**Q1. What is a probability distribution, exactly? If the values are meant to be random, how can you predict them at all?**

A probability distribution is a list of all of the possible outcomes of a random variable, along with its corresponding probability values. A probability distribution links each outcome of a random variable or process with its probability of occurrence.

Some real life examples of the use of probabilities are:

* **Finance:**By estimating the chance that a given financial asset will fall between or within a specific range, it’s possible to develop trading strategies to capture that predicted outcome.
* **Weather forecast:** Meteorologists can’t predict exactly what the weather will be, so they use tools and instruments to determine the likelihood that it will rain, snow or hail. They also examine historical data bases to estimate high and low temperatures and probable weather patterns for that day or week.
* **Insurance:** Probability plays an important role in analyzing insurance policies to determine which plans are best for customers and what deductible amounts they need.
* **Sports:** Athletes and coaches use probability to determine the best sports strategies for games and competitions. Companies like [BWin](https://www.bwin.com/" \t "_blank) have made a business out of this and you can even bet using different strategies.
* **Advertisement:** Probability is used to estimate potential customers that will be more likely to react positively to specific campaigns, based on their consumption patterns.

Numerically speaking, a probability is a number that ranges from 0 (meaning there is no way an event is going to happen) to 1 (which means the event will happen for sure), and if you take all the possible outcomes and add them up, you sum to 1. The bigger the value of the probability, the more likely the event is to occur.

So for example, you’d say that the probability of rain tomorrow is 40% (stated as “P(rain)=0,4”), or the probability of a car theft in a particular region is 2% (defined as “P(car theft)=0,02”). In the first case you’re interested in the variable “rain”, and in the second in the variable “car theft”. These variables, as well as any other that is a result of a random process, are referred to as “random variables”.

**Q2. Is there a distinction between true random numbers and pseudo-random numbers, if there is one? Why are the latter considered “good enough”?**

Any arbitrary-length sequence of pseudorandom numbers can always be replicated and predicted if you know the RNG being used and you know the initial state.

A sequence of true random numbers can’t be predicted or repeated.

This matters if it’s really really important that your random numbers can’t be predicted or repeated. Cryptography is the go-to example, but some scientific applications need real random numbers as well.

Any arbitrary-length sequence of pseudorandom numbers can always be replicated and predicted if you know the RNG being used and you know the initial state.

A sequence of true random numbers can’t be predicted or repeated.

This matters if it’s really really important that your random numbers can’t be predicted or repeated. Cryptography is the go-to example, but some scientific applications need real random numbers as well.

# import packages

import scipy.stats as stats

import seaborn as sns

import matplotlib.pyplot as plt

# generate data

data =stats.norm(scale=1, loc=0).rvs(1000)

# plotting a histogram

ax = sns.distplot(data,

                  bins=50,

                  kde=True,

                  color='red',

                  hist\_kws={"linewidth": 15,'alpha':1})

ax.set(xlabel='Normal Distribution', ylabel='Frequency')

plt.show()

**Binomial Distribution**

Under a given set of factors or assumptions, the binomial distribution expresses the likelihood that a variable will take one of two outcomes or independent values. ex: if an experiment is successful or a failure. if the answer for a question is “yes” or “no” etc… . [np.random.binomial()](https://www.geeksforgeeks.org/binomial-random-variables/) is used to generate binomial data. n refers to a number of trails and prefers the probability of each trail.

* Python3

|  |
| --- |
| # import packages  import seaborn as sns  import matplotlib.pyplot as plt  import numpy as np    # generate data  # n== number of trials,p== probability of each trial  n, p = 10, .6  data = np.random.binomial(n, p, 10000)    # plotting a histogram  ax = sns.distplot(data,                    bins=20,                    kde=False,                    color='red',                    hist\_kws={"linewidth": 15, 'alpha': 1})  ax.set(xlabel='Binomial Distribution', ylabel='Frequency')    plt.show() |

**Q3. What are the two main factors that influence the behaviour of a "normal" probability distribution** ?

As with any probability distribution, the parameters for the normal distribution define its shape and probabilities entirely. The normal distribution has two parameters, the mean and standard deviation.

Mean weight grams: 100

Standard Deviation: 15

## Parameters of the Normal Distribution

As with any probability distribution, the parameters for the normal distribution define its shape and probabilities entirely. The normal distribution has two parameters, the mean and standard deviation. The Gaussian distribution does not have just one form. Instead, the shape changes based on the parameter values, as shown in the graphs below.

### **Mean**

The mean is the central tendency of the normal distribution. It defines the location of the peak for the bell curve. Most values cluster around the mean. On a graph, changing the mean shifts the entire curve left or right on the X-axis.

### **Standard deviation**

The standard deviation is a measure of variability. It defines the width of the normal distribution. The standard deviation determines how far away from the mean the values tend to fall. It represents the typical distance between the observations and the average.

On a graph, changing the standard deviation either tightens or spreads out the width of the distribution along the X-axis. Larger standard deviations produce wider distributions.

Common Properties for All Forms of the Normal Distribution

Despite the different shapes, all forms of the normal distribution have the following characteristic properties.

* They’re all symmetric bell curves. The Gaussian distribution cannot model skewed distributions.
* The mean, median, and [mode](https://statisticsbyjim.com/glossary/mode/) are all equal.
* Half of the population is less than the mean and half is greater than the mean.
* The Empirical Rule allows you to determine the proportion of values that fall within certain distances from the mean. More on this below!

While the normal distribution is essential in statistics, it is just one of many probability distributions, and it does not fit all populations. To learn how to determine whether the normal distribution provides the best fit to your sample data, read my posts about [How to Identify the Distribution of Your Data](https://statisticsbyjim.com/hypothesis-testing/identify-distribution-data/) and [Assessing Normality: Histograms vs. Normal Probability Plots](https://statisticsbyjim.com/basics/assessing-normality-histograms-probability-plots/).

The [uniform distribution](https://statisticsbyjim.com/probability/uniform-distribution/) also models symmetric, [continuous data](https://statisticsbyjim.com/glossary/continuous-variables/), but all equal-sized ranges in this distribution have the same probability, which differs from the normal distribution.

If you have continuous data that are skewed, you’ll need to use a different distribution, such as the [Weibull](https://statisticsbyjim.com/probability/weibull-distribution/), [lognormal](https://statisticsbyjim.com/probability/lognormal-distribution/), [exponential](https://statisticsbyjim.com/probability/exponential-distribution/), or [gamma](https://statisticsbyjim.com/probability/gamma-distribution/) distribution.

**Q4. Provide a real-life example of a normal distribution**.

A fair rolling of dice is also a good example of normal distribution. In an experiment, it has been found that when a dice is rolled 100 times, chances to get '1' are 15-18% and if we roll the dice 1000 times, the chances to get '1' is, again, the same, which averages to 16.7% (1/6).

### **Height**

Height of the population is the example of normal distribution. Most of the people in a specific population are of average height. The number of people taller and shorter than the average height people is almost equal, and a very small number of people are either extremely tall or extremely short. However, height is not a single characteristic, several genetic and environmental factors influence height. Therefore, it follows the normal distribution.

### Rolling A Dice

A fair rolling of dice is also a good example of normal distribution. In an experiment, it has been found that when a dice is rolled 100 times, chances to get ‘1’ are 15-18% and if we roll the dice 1000 times, the chances to get ‘1’ is, again, the same, which averages to 16.7% (1/6). If we roll two dices simultaneously, there are 36 possible combinations. The probability of rolling ‘1’ (with six possible combinations) again averages to around 16.7%, i.e., (6/36). More the number of dices more elaborate will be the normal distribution graph.

**Q5. In the short term, how can you expect a probability distribution to behave? What do you think will happen as the number of trials grows?**

**Random Variable**

A **random variable** is a variable whose possible values are numerical outcomes of a random phenomenon. There are two types of random variables, discrete and continuous.

A **discrete random variable** is one which may take on only a countable number of distinct values and thus can be quantified. For example, you can define a random variable $X$ to be the number which comes up when you roll a fair dice. $X$ can take values : $[1,2,3,4,5,6]$ and therefore is a discrete random variable.

The probability distribution of a discrete random variable is a list of probabilities associated with each of its possible values. It is also sometimes called the **probability function** or the probability mass function. To have a mathematical sense, suppose a random variable $X$ may take $k$ different values, with the probability that $X = x\_{i}$ defined to be $P(X = x\_{i}) = p\_{i}$. Then the probabilities $p\_{i}$ must satisfy the following:

1: 0 < $p\_{i}$ < 1 for each $i$

2: $p\_{1} + p\_{2} + ... + p\_{k} = 1$.

Some examples of discrete probability distributions are Bernoulli distribution, Binomial distribution, Poisson distribution etc.

A **continuous random variable** is one which takes an infinite number of possible values. For example, you can define a random variable $X$ to be the height of students in a class. Since the continuous random variable is defined over an interval of values, it is represented by the area under a curve (or the integral).

The probability distribution of a continuous random variable, known as **probability distribution functions**, are the functions that take on continuous values. The probability of observing any single value is equal to $0$ since the number of values which may be assumed by the random variable is infinite. For example, a random variable $X$ may take all values over an interval of real numbers. Then the probability that $X$ is in the set of outcomes $A, P(A)$, is defined to be the area above $A$ and under a curve. The curve, which represents a function $p(x)$, must satisfy the following:

1: The curve has no negative values $(p(x) > 0$ for all $x$)

2: The total area under the curve is equal to $1$.

A curve meeting these requirements is often known as a **density curve**. Some examples of continuous probability distributions are normal distribution, exponential distribution, beta distribution, etc.

There’s another type of distribution that often pops up in literature which you should know about called **cumulative distribution function**. All random variables (discrete and continuous) have a cumulative distribution function. It is a function giving the probability that the random variable $X$ is less than or equal to $x$, for every value $x$. For a discrete random variable, the cumulative distribution function is found by summing up the probabilities.

In the next section, you will explore some important distributions and try to work them out in python but before that import all the necessary libraries that you'll use.

# for inline plots in jupyter

%matplotlib inline

# import matplotlib

import matplotlib.pyplot as plt

# for latex equations

from IPython.display import Math, Latex

# for displaying images

from IPython.core.display import Image

[**POWERED BY DATACAMP WORKSPACE**](https://www.datacamp.com/workspace)**COPY CODE**

# import seaborn

import seaborn as sns

# settings for seaborn plotting style

sns.set(color\_codes=True)

# settings for seaborn plot sizes

sns.set(rc={'figure.figsize':(5,5)})

**Uniform Distribution Function**

Perhaps one of the simplest and useful distribution is the uniform distribution. The probability distribution function of the continuous uniform distribution is:

Since any interval of numbers of equal width has an equal probability of being observed, the curve describing the distribution is a rectangle, with constant height across the interval and 0 height elsewhere. Since the area under the curve must be equal to 1, the length of the interval determines the height of the curve. The following figure shows a uniform distribution in interval (a,b). Notice since the area needs to be $1$. The height is set to $1/(b-a)$.

**Q6. What kind of object can be shuffled by using random.shuffle?**

In Python, you can shuffle (= randomize) a list, string, and tuple with random.shuffle() and random.sample().

random.shuffle() shuffles a list in place, and random.sample() returns a new randomized list. random.sample() can also be used for a string and tuple.

* random.shuffle() shuffles a list in place
* random.sample() returns a new shuffled list
* How to shuffle a string and tuple
* Initialize the random number generator with random.seed()

If you want to sort in ascending or descending order or reverse instead of shuffling, see the following articles.

* [Sort a list, string, tuple in Python (sort, sorted)](https://note.nkmk.me/en/python-list-sort-sorted/)
* [Reverse a list, string, tuple in Python (reverse, reversed)](https://note.nkmk.me/en/python-reverse-reversed/)

import random

l **=** list(range(5))

print(l)

# [0, 1, 2, 3, 4]

random**.**shuffle(l)

print(l)

# [1, 0, 4, 3, 2]

**random.sample() returns a new shuffled list**

random.sample() returns a new shuffled list. The original list remains unchanged.

random.sample() returns random elements from a list. Pass the list to the first argument and the number of elements to return to the second argument. See the following article for details.

* [Random sampling from a list in Python (random.choice, sample, choices)](https://note.nkmk.me/en/python-random-choice-sample-choices/)

By setting the total number of elements in the list to the second argument, random.sample() returns a new list with all elements randomly shuffled. You can get the total number of elements in the list with len().

l **=** list(range(5))

print(l)

# [0, 1, 2, 3, 4]

lr **=** random**.**sample(l, len(l))

print(lr)

# [0, 3, 1, 4, 2]

print(l)

# [0, 1, 2, 3, 4]

**Q7. Describe the math package's general categories of functions**.

# Types of Functions

The **types of functions** are defined on the basis of the domain, range, and function expression. The expression used to write the function is the prime defining factor for a function. Along with expression, the relationship between the elements of the domain set and the range set also accounts for the type of function. The classification of functions helps to easily understand and learn the different types of functions.

Every mathematical expression which has an input value and a resulting answer can be conveniently presented as a function. Here we shall learn about the types of functions and their definition, examples.

## What are the Different Types of Functions?

The function y = f(x) is classified into different types of [functions](https://www.cuemath.com/calculus/What-are-functions/), based on factors such as the [domain and range of a function](https://www.cuemath.com/calculus/domain-and-range-of-a-function/), and the function expression. The functions have a domain ***x*** value that is referred as input. The domain value can be a number, angle, decimal, fraction. Similarly, the ***y*** value or the f(x) value (is generally a numeric value) is the range. The types of functions have been classified into the following four types.

* Based on the Set Elements
* Based on Equation
* Based on Range
* Based on Domain

## Representation of Functions

There are three different forms of representation of functions. The functions need to be represented to showcase the domain values and the range values and the relationship between them. The functions can be represented with the help of Venn diagrams, graphical formats, and roster forms. The details of each of the forms of representation are as follows.

**Venn Diagram:**The [Venn diagram](https://www.cuemath.com/algebra/venn-diagram/) is an important format for representing the function. The Venn diagrams are usually presented as two circles with arrows connecting the element in each of the circles. The domain is presented in one circle and the range values are presented in another circle. And the function defines the arrows, and how the arrows connect the different elements in the two circles.

**Graphical Form:**Functions are easy to understand if they are represented in the graphical form with the help of the [coordinate axes](https://www.cuemath.com/jee/coordinate-axes-straight-lines/). Representing the function in graphical form, helps us to understand the changing behavior of the functions if the function is increasing or decreasing. The domain of the function - the x value is represented along the x-axis, and the range or the f(x) value of the function is plotted with respect to the y-axis.

**Roster Form:** [Roster notation](https://www.cuemath.com/algebra/roster-notation/) of a set is a simple mathematical representation of the set in mathematical form. The domain and range of the function are represented in flower brackets with the first element of a pair representing the domain and the second element representing the range. Let us try to understand this with the help of a simple example. For a function of the form f(x) = x2, the function is represented as {(1, 1), (2, 4), (3, 9), (4, 16)}. Here the first element is the domain or the x value and the second element is the range or the f(x) value of the function.

## Types of Functions - Based on Set Elements

These types of functions are classified based on the number of relationships between the elements in the domain and the codomain. The different types of functions based on set elements are as follows.

### **One One Function**

A one-to-one function is defined by f: A → B such that every element of set A is connected to a distinct element in set B. The one-to-one function is also called an injective function. Here every element of the domain has a distinct image or co-domain element for the given function.

### **Many to One Function**

A many to one function is defined by the function f: A → B, such that more than one element of the set A are connected to the same element in the set B. In a many to one function, more than one element has the same co-domain or image. If a many to one function, in the codomain, is a single value or the domain element are all connected to a single element, then it is called a constant function.

**Q8. What is the relationship between exponentiation and logarithms?**

You can see straight away that the logarithm function is a reflection of the exponential function in the line represented by f(x) = x. In other words, the axes have been swapped: x becomes f(x), and f(x) becomes x. The exponential function f(x) = ex is the inverse of the logarithm function f(x) = ln x.

In Python, the math.exp(x) function can be replaced with other expressions

1. math.e \*\* x – here math.e is a constant equal to the value of the exponent.
2. pow(math.e, x) – here pow() is a built-in function of the Python language.

**Example.**

# Function math.exp(x)

import math

y = math.exp(1) # y = 2.718281828459045

x = 0.0

y = math.exp(x) # y = 1.0

x = 3.85

y = math.exp(x) # y = 46.993063231579285

##### **Function math.log(x). Natural logarithm**

The math.log(x) function is designed to calculate the natural logarithm of a number with a given base.

The general form of the function is as follows

math.log(x [, base])

here

* x – the argument for which the logarithm is calculated;
* base – base of the logarithm. This function parameter is optional. If the base parameter is absent, then the number e = 2.718281…

If you try to call the log(0.0) function, the Python interpreter will throw an error

ValueError: math domain error

since the logarithm of zero does not exist.

Example.

# Function math.log(x)

import math

x = 1.0

y = math.log(x) # y = 0.0

**Q9. What are the three logarithmic functions that Python supports?**

The following are the variants of the basic log function in Python: log2(x) log(x, Base) log10(x)

## Understanding the log() functions in Python

In order to use the functionalities of Log functions, we need to **import** the math module using the below statement.

import math

We all need to take note of the fact that the **Python Log functions cannot be accessed directly.** We need to use the math module to access the log functions in the code.

math.log(x)

The math.log(x) function is used to calculate the **natural logarithmic value** i.e. **log to the base e** (Euler’s number) which is about 2.71828, of the parameter value (**numeric expression**), passed to it.

import math

print("Log value: ", math.log(2))

### 1. log2(x) - log base 2

The math.log2(x) function is used to calculate the **logarithmic value of a numeric expression of base 2**.

math.log2(numeric expression)

import math

print ("Log value for base 2: ")

print (math.log2(20))

### 2. log(n, Base) - log base n

The math.log(x,Base) function calculates the logarithmic value of x i.e. numeric expression for a **particular (desired) base value**.

math.log(numeric\_expression,base\_value)

This function accepts two arguments:

numeric expression

Base value

Note: If no base value is provided to the function, the math.log(x,(Base)) acts as a basic log function and calculates the log of the numeric expression to the base e.

import math

print ("Log value for base 4 : ")

print (math.log(20,4))

### 3. log10(x) - log base 10

The math.log10(x) function calculates the logarithmic value of the numeric expression to the **base 10**.

math.log10(numeric\_expression)

import math

print ("Log value for base 10: ")

print (math.log10(15))

In the above snippet of code, the logarithmic value of **15** to the **base** **10** is calculated.