Chi-Square Test:

The test is applied when you have two categorical variables from a single population. It is used to determine significant association between the two variables.

```
import scipy.stats as stats
import pandas as pd
import numpy as np
import seaborn as sns
```

```
dataset = sns.load_dataset('tips')
```

```
dataset.head()
```

	total_bill	tip	sex	smoker	day	time	size
0	16.99	1.01	Female	No	Sun	Dinner	2
1	10.34	1.66	Male	No	Sun	Dinner	3
2	21.01	3.50	Male	No	Sun	Dinner	3
3	23.68	3.31	Male	No	Sun	Dinner	2
4	24.59	3.61	Female	No	Sun	Dinner	4

```
dataset_table = pd.crosstab(dataset['sex'],dataset['smoker'])
```

```
print(dataset_table)
```

```
smoker  Yes  No
sex
Male     60  97
Female   33  54
```

```
#observed values
observed_values = dataset_table.values
print('Observed values:\n',observed_values)
```

```
Observed values:
 [[60 97]
 [33 54]]
```

```
val = stats.chi2_contingency(dataset_table)
```

```
val
```

```
(0.0, 1.0, 1, array([[59.84016393, 97.15983607],
        [33.15983607, 53.84016393]]))
```

```
Expected_values = val[3]
```

```
Expected_values
```

```
array([[59.84016393, 97.15983607],
       [33.15983607, 53.84016393]])
```

```
no_of_rows = len(dataset_table.iloc[0:2,0])
no_of_columns = len(dataset_table.iloc[0,0:2])
dof = (no_of_rows-1)*(no_of_columns-1)
print("Degree of Freedom:",dof)
alpha = 0.05
```

```
Degree of Freedom: 1
```

```
from scipy.stats import chi2
chi_square = sum([(o-e)**2./e for o,e in zip(observed_values,Expected_values)])
chi_square_statistic = chi_square[0]+chi_square[1]
print("chi square statistic:",chi_square_statistic)
```

```
chi square statistic: 0.001934818536627623
```

```
critical_value = chi2.ppf(q=1-alpha,df=dof)
print('critical value:',critical_value)
```

```
critical value: 3.841458820694124
```

```
#p value
p_value = 1-chi2.cdf(x=chi_square_statistic,df=dof)
print('p-value:',p_value)
print('Significance level:',alpha)
print('Degree of Freedom:',dof)
```

```
p-value: 0.964915107315732
Significance level: 0.05
Degree of Freedom: 1
```

```
if chi_square_statistic >= critical_value:
  print("Reject H0, There is a relationship betweeen 2 categorical variables")
else:
  print("Retain H0, There is no relationship between 2 categorical variables")

if p_value <= alpha:
  print("Reject H0, There is a relationship between 2 categorical variables")
else:
  print("Retain H0, There is no relationship between 2 categorical variables")
```

```
Retain H0, There is no relationship between 2 categorical variables
Retain H0, There is no relationship between 2 categorical variables
```

T Test

A t-test is a type of inferential statistic which is used to determine if there is significance differemnce between the means of two groups which may be related in certain features

T-test has 2 types: 1.one sampled t-test 2.two-sampled t-test

One-sampled T-test with Python

The test will tell us whether means of the sample and the populationare different

```
ages = [10,20,32,34,45,56,55,54,66,53,51,33,32,35,36,65,23,25,28,29,30]
```

```
len(ages)
```

```
21
```

```
from statsmodels.stats.weightstats import ztest
```

```
import numpy as np
ages_mean = np.mean(ages)
print(ages_mean)
```

```
38.666666666666664
```

```
#lets take a sample

sample_size = 8
age_sample = np.random.choice(ages,sample_size)
```

```
age_sample
```

```
array([54, 29, 65, 32, 66, 30, 36, 23])
```

```
np.mean(age_sample)
```

```
41.875
```

```
from scipy.stats import ttest_1samp
```

```
ttest, p_value = ttest_1samp(age_sample,30)
```

```
print(p_value)
```

```
0.09102724606475271
```

```
if p_value < 0.05:
  print("we are rejecting null hypothesis")
else:
  print("we fail to reject the null hypothesis")
```

```
we fail to reject the null hypothesis
```

Some more Examples

Consider the age of students in acollege and in class A

```
import numpy as np
import pandas as pd
import scipy.stats as stats
import math
np.random.seed(6)
school_ages = stats.poisson.rvs(loc=18,mu=35,size=1500)
classA_ages = stats.poisson.rvs(loc=18,mu=35,size=25)
```

```
classA_ages.mean()
```

```
53.52
```

```
_, p_value = stats.ttest_1samp(a=classA_ages,popmean=school_ages.mean())
```

```
p_value
```

```
0.8461607039964159
```

```
school_ages.mean()
```

```
53.303333333333335
```

```
if p_value < 0.05:    #alpha value is 0.05 or 5%
  print("we are rejecting null hypothesis")
else:
  print("we are fail to reject the null hypothesis")
```

```
we are fail to reject the null hypothesis
```

Two-sample T-test with Python

The independent samples t Test or 2-sample t-test compares the means of two independent groups inorder to determine whether there is statistical evidence that the associated population means are significantly different. The independent samples t-test is a parametric test. This test is also known as Independent t Test

```
np.random.seed(12)
ClassB_ages = stats.poisson.rvs(loc=18,mu=33,size=60)
ClassB_ages.mean()
```

```
50.63333333333333
```

```
_,p_value = stats.ttest_ind(a=classA_ages,b=ClassB_ages,equal_var=False)
```

```
p_value
```

```
0.037596488729299896
```

```
if p_value < 0.05:
  print("we are rejecting null hypothesis")
else:
  print("we are accepting null hypothesis")
```

```
    we are rejecting null hypothesis
```

Paired T-test with Python

When you want to check how different samples from the same group are, you can go for a paired t-test

```
weight1 = [25,30,28,34,26,29,30,26,28,32,31,45,23,46,27]
weight2 = weight1+stats.norm.rvs(scale=5,loc=-1.25,size=15)
```

```
print(weight1)
print(weight2)
```

```
    [25, 30, 28, 34, 26, 29, 30, 26, 28, 32, 31, 45, 23, 46, 27]
    [30.57926457 34.91022437 29.00444617 29.54295091 17.86201983 32.57873174
     22.3299827  18.3771395  34.36420881 38.05941216 29.93827982 42.519014
     18.42851213 46.50667769 23.32984284]
```

```
weight_df = pd.DataFrame({'weight_10':np.array(weight1),
                          'weight_20':np.array(weight2),
                          'weight_change':np.array(weight2)-np.array(weight1)})
```

```
weight_df
```

	weight_10	weight_20	weight_change
0	25	30.579265	5.579265
1	30	34.910224	4.910224
2	28	29.004446	1.004446
3	34	29.542951	-4.457049
4	26	17.862020	-8.137980
5	29	32.578732	3.578732
6	30	22.329983	-7.670017
7	26	18.377139	-7.622861
8	28	34.364209	6.364209
9	32	38.059412	6.059412
10	31	29.938280	-1.061720
11	45	42.519014	-2.480986
12	23	18.428512	-4.571488
13	46	46.506678	0.506678
14	27	23.329843	-3.670157

```
_,p_value = stats.ttest_rel(a=weight1,b=weight2)
```

```
print(p_value)
```

```
    0.5732936534411279
```

```
if p_value < 0.05:
  print("we are rejecting the null hypothesis")
else:
  print("we are accepting the null hypothesis")
```

```
    we are accepting the null hypothesis
```

Correlation

```
import seaborn as sns
df = sns.load_dataset('iris')
```

```
df.shape
```

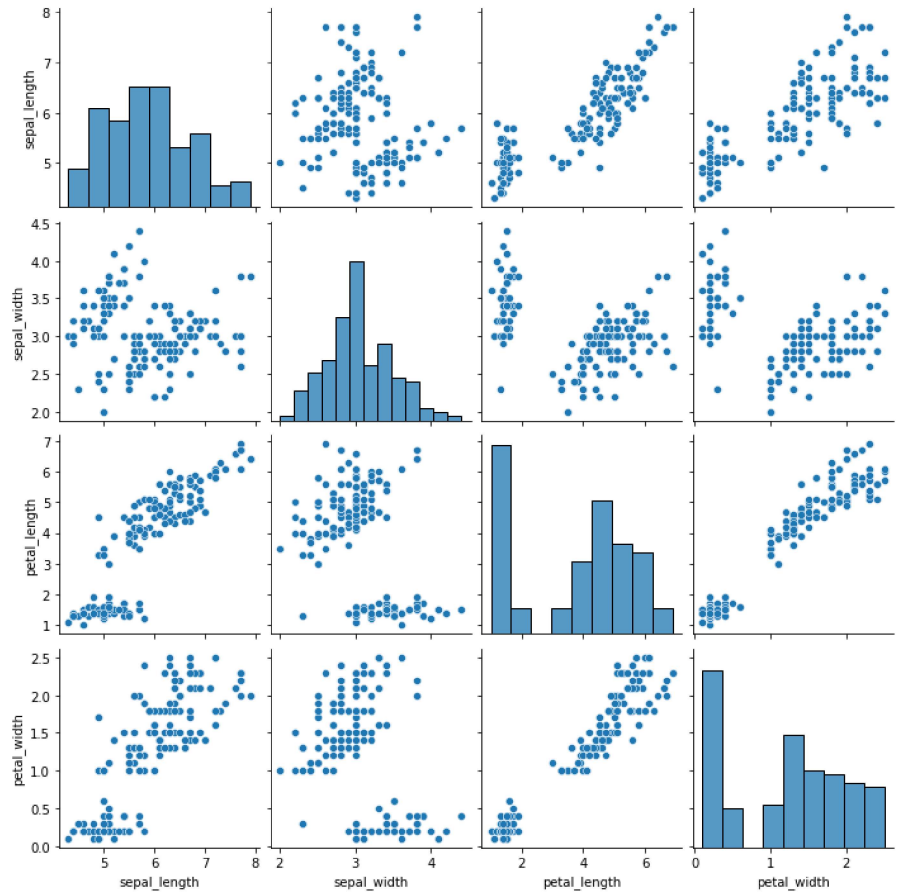
```
(150, 5)
```

```
df.corr()
```

	sepal_length	sepal_width	petal_length	petal_width
sepal_length	1.000000	-0.117570	0.871754	0.817941
sepal_width	-0.117570	1.000000	-0.428440	-0.366126
petal_length	0.871754	-0.428440	1.000000	0.962865
petal_width	0.817941	-0.366126	0.962865	1.000000

```
sns.pairplot(df)
```

```
<seaborn.axisgrid.PairGrid at 0x7f8435e04250>
```



Anova Test(F-Test)

The t-test works well when dealing with two groups, but sometimes we want to compare more than two groups at the same time. For example, if we wanted to test whether petal_width age differs based on some categorical variable like species, we have to compare the means of each level or group the varible

One Way F-test(Anova)

It tells whether two or mnore groups are similar or not based on their mean similarity and f-score. Example: there are 3 digfferent category of this flowers and their petal width and need to check all three groups are similar ar not.

```
import seaborn as sns
df1 = sns.load_dataset('iris')
```

```
df1.head()
```

	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	setosa
1	4.9	3.0	1.4	0.2	setosa
2	4.7	3.2	1.3	0.2	setosa
3	4.6	3.1	1.5	0.2	setosa
4	5.0	3.6	1.4	0.2	setosa

```
df_anova = df1[['petal_width','species']]
```

```
grps = pd.unique(df_anova.species.values)
```

```
grps
```

```
array(['setosa', 'versicolor', 'virginica'], dtype=object)
```

```
d_data = {grp:df_anova['petal_width'][df_anova.species == grp] for grp in grps}
```

```
d_data
```

```
142    1.9
143    2.3
144    2.5
145    2.3
146    1.9
147    2.0
148    2.3
149    1.8
Name: petal width, dtype: float64
```

```
F, p = stats.f_oneway(d_data['setosa'], d_data['versicolor'], d_data['virginica'])
```

```
print(p)
```

```
4.169445839443116e-85
```

```
if p < 0.05:
  print("reject null hypothesis")
else:
  print("accept null hyppothesis")
```

```
reject null hypothesis
```

```
#imports
import math
import numpy as np
from numpy.random import randn
from statsmodels.stats.weightstats import ztest

#Generate a random array of 50 numbers having mean 110 and sd 15
#similar to the IQ scores data we assume above
mean_iq = 110
sd_iq = 15/math.sqrt(50)
alpha = 0.05
null_mean = 100
data = sd_iq * randn(50) + mean_iq



#print mean and s
print('mean=%.2f stdv=%.2f' % (np.mean(data), np.std(data)))
```

```
mean=109.61 stdv=2.22
```

```
# now we perform the test. In this function , we passed data, in the valur parameter
# we passed mean value in the null hypothesis, in alternative hypothesis we check whether the mean is larger

ztest_Score, p_value = ztest(data,value = null_mean, alternative='larger')
# the function outputs a p_value and z-score corresponding to that value, we compare the
# p-value with alpha, if it is greater than alpha then we do not null hypothesis
# else we reject it.

if (p_value < alpha):
  print('Reject the null Hyppothesis')
else:
  print("Accept the null hypothesis")
```

```
Reject the null Hyppothesis
```

✓ 0s completed at 10:19 PM