Introduction to Python

MM 225: AI and Data Science

Week 4

This tutorial consists of two parts. In the first part of the tutorial, we learn the `scipy.stats` module and the use of this library for different distributions: for generating random variates, to calculate the probability mass and density functions and the p and q values. In the second part, we solve some of the problems that we discussed in the lectures using python.

Part I

`scipy.stats`

In this section, we will learn the use of `scipy.stats` module for studying the following densities and distributions:

- Continuous uniform;
- Exponential;
- Normal;
- Binomial; and,
- Poisson.

1 Continuous uniform

Let us begin with uniform continuous random variable.

1.1 Generating random deviates

Let us first generate 1000 uniform continuous random variates and plot their histogram. The following script does the job for us.

UniformContinRV.py

```
from scipy.stats import uniform
import matplotlib.pyplot as plt

r = uniform.rvs(size=1000)
plt.hist(r)
plt.show()
```

The plot generated by the script `UniformContinRV.py` is shown in Fig. 1.

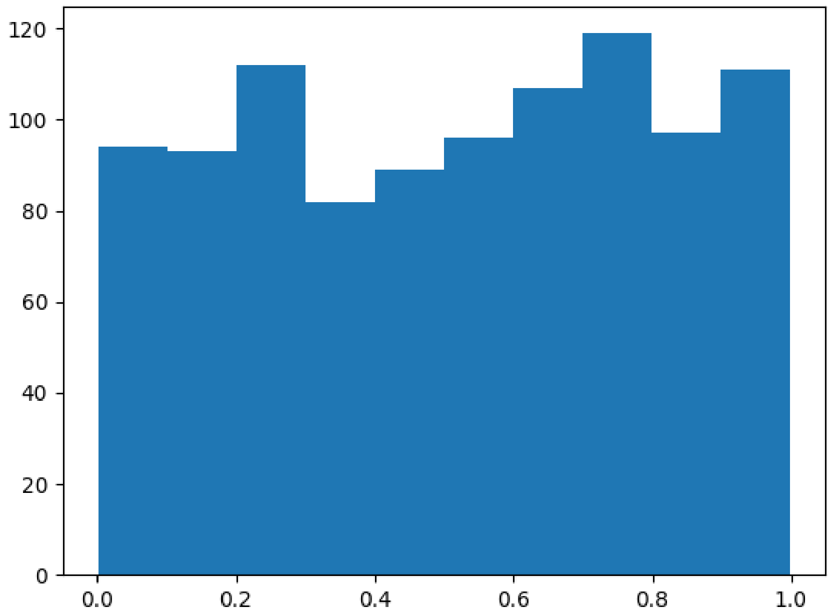


Figure 1: Histogram of thousand random variates between 0 and 1.

As you can see, by default, the random variates are between 0 and 1. But, sometimes, we need random variates in a specific range; let us say we want the random variates in the range 1 to 2. Here is the script that generates uniform variates in the range.

UniformContinRV2.py

```
from scipy.stats import uniform
import matplotlib.pyplot as plt
r = uniform.rvs(loc=1,scale=2,size=1000)
plt.hist(r)
plt.show()
```

Thus, `loc` and `scale`, when given, the random variates generated are in $(loc, loc + scale)$. The plot generated by the script `UniformContinRV2.py` is shown in Fig. 2.

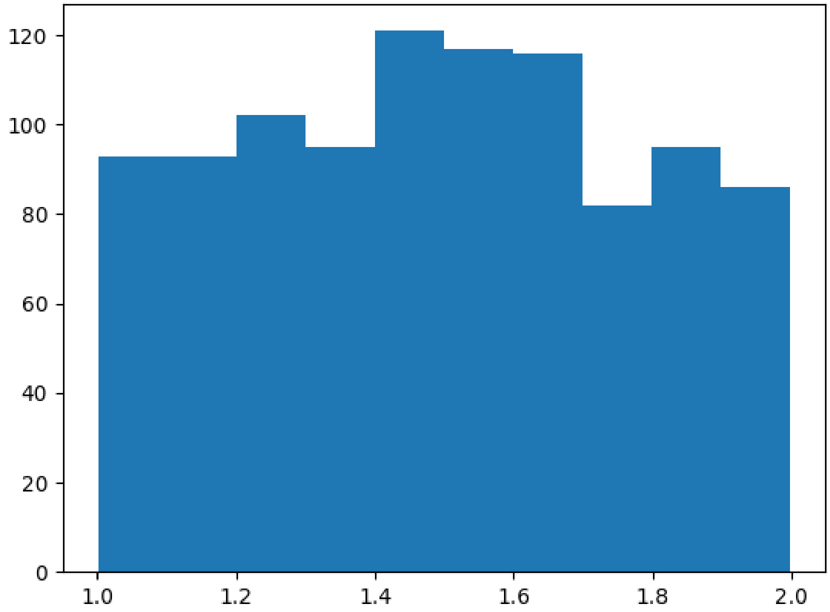


Figure 2: Histogram of thousand random variates between 1 and 2.

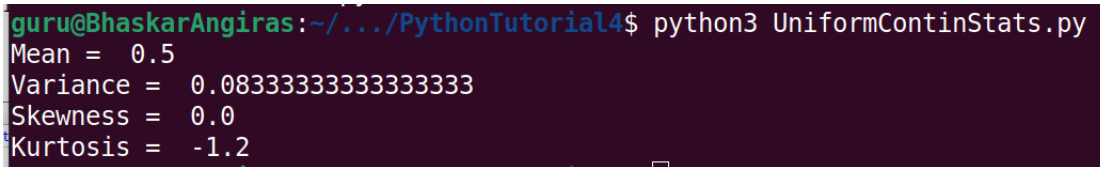
1.2 Statistics

It is possible to do calculate the mean, variance, skewness and kurtosis, the so-called moments (first, second, third and fourth, respectively) of the uniform random distribution. We will learn more about these quantities in one of the lectures in our class. Here is the script that calculates these four quantities and prints them in screen.

UniformContinStats.py

```
from scipy.stats import uniform
import matplotlib.pyplot as plt
mean = uniform.stats(moments="m")
variance = uniform.stats(moments="v")
skewness = uniform.stats(moments="s")
kurtosis = uniform.stats(moments="k")
print("Mean = ",mean)
print("Variance = ",variance)
print("Skewness = ",skewness)
print("Kurtosis = ",kurtosis)
```

The ourput of this script is shown in Fig. 3.



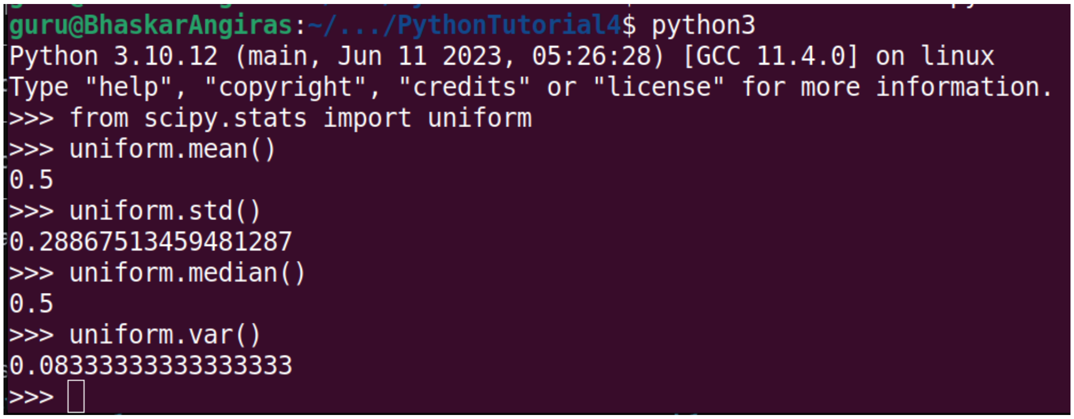
```
guru@BhaskarAngiras:~/.../PythonTutorial4$ python3 UniformContinStats.py
Mean =  0.5
Variance =  0.08333333333333333
Skewness =  0.0
Kurtosis =  -1.2
```

Figure 3: Statistics of the uniform random distribution.

It is also possible to calculate the mean, median, standard deviation and variance of the distribution using the commands as shown below.

1.3 Probability density function (pdf) and cumulative distribution function (cdf)

As we discussed in the class, the parameters of the continuous uniform random variable is the interval (a, b). We can get the probability density function



```
guru@BhaskarAngiras:~/.../PythonTutorial4$ python3
Python 3.10.12 (main, Jun 11 2023, 05:26:28) [GCC 11.4.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> from scipy.stats import uniform
>>> uniform.mean()
0.5
>>> uniform.std()
0.28867513459481287
>>> uniform.median()
0.5
>>> uniform.var()
0.08333333333333333
>>> 
```

Figure 4: Statistics of the uniform continuous random variable.

of the uniform random variable using the `uniform.pdf` command as shown in Fig. 5 below. Note that as seen from Fig. 5, the probability density is consistent with our definition:

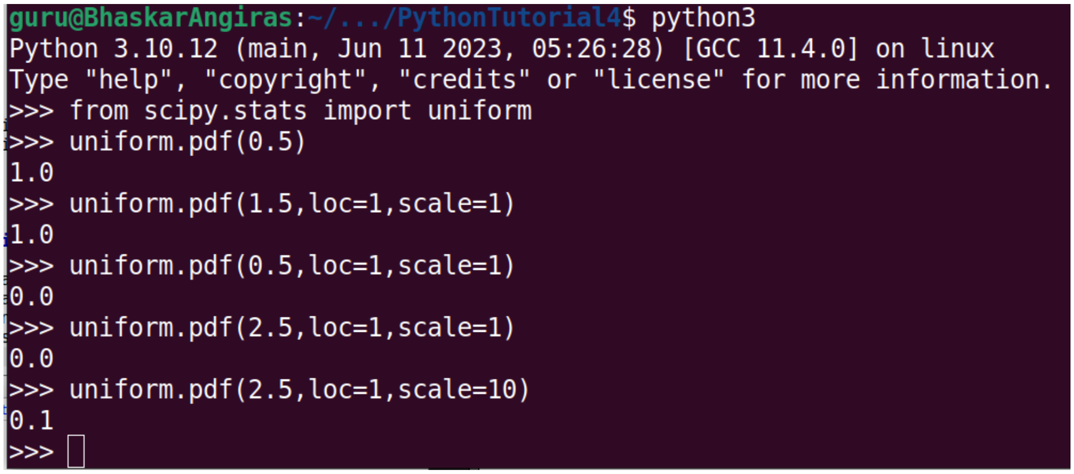
$$f(\omega) = \begin{cases} \frac{1}{b-a} & \text{if } a \leq \omega \leq b \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

Let us also generate and plot the cumulative distribution function (cdf) for the uniform continuous random variable. The script is as shown below and the plot of the cdf is shown in Fig. 6. Note that given Eq. (1), the CDF plot is consistent with our definition, namely, $F(x) = \int f(x)dx$.

UnifContinCDF.py

```
import numpy as np
from scipy.stats import uniform
import matplotlib.pyplot as plt

x = np.linspace(0,1,100)
y = uniform.cdf(x)
plt.plot(x,y)
plt.show()
```



```
guru@BhaskarAngiras:~/.../PythonTutorial4$ python3
Python 3.10.12 (main, Jun 11 2023, 05:26:28) [GCC 11.4.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> from scipy.stats import uniform
>>> uniform.pdf(0.5)
1.0
>>> uniform.pdf(1.5,loc=1,scale=1)
1.0
>>> uniform.pdf(0.5,loc=1,scale=1)
0.0
>>> uniform.pdf(2.5,loc=1,scale=1)
0.0
>>> uniform.pdf(2.5,loc=1,scale=10)
0.1
>>> 
```

Figure 5: Calculating the probability density function for uniform continuous random variable.

2 Normal

Let us repeat the exercises with normal distribution so that you can carry out similar exercises for the other distributions.

2.1 Generating random variates

Let us first generate 1000 random variates from normal distribution and plot their histogram. The following script does the job for us.

NormalRV.py

```
from scipy.stats import norm
import matplotlib.pyplot as plt

r = norm.rvs(size=1000)
plt.hist(r)
plt.show()
```

The plot generated by the script `NormalRV.py` is shown in Fig. 7.

As you can see, by default, the random variates are between 0 and 1. But, suppose we need randome variates from a normal distribution of a specific μ

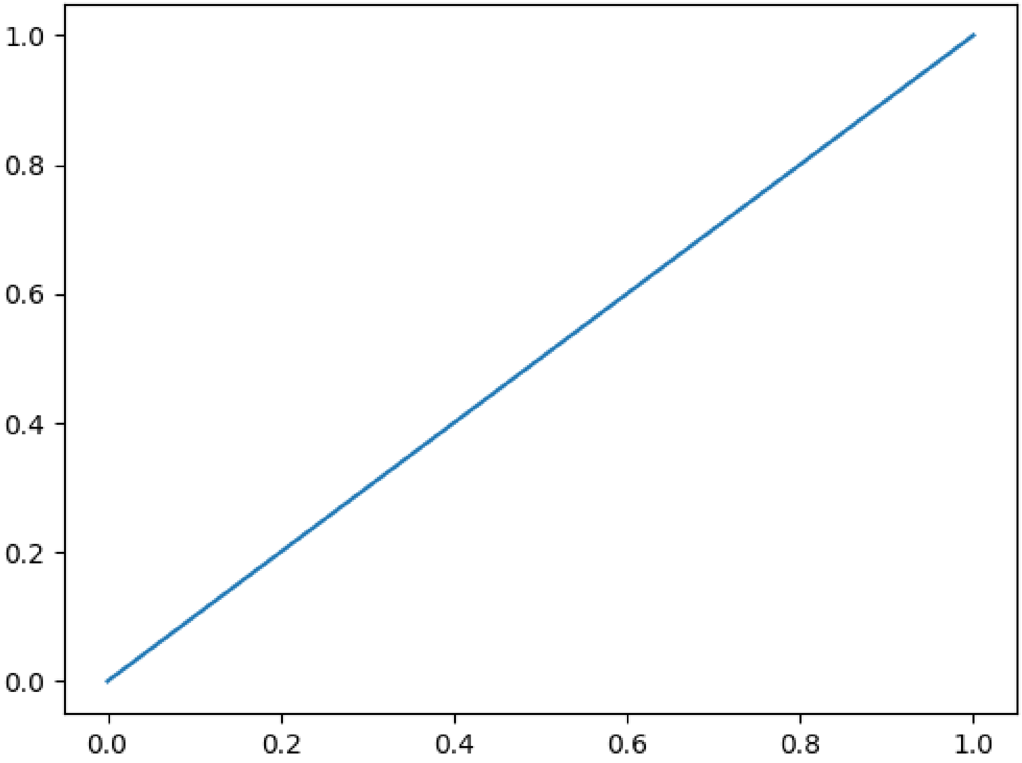


Figure 6: The CDF of the uniform continuous random variable.

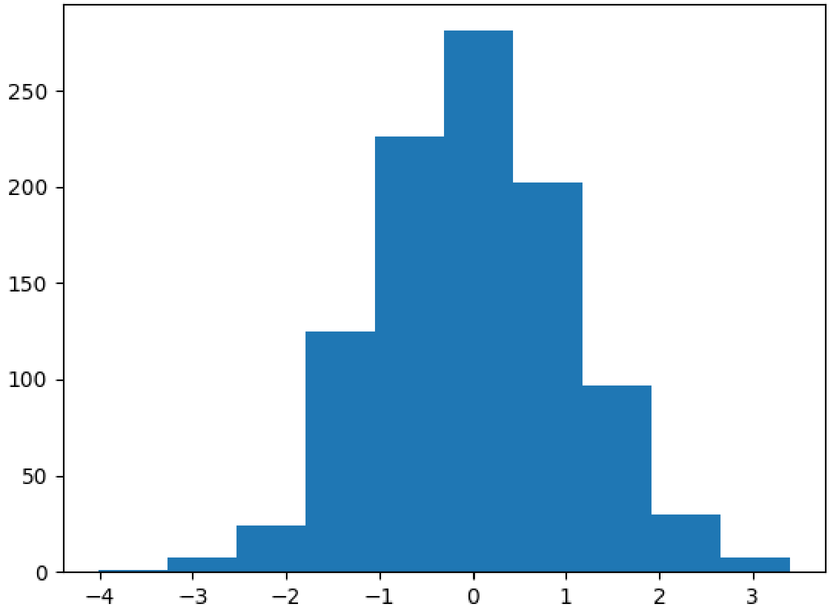


Figure 7: Histogram of thousand random variates from normal distribution. Note that by default the mean μ is assumed to be zero and the variance σ^2 is assumed to be unity.

and σ^2: say, $\mu = 10$ and $\sigma^2 = 2.5$. Here is the script that generates random variates from the normal distribution with $\mu = 10$ and $\sigma^2 = 2.5$.

NormalRV2.py

```
from scipy.stats import norm
import matplotlib.pyplot as plt

r = norm.rvs(loc=10,scale=2.5,size=1000)
plt.hist(r)
plt.show()
```

Thus, `loc` and `scale`, when given, the random variates generated are with mean *loc* and standard deviation *scale*. The plot generated by the script `NormalRV2.py` is shown in Fig. 8.

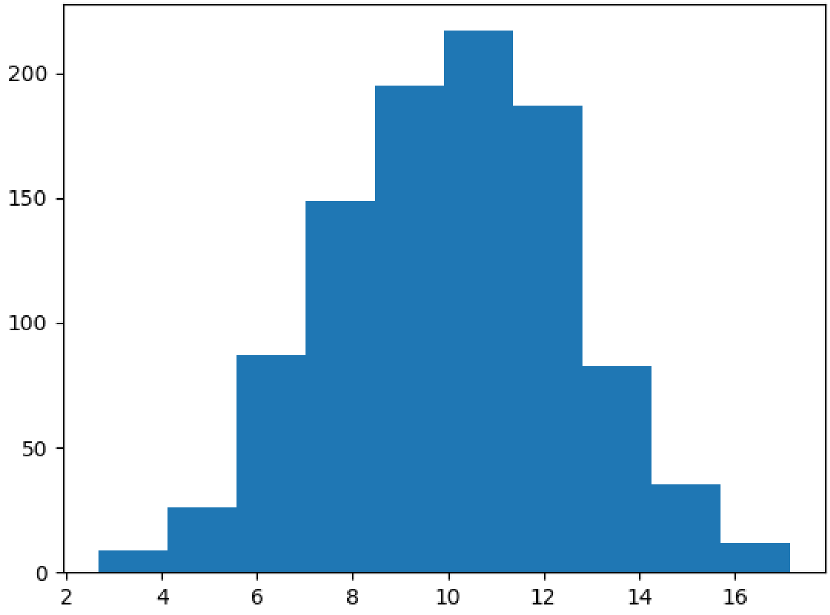


Figure 8: Histogram of thousand random deviates from normal distribution with a mean of 10 and a variance of 2.5.

2.2 Statistics

As earlier, it is possible to do some statistics on the random variates; specifically, we can calculate the mean, variance, skewness and kurtosis, the so-called moments (first, second, third and fourth, respectively). Here is the script that calculates these four quantities and prints them in screen.

UniformContinStats.py

```
from scipy.stats import norm
import matplotlib.pyplot as plt

mean = norm.stats(moments="m")
variance = norm.stats(moments="v")
skewness = norm.stats(moments="s")
kurtosis = norm.stats(moments="k")
print("Mean = ",mean)
print("Variance = ",variance)
print("Skewness = ",skewness)
print("Kurtosis = ",kurtosis)
```

2.3 Cumulative distribution function (cdf)

Let us generate and plot the cumulative distribution function (cdf) for the normal distribution. The script is as shown below and the plot of the cdf is shown in Fig. 9. As noted in our lecture, this is the error function.

NormalCDF.py

```
import numpy as np
from scipy.stats import norm
import matplotlib.pyplot as plt

x = np.linspace(-100,100,100)
y = norm.cdf(x,loc=0,scale=25)
plt.plot(x,y)
plt.show()
```

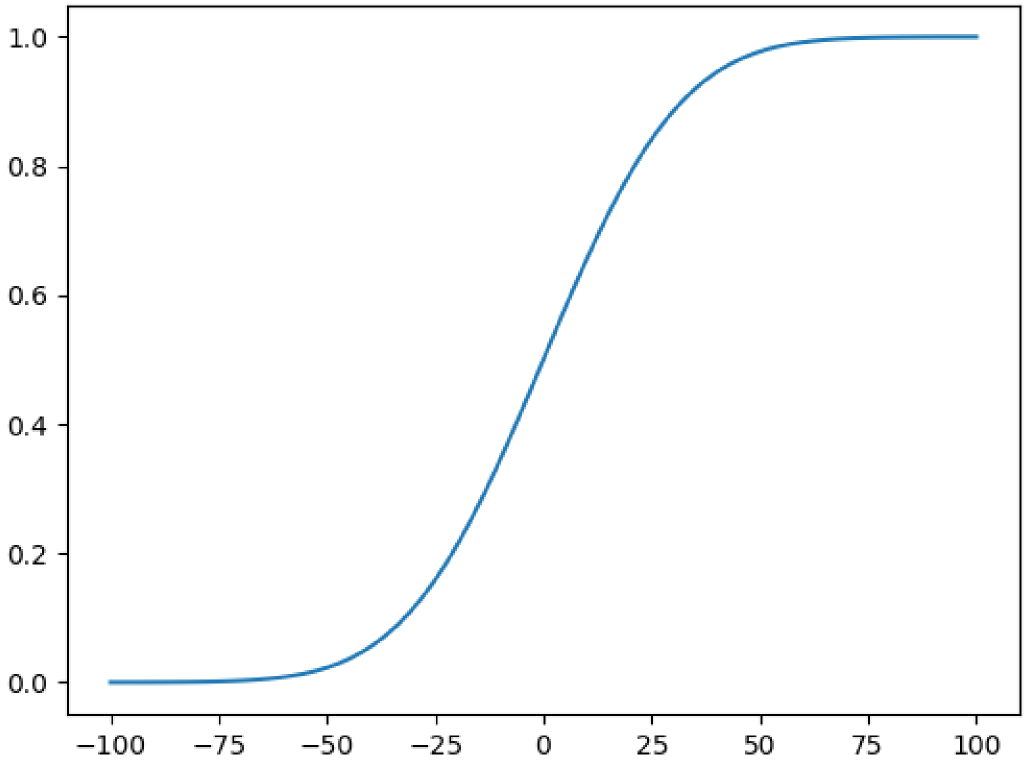


Figure 9: The CDF of the normal random variable with $\mu = 0$ and $\sigma^2 = 25$.

2.4 An aside

It is possible to carry out integration of functions numerically in python. For example, the following script integrates the normal density

$$f_X(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\left[\frac{(x-\mu)^2}{2\sigma^2}\right]} \tag{2}$$

from $-\infty$ to $+\infty$. As expected, the integral is unity.

Integration.py

```
import numpy as np
import math as m
from scipy import integrate

mu = 0
sigmasq = 1

gaussian = lambda x:\
        (1/m.sqrt(2*m.pi*sigmasq))*\
        m.exp(-(x-mu)*(x-mu)/(2*sigmasq))

P, err = integrate.quad(gaussian,-np.inf,np.inf)
print(P)
```

3 Exercise

1. Write a python script to calculate the first four moments, the median, and the standard deviation for a normal distribution with mean $\mu = 100$ and variance $\sigma^2 = 25$. Upload your script in moodle.

2. As discussed in the lecture, the integration of the Eq. (2) from $-\infty$ to x is defined as the error function (denoted as $\mathrm{erf}(x)$). Write a script to numerically integrate

$$f_X(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\left[\frac{(x-\mu)^2}{2\sigma^2}\right]} \tag{3}$$

for $\mu = 0$ and $\sigma^2 = 25$ for various values of x between -100 and 100. Plot the numerical results along with the corresponding cdf in the same plot. Label the two curves. Upload your script and the plot in moodle.

3. Recall the exponential distribution function described in our lecture. Consider the exponential distribution function with $\lambda = 5$. Write a python script which (a) generates 10 random variates from this distribution; (b) prints out the first four moments, the median and the standard deviation of this distribution; and (c) plots the probability density function and the cumulative distribution function for this distribution. Upload your script and the plots in moodle.

   Hint: You need to import from `scipy.stats` the module `expon` and for $\lambda \exp(-\lambda x)$ we need to use the parameter `scale` to be $\frac{1}{\lambda}$.

4. Recall the binomial distribution function described in our lecture. Consider the binomial distribution function with $n = 20$ and $p = 0.5$. Write a python script which (a) generates 10 random variates from this distribution; (b) prints out the first four moments, the median and the standard deviation of this distribution; and (c) plots the probability density function and the cumulative distribution function for this distribution. Upload your script and the plots in moodle.

   Hint: You need to import from `scipy.stats` the module `binom`.

5. Recall the Poisson distribution function described in our lecture. Consider the Poisson distribution function with $\lambda = 0.5$. Write a python script which (a) generates 10 random variates from this distribution; (b) prints out the first four moments, the median and the standard deviation of this distribution; and (c) plots the probability density function and the cumulative distribution function for this distribution. Upload your script and the plots in moodle.

   Hint: You need to import from `scipy.stats` the module `poisson`.

Part II

Problems based on lectures

You may use the previous tutorials and the class lecture slides available in moodle as well as the text of Grinstead and Snell for this part of the tutorial.

4 Exercise

1. Suppose in TechFest, G keeps a stall. The visitors can toss a coin ten times. They get Rs. 2 if H or lose Re. 1 if T. Write a python function to calculate the loss or gain that G makes given an entry fee amount. Write a python script which (a) calls this function assuming for the following sixteen different entry fee amounts: Rs. 5 to Rs. 15 with an increment of Re 1 and (b) plots the loss/gain as a function of entry fee. Upload script and your plot in moodle.

2. Write a python script to simulate the toss of a fair coin a large number of times. Using the script, calculate the number of times the head comes up after an (a) odd and (b) even number of times and plot them as a pie chart. Does the simulation results agree with the analytical expression? Upload your answer, script and the plot in moodle.

3. Write a python script to evaluate π by carrying out Buffon's needle experiment on the computer. What is the error? How does it change with the number of needle throws? Upload your answer, the script and any figures that you may generate in moodle.

4. Write a python function which, given an N, generates N random numbers between [0,1] and adds them up and plots the histogram of the result. Write a python script which calls this function with input values of 10, 100, 1000 and 10000. Upload your script and any figures that you generate in moodle.

5. Let Z be the random variable obtained by adding two real numbers X and Y chosen at random from [0,1] with uniform probability. Write a python script to plot the cumulative distribution and density functions for Z. Upload your script and plots in moodle.