Open in Colab

Confidence Intervals

This notebook will guide you through the process of understanding and calculating **confidence intervals** for the mean in real-world quality control settings.

We'll begin with theory, then mathematical formulation, and finally apply it to your examples.

1. Theory: Why Use Confidence Intervals?

In statistics, a **confidence interval** gives an estimated range of values likely to include an unknown population parameter. This is particularly important in quality control where decisions must be made based on sample data.

For example, suppose a production line is meant to fill tubes with exactly 6 oz. of toothpaste. Due to natural variability, each tube may not have exactly 6 oz. Confidence intervals allow us to assess whether the filling process is within an acceptable range.

2. Mathematical Background

For a **population mean μ** with known standard deviation **σ**, and a sample of size **n**, the confidence interval is given by:

$$\bar{x} \pm Z_{\alpha/2} \cdot \frac{\sigma}{\sqrt{n}}$$

Where:

- $\bar{x}$ is the sample mean
- $Z_{\alpha/2}$ is the z-value for your desired confidence level (e.g. 1.96 for 95%)
- σ is the population standard deviation
- n is the sample size

If the population standard deviation is not known, the sample standard deviation **s** is used instead, and the t-distribution is applied:

$$\bar{x} \pm t_{\alpha/2,n-1} \cdot \frac{s}{\sqrt{n}}$$

3. Example 1: Toothpaste Tube Filling

A production line is designed to fill tubes with 6 oz. of toothpaste.

You randomly sample 30 tubes:

- Sample mean: 6.1 oz
- Known population standard deviation: 0.2 oz
- Significance level: 0.03 (corresponds to a 97% confidence level)

Should you stop the production line? Let's calculate the confidence interval and see whether 6 oz falls within it.

In [ ]:
```python
import scipy.stats as stats
import numpy as np

# Given data
x_bar = 6.1         # sample mean
sigma = 0.2         # population std dev
n = 30              # sample size
alpha = 0.03

# Z-score for 97% confidence
z_score = stats.norm.ppf(1 - alpha / 2)

# Margin of error
margin_error = z_score * (sigma / np.sqrt(n))

# Confidence interval
lower = x_bar - margin_error
upper = x_bar + margin_error

print(f"Confidence Interval: ({lower:.3f}, {upper:.3f})")
print("Decision: Stop production?" if 6.0 < lower or 6.0 > upper else "Production i
```

```
Confidence Interval: (6.021, 6.179)
Decision: Stop production?
```

4. Example 2: Emergency Response Time

Goal response time: 12 min or less.

Sample of 40 emergencies:

- Sample mean: 11.35 min
- Sample standard deviation: 3.25
- Confidence level: 95%

Should the Director of the Medical Center be worried?

In [ ]:
```python
# Given data
x_bar = 11.35       # sample mean
s = 3.25            # sample std dev
```

```
n = 40               # sample size
alpha = 0.05

# t-score for 95% confidence and df = n-1
t_score = stats.t.ppf(1 - alpha / 2, df=n-1)

# Margin of error
margin_error = t_score * (s / np.sqrt(n))

# Confidence interval
lower = x_bar - margin_error
upper = x_bar + margin_error

print(f"Confidence Interval: ({lower:.3f}, {upper:.3f})")
print("Decision: Response time is exceeding target" if upper > 12 else "Response ti
```

```
Confidence Interval: (10.311, 12.389)
Decision: Response time is exceeding target
```

5. Summary and Practice

- Confidence intervals provide a way to assess variability and control processes.
- Use **z-distribution** when population standard deviation is known, **t-distribution** when unknown.
- Apply the formula to determine whether key values fall within confidence ranges.

Your Turn

Try creating a similar example using your own data, or modify the above examples by changing the sample mean, standard deviation, or confidence level. You could also use the df.describe results you obtained from previous sessions.

Write an interpretation of your results.