**Probability distributions in generative AI Task**

NAME : TEJASRI .V

### **Probability distributions and data generation :**

Generative AI models aim to learn the underlying probability distribution of the data they are trained on, allowing them to generate new samples that resemble the training data. The key idea is to capture the inherent structure and patterns in the input data by approximating the complex distribution that governs it. Once the model has learned this distribution, it can sample from it to create novel data points that share similar properties with the original data.

The probability distribution is divided into two parts:

1. Discrete Probability Distributions
2. Continuous Probability Distributions

### **Discrete Probability Distribution :**

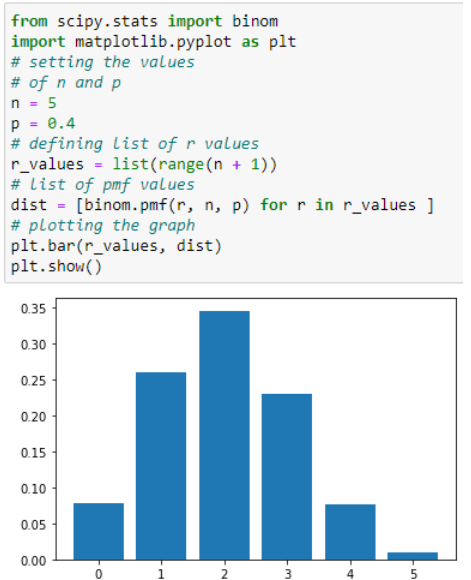
A discrete distribution describes the probability of occurrence of each value of a discrete random variable . Each possible value of the discrete random variable can be associated with a non-zero probability in a discrete probability distribution.

#### Binomial Distribution

The binomial distribution is a discrete distribution with a finite number of possibilities. When observing a series of what are known as Bernoulli trials, the binomial distribution emerges. A Bernoulli trial is a scientific experiment with only two outcomes: success or failure.

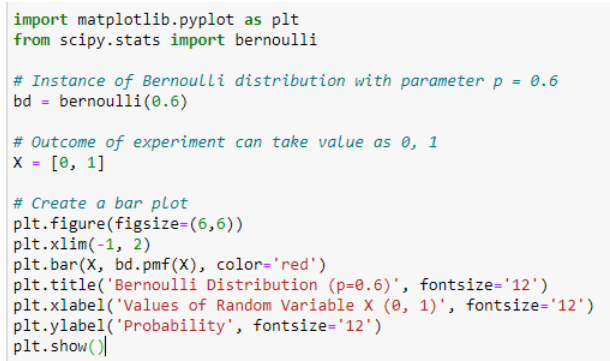
Consider a random experiment in which you toss a biased coin six times with a 0.4 chance of getting head. If 'getting a head' is considered a ‘success’, the binomial distribution will show the probability of r successes for each value of r.

The binomial random variable represents the number of successes (r) in n consecutive independent Bernoulli trials.



#### Bernoulli's Distribution

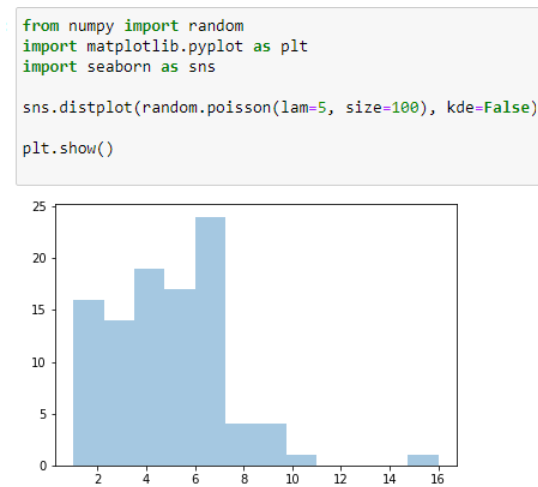
The Bernoulli distribution is a variant of the Binomial distribution in which only one experiment is conducted, resulting in a single observation. As a result, the Bernoulli distribution describes events that have exactly two outcomes.



#### Poisson Distribution

A Poisson distribution is a probability distribution used in statistics to show how many times an event is likely to happen over a given period of time. To put it another way, it's a count distribution. Poisson distributions are frequently used to comprehend independent events at a constant rate over a given time interval.

The below-given Python code generates the 1x100 distribution for occurrence 5 :



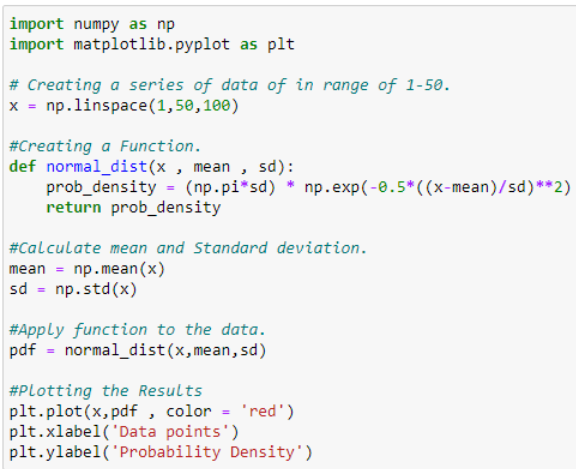
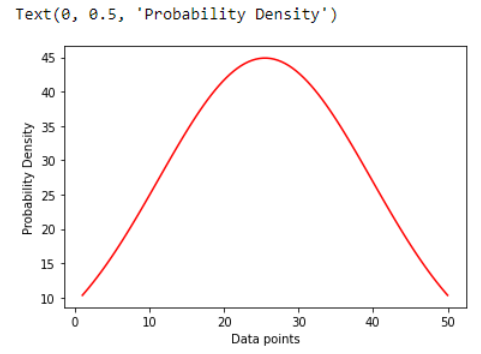
### **Continuous Probability Distributions :**

A continuous distribution describes the probabilities of a continuous random variable's possible values. A continuous random variable has an infinite and uncountable set of possible values (known as the range). The area under the curve of a continuous random variable's PDF is used to calculate its probability. As a result, only value ranges can have a non-zero probability.

1. Normal Distribution

Normal Distribution is one of the most basic continuous distribution types. Gaussian distribution is another name for it. Around its mean value, this probability distribution is symmetrical. It also demonstrates that data close to the mean occurs more frequently than data far from it. Here, the mean is 0, and the variance is a finite value.

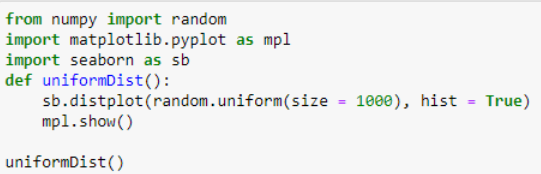
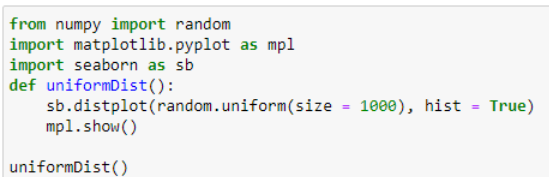
In the example, you generated 100 random variables ranging from 1 to 50. After that, you created a function to define the normal distribution formula to calculate the probability density function. Then, you have plotted the data points and probability density function against X-axis and Y-axis, respectively.

#### Continuous Uniform Distribution

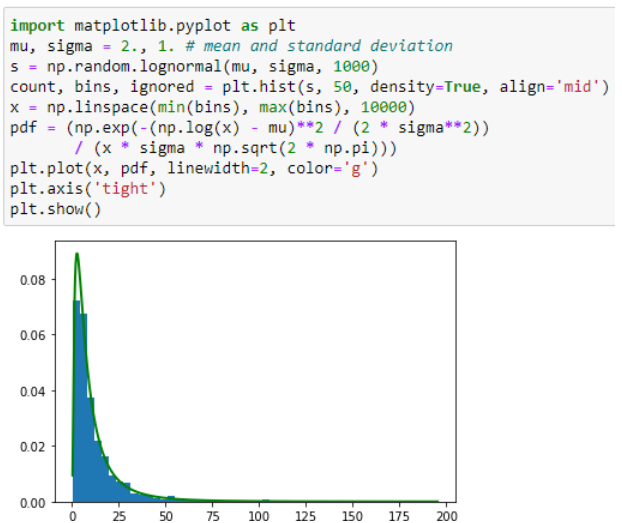
In continuous uniform distribution, all outcomes are equally possible. Each variable has the same chance of being hit as a result. Random variables are spaced evenly in this symmetric probabilistic distribution, with a 1/ (b-a) probability.

The below [Python](https://www.simplilearn.com/learn-the-basics-of-python-article) code is a simple example of continuous distribution taking 1000 samples of random variables.



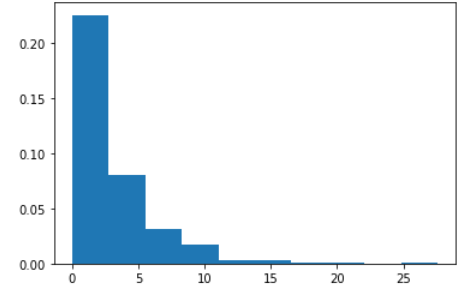
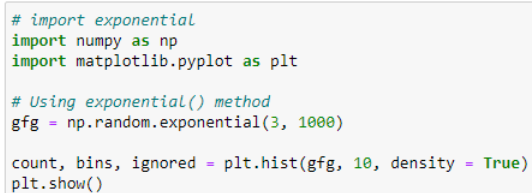
#### Log-Normal Distribution

The random variables whose logarithm values follow a normal distribution are plotted using this distribution. Take a look at the random variables X and Y. The variable represented in this distribution is Y = ln(X), where ln denotes the natural logarithm of X values.  
The size distribution of rain droplets can be plotted using log normal distribution.



#### Exponential Distribution

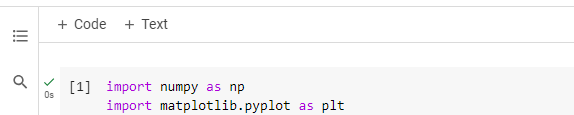
In a Poisson process, an exponential distribution is a continuous probability distribution that describes the time between events (success, failure, arrival, etc.). You can see in the below example how to get random samples of exponential distribution and return Numpy array samples by using the numpy.random.exponential() method.



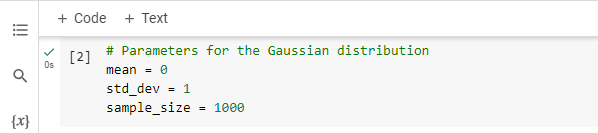
**Task:** Generating and Visualizing Probability Distributions using Python.

**Gaussian (Normal) Distribution**

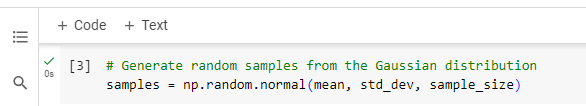
1. Import the necessary libraries: We start by importing the required libraries, numpy and matplotlib, which will help us generate random samples and visualize the distributions.



2. Set distribution parameters: Define the mean and standard deviation (std\_dev) for the Gaussian distribution, which will be used to generate the random samples. Also, specify the number of samples to generate '(sample\_size)'.

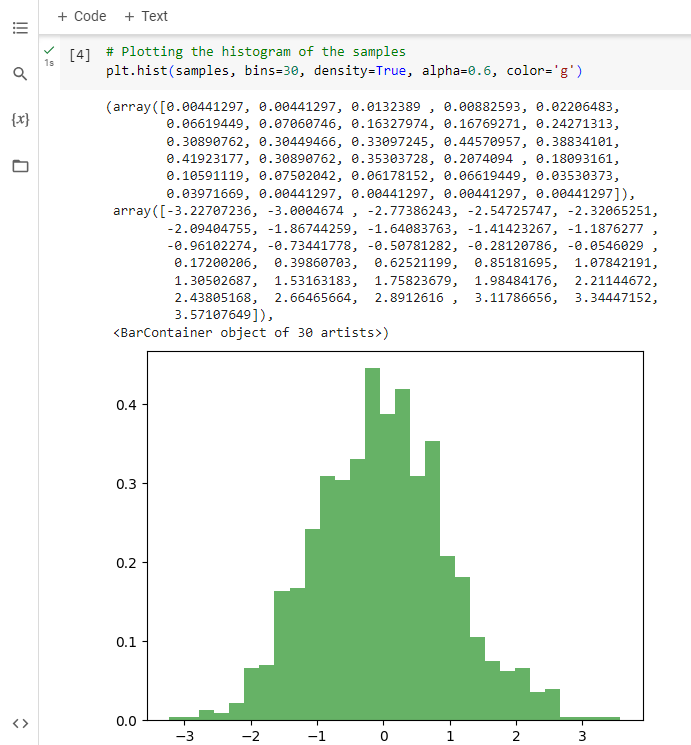


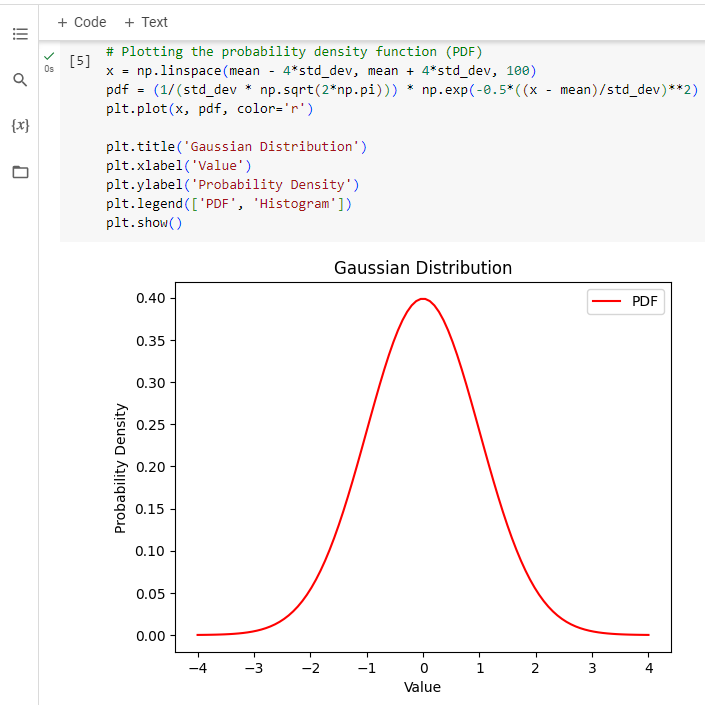
3. Generate random samples: Use the 'np.random.normal()' function to generate 'sample\_size' random samples from a Gaussian distribution with the given mean and standard deviation.



4. Plot the histogram and PDF:

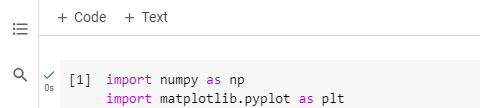
Use matplotlib to plot a histogram of the generated samples and overlay the Gaussian probability density function (PDF) on the same plot.



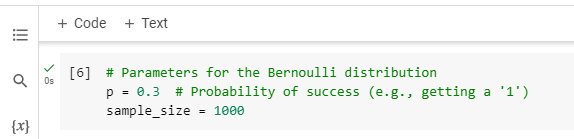


**Bernoulli Distribution**

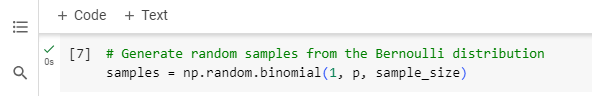
1. Import libraries: Just like before, start by importing the required libraries.



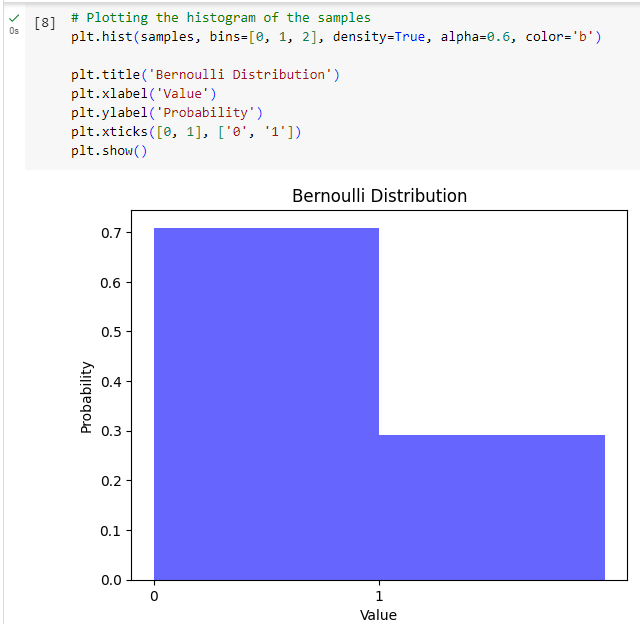
2. Set distribution parameters: Specify the probability of success '(p)' for the Bernoulli distribution and the number of samples to generate.



3. Generate random samples: Use 'np.random.binomial()' to generate random samples from a Bernoulli distribution with the given probability 'p'.

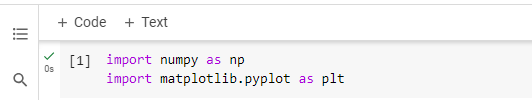


4. Plot the histogram: Plot a histogram of the generated samples, indicating the probabilities of getting 0 and 1.

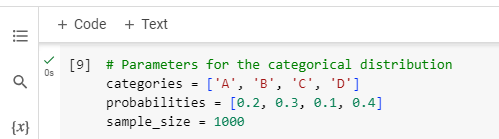


**Categorical Distribution**

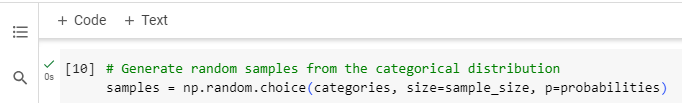
1. Import libraries: Import the necessary libraries as before.



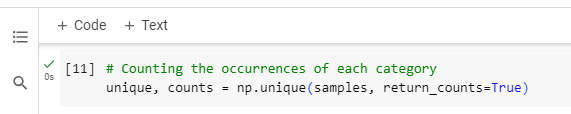
2. Set distribution parameters: Define the categories, their corresponding probabilities, and the sample size.



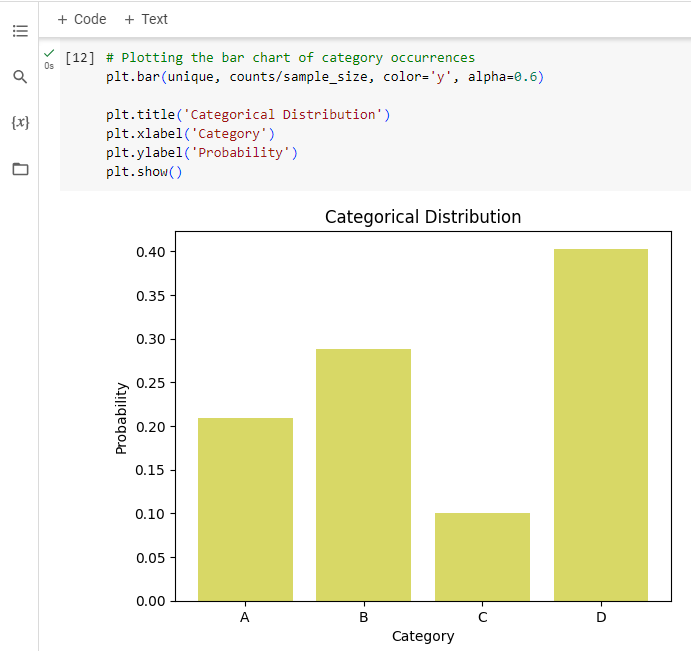
3. Generate random samples: Use 'np.random.choice()' to generate random samples from the categorical distribution with the specified probabilities.



4. Count occurrences and plot: Count the occurrences of each category and plot a bar chart of category probabilities.

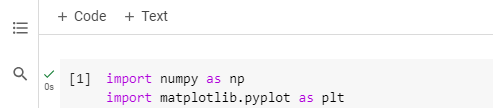


* 'np.unique()' with 'return\_counts=True' provides the count of occurrences for each unique category.
* The bar chart is created using' plt.bar()' with normalized probabilities.

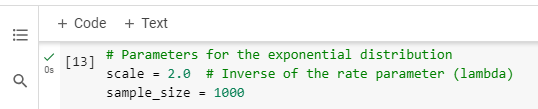


**Exponential Distribution**

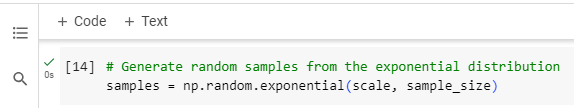
1. Import libraries: Import the required libraries.



2. Set distribution parameters: Specify the scale parameter' (scale)' and the sample size.



3. Generate random samples: Use 'np.random.exponential()' to generate random samples from an exponential distribution with the given scale.



4. Plot the histogram and PDF: Plot a histogram of the generated samples and overlay the exponential probability density function (PDF) on the same plot.

* Similar to the Gaussian distribution, 'plt.hist()' plots the histogram, and 'plt.plot()' overlays the PDF.
* The exponential PDF formula calculates the probability density at each x-value.

