WAVOO WAJEEHA WOMEN'S COLLEGE
OF ARTS & SCIENCE - KAYALPATNAM
(Affiliated to Manonmanium Sundaranar University, Tirunelveli)
Run by : Wavoo SAR Educational Trust
(minority institution)

Department of Information Technology
Statistics with Python

Module 3: **Statistics Module in Python**
Descriptive statistics, Measures of Central Tendency- Mean, Geometric Mean, Harmonic Mean, Median, Mode, Measures of Variability- Range, Variance, Standard Deviation, Statistics built-in Module.

Descriptive Statistics

Descriptive statistics is about describing and summarizing data. It uses two main approaches:
The quantitative approach describes and summarizes data numerically.

The visual approach illustrates data with charts, plots, histograms, and other graphs.

You can apply descriptive statistics to one or many datasets or variables. When you describe and summarize a single variable, you're performing **univariate analysis**. When you search for statistical relationships among a pair of variables, you're doing a **bivariate analysis**. Similarly, a **multivariate analysis** is concerned with multiple variables at once.

Types of Measures

- **Central tendency** tells you about the centers of the data. Useful measures include the mean, median, and mode.
- **Variability** tells you about the spread of the data. Useful measures include variance and standard deviation.
- **Correlation or joint variability** tells you about the relation between a pair of variables in a dataset.

Calculating Descriptive Statistics
Start by importing all the packages you'll need:

```
>>> import math
>>> import statistics
>>> import numpy as np
>>> import scipy.stats
>>> import pandas as pd
```

These are all the packages you'll need for Python statistics calculations

Measures of Central Tendency

The **measures of central tendency** show the central or middle values of datasets. There are several definitions of what's considered to be the center of a dataset. In this tutorial, you'll learn how to identify and calculate these measures of central tendency:

a) Mean
b) Geometric mean
c) Harmonic mean
d) Median
e) Mode

a)Mean

The **sample mean**, also called the **sample arithmetic mean** or simply the **average**, is the arithmetic average of all the items in a dataset. The mean of a dataset x is mathematically expressed as $\Sigma_i x_i/n$, where $i = 1, 2, \ldots, n$. In other words, it's the sum of all the elements x_i divided by the number of items in the dataset x.

You can calculate the mean with pure Python using sum() and len(), without importing libraries:

```
>>>x=[1,2.5,4,8,28]
 mean_ = sum(x) / len(x)

>>> mean_
8.7
```

Although this is clean and elegant, we can also apply built-in Python statistics functions:

```
>>> mean = statistics.mean(x)
>>> mean
8.7

>>> mean = statistics.fmean(x)
>>> mean
8.7
```

You've called the functions mean() and fmean() from the built-in Python statistics library and got the same result as you did with pure Python. **fmean() is introduced in Python 3.8** as a faster alternative to mean(). It always returns a floating-point number.

b)Geometric mean

This task can also be performed using inbuilt function of geometric_mean().
This is new in **Python versions >= 3.8.**

```
# Python3 code to demonstrate working of  Geometric Mean of List
# using statistics.geometric_mean()
```

```
import statistics
# initialize list
test_list = [6, 7, 3, 9, 10, 15]
# printing original list
print("The original list is : " + str(test_list))
# Geometric Mean of List
# using statistics.geometric_mean()
res = statistics.geometric_mean(test_list, 1)
# printing result
print("The geometric mean of list is : " + str(res))
```

Output :
The original list is : [6, 7, 3, 9, 10, 15]
The geometric mean of list is : 7.443617568993922

Calculate Geometric Mean Using SciPy
The following code shows how to use the **gmean()** function from the SciPy library to calculate the geometric mean of an array of values:

```
from scipy.stats import gmean
#calculate geometric mean
gmean([1, 4, 7, 6, 6, 4, 8, 9])
```

Output:
4.81788719702029

This can be worked in advanced Version of Python or Google colab

c) Harmonic Mean
The **harmonic mean** is the reciprocal of the mean of the reciprocals of all items in the dataset: $n / \Sigma_i(1/x_i)$, where $i = 1, 2, \ldots, n$ and n is the number of items in the dataset x. One variant of the pure Python implementation of the harmonic mean is this:

```
>>> hmean = statistics.harmonic_mean(x)
>>> hmean
2.7613412228796843
```

If you have a nan value in a dataset, then it'll return nan. If there's at least one 0, then it'll return 0. If you provide at least one negative number, then you'll get statistics.StatisticsError:

```
>>> statistics.harmonic_mean(x_with_nan)
nan
>>> statistics.harmonic_mean([1, 0, 2])
0
>>> statistics.harmonic_mean([1, 2, -2])  # Raises StatisticsError
```

d)Geometric Mean

The **geometric mean** is the n-th root of the product of all n elements x_i in a dataset x: $\sqrt[n]{(\Pi_i x_i)}$, where $i = 1, 2, \ldots, n$.

You can implement the geometric mean in pure Python like this:

```
>>> gmean = statistics.geometric_mean(x)
>>> gmean
4.67788567485604
```

e)Median

The **sample median** is the middle element of a sorted dataset. The dataset can be sorted in increasing or decreasing order. If the number of elements n of the dataset is odd, then the median is the value at the middle position: $0.5(n + 1)$. If n is even, then the median is the arithmetic mean of the two values in the middle, that is, the items at the positions $0.5n$ and $0.5n + 1$.

For example, if you have the data points 2, 4, 1, 8, and 9, then the median value is 4, which is in the middle of the sorted dataset (1, 2, 4, 8, 9). If the data points are 2, 4, 1, and 8, then the median is 3, which is the average of the two middle elements of the sorted sequence (2 and 4).

```
# Python code to demonstrate the  working of median() function.
# importing statistics module
import statistics
# unsorted list of random integers
data1 = [2, -2, 3, 6, 9, 4, 5, -1]
# Printing median of the
# random data-set
print("Median of data-set is : % s "   % (statistics.median(data1)))
```

Output :

Median of data-set is : 3.5

f) Mode

The **sample mode** is the value in the dataset that occurs most frequently. If there isn't a single such value, then the set is **multimodal** since it has multiple modal values. For example, in the set that contains the points 2, 3, 2, 8, and 12, the number 2 is the mode because it occurs twice, unlike the other items that occur only once.

You can obtain the mode with statistics.mode() and statistics.multimode():

```
>> u=[2,3,2,8,12]
>>> mode_ = statistics.mode(u)
>>> mode_

>>> mode_ = statistics.multimode(u)
>>> mode_
[2]
```

Example :

Given data-set is : **[1, 2, 3, 4, 4, 4, 4, 5, 6, 7, 7, 7, 8]**

The mode of the given data-set is **4**

Logic: 4 is the most occurring/ most common element from the given list

```
# Python code to demonstrate the use of mode() function
```

```
# mode() function a sub-set of the statistics module
# We need to import the statistics module before doing any work

import statistics
# declaring a simple data-set consisting of real valued positive integers.
set1 =[1, 2, 3, 3, 4, 4, 4, 5, 5, 6]
# Printing out mode of given data-set
print("Mode of given data set is % s" % (statistics.mode(set1)))
```

Measures of Variability

The measures of central tendency aren't sufficient to describe data. You'll also need the **measures of variability** that quantify the spread of data points. In this section, you'll learn how to identify and calculate the following variability measures:

I. Variance
II. Standard deviation
III. Range

I.Variance

The **sample variance** quantifies the spread of the data. It shows numerically how far the data points are from the mean. You can express the sample variance of the dataset x with n elements mathematically as $s^2 = \Sigma_i(x_i - \text{mean}(x))^2 / (n - 1)$, where $i = 1, 2, \ldots, n$ and $\text{mean}(x)$ is the sample mean of x.

```
>>> var_ = statistics.variance(x)
```

```
# Python code to demonstrate the working of variance() function of Statistics Module
# Importing Statistics module
import statistics
# Creating a sample of data
sample = [2.74, 1.23, 2.63, 2.22, 3, 1.98]
# Prints variance of the sample set
print("Variance of sample set is % s"      %(statistics.variance(sample)))
```

Output :

Variance of sample set is 0.40924

II.Standard Deviation

Standard Deviation is a measure of spread in Statistics. It is used to quantify the measure of spread, variation of a set of data values.

```
# Python code to demonstrate stdev() function
# importing Statistics module
import statistics
# creating a simple data - set
sample = [1, 2, 3, 4, 5]
# Prints standard deviation
print("Standard Deviation of sample is % s " % (statistics.stdev(sample)))
```

Output :
Standard Deviation of the sample is 1.5811388300841898

III. Range
The **range of data** is the difference between the maximum and minimum element in the dataset

```
numbers = [4,10,29,33,42,-67]
n_min = min(numbers)
n_max = max(numbers)
n_range = n_max - n_min
print(n_range)
```

Output: 109

Python statistics Module

Python has a built-in module that you can use to calculate mathematical statistics of numeric data.

The statistics module was new in Python 3.4.

Statistics Methods

Method	Description
statistics.harmonic_mean()	Calculates the harmonic mean (central location) of the given data
statistics.mean()	Calculates the mean (average) of the given data
statistics.median()	Calculates the median (middle value) of the given data
statistics.median_grouped()	Calculates the median of grouped continuous data
statistics.median_high()	Calculates the high median of the given data
statistics.median_low()	Calculates the low median of the given data
statistics.mode()	Calculates the mode (central tendency) of the given numeric or nominal data
statistics.pstdev()	Calculates the standard deviation from an entire population
statistics.stdev()	Calculates the standard deviation from a sample of data
statistics.pvariance()	Calculates the variance of an entire population
statistics.variance()	Calculates the variance from a sample of data

Sources:
https://realpython.com/python-statistics/
https://www.tutorialsteacher.com/python/statistics-module