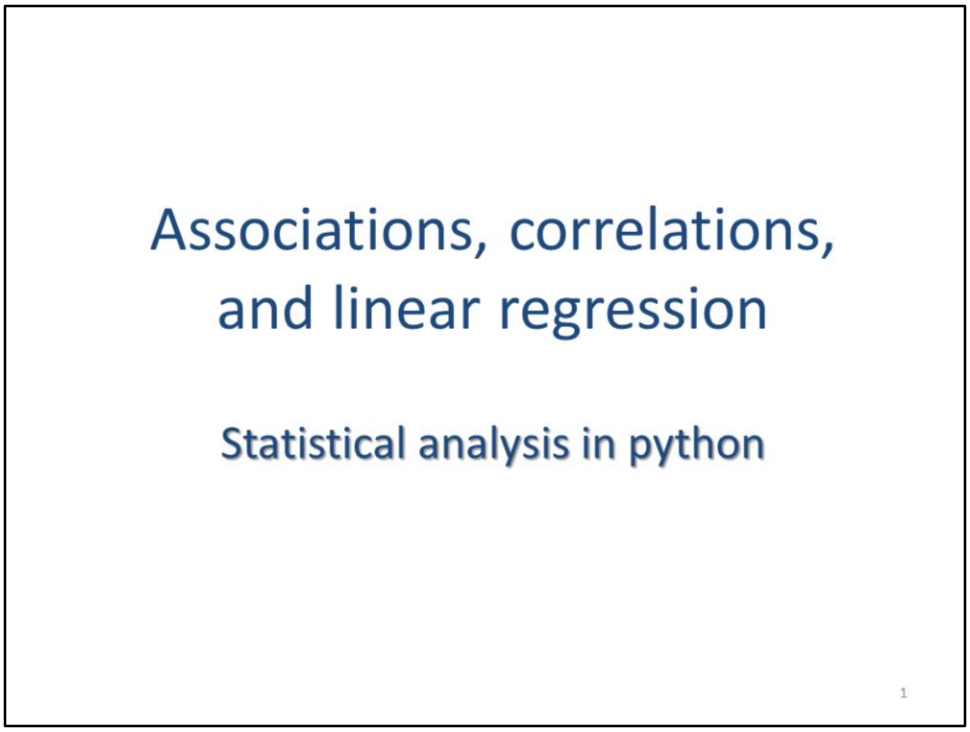
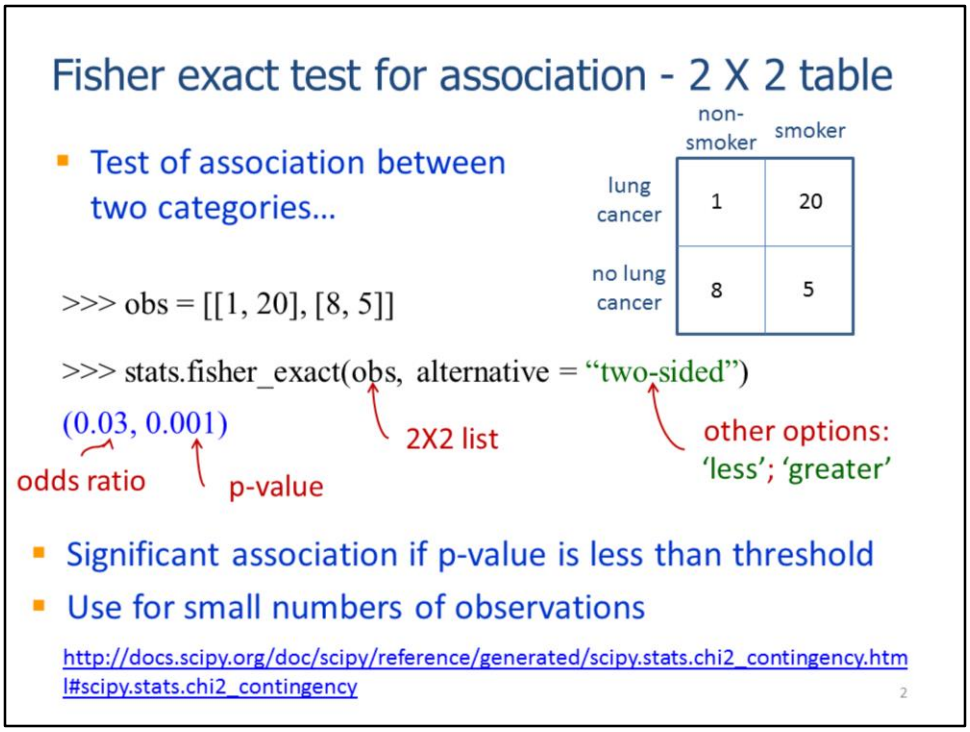


This video will discuss some scipy tools that assess associations among categorical data, correlations among continuous datasets, and linear regressions.

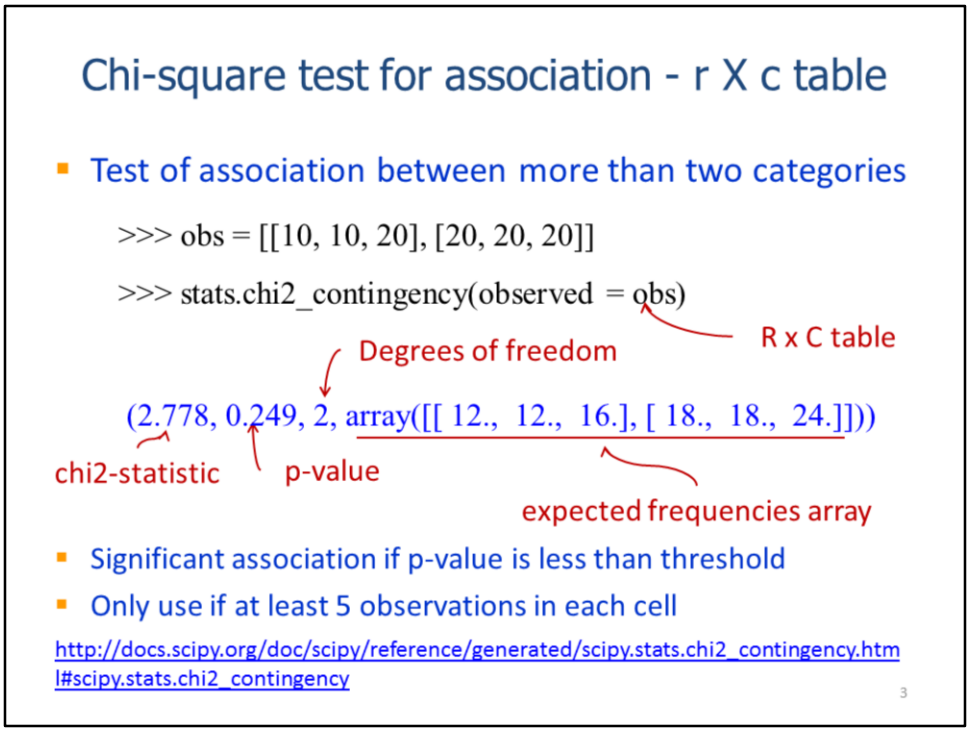


The Fisher exact test checks for associations between two different categories of data within a dataset. For example, this test could be used to check for an association between smoking and lung cancer.

For the stats module method, the data must be put in a 2 level list – note that the sub-lists correspond to the rows in the example table. Creating a diagram of the table is helpful for setting up the input list and interpreting the test results.

The **fisher_exact** method can test associations for a 2x2 list.

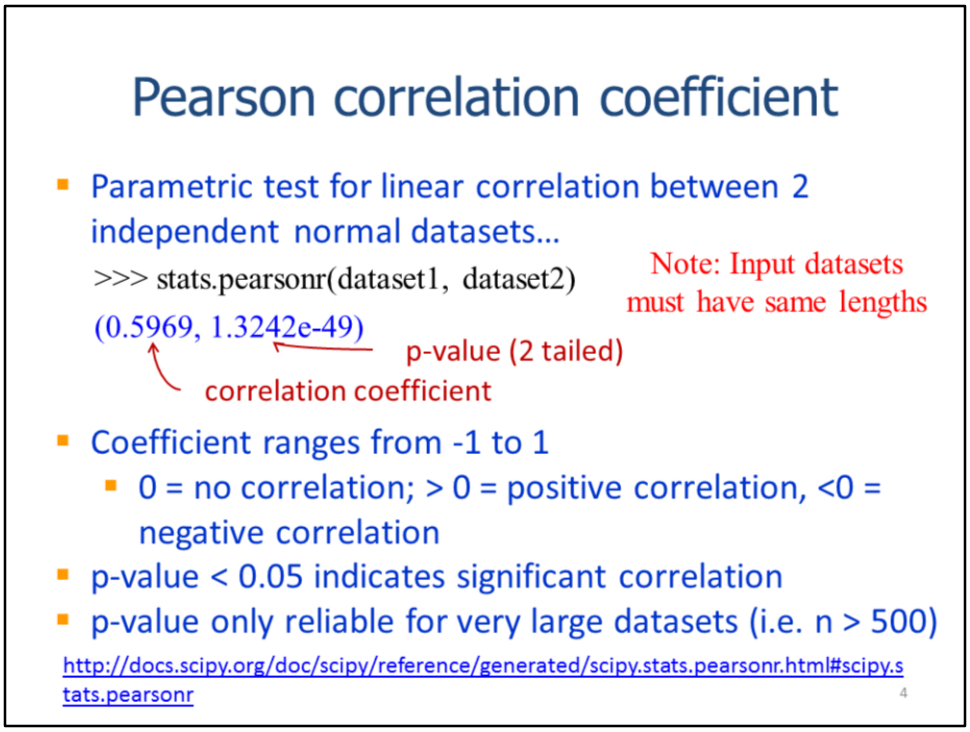
A significant p-value indicates that there is an association between the two categories. In this example, the small p-value indicates a strong association between smoking and lung cancer.



The chi squared test checks for associations among 3 or more categories of data. A table diagram illustrating the hypothesized associations is helpful when setting up the test in a script.

The **chi2_contingency** method performs the chi-squared test. The input list should have two levels with each sub-list corresponding to a row in the associations table. In this example, the test is performed for a table that has 2 rows and 3 columns.

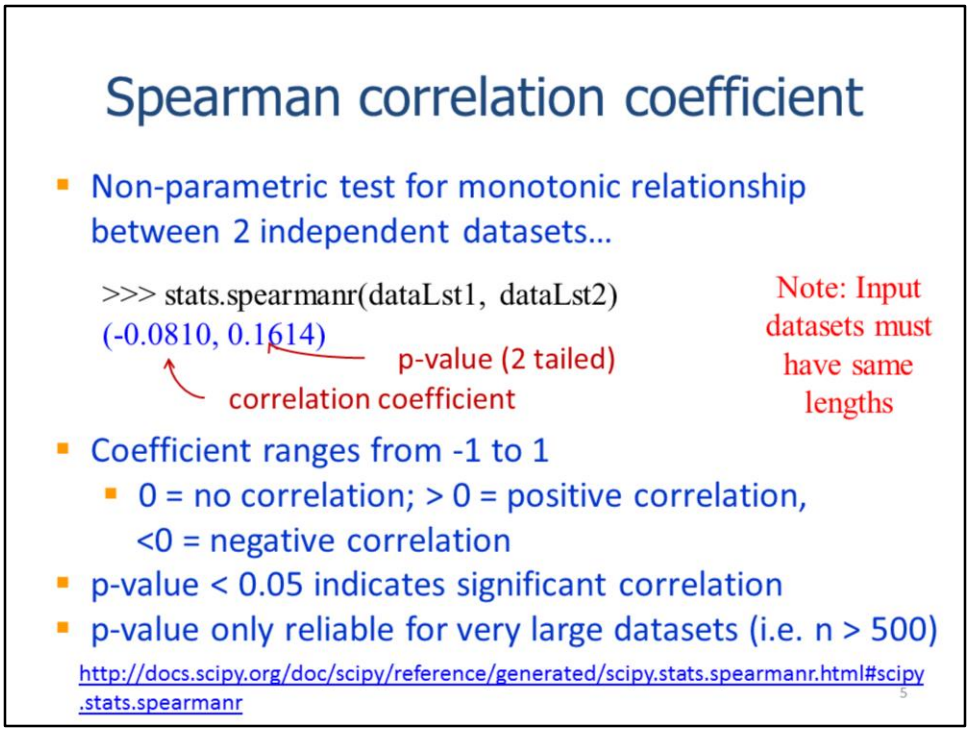
A significant p-value indicates there is a statistically significant association among the categories. For the test results to be valid, there should be a minimum of 5 observations for each cell in the associations table.



The Pearson correlation coefficient tests for linear correlations between two independent datasets that are normally distributed.

For the **pearsonr** method, the two input datasets must have the same number of values.

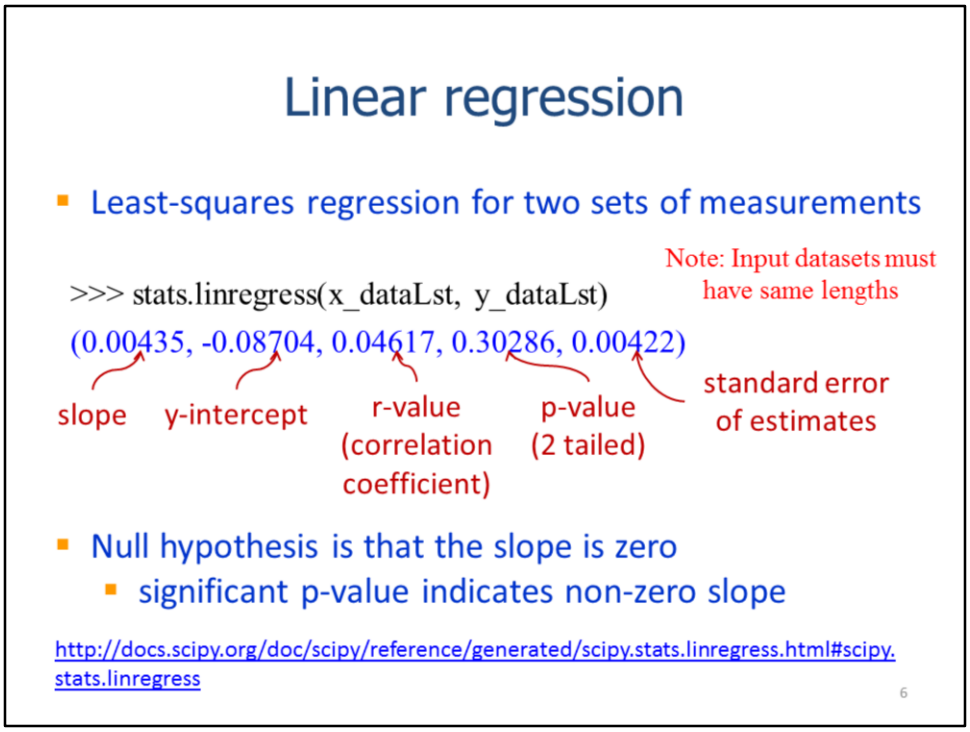
The test results range from -1 to 1. Negative coefficients indicate the data are inversely correlated; positive coefficients indicate data that are positively correlated. Values that are near zero indicate there is little or no correlation. A significant p-value indicates the correlation is significant; however, according to scipy's documentation, the p-value is only reliable for very large datasets. Generally, it is more practical to gauge of significance by how far the coefficient is from zero - the further it is from zero, the more likely the it is to be significant.



The spearman correlation coefficient also tests for monotonic relationships between two independent datasets and is suitable for data that are not normally distributed (i.e. non-parametric). Note that a monotonic relationship is one that is either always increasing or always decreasing.

The **spearmanr** method requires both datasets to have the same number of values.

The coefficient values range from -1 to 1. Values near zero indicate no correlation; negative and positive coefficients indicate negative and positive correlations, respectively. The further the coefficient is from zero, the stronger and more significant the correlation.



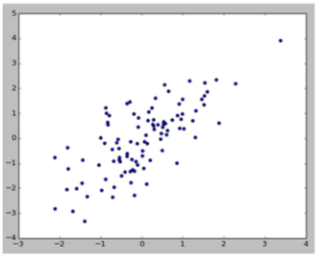
Linear regression analysis is used to assess the correlation and quantify the relationship between two datasets.

The **linregress** method calculates a linear least-squared regression. The input datasets must contain the same number of values. The method returns the **slope** and **y-intercept** of the regression equation as well as the **r-value**, the **p-value**, and the **standard error**. Note that an **r-squared** value is typically used to indicate how well the regression equation fits the data – the better the data fit the regression line, the higher the r-squared. The r-squared can be calculated by squaring the **r-value** provided by the **linregress** method.

A significant p-value indicates that the slope of the relationship is not zero. However, when working with large datasets a linear regression may give a significant p-value even when the slope is near zero – in such cases, the relationship may be too weak to be of practical importance.

Example script: linear regression

```
import matplotlib.pyplot as figure
import scipy.stats as stats
xData = stats.norm.rvs(0, 1, 100)
yData = xData+stats.norm.rvs(0, 1, 100)
figure.scatter(xData , yData)
slope, intercept, r, p, stderr = stats.linregress(xData, yData)
y = slope * xData + intercept
```



7

The next two slides will show an example of a script that calculates a linear regression and plots the result.

These statements create two random datasets using the stats module's **rvs** method.

This statement uses pyplot to create a scatter plot of the two datasets.

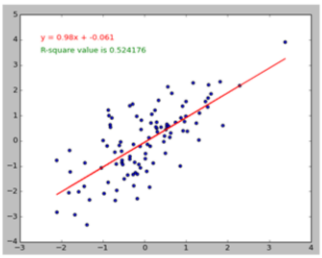
The stats module's lineregress method is used to run a linear regression on the two datasets.

The slope and y-intercept from the linear regression are used to calculate the y-values predicted by the regression. Note that xData is an array and math operators are applied to each value in an array.

Example script: linear regression

```
figure.plot(xData, y, color="r")
xloc=figure.xticks()[0][0]
yloc=figure.yticks()[0][-1]
figure.text(xloc+0.5, yloc-1, 'y = %sx + %s' % (slope, intercept),
        color = 'r')
figure.text(xloc+0.5, yloc-1.5, 'r-square value is %s' % (r**2),
        color = 'g')
figure.show()
```

get axis coordinates from xtick locations (for text placement)



8

The regression data are plotted on the figure using the original xData and the predicted yData.

These statements get the location of the first xtick and ytick. These locations will be used to specify the location of text that will be added to the figure.

The text statements add text boxes to the figure to indicate the regression equation and r-squared value. Note that the r-value, calculated by the linregress method, must be squared to provide an r-squared value.

The script created figure shown here.