statistics-advance-7

August 13, 2023

[ ]:
```
# Q1. Write a Python function that takes in two arrays of data and calculates␣
 ↪the F-value for a variance ratio test.

# The function should return the F-value and the corresponding p-value for the␣
 ↪test.
```

[1]:
```
import numpy as np
from scipy.stats import f

def variance_ratio_test(data1, data2):
    # Calculate the variances of the two datasets
    var1 = np.var(data1, ddof=1)  # Using ddof=1 for unbiased sample variance
    var2 = np.var(data2, ddof=1)

    # Calculate the F-value
    f_value = var1 / var2

    # Degrees of freedom for the F-distribution
    df1 = len(data1) - 1
    df2 = len(data2) - 1

    # Calculate the p-value
    p_value = 1 - f.cdf(f_value, df1, df2)

    return f_value, p_value

# Example usage
data1 = np.array([12, 15, 18, 20, 25])
data2 = np.array([10, 14, 16, 22, 24])
f_value, p_value = variance_ratio_test(data1, data2)
print("F-value:", f_value)
print("p-value:", p_value)
```

```
F-value: 0.7379518072289156
p-value: 0.6122279427198223
```

[ ]:
```
# Q2. Given a significance level of 0.05 and the degrees of freedom for the␣
↪numerator and denominator of an F-distribution.

# Write a Python function that returns the critical F-value for a two-tailed␣
↪test.
```

[2]:
```
from scipy.stats import f

def critical_f_value(significance_level, df_num, df_den):
    # Calculate the critical F-value for a two-tailed test
    alpha = significance_level / 2  # Divide by 2 for a two-tailed test
    crit_f_value = f.ppf(1 - alpha, df_num, df_den)

    return crit_f_value

# Given values
significance_level = 0.05
df_num = 3  # Degrees of freedom for the numerator
df_den = 12  # Degrees of freedom for the denominator

crit_f = critical_f_value(significance_level, df_num, df_den)
print("Critical F-value:", crit_f)
```

```
Critical F-value: 4.474184809637748
```

[ ]:
```
# Q3. Write a Python program that generates random samples from two normal␣
↪distributions with known variances and uses an F-test to determine if the␣
↪variances are equal.

# The program should output the F value, degrees of freedom, and p-value for␣
↪the test.
```

[3]:
```
import numpy as np
from scipy.stats import f

def f_test_equal_variances(sample1, sample2):
    # Calculate the variances of the two samples
    var1 = np.var(sample1, ddof=1)  # Using ddof=1 for unbiased sample variance
    var2 = np.var(sample2, ddof=1)

    # Calculate the F-value
    f_value = var1 / var2

    # Degrees of freedom for the F-distribution
    df1 = len(sample1) - 1
    df2 = len(sample2) - 1

```

```
    # Calculate the p-value
    p_value = 2 * min(f.cdf(f_value, df1, df2), 1 - f.cdf(f_value, df1, df2))

    return f_value, df1, df2, p_value

# Generate random samples from two normal distributions with known variances
np.random.seed(42)
sample_size = 50
mean1, var1 = 0, 1
mean2, var2 = 0, 1.5
sample1 = np.random.normal(mean1, np.sqrt(var1), sample_size)
sample2 = np.random.normal(mean2, np.sqrt(var2), sample_size)

# Perform the F-test for equal variances
f_value, df1, df2, p_value = f_test_equal_variances(sample1, sample2)

print("F-value:", f_value)
print("Degrees of freedom (numerator):", df1)
print("Degrees of freedom (denominator):", df2)
print("p-value:", p_value)
```

```
F-value: 0.7602363589291505
Degrees of freedom (numerator): 49
Degrees of freedom (denominator): 49
p-value: 0.3405506021326978
```

[ ]:
```
# Q4.The variances of two populations are known to be 10 and 15. A sample of 12␣
↪observations is taken from each population.
# Conduct an F-test at the 5% significance level to determine if the variances␣
↪are significantly different
```

[4]:
```
import scipy.stats as stats

# Given data
variance1 = 10
variance2 = 15
sample_size = 12
significance_level = 0.05

# Calculate the F-statistic
f_statistic = variance1 / variance2

# Calculate degrees of freedom
df1 = sample_size - 1
df2 = sample_size - 1

# Calculate critical F-value
```

```
critical_f_value = stats.f.ppf(1 - significance_level / 2, df1, df2)

# Perform the F-test
if f_statistic > critical_f_value:
    print("Reject the null hypothesis. Variances are significantly different.")
else:
    print("Fail to reject the null hypothesis. Variances are not significantly␣
 ↪different.")

print("Calculated F-statistic:", f_statistic)
print("Critical F-value:", critical_f_value)
```

```
Fail to reject the null hypothesis. Variances are not significantly different.
Calculated F-statistic: 0.6666666666666666
Critical F-value: 3.473699051085809
```

[ ]:
```
# Q5. A manufacturer claims that the variance of the diameter of a certain␣
 ↪product is 0.005.

# A sample of 25 products is taken, and the sample variance is found to be 0.
 ↪006.

# Conduct an F-test at the 1% significance level to determine if the claim is␣
 ↪justified
```

[5]:
```
import scipy.stats as stats

# Given data
claimed_variance = 0.005
sample_variance = 0.006
sample_size = 25
significance_level = 0.01

# Calculate the F-statistic
f_statistic = sample_variance / claimed_variance

# Calculate degrees of freedom
df1 = sample_size - 1
df2 = sample_size - 1

# Calculate critical F-values
critical_f_lower = stats.f.ppf(significance_level / 2, df1, df2)
critical_f_upper = stats.f.ppf(1 - significance_level / 2, df1, df2)

# Perform the F-test
if f_statistic > critical_f_upper or f_statistic < 1 / critical_f_upper:
    print("Reject the null hypothesis. Claimed variance is not justified.")
```

```
else:
    print("Fail to reject the null hypothesis. Claimed variance is justified.")

print("Calculated F-statistic:", f_statistic)
print("Critical F-values (lower and upper):", critical_f_lower,␣
 ↪critical_f_upper)
```

```
Fail to reject the null hypothesis. Claimed variance is justified.
Calculated F-statistic: 1.2
Critical F-values (lower and upper): 0.3370701342685674 2.966741631292762
```

[ ]:
```
# Q6. Write a Python function that takes in the degrees of freedom for the␣
 ↪numerator and denominator of an F-distribution and calculates the mean and␣
 ↪variance of the distribution.

# The function should return the mean and variance as a tuple.
```

[14]:
```
def f_distribution_mean_variance(df_numerator, df_denominator):
    if df_numerator <= 0 or df_denominator <= 0:
        raise ValueError("Degrees of freedom must be positive.")

    if df_denominator == 1:
        raise ValueError("For F-distribution, denominator degrees of freedom␣
 ↪should be greater than 1.")

    mean = df_denominator / (df_denominator - 2)
    if df_denominator <= 4:
        variance = float('inf')  # Variance is undefined for df_denominator <= 4
    else:
        variance = (2 * (df_denominator ** 2) * (df_numerator + df_denominator␣
 ↪- 2)) / (df_numerator * (df_denominator - 2) ** 2 * (df_denominator - 4))

    return mean, variance

# Example usage
numerator_df = 5
denominator_df = 15
mean, variance = f_distribution_mean_variance(numerator_df, denominator_df)
print(f"Mean: {mean}, Variance: {variance}")
```

```
Mean: 1.1538461538461537, Variance: 0.8714362560516407
```

The mean of the F-distribution is calculated using the formula: mean = df_denominator / (df_denominator - 2)

The variance of the F-distribution is calculated using a formula that depends on the degrees of freedom for both the numerator and denominator.

variance = (2 * (df_denominator^2) * (df_numerator + df_denominator - 2)) / (df_numerator * (df_denominator - 2)^2 * (df_denominator - 4))

[ ]:
```
# Q7. A random sample of 10 measurements is taken from a normal population with␣
↪unknown variance.

# The sample variance is found to be 25.

# Another random sample of 15 measurements is taken from another normal␣
↪population with unknown variance, and the sample variance is found to be 20.

# Conduct an F-test at the 10% significance level to determine if the variances␣
↪are significantly different.
```

[7]:
```
import scipy.stats as stats

# Given data
sample_variance1 = 25
sample_variance2 = 20
sample_size1 = 10
sample_size2 = 15
significance_level = 0.10

# Calculate the F-statistic
f_statistic = sample_variance1 / sample_variance2

# Calculate degrees of freedom
df1 = sample_size1 - 1
df2 = sample_size2 - 1

# Calculate critical F-values
critical_f_lower = stats.f.ppf(significance_level / 2, df1, df2)
critical_f_upper = stats.f.ppf(1 - significance_level / 2, df1, df2)

# Perform the F-test
if f_statistic > critical_f_upper or f_statistic < 1 / critical_f_upper:
    print("Reject the null hypothesis. Variances are significantly different.")
else:
    print("Fail to reject the null hypothesis. Variances are not significantly␣
↪different.")

print("Calculated F-statistic:", f_statistic)
print("Critical F-values (lower and upper):", critical_f_lower,␣
↪critical_f_upper)
```

```
Fail to reject the null hypothesis. Variances are not significantly different.
Calculated F-statistic: 1.25
Critical F-values (lower and upper): 0.3305268601412525 2.6457907352338195
```

[ ]:
```
# Q8. The following data represent the waiting times in minutes at two
↪different restaurants on a Saturday night:

# Restaurant A: 24, 25, 28, 23, 22, 20, 27;

# Restaurant B: 31, 33, 35, 30, 32, 36.

# Conduct an F-test at the 5% significance level to determine if the variances
↪are significantly different.
```

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

# Given data
waiting_times_restaurant_A = [24, 25, 28, 23, 22, 20, 27]
waiting_times_restaurant_B = [31, 33, 35, 30, 32, 36]
significance_level = 0.05

# Calculate sample variances
sample_variance_A = sum([(x - sum(waiting_times_restaurant_A)/
↪len(waiting_times_restaurant_A))**2 for x in waiting_times_restaurant_A]) /
↪(len(waiting_times_restaurant_A) - 1)
sample_variance_B = sum([(x - sum(waiting_times_restaurant_B)/
↪len(waiting_times_restaurant_B))**2 for x in waiting_times_restaurant_B]) /
↪(len(waiting_times_restaurant_B) - 1)

# Calculate the F-statistic
f_statistic = sample_variance_A / sample_variance_B

# Calculate degrees of freedom
df1 = len(waiting_times_restaurant_A) - 1
df2 = len(waiting_times_restaurant_B) - 1

# Calculate critical F-values
critical_f_lower = stats.f.ppf(significance_level / 2, df1, df2)
critical_f_upper = stats.f.ppf(1 - significance_level / 2, df1, df2)

# Perform the F-test
if f_statistic > critical_f_upper or f_statistic < 1 / critical_f_upper:
    print("Reject the null hypothesis. Variances are significantly different.")
else:
    print("Fail to reject the null hypothesis. Variances are not significantly
↪different.")

print("Calculated F-statistic:", f_statistic)
print("Critical F-values (lower and upper):", critical_f_lower,
↪critical_f_upper)
```

```
Fail to reject the null hypothesis. Variances are not significantly different.
Calculated F-statistic: 1.4551907719609583
Critical F-values (lower and upper): 0.16701279718024772 6.977701858535566
```

[ ]:
```python
# Q9. The following data represent the test scores of two groups of students:

# Group A: 80, 85, 90, 92, 87, 83;

# Group B: 75, 78, 82, 79, 81, 84.

# Conduct an F-test at the 1% significance level to determine if the variances␣
↪are significantly different
```

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

# Given data
test_scores_group_A = [80, 85, 90, 92, 87, 83]
test_scores_group_B = [75, 78, 82, 79, 81, 84]
significance_level = 0.01

# Calculate sample variances
sample_variance_group_A = sum([(x - sum(test_scores_group_A)/
↪len(test_scores_group_A))**2 for x in test_scores_group_A]) /␣
↪(len(test_scores_group_A) - 1)
sample_variance_group_B = sum([(x - sum(test_scores_group_B)/
↪len(test_scores_group_B))**2 for x in test_scores_group_B]) /␣
↪(len(test_scores_group_B) - 1)

# Calculate the F-statistic
f_statistic = sample_variance_group_A / sample_variance_group_B

# Calculate degrees of freedom
df1 = len(test_scores_group_A) - 1
df2 = len(test_scores_group_B) - 1

# Calculate critical F-values
critical_f_lower = stats.f.ppf(significance_level / 2, df1, df2)
critical_f_upper = stats.f.ppf(1 - significance_level / 2, df1, df2)

# Perform the F-test
if f_statistic > critical_f_upper or f_statistic < 1 / critical_f_upper:
    print("Reject the null hypothesis. Variances are significantly different.")
else:
    print("Fail to reject the null hypothesis. Variances are not significantly␣
↪different.")

print("Calculated F-statistic:", f_statistic)
```

```python
print("Critical F-values (lower and upper):", critical_f_lower, critical_f_upper)
```

```
Fail to reject the null hypothesis. Variances are not significantly different.
Calculated F-statistic: 1.9442622950819677
Critical F-values (lower and upper): 0.066936171954696 14.939605459912224
```

[ ]:

```python

```