t-tests

May 26, 2024

[62]:
```
import scipy.stats as stats
import pandas as pd
import numpy as np
import statsmodels.api as sm
import pylab
```

[6]:
```
df = pd.read_csv('/content/titanic.csv')
```

[7]:
```
df.head()
```

[7]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th…	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S

[8]:
```
df.shape
```

[8]:
```
(891, 12)
```

1 Single Sample t-test

- H0 : Population_mean = 35
- H1 : Population_mean <=35
- Alpha : 0.05

[11]:
```
population_age = df['Age']
```

[13]:
```
population_age.dropna()
```

[13]:
```
0      22.0
1      38.0
2      26.0
3      35.0
4      35.0
       …
885    39.0
886    27.0
887    19.0
889    26.0
890    32.0
Name: Age, Length: 714, dtype: float64
```

[23]:
```
sample_age = population_age.dropna()
sample_age1 = sample_age.sample(30)
```

[24]:
```
sample_age1
```

[24]:
```
182     9.0
510    29.0
265    36.0
211    35.0
79     30.0
735    28.5
323    22.0
373    22.0
848    28.0
398    23.0
110    47.0
407     3.0
551    27.0
746    16.0
615    24.0
56     21.0
782    29.0
267    25.0
119     2.0
688    18.0
```

```
429    32.0
272    41.0
63      4.0
315    26.0
289    22.0
131    20.0
663    36.0
115    21.0
439    31.0
400    39.0
Name: Age, dtype: float64
```

[25]:
```
sample_age1.shape
```

[25]:
```
(30,)
```

1.1 Assumptions to perform Single-Sample-t-test

- Normality : Since we have a sample size of 30 a/c to central limit theorem we can assume it to be normal
- Independence of sample
- Random Sampling
- Unknown Population Std. deviation

1.1.1 Performing Shapiro-Wilk test

- This test is used to check normality

[26]:
```
from scipy.stats import shapiro
```

[31]:
```
pop_mean = 35
shapiro(sample_age1)
```

[31]:
```
ShapiroResult(statistic=0.9610083699226379, pvalue=0.3286316990852356)
```

1.1.2 Since the p-value is greater than 0.05, therefore the sample data is normally distributed

1.1.3 Basically it was not necessary to perform this test, but still I did it.

[32]:
```
t_statistic , p_value = stats.ttest_1samp(sample_age1,pop_mean)
```

[33]:
```
print('t-statistic: ',t_statistic)
```

```
t-statistic:  -5.136536349467585
```

[36]:
```
print('p_value: ',p_value/2)
```

```
p_value:  8.677479419803319e-06
```

1.1.4 Since it is one tailed , we divide p-value by 2

[37]:
```
alpha = 0.05
if p_value < alpha:
  print("Reject Null Hypothesis")
else:
  print("Accept Null Hypothesis")
```

```
Reject Null Hypothesis
```

1.1.5 Therefore our null hypothesis of mean age being 35 is wrong

- i.e. the population mean age is less than 35

1.1.6 Checking whether our process is right or wrong

[40]:
```
df['Age'].mean()
```

[40]:
```
29.69911764705882
```

1.1.7 There mean age is actually less than 35 i.e 29.70 , which proves our test is right

2 Independent two-sample test

2.1 Assumptions for the test

- Independence of obeservations
- Normality
- Equal Variances : The variances of two independent groups should be equal
- Random Sampling

- H0 : The average age of male and female has no difference
- H1 : The average age of male is significantly higher than average age of female.
- Alpha = 0.05

[46]:
```
df[['Age','Sex']].dropna()
```

[46]:

	Age	Sex
0	22.0	male
1	38.0	female
2	26.0	female
3	35.0	female
4	35.0	male
..	…	…
885	39.0	female
886	27.0	male
887	19.0	female

```
889  26.0    male
890  32.0    male

[714 rows x 2 columns]
```

[47]:
```
male_pop = df[df['Sex'] == 'male']['Age'].dropna()
```

[48]:
```
female_pop = df[df['Sex'] == 'female']['Age'].dropna()
```

[51]:
```
male_pop.head()
```

[51]:
```
0     22.0
4     35.0
6     54.0
7      2.0
12    20.0
Name: Age, dtype: float64
```

[52]:
```
female_pop.head()
```

[52]:
```
1    38.0
2    26.0
3    35.0
8    27.0
9    14.0
Name: Age, dtype: float64
```

[55]:
```
print(female_pop.shape)
print(male_pop.shape)
```

```
(261,)
(453,)
```

[77]:
```
sample_male = male_pop.sample(30)
sample_female = female_pop.sample(30)
```

[78]:
```
sample_male.shape
```

[78]:
```
(30,)
```

[79]:
```
sample_female.shape
```

[79]:
```
(30,)
```

2.1.1 Shapiro test to check for normality

[80]:
```
print(shapiro(sample_male))
print(shapiro(sample_female))
```

```
ShapiroResult(statistic=0.9784983992576599, pvalue=0.7843736410140991)
ShapiroResult(statistic=0.9559767842292786, pvalue=0.24363601207733154)
```

2.1.2 since p_value of both sample is greater than 0.05, they are normal

2.1.3 Levene test to check the whether both groups have equal variances

[81]:
```
from scipy.stats import levene
```

[82]:
```
levene(sample_male , sample_female)
```

[82]:
```
LeveneResult(statistic=2.9567637031766094, pvalue=0.09085181613089263)
```

2.1.4 Since p-value > 0.05 , we donot reject the null hypothesis of the variances of both group being same.

3 F-test to check equality of variances

[83]:
```
variance1 = np.var(sample_male, ddof=1)
variance2 = np.var(sample_female, ddof=1)
```

[84]:
```
f_value = variance1 / variance2
```

[85]:
```
df1 = len(sample_male) - 1
df2 = len(sample_female) - 1
```

[86]:
```
p_value = stats.f.cdf(f_value, df1, df2)
```

[87]:
```
print('Degree of freedom 1:',df1)
print('Degree of freedom 2:',df2)
print("F-statistic:", f_value)
print("p-value:", p_value)
```

```
Degree of freedom 1: 29
Degree of freedom 2: 29
F-statistic: 1.7173672839111591
p-value: 0.9243640868083862
```

3.0.1 Since p-value is > 0.05 we accept the null hypothesis

[88]:
```
t_statistic , p_value = stats.ttest_ind(sample_male,sample_female)
```

[89]:
```
print('t-statistic: ',t_statistic)
print('p_value: ',p_value/2)
```

```
t-statistic:  2.44583001377923
p_value:  0.008753503445615367
```

[90]:
```
alpha = 0.05
if p_value < alpha:
  print("Reject Null Hypothesis")
else:
  print("Accept Null Hypothesis")
```

```
Reject Null Hypothesis
```

3.0.2 Therefore we have to accept the null hypothesis.

3.0.3 Welch t-test (if we donot have equal varinaces)

- Same as two-sample t test , only we have to set the parameter, equal_var=False

4 Paired two-sample t-tests

4.0.1 Assumptions

- Paired Observations
- Normality : The difference between paired observations should be normal
- Independence of pairs
- H0 : There is no differnce of wieghts before and after treatment
- H1 : There is a significant difference
- Alpha : 0.05

[96]:
```
weight_before = np.array([80, 92, 75, 68, 85, 78, 73, 90, 70, 88, 76, 84, 82,␣
↪77, 91])
weight_after = np.array([78, 93, 81, 67, 88, 76, 74, 91, 69, 88, 77, 81, 80,␣
↪79, 88])
```

[97]:
```
weight_differences = weight_after - weight_before
```

[99]:
```
shapiro_test = stats.shapiro(weight_differences)
print("Shapiro-Wilk test:", shapiro_test)
```

```
Shapiro-Wilk test: ShapiroResult(statistic=0.9220570921897888,
pvalue=0.20704729855060577)
```

4.0.2 We accept the null hypothesis of difference of paired obeseravtions is normal

[101]:
```
mean_diff = np.mean(weight_differences)
std_diff = np.std(weight_differences, ddof=1)
```

[102]:
```
n = len(weight_differences)
t_statistic = mean_diff / (std_diff / np.sqrt(n))
df = n - 1
```

[103]:
```
t_statistic
```

[103]: 0.10482848367219182

[104]:
```
alpha = 0.05
p_value = stats.t.cdf(t_statistic, df)
```

[105]:
```
p_value
```

[105]: 0.5410005146857456

4.0.3 Since p > alpha we accept the null hypothesis

4.0.4 Verification

[107]:
```
weight_before.mean()
```

[107]: 80.6

[109]:
```
weight_after.mean()
```

[109]: 80.66666666666667

[ ]:
```

```