4.0 Statistical Inference

4.1 Sample & Population

4.2 Standard error

4.3 Estimator & estimate

4.4 Null and alternative hypothesis

4.5 Steps in hypothesis testing

4.6 Type-I and Type-II error

4.7 Specificity and sensitivity

4.8 P-value

4.9 One sample & two sample ztest

Statistical Inference:

```
Statistical inference is the process of drawing conclusions about a
population based on a sample of data. It involves making educated
guesses or estimates about population parameters, such as means,
proportions, or variances, using information obtained from a sample.
```

4.1 Sample & Population :

```
[i]Population: The entire group or set of individuals, items, or data
points that are the subject of interest in a statistical study. It's
often denoted as "N."
[ii]Sample: A subset of the population that is selected for the
purpose of the study. It's typically denoted as "n."
```

"In Python, you can create a random sample from a population using libraries like NumPy. Here's an example: "

In [11]:
```
import numpy as np

# Define a population (for demonstration purposes)
population = np.array([10, 15, 20, 25, 30, 35, 40, 45, 50])
print('population = ',population)
```

```
# Randomly select a sample from the population
sample_size = 3
print('sample_size = ',sample_size)
sample = np.random.choice(population, sample_size, replace=False)
print('sample = ',sample)
```

```
population =  [10 15 20 25 30 35 40 45 50]
sample_size =  3
sample =  [30 10 40]
```

Explanation:

We import NumPy and create a population array. We use np.random.choice to select a random sample of size sample_size from the population without replacement.

4.2 Standard Error

Standard error measures the variability of a sample statistic (e.g., sample mean) across different random samples from the same population. It quantifies the uncertainty associated with estimating a population parameter.
Formula for standard error (SE): SE = (σ / √n), where σ is the population standard deviation.

'To calculate the standard error, you need the sample data and the population standard deviation. Here's an example:'

In [26]:
```
import numpy as np

# Sample data
data = np.array([21, 25, 29, 32, 35])
print('data = ',data)
# Calculate sample size and sample mean
sample_size = len(data)
print('sample_size = ',sample_size)
sample_mean = np.mean(data)
print('sample_mean = ',sample_mean)

# Assume we know the population standard deviation (for demonstration)
population_stddev = 5
print('population_stddev = ',population_stddev)
# Calculate standard error
standard_error = population_stddev / np.sqrt(sample_size)
print('standard_error = ',standard_error)

"""

Explanation:
    We calculate the sample mean and size using NumPy functions.We assume knowledge of
    population standard deviation (which is typically unknown in practice) and use it
    compute the standard error."""
print(__doc__)
```

```
data =  [21 25 29 32 35]
sample_size =  5
sample_mean =  28.4
population_stddev =  5
standard_error =  2.23606797749979


Explanation:
    We calculate the sample mean and size using NumPy functions.We assume knowledge o
f the
    population standard deviation (which is typically unknown in practice) and use it
to
    compute the standard error.
```

4.3 Estimator & Estimate :

In statistical inference, an estimator is a statistic used to estimate a population parameter, and an estimate is the specific value obtained when using that estimator with sample data.

[i]Estimator: A statistic (e.g., sample mean, sample proportion) that is used to make an inference about an unknown population parameter.
[ii]Estimate: The specific numerical value obtained from the sample data when using an estimator. For example, the sample mean is an estimator, and its value from a particular sample is the estimate.

Let's illustrate this with a simple example using the sample mean as an estimator:

In [28]:
```python
import numpy as np

# Sample data
data = np.array([10, 15, 20, 25, 30])
print('data = ',data)
# Calculate the sample mean (estimator)
sample_mean = np.mean(data)

print("Sample Mean (Estimate):", sample_mean)
```

```
data =  [10 15 20 25 30]
Sample Mean (Estimate): 20.0
```

4.4 Null and Alternative Hypothesis:

Null Hypothesis (H0):

A statement that there is no significant effect or difference in the population. It represents the status quo or no change.

Alternative Hypothesis (H1 or Ha):

A statement that contradicts the null hypothesis, suggesting a significant effect or difference in the population.

Hypotheses are statements used in hypothesis testing to make claims about a population. Here's an example of defining null and alternative hypotheses:

In [31]:
```
# Define hypotheses
null_hypothesis = "The population mean is equal to 100."
alternative_hypothesis = "The population mean is not equal to 100."

print("Null Hypothesis:", null_hypothesis)
print("Alternative Hypothesis:", alternative_hypothesis)

"""
Explanation:

In this code, we define null and alternative hypotheses. The null hypothesis typically
no effect or no difference, while the alternative hypothesis contradicts the null hypo
print(__doc__)
```

```
Null Hypothesis: The population mean is equal to 100.
Alternative Hypothesis: The population mean is not equal to 100.

Explanation:

In this code, we define null and alternative hypotheses. The null hypothesis typicall
y represents
no effect or no difference, while the alternative hypothesis contradicts the null hyp
othesis.
```

4.5 Steps in Hypothesis Testing :

```
Formulate hypotheses (H0 and H1).
[i]  Collect and analyze data.
[ii] Calculate a test statistic (e.g., t-statistic, z-statistic).
[iii]Determine the significance level (alpha, usually set at 0.05).
[iv] Compare the test statistic to a critical value or calculate a p-
value.
[v]  Make a decision: Reject H0 if p-value ≤ alpha; otherwise, fail
to reject H0.
```

In [42]:
```
import numpy as np
from scipy import stats

# Step 1: Formulate hypotheses (H0 and H1)
# Null Hypothesis (H0): The population mean is 100.
# Alternative Hypothesis (H1): The population mean is not 100.

# Step 2: Collect and analyze data
"""
In this step, you gather data through experiments, surveys, observations, or any other
appropriate method. The data should be collected and prepared carefully to ensure its
and relevance.
"""
# Sample data (assuming it's normally distributed)
data = np.array([98.6, 98.7, 98.5, 98.8, 98.7, 98.9, 98.6, 98.7, 98.5, 98.6])
print('data = ',data)
```

```
# Step 3: Calculate a test statistic (t-statistic) and p-value
"""
Depending on your research question and the type of data you have, you choose an appro
statistical test (e.g., t-test, z-test, chi-square test) to calculate a test statistic
This statistic quantifies the difference or effect observed in your sample data.
"""

null_mean = 100  # Hypothesized population mean (H0)
print('null_mean = ',null_mean)
t_statistic, p_value = stats.ttest_1samp(data, null_mean)
print('t_statistic = ',t_statistic)
print('p_value = ',p_value)


# Step 4: Determine the significance level (alpha)
"""
The significance level (α) is a predetermined threshold that represents the probabilit
a Type-I error (incorrectly rejecting a true null hypothesis). Commonly used values fo
0.05 (5%) or 0.01 (1%).
"""

alpha = 0.05  # Significance level

# Step 5: Compare the test statistic to a critical value or calculate a p-value
# In this case, we calculate the p-value.
"""
In this step, you either compare the calculated test statistic to a critical value fro
statistical table (for traditional hypothesis testing) or calculate a p-value
(for modern hypothesis testing). The critical value represents the cutoff point beyond
you would reject the null hypothesis. The p-value represents the probability of obtair
observed result (or more extreme results) under the assumption that the null hypothesi
"""
print("Step 3: Calculate Test Statistic and P-Value")
print("T-Statistic:", t_statistic)
print("P-Value:", p_value)

# Step 6: Make a decision
"""
Reject H0 if p-value ≤ alpha; otherwise, fail to reject H0:

If the p-value obtained in step 5 is less than or equal to the chosen significance lev
you reject the null hypothesis. This means you have evidence to support the alternativ
If the p-value is greater than α, you fail to reject the null hypothesis. This means t
not enough evidence to support the alternative hypothesis.

"""
if p_value < alpha:
    decision = "Reject the null hypothesis"
else:
    decision = "Fail to reject the null hypothesis"

print("\nStep 6: Make a Decision")
print(decision)
"""
Explanation:

In this code, we follow the steps you provided for hypothesis testing.
We use the scipy.stats.ttest_1samp function to calculate the t-statistic
and p-value for a one-sample t-test comparing the sample data to the null
```

```
hypothesis mean (100).We set a significance level (alpha) of 0.05, which is
a common choice. Based on the p-value and alpha, we make a decision to either
reject or fail to reject the null hypothesis.
"""
print(__doc__)
```

```
data =  [98.6 98.7 98.5 98.8 98.7 98.9 98.6 98.7 98.5 98.6]
null_mean =  100
t_statistic =  -33.499999999999545
p_value =  9.274179562846231e-11
Step 3: Calculate Test Statistic and P-Value
T-Statistic: -33.499999999999545
P-Value: 9.274179562846231e-11

Step 6: Make a Decision
Reject the null hypothesis

Explanation:

In this code, we follow the steps you provided for hypothesis testing.
We use the scipy.stats.ttest_1samp function to calculate the t-statistic
and p-value for a one-sample t-test comparing the sample data to the null
hypothesis mean (100).We set a significance level (alpha) of 0.05, which is
a common choice. Based on the p-value and alpha, we make a decision to either
reject or fail to reject the null hypothesis.
```

4.6 Type-I and Type-II Error :

1.Type-I Error (False Positive): This error occurs when you incorrectly reject the null hypothesis when it is actually true.

2.Type-II Error (False Negative): This error occurs when you incorrectly fail to reject the null hypothesis when it is actually false.

We'll use a one-sample z-test as an example and create scenarios to illustrate both types of errors.

In [44]:
```
import numpy as np
import statsmodels.api as sm
from scipy import stats

# Population parameters (for demonstration purposes)
population_mean = 100
population_stddev = 10

# Sample data (assuming it's normally distributed)
sample_size = 30
sample_data = np.random.normal(population_mean, population_stddev, sample_size)

# Hypotheses:
# Null Hypothesis (H0): The population mean is 100.
# Alternative Hypothesis (H1): The population mean is not 100 (two-tailed test).

# Step 3: Calculate a test statistic (z-statistic) and p-value
null_mean = 100  # Hypothesized population mean (H0)
```

```
z_statistic, p_value = sm.stats.ztest(sample_data, value=null_mean)

# Step 6: Make a decision
alpha = 0.05  # Significance level

print("Step 3: Calculate Test Statistic and P-Value")
print("Z-Statistic:", z_statistic)
print("P-Value:", p_value)

if p_value < alpha:
    decision = "Reject the null hypothesis"
else:
    decision = "Fail to reject the null hypothesis"

print("\nStep 6: Make a Decision")
print(decision)

# Simulate Type-I Error (Rejecting a true null hypothesis)
if decision == "Reject the null hypothesis":
    print("\nSimulating Type-I Error (False Positive)")
    random_data = np.random.normal(population_mean, population_stddev, sample_size)
    _, p_value_type1 = sm.stats.ztest(random_data, value=null_mean)
    if p_value_type1 < alpha:
        print("Type-I Error Occurred: Null Hypothesis Rejected Incorrectly")
    else:
        print("No Type-I Error")

# Simulate Type-II Error (Failing to reject a false null hypothesis)
print("\nSimulating Type-II Error (False Negative)")
true_mean = 110  # True population mean (different from null hypothesis)
sample_data_type2 = np.random.normal(true_mean, population_stddev, sample_size)
_, p_value_type2 = sm.stats.ztest(sample_data_type2, value=null_mean)
if p_value_type2 < alpha:
    print("No Type-II Error")
else:
    print("Type-II Error Occurred: Null Hypothesis Not Rejected Incorrectly")
```

```
Step 3: Calculate Test Statistic and P-Value
Z-Statistic: 0.33653421340768963
P-Value: 0.7364680559629394

Step 6: Make a Decision
Fail to reject the null hypothesis

Simulating Type-II Error (False Negative)
No Type-II Error
```

Explanation: *We first generate a sample dataset and define null and alternative hypotheses as usual. *We calculate the z-statistic and p-value for the sample data.Based on the p-value and significance level (alpha), we make a decision to either reject or fail to reject the null hypothesis. *To simulate a Type-I Error,we generate random data from the same population distribution and check if the null hypothesis is rejected incorrectly. *To simulate a Type-II Error, we generate a new sample with a population mean different from the null hypothesis and check if the null hypothesis is not rejected incorrectly.

4.7 Specificity and Sensitivity

Specificity:

The proportion of true negatives among all the actual negatives. It measures the ability of a test to correctly identify negative cases.

Sensitivity:

The proportion of true positives among all the actual positives. It measures the ability of a test to correctly identify positive cases.

In [45]:
```
from sklearn.metrics import confusion_matrix

# True labels (actual outcomes)
true_labels = [1, 0, 1, 0, 1, 0, 1, 0, 1, 0]

# Predicted labels (model's predictions)
predicted_labels = [1, 1, 1, 0, 1, 0, 0, 0, 1, 0]

# Calculate confusion matrix
conf_matrix = confusion_matrix(true_labels, predicted_labels)

# Extract values from confusion matrix
tn, fp, fn, tp = conf_matrix.ravel()

# Calculate Specificity and Sensitivity
specificity = tn / (tn + fp)
sensitivity = tp / (tp + fn)

print("Specificity (True Negative Rate):", specificity)
print("Sensitivity (True Positive Rate):", sensitivity)
```

```
Specificity (True Negative Rate): 0.8
Sensitivity (True Positive Rate): 0.8
```

Explanation: *We start by importing the confusion_matrix function from the sklearn.metrics module. *We define the true labels and predicted labels. In this example, 1 represents a positive outcome, and 0 represents a negative outcome. *We calculate the confusion matrix using confusion_matrix, which gives us the counts of true negatives (tn), false positives (fp), false negatives (fn), and true positives (tp). *We use these values to calculate specificity (the proportion of true negatives among actual negatives) and sensitivity (the proportion of true positives among actual positives).

P-Value:

The p-value is the probability of obtaining a test statistic as extreme as or more extreme than the one calculated from the sample data, assuming the null hypothesis is true. A smaller p-value provides stronger evidence against the null hypothesis.

To calculate and interpret the p-value in hypothesis testing using Python, you can use statistical libraries such as SciPy. Let's illustrate how to calculate the p-value for a one-sample t-test as an example:

In [49]:
```
import numpy as np
from scipy import stats

# Sample data
data = np.array([68, 72, 75, 70, 74, 73, 72])
```

```
print('data = ',data)
# Null hypothesis: Population mean is 70
null_mean = 70
print('null_mean = ',null_mean)
# Perform a one-sample t-test
t_statistic, p_value = stats.ttest_1samp(data, null_mean)

print("T-Statistic:", t_statistic)
print("P-Value:", p_value)
```

```
data =  [68 72 75 70 74 73 72]
null_mean =  70
T-Statistic: 2.2228757209048453
P-Value: 0.06792854992730868
```

Explanation: * We import NumPy and SciPy's stats module. * We define the sample data, which is the dataset you want to test against the null hypothesis. * We set the null hypothesis mean (in this case, 70). * We use the ttest_1samp function to perform a one-sample t-test, comparing the sample data to the null hypothesis mean. * The p_value obtained represents the probability of observing the sample data (or more extreme results) if the null hypothesis is true. ##### Interpretation: * If the p-value is less than your chosen significance level (alpha), typically 0.05, you would reject the null hypothesis. This suggests that there is enough evidence to support the alternative hypothesis. * If the p-value is greater than alpha, you would fail to reject the null hypothesis, indicating that there is not enough evidence to support the alternative hypothesis. The p-value helps you determine the strength of evidence against the null hypothesis. A smaller p-value provides stronger evidence against the null hypothesis, indicating that your results are less likely to be due to random chance.

4.9 One sample & two sample ztest :

One-sample and two-sample z-tests are statistical tests used to compare a sample mean (or proportion) to a known population mean (or proportion) or to compare the means (or proportions) of two independent samples when the population standard deviation is known. Here's how you can perform both tests using Python:

One-Sample Z-Test:

In a one-sample z-test, you compare the sample mean to a known population mean. The formula for the one-sample z-test is:

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

Where:

- $\bar{x}$ is the sample mean.
- μ is the known population mean.
- σ is the population standard deviation.
- n is the sample size.

Here's an example of a one-sample z-test using Python:

In [51]:
```
import numpy as np
import statsmodels.api as sm

# Sample data
sample_data = np.array([68, 72, 75, 70, 74, 73, 72])

# Population parameters (for demonstration purposes)
population_mean = 70
population_stddev = 5

# Perform one-sample z-test
z_statistic, p_value = sm.stats.ztest(sample_data, value=population_mean)

print("Z-Statistic:", z_statistic)
print("P-Value:", p_value)
```

```
Z-Statistic: 2.2228757209048453
P-Value: 0.026224181348738
```