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

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

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 $fx \mapsto \frac{1}{o\sqrt{2\pi}} * \exp^{-\frac{1}{2}*(\frac{x-m}{o})^2}$
- Paramètre que l'utilisateur va rentrer : m et o

In [2]:
```
import math
def f(x,m,o):
    if(o>0):
        premier=(1/(o*math.sqrt(2*math.pi)))
        deuxieme=math.exp(-1/2*((x-m)/o)**2)
        return premier*deuxieme
    else:
        print("Paramètre o ne peut pas être inférieur à 0")
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
```

Attention : Ici c'est une loi normale de type centrée et réduite nous avons pas encore fait un changement de variable pour la ramener à une loi normale de type centrée et réduite en cas de changement de m et o

- voila la condition pour le changement de variable:
  - si m != 0 et o !=1 il faut faire t=(t-m)/o

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 $fx \mapsto \frac{b-a}{n}\sum_{k=0}^{k=n} f(ak)$

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).
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

In [5]:
```
def methode_rect(f,t):
    n=1000000     #n: diviser en intervalle
    a=0           #a: borne minimum à 0 car pas de bornes minimum
    b=t           #b: borne maximum devient t car P(X<t)
    prem=(b-a)/n  #calcul d'avant la somme
    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)
    somme=0       #somme: pour le calcul de l'air en dessous la courbe
    for i in range(n):  #Boucle pour la somme
        somme=somme+(f(x,m,o))  #On somme la fonction de la loi normale
```

```
        x=x+prem                #On avance pas à pas
    return somme*prem+0.5       #On return +0.5

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

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 $fx \mapsto \frac{b-a}{2n}\left(f(a)+f(b)+2\sum_{k=0}^{k=n} f(ak)\right)$

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).
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.

In [6]:
```
def methode_trap(f,t):
    n=1000000     #n: diviser en intervalle
    a=0           #a: borne minimum à 0 car pas de bornes minimum
    b=t           #b: borne maximum devient t car P(X<t)
    prem=(b-a)/n  #calcul d'avant la parenthèse sans le 2 qui divise
    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)
    somme=0       #somme: pour le calcul de l'air en dessous la courbe
    f_a_b=f(x,m,o)+f(b,m,o)  #addition des fonctions avant la somme
    for k in range(n):       #boucle de la somme
        somme=somme+f(x,m,o)  #somme de la fonction f(a)
        x=x+prem             #pas à pas
    return((prem/2)*(f_a_b+2*somme)+0.5)   #return la formule
print(methode_trap(f,1.2))
```

0.8849308085078852

Cette méthode des trapèzes retourne un résultat plus précis

Choix fait pour interfacer avec le web

Code pour intégrer le python au php :

In [ ]:
```
<?php
    echo "<h1>Module de probabilité</h1><br>";

    echo "<style>
    form{font-weight: normal;font-size: large;}
</style>";

    echo "<form method='post'>
    m = <input type='number' name='m'><br>
    o = <input type='number' name='o'><br>
    <input type='submit' value='Valider' name='v'>

</form>";

    if(isset($_POST['v'])){

        $m = $_POST['m'];
        $o = $_POST['o'];
        $c = "/Documents/Scripts/pip.exe";//répèrtoire de l'interpreteur python
        $f ="/script_python/loi_normale.py"; // chemin d'acces scripts python
        $result = exec("/Documents/Scripts/pip.exe /script_python/loi_normale.py 3 5");

        echo $result;
    }


?>
```

Code pour récupérer les arguments saisis par l'utilisateur sur l'application

In [ ]:
```
import sys

def main():
    if len(sys.argv) != 3:
        print ("pas assez d'arguments")
        return

    print sys.argv[1] //affiche arg1
```

```
    print sys.argv[2] //affiche arg2

main()
```

Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js