, the sample mean) is from the population mean**

- We need to find how many standard deviation away the sample mean is from population mean

Z score will be:

$$Z = \frac{\bar{X} - \mu}{\frac{\sigma}{\sqrt{n}}}$$

Where,

- $\bar{X}$ = sample mean
- μ - population mean
- $\sigma/\sqrt{n}$ = standard error

In our sample S that we are looking above the **, our population mean will lie between z scores i.e. z1 and z2 which is -1.96 and 1.96**.

Between these two points, we will have our 95% confidence interval.

$$Z_1 < \frac{\bar{X}_s - \mu}{\frac{\sigma}{\sqrt{n}}} < Z_2 \text{ (equation 1)}$$

- $\bar{X}_s$ represents mean of sample S

Now, if we compare left side: $Z_1 < \frac{\bar{X}_s - \mu}{\frac{\sigma}{\sqrt{n}}}$

- We will get: $\bar{X}_s - Z_1 * (\frac{\sigma}{\sqrt{n}}) < \mu$ (equation 2)

Now, if we compare right side: $\frac{\bar{X}_s - \mu}{\frac{\sigma}{\sqrt{n}}} < Z_2$

- We will get: $\mu > \bar{X}_s + Z_2 * (\frac{\sigma}{\sqrt{n}})$ (equation 3)

Now, from equation 1, 2 and 3 if we want the μ value only,

- $\bar{X}_s - Z_1 * (\frac{\sigma}{\sqrt{n}}) < \mu < \bar{X}_s + Z_2 * (\frac{\sigma}{\sqrt{n}})$

> **Q1. So, what will be the range where original μ is lying?**

Confidence Interval = $\bar{X} \pm Z \left(\frac{\sigma}{\sqrt{n}} \right)$

OR

$[\bar{X} - Z * (\frac{\sigma}{\sqrt{n}})\ ,\ \bar{X} + Z * (\frac{\sigma}{\sqrt{n}})]$

where:

$\bar{X}$: sample mean

z: is the z-score corresponding to the desired confidence level.

σ: population standard deviation

n: sample size

If you take $(Z * \frac{\sigma}{\sqrt{n}})$, it is referred as the the **margin of error** in the context of confidence intervals.

Now let's try to solve one example

Example on confidence interval

```
The mean height of a sample of 100 adults was found to be 65 inches, with a standard devi
 Compute 95% confidence interval
```

Solution:

Given,

- Sample size "n" = 100
- Sample mean = 65
- Standard deviation = 2.5

First, let's calculate the standard error:

```
# std deviation  sigma/sqrt(n)
std_error = 2.5/np.sqrt(100)
std_error
```

```
0.25
```

Now the values of Z for 95% confidence will be,

we have calculated above

```
# z1 will be
z1 = norm.ppf(0.025)
z1
```

```
    -1.9599639845400545
```

```
# z2 will be
z2 = norm.ppf(1 - 0.025)  # we can also use norm.ppf(0.975)
z2
```

```
    1.959963984540054
```

Now, How to get the data points for z1 which is on left side and z2 which is on right side,

$X_1 = \mu + Z1 * \sigma$ and,

$X_2 = \mu + Z2 * \sigma$

```
x1 = 65 + z1 * std_error
x1
```

```
    64.51000900386498
```

```
x2 = 65 + z2 * std_error
x2
```

```
    65.48999099613502
```

```
So the range of 95% confidence interval --> [64.51, 65.48]
```

Conclusion:

We can claim that the population mean will lie between the value 64.51 and 65.48 with 95% confidence.

There is directly one function available which will calculate interval by using just a single formula

- Using **norm.interval()**

You have to pass three attributes :

- **norm.interval(confidence, loc=0, scale=1)**
  - confidence: how much confidence you want
  - loc: pass the **mean** value here (by default it is 0)
  - scale: pass the **standard error** here (by default it is 1)

```
norm.interval(0.95, loc=65, scale=std_error)
```

```
(64.51000900386498, 65.48999099613502)
```

Confidence interval using Bootstrap

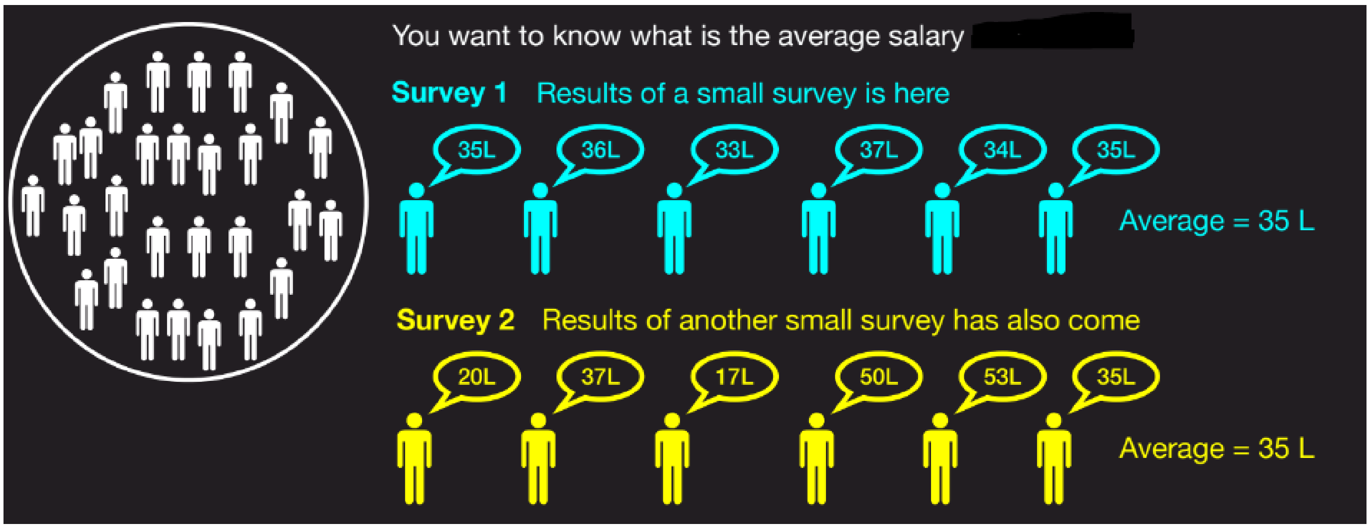
- Suppose you have very little data and you want to compute a confidence interval for some other statistics like median then the most common technique is bootstrapping.

Let's start with one example:

Example: Salary Survey

```
Imagine we want to analyse and learn about the data scientist salaries at Google.
```

We have 2 surveys,


You want to know what is the average salary [redacted]
Survey 1 Results of a small survey is here
35L
36L
33L
37L
34L
35L
Average = 35 L
Survey 2 Results of another small survey has also come
20L
37L
17L
50L
53L
35L
Average = 35 L

We don't have a population mean here but we have 2 samples, so, we can calculate the sample means.

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

```
35.0
```

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

```
35.0
```

> **Q1. Which of the two surveys is better for estimating the population parameter or which survey is more reliable?**

By observing the samples we can conclude that values in survey 1 is much closer to the mean values so survey 1 will be more accurate for estimation

> **Q2. Now, can we simulate more and more sets of samples like the ones above?**

For this statisticians come up with something which has reasonable amount of accuracy.

Sample With Replacement

- Statistician suggested that take your survey and then create more samples from the same survey only using **replacement**.
- Bootstrapping is a statistical procedure that resamples a single dataset to create many simulated samples.

```
n = 6
bootstrapped_samples = np.random.choice(survey_1, size=n)
bootstrapped_samples
```

```
    array([35, 35, 34, 34, 35, 35])
```

Here we will get an array of length 6 where each element is one of the original data points from survey 1 which is randomly chosen.

Every time we run this code, we will get a different array so we can observe that the mean of this newly constructed array will also be different.

```
np.mean(bootstrapped_samples)
```

```
    34.666666666666664
```

```
bootstrapped_samples = np.random.choice(survey_2, size=n)
np.mean(bootstrapped_samples)
```

```
    31.0
```

Let's observe the difference between survey_1 and survey_2 by running the code several times.

- We can observe that in survey 1, the mean value is always close to 35 But in survey 2, it sometimes comes to 35, sometimes 40, sometimes 39 so there is more variance in survey 2.

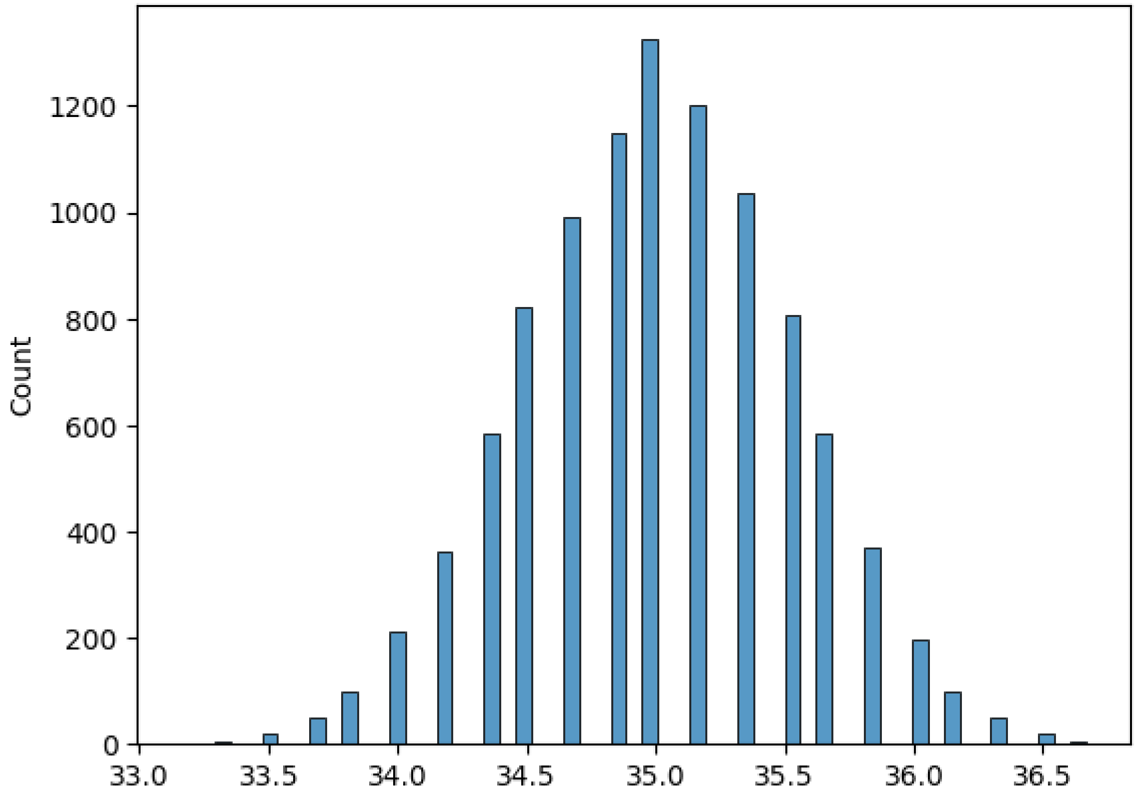
So we will go with survey_1 as it has the higher confidence because the variance in survey_1 is less.

Let's draw a histogram of this survey

```
bootstrapped_means_survey_1 = []
for reps in range(10000):
    bootstrapped_samples = np.random.choice(survey_1, size=n)
    bootstrapped_mean = np.mean(bootstrapped_samples)
    bootstrapped_means_survey_1.append(bootstrapped_mean)
```

```
sns.histplot(bootstrapped_means_survey_1)
```

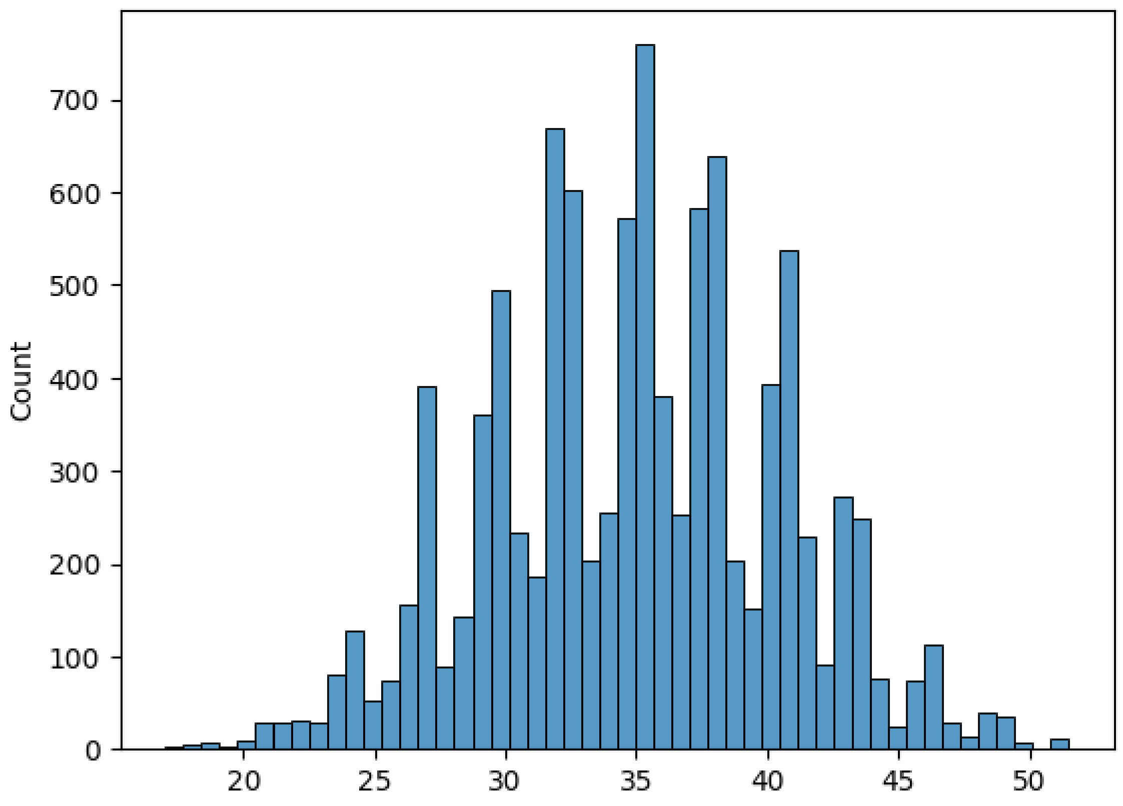
```
<Axes: ylabel='Count'>
```



```
bootstrapped_means_survey_2 = []
for reps in range(10000):
    bootstrapped_samples = np.random.choice(survey_2, size=n)
    bootstrapped_mean = np.mean(bootstrapped_samples) # Replace by any statistic
    bootstrapped_means_survey_2.append(bootstrapped_mean)
```

```
sns.histplot(bootstrapped_means_survey_2)
```

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



> **Let's compare these two histograms, what can we observe?**

- **We can observe that in survey_2, the interval or range is somewhere between 20-50**

  While in survey_1, it is between 33-36 which is very close to the actual mean

So we can conclude that survey 1 is more accurate than survey 2

How to compute the condidence interval?

We can calculate the percentile of **bootstrapped mean**

- 2.5th percentile will give me lower bound (x1)
- 97.5th percentile will give me upper bound (x2)

Then, confidence inteval will be [x1, x2]

```
len(bootstrapped_means_survey_1)
```

```
10000
```

```
x1 = np.percentile(bootstrapped_means_survey_1, 2.5)
x1
```

```
    34.0
```

```
x2 = np.percentile(bootstrapped_means_survey_1, 97.5)
x2
```

```
    36.0
```

We can observe that 95% of the numbers lies between 34 & 36 so
Confidence Interval: $(x1, x2)$

AS this process is random, this will be slight change in CI everytime

```
len(bootstrapped_means_survey_2)
```

```
    10000
```

```
x1 = np.percentile(bootstrapped_means_survey_2, 2.5)
x1
```

```
    24.0
```

```
x2 = np.percentile(bootstrapped_means_survey_2, 97.5)
x2
```

```
    45.83749999999994
```

Here also CI will be
Confidence Interval = $(x1, x2)$, this is how we can calculare Confidence Interval using