Saeth Wannasuphoprasit
Student ID: 201585689
Maths and Stats for AI and Data Science Assignment 3 (Programming Assignment)

(T1) In the first task the focus is on Binomial distribution in which the probability mass distribution for n trials where the probability of success is given by function $f(n,k,p) = \binom{n}{k} p^k (1-p)^{n-k}$, for all integer $0 \leq k \leq n$.

We know that the recursive definition of $\binom{n}{k}$ is

A - $\binom{n}{0} = 1$

B - $\binom{n}{n} = 1$

C - $\binom{n}{k} = \binom{n-1}{k-1} + \binom{n-1}{k}$ in any other case.

(20 Pts) Establish similar recursive relationship for $f(n,k)$ including

A - $f(n,0,p) = ?$

B - $f(n,n,p) = ?$

C - $f(n,k,p) = ? \bullet f(n-1,k-1,p) + ? \bullet f(n-1,k,p)$ in any other case.

A:

$$f(n,0,p) = \binom{n}{0} p^0 (1-p)^{n-0} = (1)(1)\,(1-p)^{n-0} = (1-p)^n$$

B:

$$f(n,n,p) = \binom{n}{n} p^n (1-p)^{n-n} = (1)\,(p^n)\,(1-p)^0 = p^n$$

C:

$$f(n,k,p) = \binom{n}{k} p^k (1-p)^{n-k} = (C1)\,[f(n-1, k-1, p)] + (C2)[f(n-1, k, p)]$$

$$= (C1)\,[\binom{n-1}{k-1} p^{k-1} (1-p)^{(n-1)-(k-1)}] + (C2)\,[\binom{n-1}{k} p^k (1-p)^{n-1-k}]$$

$$= (C1)\,[\binom{n-1}{k-1} p^{k-1} (1-p)^{n-k}] + (C2)\,[\binom{n-1}{k} p^k (1-p)^{n-k-1}]$$

From $\binom{n}{k} = \binom{n-1}{k-1} + \binom{n-1}{k}$;

we can get the desired outcome f(n, k, p) by

$$[\binom{n-1}{k-1} + \binom{n-1}{k}][p^k(1-p)^{n-k}] = \binom{n}{k}p^k(1-p)^{n-k}$$

So, C1 is p and C2 is 1 - p because

$$(p)\,[\binom{n-1}{k-1}p^{k-1}(1-p)^{n-k}] + (1-p)\,[\binom{n-1}{k}p^k(1-p)^{n-k-1}]$$

is equal to

$$\binom{n-1}{k-1}p^k(1-p)^{n-k} + \binom{n-1}{k}p^k(1-p)^{n-k}$$

which is equal to

$$[\binom{n-1}{k-1} + \binom{n-1}{k}][p^k(1-p)^{n-k}] = \binom{n}{k}p^k(1-p)^{n-k}$$

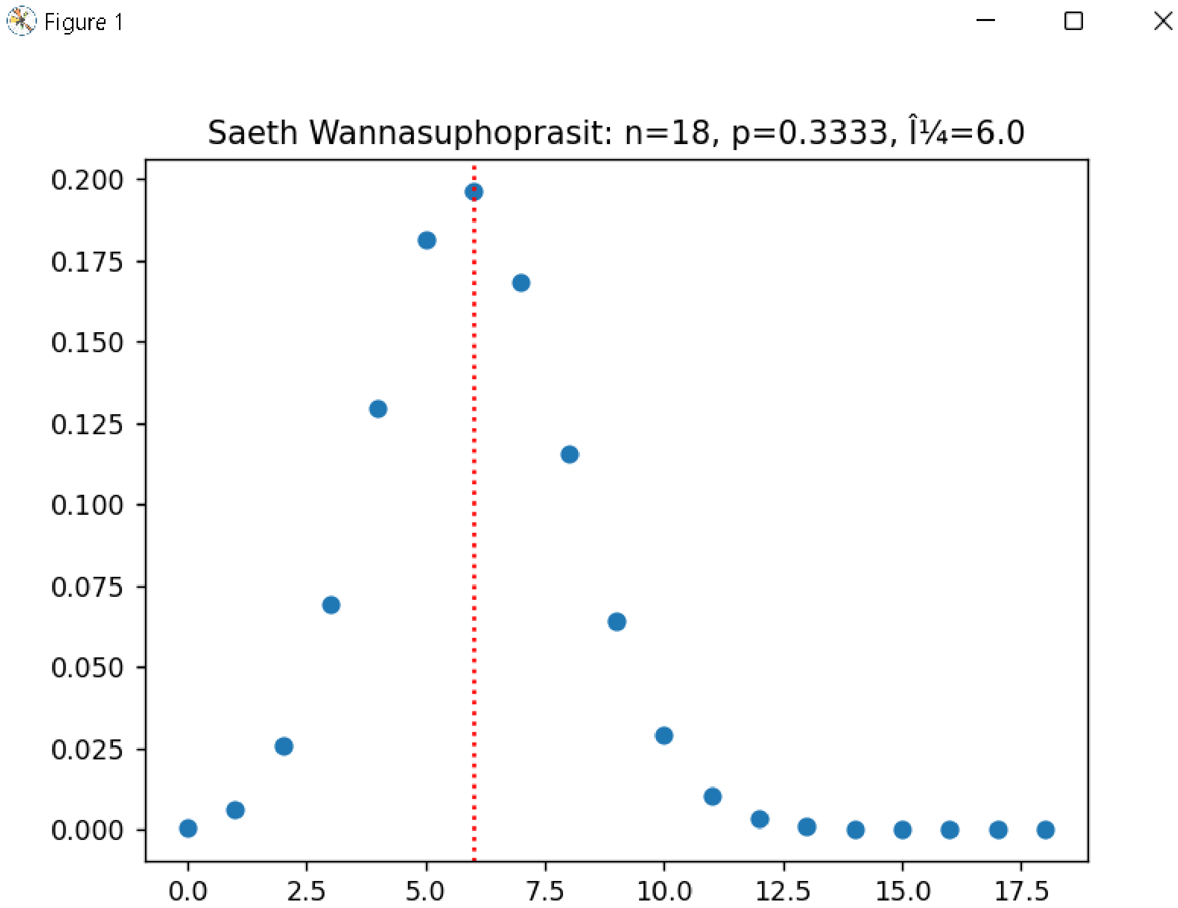
So,

$$f(n,k,p) = p[f(n-1,k-1,p)] + (1-p)[f(n-1,k,p)]$$

(20 Pts) Amend the content of code *Binomial.py* to compute and plot the relevant probability mass distribution. Make sure you change this code only in lines with comment *# <- change here*.

Submit the amended code *Binomial.py* and a note on the recursive definition of $f(n, k, p)$ in a separate (joint for all tasks) pdf document as the solution.

The amended code is in the Binomial.py and the plot result is shown below:



The definition of the recursive f(n, k, p):

The function BinomialRec(n,k,p) will accept the input n (intiger), k(intiger), and p([0, 1])

Note that:
n is the number of trials
k is the total number of “successes”
p is the probability of successes in each trial

The output of this function is the value of f(n, k, p) which from the previous exercise is equal to

$$p[f(n-1,k-1,p)]+(1-p)[f(n-1,k,p)]$$

= p[BinomialRec(n-1, k-1, p)] + (1-p)[BinomialRec(n-1, k, p)]

In order to compute the f(n, k, p) or BinomialRec(n, k, p), as the process continues,
the BinomialRec(n-1, k-1, p) will return
p[BinomialRec(n-2, k-2, p)] + (1-p)[BinomialRec(n-2, k-1, p)]

Then, BinomialRec(n-2, k-2, p) and BinomialRec(n-2, k-1, p) will recall the function itself until the process meets the end of the calculatoin which consists of 2 cases:

1. k = 0 (will return $(1-p)^n$)
2. n = k (will return p^n)

This is also true with the BinomialRec(n-1, k, p).
It will return p[the BinomialRec(n-2, k-1, p)] + (1-p)[the BinomialRec(n-2, k, p)].
Then, BinomialRec(n-2, k-1, p) and BinomialRec(n-2, k, p) will recall the function itself until the process meets the end of the calculatoin mentioned before.

For example, BinomialRec(4, 2, p)
= p[BinomialRec(3, 1, p)] + (1-p)BinomialRec(3, 2, p)
= p[p[BinomialRec(2, 0, p)] + (1-p)[BinomialRec(2, 1, p)]] +
(1-p)[p[BinomialRec(2, 1, p)] + (1-p)[BinomialRec(2, 2, p)]]

define new colors for simple explanation:
p[p[BinomialRec(2, 0, p)] + (1-p)[BinomialRec(2, 1, p)]] +
(1-p)[p[BinomialRec(2, 1, p)] + (1-p)[BinomialRec(2, 2, p)]]

= p[p[$(1-p)^n$] + (1-p)[p[BinomialRec(1, 0, p)] + (1-p)[BinomialRec(1, 1, p)]]] + (1-p)[p[p[BinomialRec(1, 0, p)] + (1-p)[BinomialRec(1, 1, p)]] +
(1-p)[p^n]]

define new colors for simple explanation:
p[p[$(1-p)^n$] + (1-p)[p[BinomialRec(1, 0, p)] + (1-p)[BinomialRec(1, 1, p)]]] + (1-p)[p[p[BinomialRec(1, 0, p)] + (1-p)[BinomialRec(1, 1, p]] + (1-p)[p^n]]

= p[p[$(1-p)^n$] + (1-p)[p[$(1-p)^n$] + (1-p)[p^n]]] + (1-p)[p[p[$(1-p)^n$] + (1-p)[p^n]] + (1-p)[p^n]]

(T2) In the second task the focus is on Poisson distribution with the expected value λ in which the probability mass distribution is given by function $f(k, \lambda) = \frac{\lambda^k e^{-\lambda}}{k!}$, for all integer $k \geq 0$.

(20 Pts) Establish similar recursive relationship for $f(n, k)$ including

A - $f(0, \lambda) = ?$
B - $f(k, \lambda) = f(k-1, \lambda) \bullet ?$, for all integer $k > 0$.

A:

$$f(0, \lambda) = \frac{\lambda^0 e^{-\lambda}}{0!} = \frac{(1)e^{-\lambda}}{1} = e^{-\lambda}$$

B:

$$f(k, \lambda) = \frac{\lambda^k e^{-\lambda}}{k!} = f(k-1, \lambda)B = \frac{\lambda^{k-1} e^{-\lambda}}{(k-1)!}B$$

Note that k! = k(k-1)!

So,

$$\frac{\lambda^k e^{-\lambda}}{k!} = \frac{\lambda^k e^{-\lambda}}{k(k-1)!}$$

We will get

$$\frac{\lambda^k e^{-\lambda}}{k(k-1)!} = \frac{\lambda^{k-1} e^{-\lambda}}{(k-1)!}B$$

So, B is $\frac{\lambda}{k}$

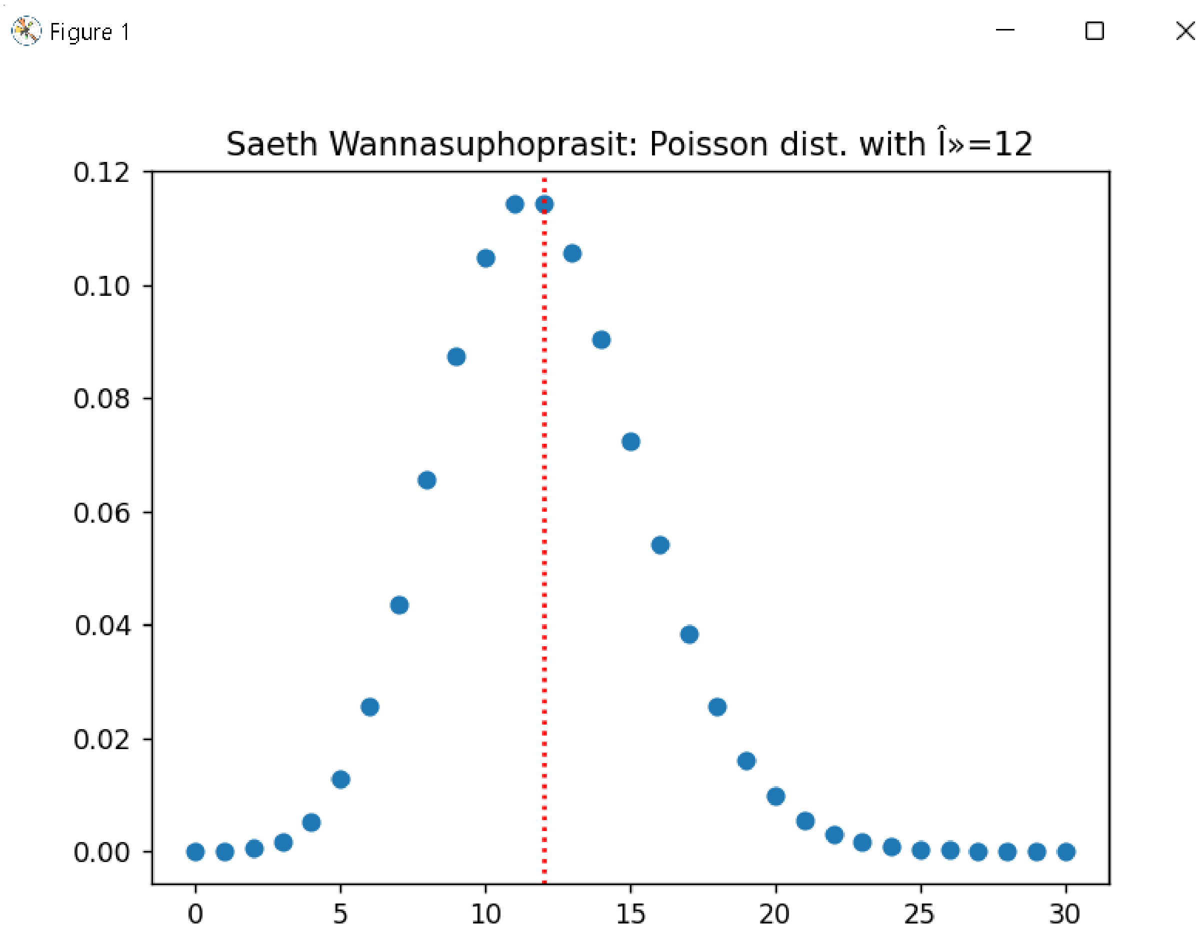
So, the answer to this question is

$$f(k, \lambda) = f(k-1, \lambda)(\tfrac{\lambda}{k})$$

(20 Pts) Amend the content of code *Poisson.py* to compute and plot the relevant probability mass distribution. Make sure you change this code only in lines with comment *# <- change here*. Please note that in the code parameter λ is represented by the variable l.

Submit the amended code *Poisson.py* and a note on the recursive definition of $f(k, \lambda)$ in a separate (joint for all tasks) pdf document as the solution.

The answer is in the Poisson.py and the plot result is shown below:



The definition of the recursive $f(k, \lambda)$:

The function Poisson(k,l) will accept the input k (intiger), and l (real number)

Note that:
k is the number of desired occurrences
l is the mean of the occurrences during the interval

The output of this function is the value of $f(k, \lambda)$ or Poisson(k,l) which from the previous exercise is equal to

$f(k, \lambda) = f(k - 1, \lambda)(\frac{\lambda}{k})$

Which is Poisson(k, l) = Poisson(k-1, l) * (l/k)

In order to compute the $f(k, \lambda)$ or Poisson(k,l), as the process continues, the Poisson(k-1, l) will return

Poisson(k-2, l)(l/(k-1))

Then, Poisson(k-2, l) will recall the function itself until the process meets the end of the calculatoin which is k = 0 (it will return e^{-l})

For example, Poisson(4, l) = Poisson(3, l)(l/4)

= Poisson(2, l)(l/3)(l/4)

define new colors for simple explanation:

Poisson(2, l)(l/3)(l/4)

= Poisson(1, l)(l/2)(l/3)(l/4)

define new colors for simple explanation:

Poisson(1, l)(l/2)(l/3)(l/4)

= Poisson(0, l)(l/1)(l/2)(l/3)(l/4)

define new colors for simple explanation:

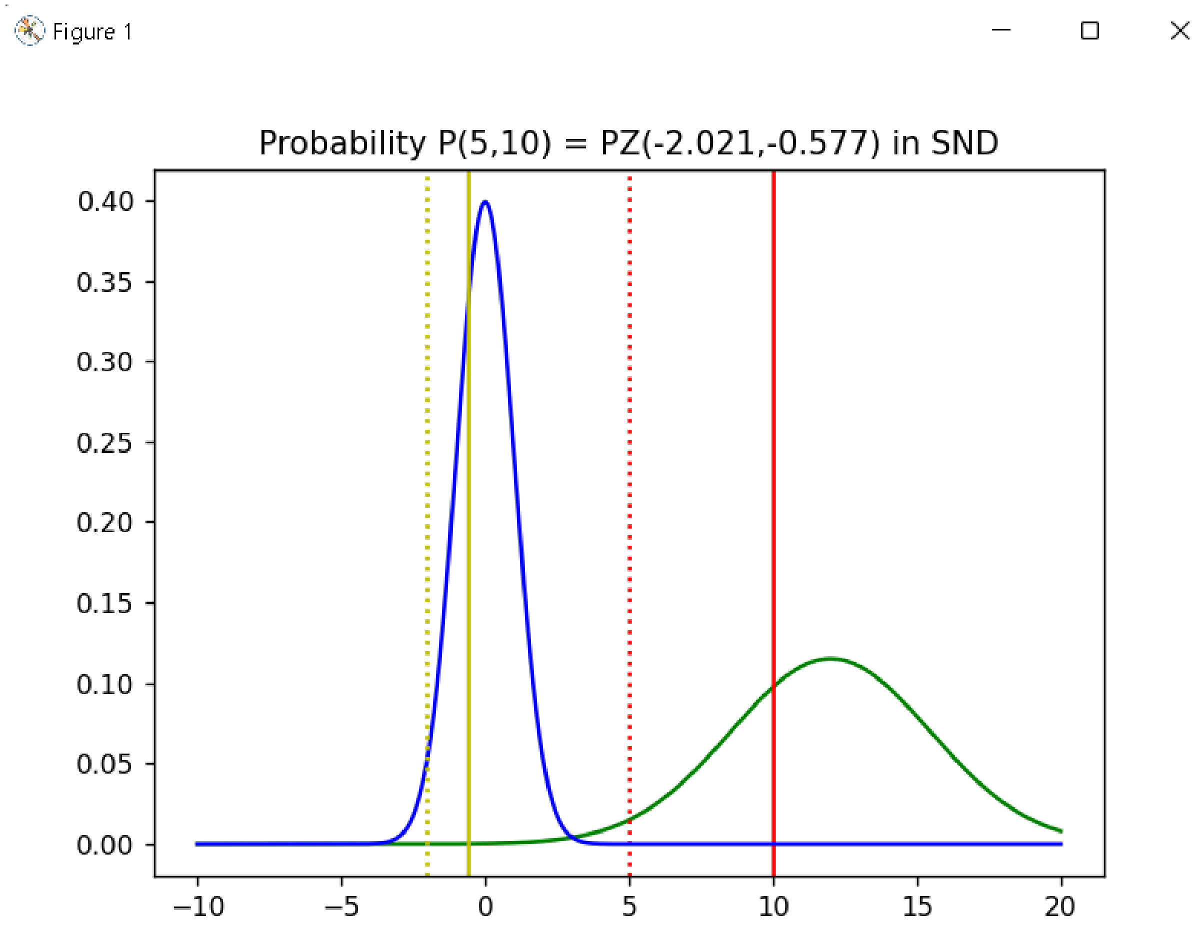
Poisson(0, l)(l/1)(l/2)(l/3)(l/4)

= e^{-l} (l/1)(l/2)(l/3)(l/4)

(T3) In the third task the focus is on Normal (Gaussian) distribution with the expected value μ (in the code mu) and the standard deviation (in the code s) in which the probability density distribution is given by formula $f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$, for all integer $k \geq 0$.

(15 Pts) Amend the content of code *Gaussoan.py* to plot the probability mass distribution in the adopted Normal distribution (AND) based on parameters mu and s determined by the code and to plot the probability mass distribution for standard Normal distribution (SND). In addition compute standardization of two points $X1 = 5$ and $X2 = 10$ in AND to $X1z$ and $X2z$ in SND. Make sure you change this code only in lines with comment *# <- change here*.

The answer is in the Normal.py and the plot result is shown below:



Note that:
The green line is the AND plot (mu = 12 and s = sqrt(mu))
The red lines are the X1 = 5 and X2 = 10 verticle lines
The blue line is the SND plot (mu = 0 and s = 1)
The yellow lines are the X1z = (X1-mu)/s and X2z = (X2-mu)/s verticle lines

(5 Pts) Compute also the probability that a random variable drawn from AND belongs to segment $(X1, X2)$ using points $X1z$ and $X2z$ in SND and precomputed probabilities for SND.

Submit the amended code *Normal.py* **and a short answer to the last answer in a separate (joint for all tasks) pdf document as the solution.**

P(X1< X <X2) = $\int_{X1}^{X2} f(x)dx$ where
X1 = 5,
X2 = 10,
$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}(\frac{x-\mu}{\sigma})^2}$,
mu = 12,
and s = sqrt(12)

So, P(X1< X <X2) =

$$\int_{X1}^{X2} \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2} dx = \int_{5}^{10} \frac{1}{\sqrt{12}\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-12}{\sqrt{12}}\right)^2} dx = 0.26$$

So, P(X1< X <X2) = 0.26

P(X1z < X < X2z) = $\int_{X1}^{X2} f'(x)dx$ where
X1z = (X1 - mu)/s = -2.021,
X2z = (X2 - mu)/s = -0.577,
f'(x) = f(x) when mu = 0 and s = 1.

So, P(X1z < X < X2z) =

$$\int_{X1z}^{X2z} \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2} dx = \int_{-2.021}^{-0.577} \frac{1}{(1)\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-0}{1}\right)^2} dx = 0.26$$

So, P(X1z < X < X2z) = 0.26

And according to Z-table, p(z = -2.021) = 0.9783, p(z = -0.577) = 0.719
So, p(X1z < X < X2z) = p(z = -2.021) - p(z = -0.577) = 0.9783 - 0.719 = 0.26