# Statistics On Python (Part - One)

By: `Veerendra Bhadrachalam`

PGD Data Science(Pursuing), M.Sc. Statistics

---

## Connect with me:

**Linkedin:** Veerendra Bhadrachalam **Github:** Click here for Source code
**Kaggle:** Connect with me on kaggle.com

---

## Hi □, Welcome to the notebook □ on *Statistics with Python.*

# Agenda:

Here we are going to see how to perform Statistical Operations □ in python most efficiently.

**Table of Content:**

---

# Definition of Statistics

There are many great statisticains who have gave different definitions for statistics, but the definition I feel close to the work we do as a Data Scientist is:

In [30]:
```
from IPython.display import Image
Image('stats.jpg')
```

Out[30]:


**Simple def:**

*A branch of mathematics dealing with the* `collection, analysis, interpretation, and presentation of masses of numerical data.`

**A bit high level:**

*The practice or science of* `collecting and analysing numerical data` in large quantities, `especially for the purpose of inferring proportions in a whole from those in a representative sample.`

---

Back to Agenda

---

# Variable and Random Variable

### Variable

---

`Mathematical definition of a Variable:`

*A **variable** is a quantity that may change within the context of a mathematical problem or experiment.*

Typically, we use a single letter to represent a variable in mathematics.

- The letters x, y, and z are common generic symbols used for variables.

`Statistical definition of Variable:`

In statistics, a **variable** has two defining characteristics:

- `A variable is an attribute that describes a person, place, thing, or idea.`
- `The value of the variable can "vary" from one entity to another.`

**For example, a person's hair color is a potential variable, which could have the value of "blond" for one person and "brunette" for another.**

### Random Variable

---

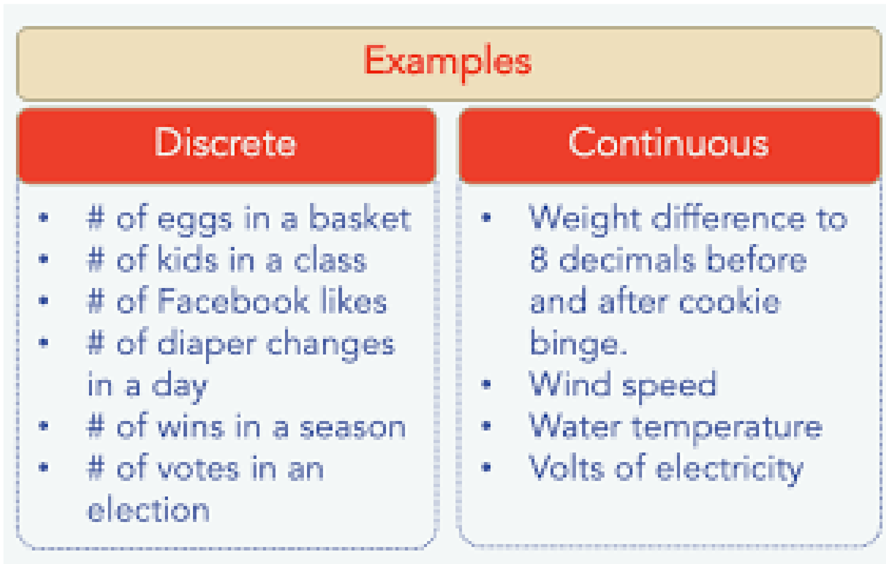
A **random variable** is a *numerical description of the* ***outcome of a statistical experiment.***

**Random variable can be classified into two types :**

1. Discrete random variable
2. Continous random variable

In [40]:
```
Image('CvD.png', width = 800)
```

Out[40]:




**Discrete Random Variable:** A random variable that may assume only a finite number or an infinite sequence of values is said to be discrete.

**Continuous Random Variable:** A random variable that may *assume any value in some interval on the real number line* is said to be continuous.

In [26]:
```
import numpy as np
import matplotlib.pyplot as plt
```

---

# Create a random integer number list.

*For this we will be using* `np.random()`

In [2]:
```
#rand() for random float values
```

```
#randint() for random integer values
#seed() to maintain one set of random numbers(where ever & when eve
r generated) through out the process.

#np.random.randint?
#np.random.seed?
```

In [24]:
```
np.random.seed(0)
a = np.random.randint(7, 10, 20)
```

In [19]:
```
np.min(a) #argument1 -> lower limt (mandatory )
```

Out[19]:
```
7
```

In [20]:
```
np.max(a) #argument2 -> higher limit (Optional)
```

Out[20]:
```
9
```

In [21]:
```
np.size(a) #argument3 -> size (no.of values to be generated in the
 array.)
                        #size[if not specified by default generates
only one random values]
```

Out[21]:
```
20
```

In [22]:
```
print(a)
```

```
[8 7 7 8 8 7 7 8 7 8 7 9 8 9 7 9 8 9 7 7]
```

---

[Back to Agenda](#)

---

# Measure of Central Tendency

> A `measure of central tendency` is a **single value that attempts to describe a set of data by identifying the central position within that set of data.**

Fun Facts:

- As such, measures of central tendency are sometimes called **measures of central location.**
- They are also classed as **summary statistics.**

The **mean (often called the average) is most likely the measure of central tendency** that you are most familiar with, but there are others, such as the **median and the mode.**

### Lets Caluclate the Measures of Centeral Tendency.

---

# Mean

#### Caluclate the mean of the above list of values

*To caluclate mean/arthematic mean we can use `np.mean()`.*

In [25]:
```
np.mean(a)
```

Out[25]:
```
7.9
```

Mean, also know as arthematic mean of this set of random values is `7.9`.

---

# Median

#### Caluclate the meadian of the above list of values.

*To caluclate median we can use `np.median()`.*

In [9]:
```
np.median(a)
```

Out[9]:
```
8.0
```

The median of this list of random numbers is `8`.

---

# Mode

**Calculate the mode of the above list of values.**

*Numpy package unfortunately don't have buit-in function to calculate mode. ~np.mode()~*

*So we will be **importing statistics package of python, and will use mode() friom it.***

In [10]:
```
import statistics as stats

stats.mode(a)
```

Out[10]:
```
8
```

**Or**

In [11]:
```
from statistics import mode

mode(a)
```

Out[11]:
```
8
```

And the mode for this set of random numbers id `8`.

---

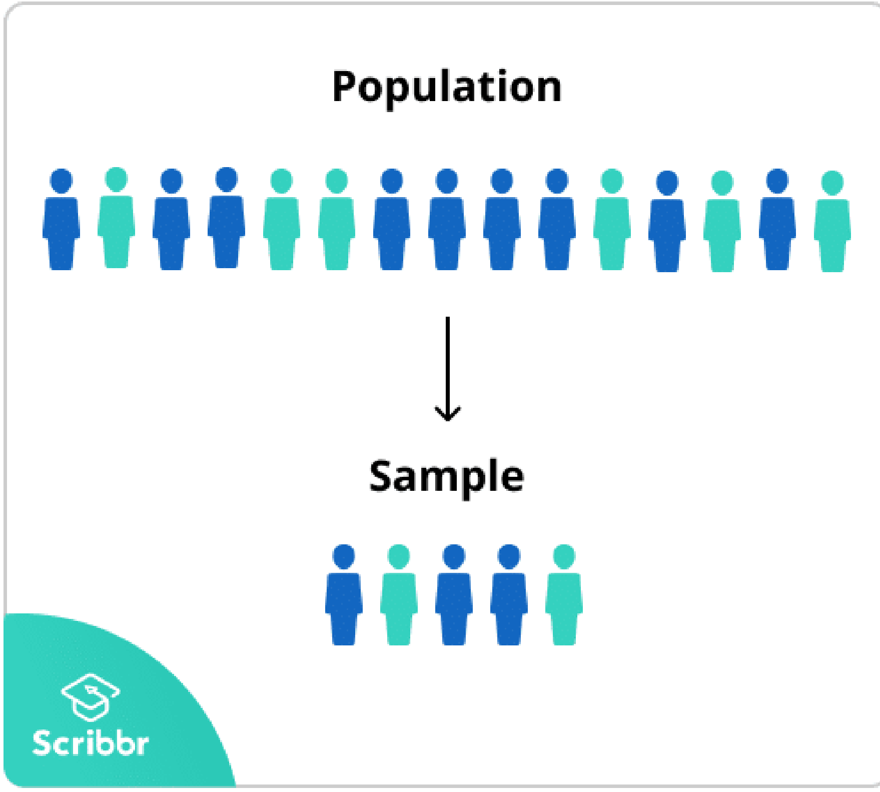
[Back to Agenda](#)

---

# Population and Sample

In [41]:
```
Image('pns.png')
```

Out[41]:




### Population:

A population is the entire group that you want to draw conclusions about.

### Sample:

A sample is the specific group that you will collect data from.

- The size of the sample is always less than the total size of the population.

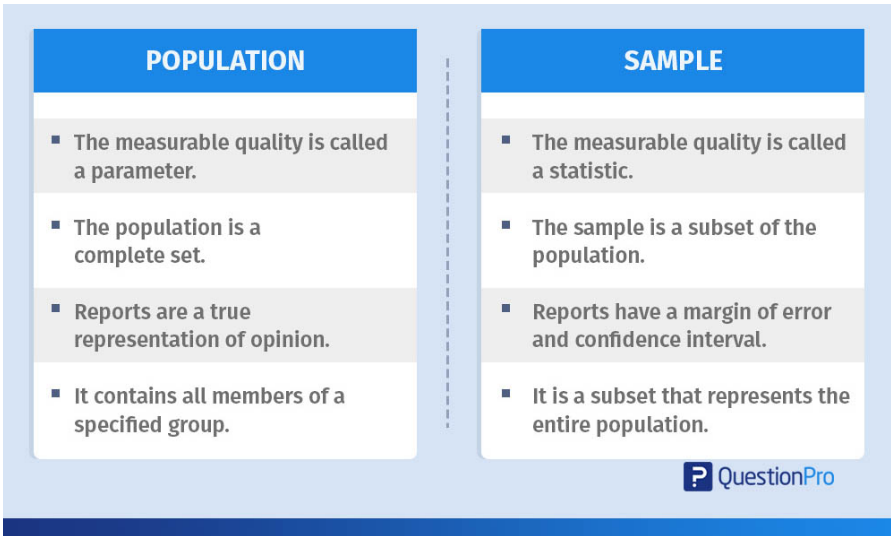
### Note:

> In research, a **population doesn't always refer to people.**
>
> - It can mean a group containing **elements of anything you want to study,** such as `objects` ,

> `events`, `organizations`, `countries`, `species`, `organisms`, etc.

In [42]:
```
Image('PopVsSam.jpg')
```

Out[42]:




# Population

### Creating a random population data set of size 120

For this we will be using our old from `np.random.randint()`.

In [86]:
```
np.random.seed(2)
pop = np.random.randint(23, 60, 120) # <- 120 random integers betwe
en 23 and 60(excluded).

print(pop)
```

```
[38 31 45 41 34 30 57 54 34 44 54 49 43 26 27 56 26 28 47 27 29
54 42 54
 25 39 35 27 49 38 31 38 31 40 45 32 49 42 55 55 49 31 35 33 57
32 29 45
 29 42 41 24 27 40 29 56 41 43 49 46 45 33 31 49 58 50 55 39 44
52 39 29
 33 40 38 53 36 49 43 45 59 50 23 51 59 34 32 30 41 24 38 50 55
32 37 53
 54 55 39 44 57 44 42 51 56 31 45 47 34 24 28 30 39 51 38 40 54
57 39 59]
```

### Mean, Median and Mode of Population

In [87]:
```
#Population Mean
pop_mean = np.mean(pop)
print("Population Mean:", pop_mean)

#Population Median
pop_median = np.median(pop)
print("Population Median:", pop_median)

#Population Mode
from statistics import mode
p_mode = mode(pop)
print("Population Mode:", p_mode)
```

```
Population Mean: 41.38333333333333
Population Median: 41.0
Population Mode: 49
```

---

[Back to Agenda](#)

---

## Sample

Sampling a random sample of size 30 values from the population - [pop](#)

For this our syntax will be `np.random.choice(population, size)`

In [93]:
```
np.random.seed(0)
samp = np.random.choice(pop, 30)

print(samp)
```

```
[57 45 57 58 39 39 51 44 51 54 38 49 30 39 41 41 43 49 50 42 55 30 29 41
 50 42 39 49 33 44]
```

Here you go, we have our random sample of size 30 [samp](#) values from the population [pop](#).

### Mean, Median and Mode of Sample

In [94]:

```
#Sample Mean
samp_mean = np.mean(samp)
print("Sample Mean:", samp_mean)

#Sample Median
samp_median = np.median(samp)
print("Sample Median:", samp_median)

#Sample Mode
from statistics import mode
p_mode = mode(samp)
print("Sample Mode:", p_mode)
```

```
Sample Mean: 44.3
Sample Median: 43.5
Sample Mode: 39
```

---

[Back to Agenda](#)

---

## Experiment

> If observed, we can see that the mean of sample and population are close to each other.

Lets take few more samples and compare the means of them with population, lets see if we can come to any conclusion on relation of sample and population.

#### Creating 4 more samples

In [104]:

```
samp_2 = np.random.choice(pop, 30)
samp_3 = np.random.choice(pop, 30)
samp_4 = np.random.choice(pop, 30)
samp_5 = np.random.choice(pop, 30)
```

In [105]:

```
# Let compute the means of these populations

samp_list = [samp, samp_2, samp_3, samp_4, samp_5]
```

```python
samp_means = [np.mean(i) for i in samp_list]

samp_means #List of sample means
```

Out[105]:

```
[44.3,
 42.166666666666664,
 42.43333333333333,


 42.233333333333334,
 38.233333333333334]
```

We have individual sample means, now lets compute mean of sample means.

Yes! You guessed it right, we can compare that to the pop_mean and can understand the relation between the sample mean and population mean.

In [106]:

```python
print("Population Mean:", np.mean(pop))

print("Mean of Sample Means:", np.mean(samp_means))
```

```
Population Mean: 41.38333333333333
Mean of Sample Means: 41.873333333333335
```

**Conclusion:**

*Here you can see **Mean of sample means and population mean approximately equal to each other.***

□□□ End of Part One □□□
*** Thank You ***

[Connect with me]()

---

**Reference:**


https://www.kaggle.com/janiobachmann/statistical-analysis-a-frequentist-approach

https://www.mymarketresearchmethods.com/data-types-in-statistics/examples-of-discrete-and-continuous-data/