Module Probabilités

Expliquer et commentaire parties de code

Nous avons choisi de faire le code en python car il nous semblait plus convenable et plus simple pour réaliser le calcul de la probabilité d'une loi Normale

Voici la fonction qui retourne la fonction de la loi normale

$$f x \mapsto \frac{1}{o\sqrt{2\pi}} * \exp^{-\frac{1}{2}*\left(\frac{x-m}{o}\right)^2}$$

Elle prend en paramètre x qui est égale à 0 et m qui sera la valeur instanciée par l'utilisateur dans la page php. Elle est la valeur significative à l'axe des abscisses et correspond au centre de la courbe de la loi normale. Elle prend aussi o en paramètre qui représente l'écart type c'est à dire l'écart de l'axe des abscisses.

Comment avons nous codé ?

- Tout d'abord nous avons fait notre code sur une plateforme web nommé Basthon.
- Nous avons commencé par utiliser une fonction bien connu x^2.
- Nous avons donc retourner cette fonction avec un paramètre pris x.
- Nous avons donc fait pareil pour la fonction de la normale $f x \mapsto \frac{1}{o\sqrt{2\pi}} * \exp^{-\frac{1}{2}*\left(\frac{x-m}{o}\right)^2}$
- Paramètre que l'utilisateur va rentrer : m et o

Entrée [2]:

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

#Fonction de la loi normale
def f(x,m,o):
    return (1/(o*math.sqrt(2*math.pi)))* math.exp(-(x-m)**2/(2*o**2))


m=0
o=1
#m=input("Entrez un nombre qui correspond à la loi normale centrée: ")
#o=input("Entrez un nombre qui correspond à l'écart : ")
f(0,m,o)
```

Out[2]:

```
0.3989422804014327
```

La fonction qui permet de calculer, avec la méthode des rectangles droits, la probabilité de loi normale de P(X<t).

Comment avons nous codé?

- Nous avons commencé par utiliser la formule adapté pour la fonction x^2.
- Nous avons ensuite codés avec la fonction qui retourne la formule de la normale avec paramètre m et o
- Tel que la formule pour cette fonction est $f x \mapsto \frac{b-a}{n} \sum_{k=0}^{k=n} f(ak)$

Elle prend en paramètre t (qui va devenir b borne max), l'utilisateur devra l'instancier dans la page du php tel que P(X<t).
Ici nous avons l'exemple de t=1,2 tel que P(X<1,2) pour N(0,1).
Cela retourne notre air sous la courbe jusqu'a 1.2 et nous avons notre probabilité et nous pouvons vérifier sur la table de la loi normale

Entrée [3]:

```python
#Version rectangle
def methode_rect(f,t,m,o):
    n=1000000  #n: diviser en intervalle
    somme=0    #somme: pour le calcul de l'air en dessous la courbe
    a=0        #a: borne minimum,  elle ne bouge pas car P(X<t)
    x=a        #x: c'est le x de la fonction de f(x), et il devient a car c'est la somme des f(a)

    if(m!=0 and o!=1):  #Si c'est pas la loi normale centrée réduite
        t=(t-m)/o  #Changement de variable pour revenir à la loi normale centrée réduite
        mc=0          #mc: Moyenne de la loi normale centrée réduite
        oc=1          #oc: ecart de la loi normale centrée réduite
        b=t           #b: borne maximum devient t car P(X<t)
        prem=(b-a)/n #calcul d'avant la somme
        for i in range(n):  #Boucle pour la somme
            somme=somme+(f(x,mc,oc))
            x=x+prem
        return somme*prem+0.5

    else:
        b=t
        prem=(b-a)/n
        for i in range(n):
            somme=somme+(f(x,m,o))
            x=x+prem
        return somme*prem+0.5

t=1.2
m=0
o=1
#t=input("Entrez un nombre t de P(X<t) : ")
print(methode_rect(f,t,m,o))
```

```
0.8849304526308839
```

La fonction qui permet de calculer, avec la méthode des trapèzes, la probabilité de loi normale de P(X<t).

Comment avons nous codé?

- Nous avons adapté par rapport à la méthode des rectangles.
- Tel que la formule pour cette fonction est $f x \mapsto \frac{b-a}{2n}\left(f(a)+f(b)+2\sum_{k=0}^{k=n}\ f(ak)\right)$

Elle prend en paramètre t (qui va devenir b borne max), l'utilisateur devra l'instancier dans la page du php tel que P(X<t).
Ici nous avons l'exemple de t=1,2 tel que P(X<1,2) pour N(0,1).
Cela retourne notre air sous la courbe jusqu'a 1.2 et nous pouvons vérifier sur la table de la loi normale.

Entrée [4]:

```python
#version trapèze
def methode_trap(f,t,m,o):
    n=1000000
    somme=0
    a=0
    x=a
    b=t
    f_a_b=f(x,m,o)+f(b,m,o)

    if(m!=0 and o!=1):
        t=(t-m)/o
        mc=0
        oc=1
        b=t
        prem=(b-a)/n
        for k in range(n):
            somme=somme+f(x,mc,oc)
            x=x+prem
        return((prem/2)*(f_a_b+2*somme)+0.5)

    else:
        prem=(b-a)/n
        for k in range(n):
            somme=somme+f(x,m,o)
            x=x+prem
        return((prem/2)*(f_a_b+2*somme)+0.5)


t=1.2
m=0
o=1
#t=input("Entrez un nombre t de P(X<t) : ")
print(methode_trap(f,t,m,o))
```

```
0.8849308085078852
```

La fonction qui permet de calculer, avec la méthode des Simpson, la probabilité de loi normale de P(X<t).

Comment avons nous codé?

- Nous avons adapté par rapport à la méthode des rectangles.
- Tel que la formule pour cette fonction est $fx \mapsto \frac{b-a}{6n}\left(f(a)+f(b)+2\sum_{k=1}^{k=n-1}\ f(a+\frac{k(b-a)}{n}) + 4\sum_{k=0}^{k=n-1}\ f(a+\frac{(2k+1)(b-a)}{2n}\right)$

Elle prend en paramètre t (qui va devenir b borne max), l'utilisateur devra l'instancier dans la page du php tel que P(X<t).
Ici nous avons l'exemple de t=1,2 tel que P(X<1,2).
Cela retourne notre air sous la courbe jusqu'a 1.2 et nous avons notre probabilité et nous pouvons vérifier sur la table de la loi normale.

Entrée [5]:

```
#version Simpson
def methode_simp(f,t,m,o):
    n=1000000
    a=0
    x=a
    b=t

    somme=0
    som=0

    f_a_b=f(x,m,o)+f(b,m,o)

    if (m!=0 and o!=1):
        t=(t-m)/o
        mc=0
        oc=1
        b=t
        prem=(b-a)/n

        for k in range(n):
            x=((k*(b-a))/n)
            somme=somme+f(x,mc,oc)

        for j in range(n):
            x=(((2 * j +1)*(b-a))/(2*n))
            som=som+f(x,mc,oc)

        return((prem/6) * (f_a_b+ 2 * somme + 4 * som)+0.5)

    else:
        prem=(b-a)/n
        for k in range(n):
            x=((k*(b-a))/n)
            somme=somme+f(x,m,o)

        for j in range(n):
            x=(((2 * j +1)*(b-a))/(2*n))
            som=som+f(x,m,o)

        return((prem/6) * (f_a_b+ 2 * somme + 4 * som)+0.5)


t=1.2
m=0
o=1
#t=input("Entrez un nombre t de P(X<t) : ")
print(methode_simp(f,t,m,o))
```

```
0.884930489355203
```

Cette méthode des simpson retourne un résultat encore plus précis

Affichage de la courbe :

Entrée [17]:

```
plt.clf()
interval=np.arange(m-4,o,0.2)   #Intervalle pour les rectangles
axe = np.linspace(m-6,m+4, 100)  #tracage de la courbe, et de l'axe
plt.plot(axe, norm.pdf(axe, m, o))

for i in range(len(interval)-1):      #Tracer rectangle pour chaque intervalle
    plt.fill_betweenx([0, norm.pdf(interval[i],m,o)],interval[i],interval[i+1], color='gray',alpha=0.5)

plt.show()
```

Test

Entrée [9]:

```python
# Test pour loi normale centrée et réduite
m = 0
o = 1

# Test the function for t = 1
t = 1
assert math.isclose(methode_rect(f, t, m, o), 0.84, rel_tol=1e-2)
assert math.isclose(methode_trap(f, t, m, o), 0.84, rel_tol=1e-2)
assert math.isclose(methode_simp(f, t, m, o), 0.84, rel_tol=1e-2)


# Test the function for t = -1
t = -1
assert math.isclose(methode_rect(f, t, m, o), 0.16, rel_tol=1e-2)
assert math.isclose(methode_trap(f, t, m, o), 0.16, rel_tol=1e-2)
assert math.isclose(methode_simp(f, t, m, o), 0.16, rel_tol=1e-2)

# Test the function for t = 1.2
t = 1.2
assert math.isclose(methode_rect(f, t, m, o), 0.89, rel_tol=1e-2)
assert math.isclose(methode_trap(f, t, m, o), 0.89, rel_tol=1e-2)
assert math.isclose(methode_simp(f, t, m, o), 0.89, rel_tol=1e-2)

# Test pour loi normale non centrée non réduite
m = 10
o = 2

# Test the function for t = 11
t = 11
assert math.isclose(methode_rect(f, t, m, o), 0.69, rel_tol=1e-2)
assert math.isclose(methode_trap(f, t, m, o), 0.69, rel_tol=1e-2)
assert math.isclose(methode_simp(f, t, m, o), 0.69, rel_tol=1e-2)

# Test the function for t = 9
t = 9
assert math.isclose(methode_rect(f, t, m, o), 0.31, rel_tol=1e-2)
assert math.isclose(methode_trap(f, t, m, o), 0.31, rel_tol=1e-2)
assert math.isclose(methode_simp(f, t, m, o), 0.31, rel_tol=1e-2)


m = 3
o = 4
t = 5
assert math.isclose(methode_rect(f, t, m, o), 0.69, rel_tol=1e-2)
assert math.isclose(methode_trap(f, t, m, o), 0.69, rel_tol=1e-2)
assert math.isclose(methode_simp(f, t, m, o), 0.69, rel_tol=1e-2)
assert not math.isclose(methode_rect(f, t, m, o), 0.84, rel_tol=1e-2) ## Le résultat entré est faux
```

Choix fait pour interfacer avec le web

On a fait le choix de le faire en plusieurs script, tout d'abord. Nous avons utilisé :

- SimFast-Module_Probabilite.php : c'est l'interface pour afficher les probabilités
- loi_normale.py : pour la fonction de la loi normale
- methode_rectangles.py : script pour la methode des rectangles
- methode_trapezes.py : script pour la methode des trapèzes
- methode_simpson.py : script pour la methode des simpsons
- graphes.png : Enrengistrement du graphe
- normal_law.php : page en php qui fait le lien entre le script python et l'interface

Code pour intégrer le python au php :

Entrée [ ]:

```php
<?php
session_start();
include '../../Config/database.php';
global $db;

//Validation du formulaire
if(isset($_POST["valider"])){
    $updatemot_dernier_module = $db->prepare("UPDATE utilisateur SET dernier_module = ? where login=?");
    $updatemot_dernier_module->execute(array("Probabilite", $_SESSION['login']));

    //Recuperer la methode et les parametre mis dans l interface
    $methode_choisie = $_POST['methode'];
    $mu = $_POST['mu'];
    $sigma = $_POST['sigma'];
    $t = $_POST['t'];

    $methode_rectangles ="/usr/bin/python3  ../python/methode_rectangles.py";
    $methode_trapezes ="/usr/bin/python3 ../python/methode_trapezes.py";
    $methode_simpsons="/usr/bin/python3 ../python/methode_simpson.py";
    //va chercher les dossiers python


    if($methode_choisie=="methode_rectangle"){
       $result = exec($methode_rectangles. ' '. $mu .' '. $sigma . ' ' . $t);
        //met les arguments récupérer dans l interface pour mettre dans le script python methode rectangles

    }elseif ($methode_choisie=="methode_trapeze"){
        $result = exec($methode_trapezes. ' '. $mu .' '. $sigma. ' ' . $t);
    }
    else{
        $result = exec($methode_simpsons. ' '. $mu .' '. $sigma. ' ' . $t);
    }

    header("Location: ../SimFast-Module_Probabilite.php?result=$result");
    //affiche le resultat dans l interface
}

```