tion-theory-and-hypothesis-testing

November 25, 2024

[1]:
```
import scipy.stats as stats

# Define parameters
mu = 60  # Population mean
sigma = 6  # Population standard deviation
n = 40   # Sample size
x_bar = 58  # Sample mean

# Calculate standard error of the mean
sem = sigma / (n**0.5)

# Calculate Z-score
z = (x_bar - mu) / sem

# Calculate probability using the Z-table or calculator
probability = stats.norm.cdf(z)

print("The probability that the mean lifetime of 40 batteries is less than 58␣
 ↪months is:", probability)
```

```
The probability that the mean lifetime of 40 batteries is less than 58 months
is: 0.017507490509831244
```

[2]:
```
import scipy.stats as stats

# Sample size
n = 40

# Sample mean
x_bar = 310

# Population standard deviation
sigma = 89

# Confidence level
alpha = 0.05

# Calculate standard error of the mean
```

```
sem = sigma / (n**0.5)

# Calculate z-score for the desired confidence level
z = stats.norm.ppf(1 - alpha/2)

# Calculate the margin of error
margin_of_error = z * sem

# Calculate the confidence interval
lower_bound = x_bar - margin_of_error
upper_bound = x_bar + margin_of_error

print("95% Confidence Interval:", (lower_bound, upper_bound))
```

```
95% Confidence Interval: (282.419121062447, 337.580878937553)
```

[8]:
```
import scipy.stats as stats

# Hypothesized population mean
mu0 = 4.5

# Sample data (replace with your actual data)
sample_data = [4.2, 4.7, 5.1, 3.8, 4.9, 4.3, 5.0, 4.6, 4.1, 4.8]

# Calculate sample statistics
sample_mean = sum(sample_data) / len(sample_data)
sample_size = len(sample_data)

# Calculate the standard error using sem function
se = stats.sem(sample_data)

# Calculate the z-statistic
z_stat = (sample_mean - mu0) / se

# Calculate the p-value for a two-tailed test
p_value = stats.norm.sf(abs(z_stat)) * 2

print("Sample mean:", sample_mean)
print("Standard error:", se)
print("Z-statistic:", z_stat)
print("P-value:", p_value)

# Set the significance level (alpha)
alpha = 0.05

# Compare the p-value to the significance level
if p_value < alpha:
```

```
    print("Reject 1  the null hypothesis. The mean waiting time has changed.")
else:
    print("Fail to reject the null hypothesis. There is not enough evidence to␣
 ↪conclude that the mean waiting time has changed.")
```

```
Sample mean: 4.55
Standard error: 0.13601470508735442
Z-statistic: 0.36760731104690264
P-value: 0.7131660625710632
Fail to reject the null hypothesis. There is not enough evidence to conclude
that the mean waiting time has changed.
```

[9]:
```
import scipy.stats as stats

# Given z-statistic
z_stat = 2.00

# Calculate the p-value for a two-tailed z-test
p_value = stats.norm.sf(abs(z_stat)) * 2

print("The p-value for the two-tailed z-test is:", p_value)
```

```
The p-value for the two-tailed z-test is: 0.04550026389635839
```

[10]:
```
import scipy.stats as stats

# Given values
sample_mean = 30000
population_std = 8000
sample_size = 400
hypothesized_mean = 29000
alpha = 0.05

# Calculate the Z-score
z_score = (sample_mean - hypothesized_mean) / (population_std / (sample_size**0.
 ↪5))

# Calculate the p-value for a one-tailed test
p_value = stats.norm.sf(z_score)

print("Z-score:", z_score)
print("P-value:", p_value)

if p_value < alpha:
    print("Reject the null hypothesis. The mean household income is␣
 ↪significantly greater than 29000 rupees.")
```

```
    print("K2 Jeans should consider launching the product line in the new␣
 ↪market.")
else:
    print("Fail to reject the null hypothesis. There is not enough evidence to␣
 ↪conclude that the mean household income is greater than 29000 rupees.")
    print("K2 Jeans should reconsider launching the product line in the new␣
 ↪market.")
```

```
Z-score: 2.5
P-value: 0.006209665325776132
Reject the null hypothesis. The mean household income is significantly greater
than 29000 rupees.
K2 Jeans should consider launching the product line in the new market.
```