Logo

Description automatically generated

**Data Visualization Techniques**

Python algorithms used in Tableau

**Helia Rahimi**

**Frederick Ditliac Atiah**

**Correlation and trend scorecard**

This scorecard shows the correlation between Gdp and suicide numbers per continent. The orange shows the movement of the suicide rate while the blue line shows the Gdp rate. It can be observed that when the Gdp of a continent increases the suicide rate decreases for all continents. With time both lines cross paths (each other) as can be seen in Oceania. Graphical user interface, chart, line chart

Description automatically generated

Python was used to get the correlation coefficient and the p-value.  Across the board or continent, the p-value was 3.3 whiles the correlation coefficient was 0.8243 which shows a strong correlation between the Gdp and suicide rate.

This does not tell the story. The clustering visualization below did show more trends and clusters.

Graphical user interface, text, application

Description automatically generated

Graphical user interface, text, application

Description automatically generated

Python Scipy package were used. Function **scipy.stats.pearsonr(*x*, *y*),** which calculates Pearson correlation coefficient and p-value for testing non-correlation.

The Pearson correlation coefficient measures the linear relationship between two datasets. The calculation of the p-value relies on the assumption that each dataset is normally distributed. Like other correlation coefficients, this one varies between -1 and +1 with 0 implying no correlation. Correlations of -1 or +1 imply an exact linear relationship.

The correlation coefficient is calculated as follows:

Text, letter

Description automatically generated

where m­­x is the mean of the vector x and my is the mean of the vector y.

Under the assumption that x and y are drawn from independent normal distributions

Text, letter

Description automatically generated

where n is the number of samples, and B is the beta function. This is sometimes referred to as the exact distribution of r. This is the distribution that is used in **pearsonr** to compute the p-value. The distribution is a beta distribution on the interval [-1, 1], with equal shape parameters a = b = n/2 - 1. In terms of SciPy’s implementation of the beta distribution, the distribution of r is:

dist **=** scipy**.**stats**.**beta**(**n**/2** **-** **1,** n**/2** **-** **1,** loc**=-1,** scale**=2)**

The p-value returned by **pearsonr** is a two-sided p-value. The p-value roughly indicates the probability of an uncorrelated system producing datasets that have a Pearson correlation at least as extreme as the one computed from these datasets. More precisely, for a given sample with correlation coefficient r, the p-value is the probability that abs(r’) of a random sample x’ and y’ drawn from the population with zero correlation would be greater than or equal to abs(r). In terms of the object dist shown above, the p-value for a given r and length n can be computed as:

p **=** **2\***dist**.**cdf**(-**abs**(**r**))**

When n is 2, the above continuous distribution is not well-defined. One can interpret the limit of the beta distribution as the shape parameters a and b approach a = b = 0 as a discrete distribution with equal probability masses at r = 1 and r = -1. More directly, one can observe that, given the data x = [x1, x2] and y = [y1, y2], and assuming x1 != x2 and y1 != y2, the only possible values for r are 1 and -1. Because abs(r’) for any sample x’ and y’ with length 2 will be 1, the two-sided p-value for a sample of length 2 is always 1.