Assignment T-test

Date: 05/05/2020 Name: D.Saravanan

TTest test the null hypothesis H_0 against the alternative hypothesis H_1.

For univariate samples, TTest performs a Student t test. The test statistic is assumed to follow a StudentTDistribution [df].

The degrees of freedom df, used to specify the distribution of the test statistic, depend on the sample size, number of samples, and in the case of two univariate samples, the results of a test for equal variances.

For the TTest, a cutoff α is chosen such that H_0 is rejected only if $p < \alpha$. The value of α used for the "TestConclusion" and "ShortTestConclusion" properties is controlled by the SignificanceLevel option. This value α is also used in diagnostic tests of assumptions, including tests for normality, equal variance, and symmetry. By default, α is set to 0.05.

1. Two sets of ten students selected at random from a college were taken. One set was given memory test as they were and the other was given the memory test after two weeks of training and the scores are given below.

Set A:	10	8	7	9	8	10	9	6	7	8
Set B:	12	8	8	10	8	11	9	8	9	9

Do you think there is a significant effect due to training?

Solution:

Null hypothesis: $H_0 : \mu 1 = \mu 2$
Alternative hypothesis: $H_1 : \mu 1 \neq \mu 2$

Set A:

Mean:

$$\bar{x}_1 = \frac{\sum\limits_{i=1}^{n1} x_{1i}}{n1} = \frac{10+8+7+9+8+10+9+6+7+8}{10} = 8.2$$

Variance:

$$s_1^2 = \frac{\sum\limits_{i=1}^{n1} (x_{1i} - \bar{x_1})^2}{n1 - 1} = \frac{\sum\limits_{i=1}^{10} (x_{1i} - 8.2)^2}{10 - 1} = 1.73333$$

Standard Deviation:

$$s_1 = \sqrt{\frac{\sum\limits_{i=1}^{n1} (x_{1i} - \bar{x_1})^2}{n1 - 1}} = \sqrt{\frac{\sum\limits_{i=1}^{10} (x_{1i} - 8.2)^2}{10 - 1}} = \sqrt{1.73333} = 1.31656$$

Set B:

Mean:

$$\bar{x}_2 = \frac{\sum\limits_{i=1}^{n2} x_{2i}}{n2} = \frac{12+8+8+10+8+11+9+8+9+9}{10} = 9.2$$

Variance:

$$s_2^2 = \frac{\sum\limits_{i=1}^{n2} (x_{2i} - \bar{x_2})^2}{n2 - 1} = \frac{\sum\limits_{i=1}^{10} (x_{2i} - 9.2)^2}{10 - 1} = 1.95556$$

Standard Deviation:

$$s_2 = \sqrt{\frac{\sum_{i=1}^{n2}(x_{2i} - \bar{x}_2)^2}{n2-1}} = \sqrt{\frac{\sum_{i=1}^{10}(x_{2i} - 9.2)^2}{10-1}} = \sqrt{1.95556} = 1.39841$$

$$p_1 = \frac{x_1}{n_1} = \frac{8.2}{10} = 0.82 \quad \& \quad q_1 = 1 - p_1 = 1 - 0.82 = 0.18$$

$$p_2 = \frac{x_2}{n_2} = \frac{9.2}{10} = 0.92 \quad \& \quad q_2 = 1 - p_2 = 1 - 0.92 = 0.08$$

Population Proportion P:

$$P = \frac{n_1 p_1 + n_2 p_2}{n_1 + n_2} = \frac{10 \times 0.82 + 10 \times 0.92}{10 + 10} = 0.87 \quad \& \quad Q = 1 - P = 1 - 0.87 = 0.13$$

Calculation of $s1/s2$:

$$\frac{s_1}{s_2} = \frac{1.31656}{1.39841} = 0.94147$$

Test statistic:

$$T = \frac{\bar{x}_1 - \bar{x}_2}{\sqrt{s_1^2/n1 + s_2^2/n2}}$$

where $n1$ and $n2$ are the sample sizes, $\bar{x}_1$ and $\bar{x}_2$ are the sample means, and s_1^2 and s_2^2 are the sample variances. If equal variances are assumed $(0.5 < s1/s2 < 2)$, then the formula reduces to:

$$T = \frac{\bar{x}_1 - \bar{x}_2}{s_p\sqrt{1/n1 + 1/n2}}$$

where

$$s_p^2 = \frac{(n1-1)s_1^2 + (n2-1)s_2^2}{n1 + n2 - 2}$$

Calculation of s_p:

$$s_p = \sqrt{\frac{(10-1) \times 1.31656^2 + (10-1) \times 1.39841^2}{10 + 10 - 2}} = 1.35810$$

Calculation of T statistic:

$$T = \frac{8.2 - 9.2}{1.35810 \times \sqrt{1/10 + 1/10}} = -1.64646$$

Calculation of Standard Error:

$$S.E = s_p\sqrt{1/n1 + 1/n2} = 1.35810 \times \sqrt{1/10 + 1/10} = 0.60736$$

Program:

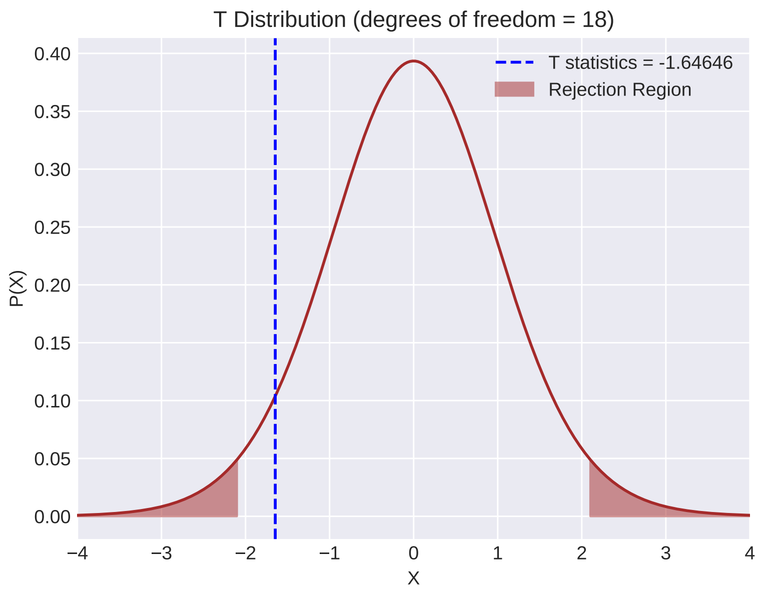
```
#!/usr/bin/env python3
import numpy as np
from scipy.stats import t
import matplotlib.pyplot as plt
plt.style.use('seaborn-darkgrid')

def similar_variance(n1,s1_mean,n2,s2_mean):
    dof = n1+n2-2
    sp = np.sqrt(((n1-1)*s1_stdv**2+(n2-1)*s2_stdv**2)/(dof))
    return (s1_mean-s2_mean)/(sp * np.sqrt(1/n1 + 1/n2)), dof, sp

def non_similar_variance(n1,s1_mean,n2,s2_mean):
    sd = np.sqrt(s1_stdv**2/n1 + s2_stdv**2/n2)
    dof = (sd**4)/(((s1_std**2/n1)**2/(n1-1)) + ((s2_std**2/n2)**2/(n2-1)))
    return (s1_mean-s2_mean)/sd, dof, sd

def ttest_and_variance(s1_stdv, s2_stdv):
    if 0.5 < s1_stdv/s2_stdv < 2:
        return similar_variance(n1,s1_mean,n2,s2_mean)
    else:
        return non_similar_variance(n1,s1_mean,n2,s2_mean)

print("__________Two-sample T-test for unpaired data__________\n")

sample1 = [float(value) for value in input("Sample_1 values: ").split()]
sample2 = [float(value) for value in input("Sample_2 values: ").split()]

# Number of Observations
n1 = len(sample1); n2 = len(sample2)

# Mean and Standard Deviation
s1_mean = np.mean(sample1); s1_stdv = np.std(sample1, ddof=1)
s2_mean = np.mean(sample2); s2_stdv = np.std(sample2, ddof=1)

# level of significance, confidence level
los = 0.05; cnl = 1 - los

# standard error (SE) of mean of sample1
std_err1 = s1_stdv/np.sqrt(n1)

print("\nSample 1: \n\t Number of Observations = {} \n\t Mean = {:.5f}".format(n1,s1_mean))
print("\t Standard Deviation = {:.5f}".format(s1_stdv))
print("\t Standard Error of the Mean = {:.5f}".format(std_err1))

# standard error (SE) of mean of sample2
std_err2 = s2_stdv/np.sqrt(n2)

print("\nSample 2: \n\t Number of Observations = {} \n\t Mean = {:.5f}".format(n2,s2_mean))
print("\t Standard Deviation = {:.5f}".format(s2_stdv))
print("\t Standard Error of the Mean = {:.5f}".format(std_err2))

# calculation of T-statistics and degrees of freedom
tstatistics, dof, sp = ttest_and_variance(s1_stdv, s2_stdv)
print("\nT statistics: {:.5f}".format(tstatistics))

# calculation of Critical values
tcritical_l = t.ppf(q = los/2, df = dof)
tcritical_u = -tcritical_l
print("\nCritical values are {:.5f}, {:.5f}".format(tcritical_l, tcritical_u))

# decision making - using T-statistics and Critical values
```

```
if tstatistics < tcritical_l or tstatistics > tcritical_u:
    print("Reject the Null hypothesis.")
else: print("Fail to reject the Null hypothesis.")

# calculation of p-value
pvalue = 2*t.cdf(tstatistics, df = dof)
print("\np-value: {:.5f}".format(pvalue))

# decision making - using p-value and level of significance
if pvalue < los: print("Reject the Null hypothesis.")
else: print("Fail to reject the Null hypothesis.")

# standard error
std_error = sp * np.sqrt(1/n1 + 1/n2)
print("\nStandard Error: {:.5f}".format(std_error))

# confidence interval
cnf_int = (s1_mean - s2_mean) + std_error * np.array([tcritical_l, tcritical_u])
print("Confidence Interval: {}".format(cnf_int))

# plot script
x1=np.linspace(-10,tcritical_l,1000); y1=t.pdf(x1,df=dof)
x2=np.linspace(tcritical_u,10,1000);  y2=t.pdf(x2,df=dof)
x3=np.linspace(-4,4,1000); y3=t.pdf(x3,df=dof)
ax=plt.figure().add_subplot(111); ax.plot(x3,y3,color='brown')
ax.fill_between(x1,y1,0,alpha=0.5,color='brown',label='Rejection Region')
ax.fill_between(x2,y2,0,alpha=0.5,color='brown'); ax.set(xlabel='X',ylabel='P(X)')
ax.axvline(x=tstatistics,ls='--',c='b',label='T statistics = {:.5f}'.format(tstatistics))
ax.set(xlim=[-4,4],title="T Distribution (degrees of freedom = {:.0f})".format(dof))
plt.legend(); plt.savefig('tscript3.pdf',dpi=72,bbox_inches='tight'); plt.show()
```

Output:

```
__________Two-sample T-test for unpaired data__________

Sample_1 values: 10 8 7 9 8 10 9 6 7 8
Sample_2 values: 12 8 8 10 8 11 9 8 9 9

Sample 1:
     Number of Observations = 10
     Mean = 8.20000
     Standard Deviation = 1.31656
     Standard Error of the Mean = 0.41633

Sample 2:
     Number of Observations = 10
     Mean = 9.20000
     Standard Deviation = 1.39841
     Standard Error of the Mean = 0.44222

T statistics: -1.64646

Critical values are -2.10092, 2.10092
Fail to reject the Null hypothesis.

p-value: 0.11702
Fail to reject the Null hypothesis.
```

```
Standard Error: 0.60736
Confidence Interval: [-2.27602071  0.27602071]
```



Program: T-test with in-built scipy.stats.ttest

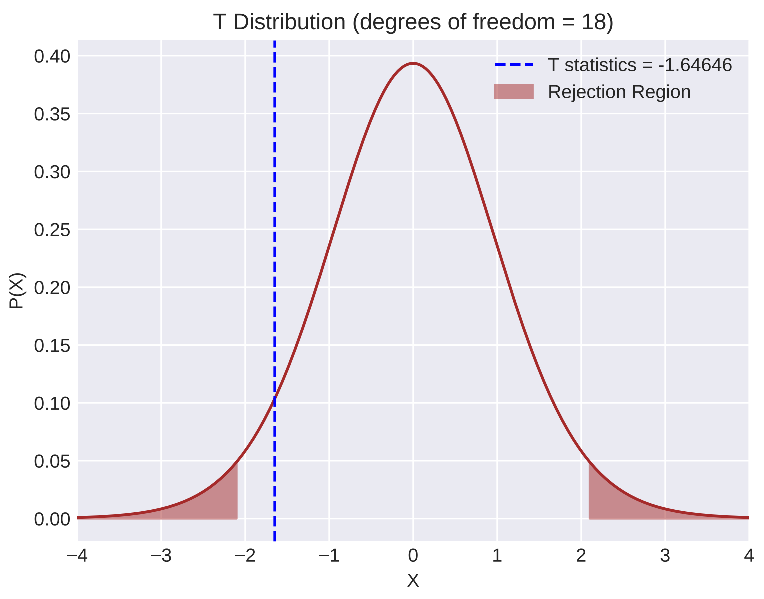
```
#!/usr/bin/env python3
import numpy as np
from scipy.stats import t, ttest_ind
import matplotlib.pyplot as plt
plt.style.use('seaborn-darkgrid')

print("__________Two-sample T-test for unpaired data__________\n")

sample1 = [float(value) for value in input("Sample_1 values: ").split()]
sample2 = [float(value) for value in input("Sample_2 values: ").split()]

# Number of Observations
n1 = len(sample1); n2 = len(sample2)

# Mean and Standard Deviation
s1_mean = np.mean(sample1); s1_stdv = np.std(sample1, ddof=1)
s2_mean = np.mean(sample2); s2_stdv = np.std(sample2, ddof=1)

# calculation of degrees of freedom and pooled variance
if 0.5 < s1_stdv/s2_stdv < 2:
    dof = n1+n2-2
    sp = np.sqrt(((n1-1)*s1_stdv**2+(n2-1)*s2_stdv**2)/(dof))
else:
    sp = np.sqrt(s1_stdv**2/n1 + s2_stdv**2/n2)
    dof = (sp**4)/(((s1_std**2/n1)**2/(n1-1)) + ((s2_std**2/n2)**2/(n2-1)))

# level of significance, confidence level
los = 0.05; cnl = 1 - los
```

```python
# standard error (SE) of mean of sample1
std_err1 = s1_stdv/np.sqrt(n1)

print("\nSample 1: \n\t Number of Observations = {} \n\t Mean = {:.5f}".format(n1,s1_mean))
print("\t Standard Deviation = {:.5f}".format(s1_stdv))
print("\t Standard Error of the Mean = {:.5f}".format(std_err1))

# standard error (SE) of mean of sample2
std_err2 = s2_stdv/np.sqrt(n2)

print("\nSample 2: \n\t Number of Observations = {} \n\t Mean = {:.5f}".format(n2,s2_mean))
print("\t Standard Deviation = {:.5f}".format(s2_stdv))
print("\t Standard Error of the Mean = {:.5f}".format(std_err2))

# calculation of T-statistics and p-value
tstatistics, pvalue = ttest_ind(sample1, sample2)
print("\nT statistics: {:.5f}".format(tstatistics))

# calculation of Critical values
tcritical_l = t.ppf(q = los/2, df = dof)
tcritical_u = -tcritical_l
print("\nCritical values are {:.5f}, {:.5f}".format(tcritical_l, tcritical_u))

# decision making - using T-statistics and Critical values
if tstatistics < tcritical_l or tstatistics > tcritical_u:
    print("Reject the Null hypothesis.")
else: print("Fail to reject the Null hypothesis.")

print("\np-value: {:.5f}".format(pvalue))

# decision making - using p-value
if pvalue < los: print("Reject the Null hypothesis.")
else: print("Fail to reject the Null hypothesis.")

# standard error
std_error = sp * np.sqrt(1/n1 + 1/n2)
print("\nStandard Error: {:.5f}".format(std_error))

# confidence interval
cnf_int = (s1_mean - s2_mean) + std_error * np.array([tcritical_l, tcritical_u])
print("Confidence Interval: {}".format(cnf_int))

# plot script
x1=np.linspace(-10,tcritical_l,1000); y1=t.pdf(x1,df=dof)
x2=np.linspace(tcritical_u,10,1000);  y2=t.pdf(x2,df=dof)
x3=np.linspace(-4,4,1000); y3=t.pdf(x3,df=dof)
ax=plt.figure().add_subplot(111); ax.plot(x3,y3,color='brown')
ax.fill_between(x1,y1,0,alpha=0.5,color='brown',label='Rejection Region')
ax.fill_between(x2,y2,0,alpha=0.5,color='brown'); ax.set(xlabel='X',ylabel='P(X)')
ax.axvline(x=tstatistics,ls='--',c='b',label='T statistics = {:.5f}'.format(tstatistics))
ax.set(xlim=[-4,4],title="T Distribution (degrees of freedom = {:.0f})".format(dof))
plt.legend(); plt.savefig('ttest2.pdf',dpi=72,bbox_inches='tight'); plt.show()
```

Output:

```
__________Two-sample T-test for unpaired data__________
```

```
Sample_1 values: 10 8 7 9 8 10 9 6 7 8
Sample_2 values: 12 8 8 10 8 11 9 8 9 9

Sample 1:
     Number of Observations = 10
     Mean = 8.20000
     Standard Deviation = 1.31656
     Standard Error of the Mean = 0.41633

Sample 2:
     Number of Observations = 10
     Mean = 9.20000
     Standard Deviation = 1.39841
     Standard Error of the Mean = 0.44222

T statistics: -1.64646

Critical values are -2.10092, 2.10092
Fail to reject the Null hypothesis.

p-value: 0.11702
Fail to reject the Null hypothesis.

Standard Error: 0.60736
Confidence Interval: [-2.27602071  0.27602071]
```



2. A group of 5 patients treated with medicine A weighs 42, 29, 48, 60 and 41 kg. A second group of 7 patients from the same hospital treated with medicine B weighs 38, 42, 56, 64, 68, 69 and 62 kg. Do you agree with the claim that medicine B increases weight significantly.

   Null hypothesis: $H_0 : \mu 1 = \mu 2$
   Alternative hypothesis: $H_1 : \mu 1 \neq \mu 2$

   Medicine A:

   Mean:

$$\bar{x}_1 = \frac{\sum_{i=1}^{n1} x_{1i}}{n1} = \frac{42 + 29 + 48 + 60 + 41}{5} = 44.0$$

   Variance:

$$s_1^2 = \frac{\sum_{i=1}^{n1} (x_{1i} - \bar{x_1})^2}{n1 - 1} = \frac{\sum_{i=1}^{5} (x_{1i} - 44.0)^2}{5 - 1} = 127.5$$

   Standard Deviation:

$$s_1 = \sqrt{\frac{\sum_{i=1}^{n1} (x_{1i} - \bar{x_1})^2}{n1 - 1}} = \sqrt{\frac{\sum_{i=1}^{5} (x_{1i} - 44.0)^2}{5 - 1}} = \sqrt{127.5} = 11.29159$$

   Medicine B:

   Mean:

$$\bar{x}_1 = \frac{\sum_{i=1}^{n1} x_{1i}}{n1} = \frac{38 + 42 + 56 + 64 + 68 + 69 + 62}{7} = 57.0$$

   Variance:

$$s_1^2 = \frac{\sum_{i=1}^{n1} (x_{1i} - \bar{x_1})^2}{n1 - 1} = \frac{\sum_{i=1}^{7} (x_{1i} - 57.0)^2}{7 - 1} = 154.3$$

   Standard Deviation:

$$s_1 = \sqrt{\frac{\sum_{i=1}^{n1} (x_{1i} - \bar{x_1})^2}{n1 - 1}} = \sqrt{\frac{\sum_{i=1}^{7} (x_{1i} - 57.0)^2}{7 - 1}} = \sqrt{154.3} = 12.42310$$

   Calculation of $s1/s2$:

$$\frac{s_1}{s_2} = \frac{11.29159}{12.42310} = 0.90892$$

   Test statistic:

$$T = \frac{\bar{x}_1 - \bar{x}_2}{\sqrt{s_1^2/n1 + s_2^2/n2}}$$

   If equal variances are assumed $(0.5 < s1/s2 < 2)$, then the formula reduces to:

$$T = \frac{\bar{x}_1 - \bar{x}_2}{s_p\sqrt{1/n1 + 1/n2}}$$

where

$$s_p^2 = \frac{(n1-1)s_1^2 + (n2-1)s_2^2}{n1+n2-2}$$

Calculation of s_p:

$$s_p = \sqrt{\frac{(5-1)\times 11.29159^2 + (7-1)\times 12.42310^2}{5+7-2}} = 11.98332$$

Calculation of T statistic:

$$T = \frac{44.0 - 57.0}{11.98332 \times \sqrt{1/5 + 1/7}} = -1.85272$$

Calculation of Standard Error:

$$S.E = s_p\sqrt{1/n1 + 1/n2} = 11.98332 \times \sqrt{1/5 + 1/7} = 7.01671$$

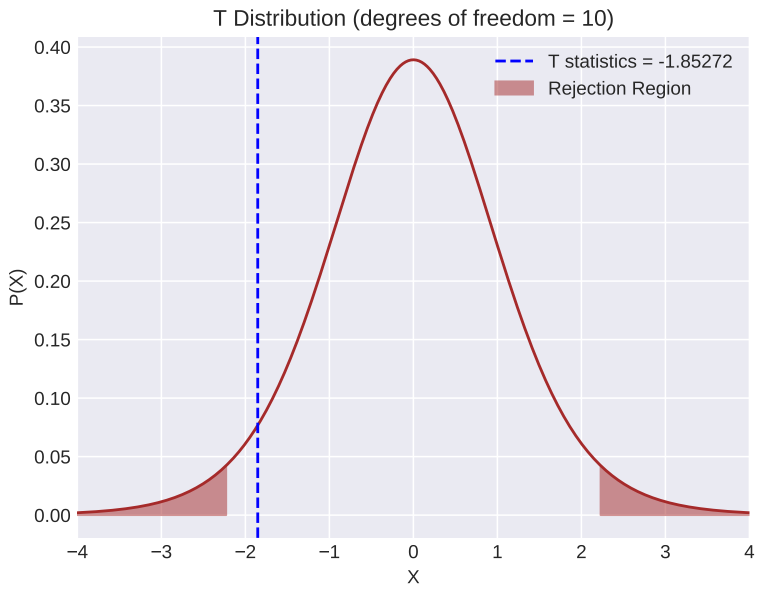
Program:

```
#!/usr/bin/env python3
import numpy as np
from scipy.stats import t
import matplotlib.pyplot as plt
plt.style.use('seaborn-darkgrid')

def similar_variance(n1,s1_mean,n2,s2_mean):
    dof = n1+n2-2
    sp = np.sqrt(((n1-1)*s1_stdv**2+(n2-1)*s2_stdv**2)/(dof))
    return (s1_mean-s2_mean)/(sp * np.sqrt(1/n1 + 1/n2)), dof, sp

def non_similar_variance(n1,s1_mean,n2,s2_mean):
    sd = np.sqrt(s1_stdv**2/n1 + s2_stdv**2/n2)
    dof = (sd**4)/(((s1_std**2/n1)**2/(n1-1)) + ((s2_std**2/n2)**2/(n2-1)))
    return (s1_mean-s2_mean)/sd, dof, sd

def ttest_and_variance(s1_stdv, s2_stdv):
    if 0.5 < s1_stdv/s2_stdv < 2:
        return similar_variance(n1,s1_mean,n2,s2_mean)
    else:
        return non_similar_variance(n1,s1_mean,n2,s2_mean)

print("__________Two-sample T-test for unpaired data__________\n")

sample1 = [float(value) for value in input("Sample_1 values: ").split()]
sample2 = [float(value) for value in input("Sample_2 values: ").split()]

# Number of Observations
n1 = len(sample1); n2 = len(sample2)

# Mean and Standard Deviation
s1_mean = np.mean(sample1); s1_stdv = np.std(sample1, ddof=1)
s2_mean = np.mean(sample2); s2_stdv = np.std(sample2, ddof=1)

# level of significance, confidence level
los = 0.05; cnl = 1 - los

# standard error (SE) of mean of sample1
std_err1 = s1_stdv/np.sqrt(n1)

print("\nSample 1: \n\t Number of Observations = {} \n\t Mean = {:.5f}".format(n1,s1_mean))
print("\t Standard Deviation = {:.5f}".format(s1_stdv))
print("\t Standard Error of the Mean = {:.5f}".format(std_err1))
```

```
# standard error (SE) of mean of sample2
std_err2 = s2_stdv/np.sqrt(n2)

print("\nSample 2: \n\t Number of Observations = {} \n\t Mean = {:.5f}".format(n2,s2_mean))
print("\t Standard Deviation = {:.5f}".format(s2_stdv))
print("\t Standard Error of the Mean = {:.5f}".format(std_err2))

# calculation of T-statistics and degrees of freedom
tstatistics, dof, sp = ttest_and_variance(s1_stdv, s2_stdv)
print("\nT statistics: {:.5f}".format(tstatistics))

# calculation of Critical values
tcritical_l = t.ppf(q = los/2, df = dof)
tcritical_u = -tcritical_l
print("\nCritical values are {:.5f}, {:.5f}".format(tcritical_l, tcritical_u))

# decision making - using T-statistics and Critical values
if tstatistics < tcritical_l or tstatistics > tcritical_u:
    print("Reject the Null hypothesis.")
else: print("Fail to reject the Null hypothesis.")

# calculation of p-value
pvalue = 2*t.cdf(tstatistics, df = dof)
print("\np-value: {:.5f}".format(pvalue))

# decision making - using p-value and level of significance
if pvalue < los: print("Reject the Null hypothesis.")
else: print("Fail to reject the Null hypothesis.")

# standard error
std_error = sp * np.sqrt(1/n1 + 1/n2)
print("\nStandard Error: {:.5f}".format(std_error))

# confidence interval
cnf_int = (s1_mean - s2_mean) + std_error * np.array([tcritical_l, tcritical_u])
print("Confidence Interval: {}".format(cnf_int))

# plot script
x1=np.linspace(-10,tcritical_l,1000); y1=t.pdf(x1,df=dof)
x2=np.linspace(tcritical_u,10,1000);  y2=t.pdf(x2,df=dof)
x3=np.linspace(-4,4,1000); y3=t.pdf(x3,df=dof)
ax=plt.figure().add_subplot(111); ax.plot(x3,y3,color='brown')
ax.fill_between(x1,y1,0,alpha=0.5,color='brown',label='Rejection Region')
ax.fill_between(x2,y2,0,alpha=0.5,color='brown'); ax.set(xlabel='X',ylabel='P(X)')
ax.axvline(x=tstatistics,ls='--',c='b',label='T statistics = {:.5f}'.format(tstatistics))
ax.set(xlim=[-4,4],title="T Distribution (degrees of freedom = {:.0f})".format(dof))
plt.legend(); plt.savefig('tscript3.pdf',dpi=72,bbox_inches='tight'); plt.show()
```

Output:

```
__________Two-sample T-test for unpaired data__________


Sample 1:
     Number of Observations = 5
     Mean = 44.00000
     Standard Deviation = 11.29159
     Standard Error of the Mean = 5.04975

Sample 2:
     Number of Observations = 7
     Mean = 57.00000
     Standard Deviation = 12.42310
     Standard Error of the Mean = 4.69549

T statistics: -1.85272

Critical values are -2.22814, 2.22814
Fail to reject the Null hypothesis.

p-value: 0.09363
Fail to reject the Null hypothesis.

Standard Error: 7.01671
Confidence Interval: [-28.63421472   2.63421472]
```



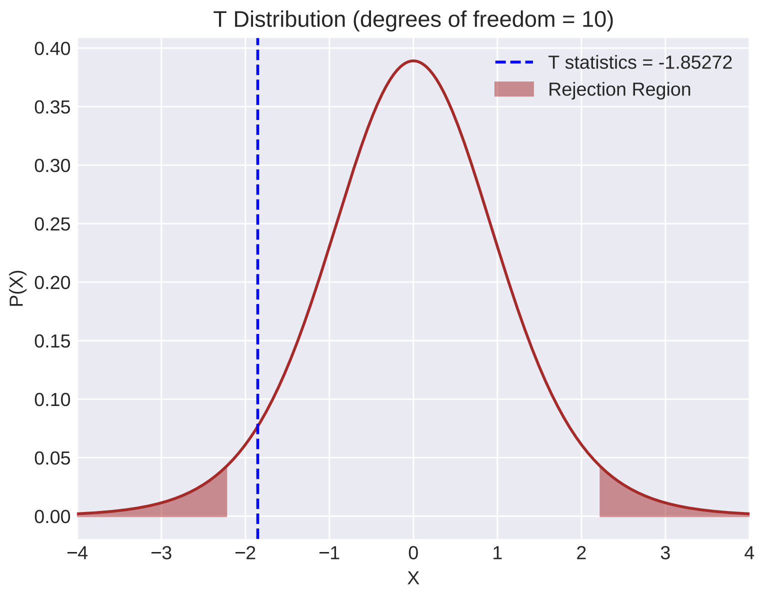
Program: T-test with in-built scipy.stats.ttest

```
#!/usr/bin/env python3
import numpy as np
from scipy.stats import t, ttest_ind
import matplotlib.pyplot as plt
plt.style.use('seaborn-darkgrid')

print("__________Two-sample T-test for unpaired data__________\n")

sample1 = [float(value) for value in input("Sample_1 values: ").split()]
sample2 = [float(value) for value in input("Sample_2 values: ").split()]

# Number of Observations
n1 = len(sample1); n2 = len(sample2)

# Mean and Standard Deviation
s1_mean = np.mean(sample1); s1_stdv = np.std(sample1, ddof=1)
s2_mean = np.mean(sample2); s2_stdv = np.std(sample2, ddof=1)

# calculation of degrees of freedom and pooled variance
if 0.5 < s1_stdv/s2_stdv < 2:
    dof = n1+n2-2
    sp = np.sqrt(((n1-1)*s1_stdv**2+(n2-1)*s2_stdv**2)/(dof))
else:
    sp = np.sqrt(s1_stdv**2/n1 + s2_stdv**2/n2)
    dof = (sp**4)/(((s1_std**2/n1)**2/(n1-1)) + ((s2_std**2/n2)**2/(n2-1)))

# level of significance, confidence level
los = 0.05; cnl = 1 - los

# standard error (SE) of mean of sample1
std_err1 = s1_stdv/np.sqrt(n1)

print("\nSample 1: \n\t Number of Observations = {} \n\t Mean = {:.5f}".format(n1,s1_mean))
print("\t Standard Deviation = {:.5f}".format(s1_stdv))
print("\t Standard Error of the Mean = {:.5f}".format(std_err1))

# standard error (SE) of mean of sample2
std_err2 = s2_stdv/np.sqrt(n2)

print("\nSample 2: \n\t Number of Observations = {} \n\t Mean = {:.5f}".format(n2,s2_mean))
print("\t Standard Deviation = {:.5f}".format(s2_stdv))
print("\t Standard Error of the Mean = {:.5f}".format(std_err2))

# calculation of T-statistics and p-value
tstatistics, pvalue = ttest_ind(sample1, sample2)
print("\nT statistics: {:.5f}".format(tstatistics))

# calculation of Critical values
tcritical_l = t.ppf(q = los/2, df = dof)
tcritical_u = -tcritical_l
print("\nCritical values are {:.5f}, {:.5f}".format(tcritical_l, tcritical_u))

# decision making - using T-statistics and Critical values
if tstatistics < tcritical_l or tstatistics > tcritical_u:
    print("Reject the Null hypothesis.")
else: print("Fail to reject the Null hypothesis.")

print("\np-value: {:.5f}".format(pvalue))

# decision making - using p-value
if pvalue < los: print("Reject the Null hypothesis.")
else: print("Fail to reject the Null hypothesis.")

# standard error
std_error = sp * np.sqrt(1/n1 + 1/n2)
print("\nStandard Error: {:.5f}".format(std_error))

# confidence interval
cnf_int = (s1_mean - s2_mean) + std_error * np.array([tcritical_l, tcritical_u])
print("Confidence Interval: {}".format(cnf_int))

# plot script
x1=np.linspace(-10,tcritical_l,1000); y1=t.pdf(x1,df=dof)
x2=np.linspace(tcritical_u,10,1000);  y2=t.pdf(x2,df=dof)
x3=np.linspace(-4,4,1000); y3=t.pdf(x3,df=dof)
ax=plt.figure().add_subplot(111); ax.plot(x3,y3,color='brown')
ax.fill_between(x1,y1,0,alpha=0.5,color='brown',label='Rejection Region')
ax.fill_between(x2,y2,0,alpha=0.5,color='brown'); ax.set(xlabel='X',ylabel='P(X)')
ax.axvline(x=tstatistics,ls='--',c='b',label='T statistics = {:.5f}'.format(tstatistics))
ax.set(xlim=[-4,4],title="T Distribution (degrees of freedom = {:.0f})".format(dof))
plt.legend(); plt.savefig('ttest2.pdf',dpi=72,bbox_inches='tight'); plt.show()
```

Output:

```
__________Two-sample T-test for unpaired data__________
```

```
Sample 1:
     Number of Observations = 5
     Mean = 44.00000
     Standard Deviation = 10.09950
     Standard Error of the Mean = 4.51664

Sample 2:
     Number of Observations = 7
     Mean = 57.00000
     Standard Deviation = 11.50155
     Standard Error of the Mean = 4.34718

T statistics: -1.85272

Critical values are -2.22814, 2.22814
Fail to reject the Null hypothesis.

p-value: 0.09363
Fail to reject the Null hypothesis.

Standard Error: 6.41885
Confidence Interval: [-27.30208861   1.30208861]
```



3. Samples of two types of electric bulbs were tested for length of life and the following data were obtained

	Type I	Type II
No. of Samples:	8	7
Mean (hours):	1134	1024
SD (hours):	35	40

Test at 5 percent level, whether the difference in sample mean is significant.

Null hypothesis: $H_0 : \mu 1 = \mu 2$
Alternative hypothesis: $H_1 : \mu 1 \neq \mu 2$

Type I:
 Mean = 1134

Variance = 1225
Standard Deviation = 35

Type II:
Mean = 1024
Variance = 1600
Standard Deviation = 40

Calculation of $s1/s2$:

$$\frac{s_1}{s_2} = \frac{35}{40} = 0.875$$

Test statistic:

$$T = \frac{\bar{x}_1 - \bar{x}_2}{\sqrt{s_1^2/n1 + s_2^2/n2}}$$

where $n1$ and $n2$ are the sample sizes, $\bar{x}_1$ and $\bar{x}_2$ are the sample means, and s_1^2 and s_2^2 are the sample variances.
If equal variances are assumed $(0.5 < s1/s2 < 2)$, then the formula reduces to:

$$T = \frac{\bar{x}_1 - \bar{x}_2}{s_p\sqrt{1/n1 + 1/n2}}$$

where

$$s_p^2 = \frac{(n1-1)s_1^2 + (n2-1)s_2^2}{n1 + n2 - 2}$$

Calculation of s_p:

$$s_p = \sqrt{\frac{(8-1) \times 35^2 + (7-1) \times 40^2}{8 + 7 - 2}} = 37.39087$$

Calculation of T statistic:

$$T = \frac{1134 - 1024}{37.39087 \times \sqrt{1/8 + 1/7}} = 5.68428$$

Calculation of Standard Error:

$$S.E = s_p\sqrt{1/n1 + 1/n2} = 37.39087 \times \sqrt{1/8 + 1/7} = 19.35161$$