How can a binomial confidence interval be calculated in Python?

Authored by

stats writer

June 29, 2024

A binomial confidence interval refers to a statistical method used to estimate the true proportion or percentage of success in a population based on a sample. In Python, this can be calculated using the binomial distribution function, which takes into account the sample size, the observed number of successes, and a chosen level of confidence. By inputting these variables into the function, a range of values can be determined within which the true proportion of success in the population is likely to fall with a certain level of confidence. This method allows for the estimation of population parameters and can be a useful tool in decision making and data analysis.

Calculate a Binomial Confidence Interval in Python

A confidence interval for a binomial probability is calculated using the following formula:

Confidence Interval = p +/- z*(√p(1-p) / n)

where:

p: proportion of "successes"z: the chosen z-valuen: sample size

The easiest way to calculate this type of confidence interval in Python is to use the proportion_confint() function from the statsmodels package:

proportion_confint(count, nobs, alpha=0.05, method='normal')

where:

count: Number of successesnobs: Total number of trialsalpha: Significance level (default is 0.05)method: Method to use for confidence interval (default is "normal")

The following example shows how to use this function in practice.

Example: Calculate Binomial Confidence Interval in Python

Suppose we want to estimate the proportion of residents in a county that are in favor of a certain law.

We decide to select a random sample of 100 residents and find that 56 of them are in favor of the law.

We can use the proportion_confint() function to calculate the 95% confidence interval for the true proportion of residents who suppose this law in the entire county:

```
from statsmodels.stats.proportionimport proportion_confint

#calculate 95% confidence interval with 56 successes in 100 trials
```

```
proportion_confint(count=56, nobs=100)

(0.4627099463758483, 0.6572900536241518)
```

The 95% confidence interval for the true proportion of residents in the county that support the law is .

By default, this function uses the asymptotic normal approximation to calculate the confidence interval. However, we can use the method argument to use a different method.

For example, the default function used in the R programming language to calculate a binomial confidence interval is the Wilson Score Interval.

```
from statsmodels.stats.proportionimport proportion_confint

#calculate 95% confidence interval with 56 successes in 100 trials
proportion_confint(count=56, nobs=100, method='wilson')

(0.4622810465167698, 0.6532797336983921)
```

This tells us that the 95% confidence interval for the true proportion of residents in the county that support the law is .

This confidence interval is just slightly different than the one calculated using the normal approximation.

Note that we can also adjust the alpha value to calculate a different confidence interval.

For example, we can set alpha to be 0.10 to calculate a 90% confidence interval:

```
from statsmodels.stats.proportionimport proportion_confint

#calculate 90% confidence interval with 56 successes in 100 trials
proportion_confint(count=56, nobs=100, alpha=0.10, method='wilson')
```

(0.4778381449964741 5, 0.6390007285095451)

This tells us that the 90% confidence interval for the true proportion of residents in the county that support the law is .

Note: You can find the complete documentation for the proportion_confint() function .

Additional Resources

The following tutorials explain how to perform other common operations in Python: