Rendu du projet de statistique

Khadim CISSE

Seyni KAIRE

Abdoulahi MBENGUE

Réponses aux questions

Question 1 : simulation de lois

1- Loi Binomiale

```
 import numpy.random as npr
import matplotlib.pyplot as plt
import numpy as np
nbre=10000                          #TAILLE
DE L'ECHANTILLON DE LA LOI
B(30,0.2)
t=npr.binomial(30,0.2,nbre)
print(t)
p=[0]*30
for i in range(1,30):
    p[i]=t.tolist().count(i)/nbre
#CALCUL DE LA FREQUENCE
D'APPARUTION DE CHAQUE
RESULTAT
```

```
    plt.hist(t,29,facecolor='green',range=
    (0.5,29.5))
    plt.title("échantillon de taille 10000
    suivant une loibinomiale
    B(30,0.2)".format(nbre))
    plt.show()
```

2- Loi Normale

```
from math import *
#a=int(input("moyenne"))
#b=int(input("l'ecart type"))
x=np.linspace(-10, 20, 10000)
k=1/(2*sqrt(2*pi))
l=(-1/8)*((x-3)**2)
print(k)
puissance=np.exp(l)
y=k*puissance
print(x)
plt.figure()
plt.plot(x,y, label='loi normale')
plt.title('un échantillon de taille 10000
suivant une loi normale N(3,.4)')
plt.xlabel('axe x')
plt.ylabel('axe y')
plt.legend()
```

```
plt.show()
print(y)
```

3- Simulation Loi Gamma

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from scipy.stats import gamma
gamma_distribution = gamma(loc = 10, scale = 10, a = 5)
x = np.linspace(-5, 30, 10000)
_, ax = plt.subplots(1, 1)
ax.plot(x, gamma_distribution.pdf(x), 'r', lw=4)
print(x)
plt.title(' un échantillon de taille 10000 suivant une loi gamma γ(10,.5).')
plt.show()
print(gamma_distribution.pdf(x))
```

Question 2 : Regression Lineaire simple

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
data = pd.read_excel('Couple.xlsx')
data.head()
#1
```

```
import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

data = pd.read_excel('Couple.xlsx')

xi=[18,7,14,31,21,5,11,16,26,29]

yi=[55,17,36,85,62,18,33,41,63,87]

plt.plot(xi,yi,c="gray")

plt.show()

#2

# A la vue de cette représentation, nous pouvons soupconnner une relation linéaire entre les deux variables xi et yi.

# Nous pouvons constater des points de contact des deux vaiables en différents points.

import numpy as np

import matplotlib.pyplot as plt

xi=[18,7,14,31,21,5,11,16,26,29]

yi=[55,17,36,85,62,18,33,41,63,87]

coeff2=np.polyfit(xi,yi,1)

print("les coefficients de la droite des moindres carrées:"+str(coeff2))

f=np.poly1d(coeff2)

print("la droite de régression:"+str(f))

listeXi=[]

listeYi=[]

for i in np.arange(1,30,0.05):

    listeXi.append(i)

    listeYi.append(f(i))

plt.plot(listeXi,listeYi,c="red",label="regression")
```

```
plt.plot(xi,yi,c="gray",label="representation")
plt.legend()
plt.show()
#3:
#les coefficients de la droite des moindres carrées: [2.7347561 ; 1.02134146]

#4:
#Les ordonées de yi:
#La droite de régression est 2.735x+1.021
#Si x=5 alors y=14.7; si x=15 alors y=42.05

#5:
#La droite des moindres carrées est tracée sur le meme graphe en rouge

#6
#une estimation plausible de Y à xi=21 est 62 suivant le tableau [xi,yi]

#La valeur estimé avec la droite de régression en xi=21:
yi=2.2735*21+0.21
print("la valeur estimée avec la droite de régression est :"+str(yi))
#7
#L'écart entre la valeur observé en Y et
#la valeur estimé avec la dreoite de régression est :
P=62
p=47.95
```

```
print("l'écart entre les deux valeurs est:"+str(P-p))

print("cet écart est appelé RESIDU DE LA REGRESSION")


#8

#on cherche le point (¯ x, y¯)

print("moyenne des xi:"+str(np.mean(xi)))

print("moyenne des yi:"+str(np.mean(yi)))

yi=2.2735*17.8+0.21

print("coordonnée de yi si xi=17.8 est:"+str(yi))

#La droite des moindres carrées ne passe pas par le point(¯ x, y¯)

#A n'importe laquelle droite de régression,nous pouvons généraliser cette conclusion
```

Question 3: Donnees Reelles

```
#Importation des bibliotheque de numpy

import numpy as np

#Importation des bibliotheque de matplotlib

import matplotlib.pyplot as plt

#Importation des bibliotheque de pandas

import pandas as pd

#Enregistrement des donnees sous forme csv

pd.read_csv('donnees_smp.csv',sep=";")

#on a une structure de 799 observations et 26 variables.

# et chargement des donnees dans un autre variable appelee

data

data=pd.read_csv('donnees_smp.csv',sep=";")
```

```
#la moyenne, la variance, et l’écart type de la variable age
data.age.describe()
#la moyenne, la variance, et l’écart type de la variable age
data['n.enfant'].describe()

#Construction du plot de la variable age
data['age'].plot()

#verifications des types de variables obtenues
data.info()

#la mediane, la moyenne, l'ecartype de "n.enfant"
data['n.enfant'].describe()

#la mediane, la moyenne, l'ecartype de "n.fratrie"
data['n.fratrie'].describe()

#la mediane, la moyenne, l'ecartype de "dur.interv"
data['dur.interv'].describe()

#les quantiles de "age"
data['age'].quantile(0.25)

#les quantiles de "age"
data['age'].quantile(0.5)
```

```
#les quantiles de "age"
data['age'].quantile(0.75)

#les quantiles de "age"
data.loc[data['prof']=="agriculteur",:]

#definition des agriculteur qui ont plus de 2 enfants
print(data.loc[(data['prof']=="agriculteur")&(data['n.enfant']>
2),:])

#Construction du plot de la variable age
data.boxplot(column='age')

#Construction du diagramme en secteurs  de la variable 'prof'
data['prof'].value_counts().plot.pie()

# la moyenne des ages par professions
data.groupby(['age','prof']).mean()

print(data['prof'])

data['ageQ']=pd.qcut(data.age,3,labels=['age1','age2','age3'])
data['ageQ'].describe()
df=data.dropna(axis=0)
```

```
df.info()

df.shape

data['age'].hist()

data['age'].hist()&data['age'].d
```

Question 4: Methode de Monte Carlo

```
import random

from math import sqrt

from matplotlib.pyplot import*

import matplotlib.pyplot as plt

from scipy.stats import uniform


#permet d'importer toutes les variables aleatoires disponible


def integrale_mc(f, a, b, n):
    somme = 0
    for i in range(0, n):
        x = random.uniform(a, b)
        somme += f(x)


    return somme / n


# On estime la valeur de l'integrale de I2.

a=integrale_mc(lambda x: sqrt(1-x**2),0,1,10000)
```

```
print("une estimation de I2 nous donne",a)
```