Student t-test

1. One sample t-test
2. Two sample t-test
   A. Un-paired or independent t-test
   B. Paired or correlation/dependent t-test

One sample t-test (Comparison of one categorical and one numeric variable)

Test a sample with known standard value.\

Assumptions

- Observations in a sample are independent and and identically distributed.
- Observations in a sample are normaly distributed.

Interpretation

> **HO:** The mean of the sample is equal to known value.\ **H1:** The mean of the sample is unequal to known value.\

- Python Code is Here:

In [ ]:
```
# import libraries
import pandas as pd
import seaborn as sns
from scipy.stats import ttest_1samp
# load dataset
df = sns.load_dataset('titanic')
df.head()
```

Out[ ]:

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	deck
0	0	3	male	22.0	1	0	7.2500	S	Third	man	True	NaN
1	1	1	female	38.0	1	0	71.2833	C	First	woman	False	C
2	1	3	female	26.0	0	0	7.9250	S	Third	woman	False	NaN
3	1	1	female	35.0	1	0	53.1000	S	First	woman	False	C
4	0	3	male	35.0	0	0	8.0500	S	Third	man	True	NaN

In [ ]:
```
# subsetting the dataset
df1 = df[['sex','age','fare','class']]
df1.head()
```

Out[ ]:

	sex	age	fare	class
0	male	22.0	7.2500	Third
1	female	38.0	71.2833	First
2	female	26.0	7.9250	Third
3	female	35.0	53.1000	First
4	male	35.0	8.0500	Third

In [ ]:
```
# data description
df1.describe()
```

Out[ ]:

	age	fare

	age	fare
count	714.000000	891.000000
mean	29.699118	32.204208
std	14.526497	49.693429
min	0.420000	0.000000
25%	20.125000	7.910400
50%	28.000000	14.454200
75%	38.000000	31.000000

In [ ]:
```
# Check the age and compare with the known value of 45 years
ttest_1samp(df1['age'], 45)
stat, p = ttest_1samp(df1['age'], 45)

print('stat=%0.3f ,p=%0.3f' % (stat, p))
# addind conditinal arguments for ease
if p>0.05:
    print('Probably the same Distribution')
else:
    print('Probably different Distribution')
# This shows nan values..... as we can see that std value is 14 means that data is hi
# The normal SD value is +/- 2.
```

```
stat=nan ,p=nan
Probably different Distribution
```

Two sample t-test

Independent/Unpaired student's t-test

Assumptions

- Observations in each sample are independent and identically distributed.
- Observations in each sample are normaly distributed.
- Observations in each sample have the same variance.

Interpretation

> **HO:** The means of the sample are equal to known value.\ **H1:** The means of the sample are unequal to known value.

- Python Code is Here:

In [ ]:
```
# we will compare age and fare of male and female passengers
# splitting dataset

df1_male = df1.loc[df1['sex']=='male']
df1_female = df1.loc[df1['sex']=='female']

# import library
from scipy.stats import ttest_ind
stat, p = ttest_ind(df1_male['age'], df1_female['age'])
print('stat=%.3f, p=%.3f' % (stat, p))

# addind conditinal arguments for ease
if p>0.05:
    print('Probably the same Distributions')
else:
    print('Probably different Distributions')
```

```
stat=nan, p=nan
Probably different Distributions
```

In [ ]:
```
df1_female.describe()
```

Out[ ]:

	age	fare

	age	fare
count	261.000000	314.000000
mean	27.915709	44.479818
std	14.110146	57.997698
min	0.750000	6.750000
25%	18.000000	12.071875
50%	27.000000	23.000000
75%	37.000000	55.000000

In [ ]:
```
df1_male.describe()
```

Out[ ]:

	age	fare
count	453.000000	577.000000
mean	30.726645	25.523893
std	14.678201	43.138263
min	0.420000	0.000000
25%	21.000000	7.895800
50%	29.000000	10.500000
75%	39.000000	26.550000
max	80.000000	512.329200

Two sample t-test

Paired/Relational student's t-test (Comparison of two categorical and one numeric variable)\
Tests whether the means of two paired samples are significantly different.\ **Assumptions**

- Observations in each sample are independent and identically distributed.
- Observations in each sample are normaly distributed.
- Observations in each sample have the same variance.
- Observations across each sample are paired

Interpretation

> **HO:** The means of the sample are equal.\ **H1:** The means of the sample are unequal.

- Python Code is Here:

In [ ]:
```
# select only male's data

df1_male = df1.loc[df1['sex']=='male']
df1_male.tail()
```

Out[ ]:

	sex	age	fare	class
883	male	28.0	10.50	Second
884	male	25.0	7.05	Third
886	male	27.0	13.00	Second
889	male	26.0	30.00	First
890	male	32.0	7.75	Third

In [ ]:
```
# select only two classes
df1_male_first = df1_male.loc[df1_male['class']=='First']
df1_male_second = df1_male.loc[df1_male['class']=='Second']
df1_male_third = df1_male.loc[df1_male['class']=='Third']
```

In [ ]:
```
df1_male_first.head()
```

Out[ ]:

	sex	age	fare	class
6	male	54.0	51.8625	First
23	male	28.0	35.5000	First
27	male	19.0	263.0000	First
30	male	40.0	27.7208	First
34	male	28.0	82.1708	First

In [ ]:
```
df1_male_second.head()
```

Out[ ]:

	sex	age	fare	class
17	male	NaN	13.0	Second
20	male	35.0	26.0	Second
21	male	34.0	13.0	Second
33	male	66.0	10.5	Second
70	male	32.0	10.5	Second

In [ ]:
```
df1_male_third.head()
```

Out[ ]:

	sex	age	fare	class
0	male	22.0	7.2500	Third
4	male	35.0	8.0500	Third
5	male	NaN	8.4583	Third
7	male	2.0	21.0750	Third
12	male	20.0	8.0500	Third

In [ ]:
```
# import library
from scipy.stats import ttest_rel
# apply test to compare class-1 and class-3
ttest_rel(df1_male_first['age'], df1_male_third['age'])
# There is unequl length of compared classes
# In order to compare any of two or more arrays they should have equal instances
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-29-7ea5237ec650> in <module>
      2 from scipy.stats import ttest_rel
      3 # apply test to compare class-1 and class-3
----> 4 ttest_rel(df1_male_first['age'], df1_male_third['age'])
      5 # There is unequl length of compared classes
      6 # In order to compare any of two or more arrays they should have equal instan
ces

c:\Users\kalee\anaconda3\lib\site-packages\scipy\stats\stats.py in ttest_rel(a, b, ax
is, nan_policy, alternative)
   5889     nb = _get_len(b, axis, "second argument")
   5890     if na != nb:
-> 5891         raise ValueError('unequal length arrays')
   5892
   5893     if na == 0:
```

In [ ]:
```
print('shape of first class=', (df1_male_first.shape))
print('shape of second class=', (df1_male_second.shape))
print('shape of third class=', (df1_male_third.shape))
```

```
shape of first class= (122, 4)
shape of second class= (108, 4)
shape of third class= (347, 4)
```

In [ ]:
```
# to make instances equal we should follow this step
df1_1st = df1_male_first.sample(n=100)
df1_2nd = df1_male_second.sample(n=100)
df1_3rd = df1_male_third.sample(n=100)
print('shape of first class=', (df1_1st.shape))
print('shape of second class=', (df1_2nd.shape))
print('shape of third class=', (df1_3rd.shape))
```

```
shape of first class= (100, 4)
shape of second class= (100, 4)
shape of third class= (100, 4)
```

In [ ]:
```
# import library
from scipy.stats import ttest_rel
# apply test to compare class-1 and class-3
stat, p = ttest_rel(df1_1st['age'], df1_3rd['age'])
print('stat=%.3f, p=%.3f' % (stat, p))

# addind conditinal arguments for ease
if p>0.05:
    print('Probably the same Distributions')
else:
    print('Probably different Distributions')
```

```
stat=nan, p=nan
Probably different Distributions
```

In [ ]:
```

```