Statistics Advanced - 2| Assignment

Question 1: What is hypothesis testing in statistics?

Hypothesis testing in statistics is a method used to make decisions or inferences about a population based on sample data. It involves testing an assumption (called a hypothesis) about a population parameter, such as the mean or proportion, to determine whether there is enough evidence to accept or reject that assumption.

Key Points:

1. Null Hypothesis (H_0): The default assumption or statement to be tested (e.g., "The average sales is 250").
2. Alternative Hypothesis (H_1 or Ha): The statement that contradicts the null hypothesis (e.g., "The average sales is not 250").
3. Test Statistic: A value calculated from the sample data used to make the decision.
4. P-value: The probability of obtaining the observed results if the null hypothesis is true.
5. Decision: If the p-value is less than a chosen significance level (α, usually 0.05), we reject the null hypothesis; otherwise, we fail to reject it.

Example: Suppose a company claims its average daily sales are 250 units. Using a sample of sales data, hypothesis testing can help determine whether this claim is statistically supported.

Question 2: What is the null hypothesis, and how does it differ from the alternative hypothesis?

In hypothesis testing, the null hypothesis (H_0) and the alternative hypothesis (H_1 or Ha) are two opposing statements about a population parameter.

1. Null Hypothesis (H_0):

    It is the default assumption or the statement to be tested.

    It usually represents no effect, no difference, or status quo.

    Example: "The average daily sales of the store is 250 units."

2. Alternative Hypothesis (H_1 or Ha):

   It is the statement that contradicts the null hypothesis.

   It represents the effect, difference, or change the researcher wants to test.

   Example: "The average daily sales of the store is not 250 units."

Difference:

H_0 assumes nothing has changed and serves as a baseline.

H_1 represents the claim or effect we are trying to provide evidence for.

Question 3: Explain the significance level in hypothesis testing and its role in deciding the outcome of a test.

The significance level in hypothesis testing, denoted by α, is the probability of rejecting the null hypothesis (H_0) when it is actually true. It represents the tolerance for making a Type I error (false positive).

Key Points:

1. Common significance levels are 0.05 (5%), 0.01 (1%), or 0.10 (10%).
2. It sets the threshold for decision-making:

   If the p-value $\leq \alpha$, we reject H_0.

   If the p-value $> \alpha$, we fail to reject H_0.

3. A lower significance level (e.g., 0.01) makes it harder to reject H_0, reducing the chance of a false positive but increasing the chance of a Type II error (false negative).

Role in deciding the outcome:
The significance level determines the cutoff point for deciding whether the observed sample data provides strong enough evidence against the null hypothesis. For example, with $\alpha = 0.05$, there is a 5% risk of concluding that an effect exists when it actually does not.

In short, α acts as a benchmark to control the reliability of conclusions drawn from hypothesis testing.

Question 4: What are Type I and Type II errors? Give examples of each.

In hypothesis testing, errors can occur because decisions are based on sample data rather than the entire population. These are classified as Type I and Type II errors.

Type I Error (False Positive):

Occurs when the null hypothesis (H_0) is true, but we reject it.

The probability of making a Type I error is the significance level (α).

Example: A company claims its average daily sales are 250 units. If we conclude that the average is not 250 when it actually is, that is a Type I error.

Type II Error (False Negative):

Occurs when the null hypothesis (H_0) is false, but we fail to reject it.

The probability of making a Type II error is denoted by β.

Example: Using the same sales data, if the actual average daily sales are 260 units but we conclude that the average is 250 units (fail to reject H_0), that is a Type II error.

Summary:

Type I: Reject H_0 when it is true → false alarm.

Type II: Fail to reject H_0 when it is false → missed detection.

These errors highlight the uncertainty inherent in statistical decisions and are considered when choosing sample size and significance levels.

Question 5: What is the difference between a Z-test and a T-test? Explain when to use each.

A Z-test and a T-test are both statistical tests used to compare sample data to population parameters or to compare two samples, but they differ based on sample size and knowledge of population standard deviation.

Z-test:

1. Used when the population standard deviation (σ) is known.
2. Suitable for large sample sizes (usually $n > 30$).
3. Assumes that the data is approximately normally distributed.

4. Test statistic formula:
   [
   Z = \frac{\bar{X} - \mu}{\sigma / \sqrt{n}}
   ]
5. Example: Checking whether the average height of students differs from a known population mean when the population standard deviation is known.

T-test:

1. Used when the population standard deviation is unknown and the sample standard deviation (s) is used.
2. Suitable for small sample sizes (usually $n \leq 30$).
3. Assumes that the data is approximately normally distributed.
4. Test statistic formula:
   [
   t = \frac{\bar{X} - \mu}{s / \sqrt{n}}
   ]
5. Example: Testing whether the average daily sales of a store differ from a hypothesized value when the population standard deviation is unknown and the sample size is small.

Summary:

- Z-test: Known σ, large sample → precise.
- T-test: Unknown σ, small sample → uses sample standard deviation.

Question 6: Write a Python program to generate a binomial distribution with n=10 and p=0.5, then plot its histogram. (Include your Python code and output in the code box below.) Hint: Generate random number using random function.

Here's a Python program to generate a binomial distribution with (n = 10) trials and probability of success (p = 0.5), and then plot its histogram using NumPy and Matplotlib:

```
# Import necessary libraries

import numpy as np

import matplotlib.pyplot as plt
```

```
# Parameters for binomial distribution
n = 10       # Number of trials
p = 0.5      # Probability of success
size = 1000   # Number of random samples

# Generate binomial random numbers
binomial_data = np.random.binomial(n, p, size)

# Print first 10 values (optional)
print("First 10 generated values:", binomial_data[:10])

# Plot histogram
plt.hist(binomial_data, bins=n+1, color='lightgreen', edgecolor='black', rwidth=0.8)
plt.title("Histogram of Binomial Distribution (n=10, p=0.5)")
plt.xlabel("Number of Successes")
plt.ylabel("Frequency")
plt.xticks(range(n+1))
plt.show()
```

Explanation:

- np.random.binomial(n, p, size) generates random numbers following a binomial distribution.
- bins=n+1 ensures each possible number of successes (0 to n) has its own bin in the histogram.
- plt.xticks(range(n+1)) labels the x-axis with the discrete number of successes.

When you run this code, you'll see a histogram showing the frequencies of 0–10 successes across 1000 trials, with a symmetric distribution centered around 5 (since (n \times p = 5)).

Question 7: Implement hypothesis testing using Z-statistics for a sample dataset in Python. Show the Python code and interpret the results. sample_data = [49.1, 50.2, 51.0, 48.7, 50.5, 49.8, 50.3, 50.7, 50.2, 49.6, 50.1, 49.9, 50.8, 50.4, 48.9, 50.6, 50.0, 49.7, 50.2, 49.5, 50.1, 50.3, 50.4, 50.5, 50.0, 50.7, 49.3, 49.8, 50.2, 50.9, 50.3, 50.4, 50.0, 49.7, 50.5, 49.9]

Here's how you can perform hypothesis testing using Z-statistics in Python for the given sample data:

Problem Setup:

Null Hypothesis (H_0): The population mean $\mu = 50$

Alternative Hypothesis (H_1): The population mean $\mu \neq 50$ (two-tailed test)

Significance Level (α): 0.05

Python Code:

```
import numpy as np

from scipy import stats


# Sample data

sample_data = [49.1, 50.2, 51.0, 48.7, 50.5, 49.8, 50.3, 50.7, 50.2, 49.6,

        50.1, 49.9, 50.8, 50.4, 48.9, 50.6, 50.0, 49.7, 50.2, 49.5,

        50.1, 50.3, 50.4, 50.5, 50.0, 50.7, 49.3, 49.8, 50.2, 50.9,

        50.3, 50.4, 50.0, 49.7, 50.5, 49.9]


# Population mean (H0)

mu = 50


# Sample mean and standard deviation

x_bar = np.mean(sample_data)
```

```
s = np.std(sample_data, ddof=1)
n = len(sample_data)

# Standard error
se = s / np.sqrt(n)

# Z-statistic
z_stat = (x_bar - mu) / se

# Two-tailed p-value
p_value = 2 * (1 - stats.norm.cdf(abs(z_stat)))

print(f"Sample Mean: {x_bar:.3f}")
print(f"Sample Standard Deviation: {s:.3f}")
print(f"Z-Statistic: {z_stat:.3f}")
print(f"P-Value: {p_value:.4f}")

# Interpretation
alpha = 0.05
if p_value < alpha:
    print("Reject the null hypothesis: Evidence suggests the mean is different from 50.")
else:
    print("Fail to reject the null hypothesis: No sufficient evidence to say the mean is different from 50.")
```

Explanation:

1. Calculate the sample mean and sample standard deviation.
2. Compute the standard error: ($SE = s / \sqrt{n}$).
3. Compute the Z-statistic: ($Z = (\bar{x} - \mu) / SE$).
4. Compute the two-tailed p-value using the normal cumulative distribution function.
5. Compare the p-value with the significance level ($\alpha = 0.05$) to accept or reject H_0.

Question 8: Write a Python script to simulate data from a normal distribution and calculate the 95% confidence interval for its mean. Plot the data using Matplotlib. (Include your Python code and output in the code box below.)

Here's a complete Python script to simulate data from a normal distribution, calculate the 95% confidence interval for its mean, and plot the data using Matplotlib:

```
# Import necessary libraries
import numpy as np
import matplotlib.pyplot as plt
from scipy import stats

# Parameters for normal distribution
mean = 50
std_dev = 5
sample_size = 100

# Simulate data
```

```
data = np.random.normal(loc=mean, scale=std_dev, size=sample_size)

# Calculate sample mean and standard deviation
x_bar = np.mean(data)
s = np.std(data, ddof=1)

# Standard error
se = s / np.sqrt(sample_size)

# 95% confidence interval using t-distribution
confidence_level = 0.95
t_score = stats.t.ppf((1 + confidence_level) / 2, df=sample_size - 1)
ci_lower = x_bar - t_score * se
ci_upper = x_bar + t_score * se

print(f"Sample Mean: {x_bar:.3f}")
print(f"Sample Standard Deviation: {s:.3f}")
print(f"95% Confidence Interval: ({ci_lower:.3f}, {ci_upper:.3f})")

# Plot histogram
plt.hist(data, bins=15, color='skyblue', edgecolor='black')
plt.title("Histogram of Simulated Normal Data")
plt.xlabel("Value")
plt.ylabel("Frequency")
plt.axvline(x=ci_lower, color='red', linestyle='--', label=f"95% CI Lower = {ci_lower:.2f}")
plt.axvline(x=ci_upper, color='green', linestyle='--', label=f"95% CI Upper = {ci_upper:.2f}")
plt.axvline(x=x_bar, color='orange', linestyle='-', label=f"Mean = {x_bar:.2f}")

plt.legend()

plt.show()
```

Explanation:

1. np.random.normal() generates normally distributed random data.
2. Sample mean (x̄) and standard deviation (s) are calculated.
3. Standard error: (SE = s / \sqrt{n}).
4. 95% confidence interval is calculated using t-distribution because the population standard deviation is unknown.
5. The histogram shows the data distribution with mean and confidence interval marked.

Question 9: Write a Python function to calculate the Z-scores from a dataset and visualize the standardized data using a histogram. Explain what the Z-scores represent in terms of standard deviations from the mean.

Here’s a Python solution to calculate Z-scores from a dataset, visualize them, and explain their significance:

Python Code:

```
# Import necessary libraries

import numpy as np

import matplotlib.pyplot as plt


# Function to calculate Z-scores

def calculate_z_scores(data):
    mean = np.mean(data)
    std_dev = np.std(data, ddof=1)  # sample standard deviation
    z_scores = (data - mean) / std_dev
    return z_scores

# Sample dataset
data = [49.1, 50.2, 51.0, 48.7, 50.5, 49.8, 50.3, 50.7, 50.2, 49.6,
        50.1, 49.9, 50.8, 50.4, 48.9, 50.6, 50.0, 49.7, 50.2, 49.5]

# Calculate Z-scores
z_scores = calculate_z_scores(data)

print("Z-scores:", np.round(z_scores, 3))

# Plot histogram of Z-scores
plt.hist(z_scores, bins=10, color='lightcoral', edgecolor='black')
plt.title("Histogram of Z-scores")
plt.xlabel("Z-score")
plt.ylabel("Frequency")
plt.axvline(0, color='blue', linestyle='--', label='Mean (0)')
plt.legend()
plt.show()
```

Explanation of Z-scores:

1. A Z-score measures how many standard deviations a data point is from the mean.

   Formula: (Z = \frac{X - \bar{X}}{s})

X = data point

$\bar{x}$ = sample mean

s = sample standard deviation

Interpretation:

Z = 0 → data point equals the mean

Z > 0 → data point is above the mean

Z < 0 → data point is below the mean

|Z| = 1 → data point is one standard deviation away from the mean