In [2]:
```
import pandas as pd
import numpy as np
import matplotlib.pylab as plt
from datetime import datetime
import seaborn as sns
from scipy import stats as st

#importation du fichier df2
df1 = pd.read_csv("df2.csv")
df1.head()
```

Out[2]:

	Zone	Disponibilite alimentaire (Kcal/personne/an)	Disponibilite alimentaire en quantite (kg/personne/an)	Disponibilite de proteines animale en quantite (kg/personne/an)	Disponibilite interieure (kg/an)	Disponibilite de proteines vegetale en quantite (kg/personne/an)	Proteine totale	%Proteine animale	% Evolution13_14	PIB
0	Afghanistan	78840.0	79.92	4.45665	2.631000e+09	46.05	50.50665	8.823888	9.226218	1.962180e+10
1	Afrique du Sud	177755.0	139.63	13.28235	7.817000e+09	48.94	62.22235	21.346590	3.350360	3.666449e+11
2	Albanie	359525.0	389.05	21.68465	1.432000e+09	51.96	73.64465	29.444977	-8.720296	1.277628e+10
3	Algerie	137970.0	176.78	9.11770	7.120000e+09	66.94	76.05770	11.987872	-0.725128	2.097550e+11
4	Allemagne	380330.0	388.70	22.44385	3.540900e+10	39.90	62.34385	36.000103	-1.543175	3.732743e+12

In [3]:
```
#Pour verifier la distribution normale de la variable j'utilise le test de shapiro
from scipy.stats import shapiro

stat, p = shapiro(df1['Proteine totale'])

print('Statistics=%.3f, p=%.3f' % (stat, p))

#Les hypotheses:
#H0: La distribution de la variable choisi suit une loi normale/gaussien
#H1: La distribution de la variable choisi ne suit pas une loi nrmale
#Si la p-value > 0.05, le niveau de significance choisi, nous ne pouvons pas rejeter H0,
#la distribution de la var suit la loi normale

#Si la p-value < 0.05 le niveau de significance choisi, nous rejetons H0 et appliquant H1.
#La distribution de la variable ne suit pas la loi normale dans ce cas

#Interprétation
alpha = 0.05
if p > alpha:
    print('On ne peut pas rejeter H0 car p > alpha')
else:
    print('H0 est rejetée pour une p < alpha')
```

```
Statistics=0.992, p=0.409
On ne peut pas rejeter H0 car p > alpha
```

In [4]:
```
#help(shapiro)
```

Dans la suite je compare la var "Proteine Totale" qui suit une loi normale de deux cluster differents. Pour ceci j'utilise le test de Bartlett. Ce test verifie la heterogeneité de la variance pour pouvoir conclure si les deux echantillons provient de la meme population ou non. Dans notre cas c'est interessant a cause du choix du cluster.

In [6]:
```
df2 = pd.read_csv("df_cls.csv")
df2.head()
```

Out[6]:

	Zone	Disponibilite alimentaire (Kcal/personne/an)	Disponibilite alimentaire en quantite (kg/personne/an)	Disponibilite de proteines animale en quantite (kg/personne/an)	Disponibilite interieure (kg/an)	Disponibilite de proteines vegetale en quantite (kg/personne/an)	Proteine totale	%Proteine animale	% Evolution13_14	PIB	cluster
0	Afrique du Sud	177755.0	139.63	13.28235	7.817000e+09	48.94	62.22235	21.346590	3.350360	3.666449e+11	0
1	Albanie	359525.0	389.05	21.68465	1.432000e+09	51.96	73.64465	29.444977	-8.720296	1.277628e+10	2
2	Algerie	137970.0	176.78	9.11770	7.120000e+09	66.94	76.05770	11.987872	-0.725128	2.097550e+11	0
3	Allemagne	380330.0	388.70	22.44385	3.540900e+10	39.90	62.34385	36.000103	-1.543175	3.732743e+12	2
4	Arabie saoudite	189435.0	174.86	14.57080	4.897000e+09	51.67	66.24080	21.996715	7.241309	7.466471e+11	0

In [9]:
```
#choix au hazard du cluster 0 et 2 pour la var 'Proteine totale'
cluster1 = df2['Proteine totale'][df2['cluster']==0]
cluster2 = df2['Proteine totale'][df2['cluster']==2]
```

In [11]:
```
#On teste tout d'abord l'égalité des variances à l'aide de la commande
from scipy.stats import bartlett
stat, p = bartlett(cluster1, cluster2)
print('Statistics=%.3f, p=%.3f' % (stat, p))

#Les hypotheses:
#H0: La variable choisi affiche la meme variance dans les deux cluster
#H1: La variable choisi n'affiche pas la meme variance dans les deux cluster
#Si la p > 0.05, le niveau de significance choisi, nous ne pouvons pas rejeter H0,
#La variable choisi affiche la meme variance dans les deux cluster, les var des deux cluster
#provient de la meme population

#Si la p < 0.05 le niveau de significance choisi, nous rejetons H0 et appliquant H1.
#La variable choisi n'affiche pas la meme variance dans les deux cluster dans ce cas,
#les var des deux cluster ne provient pas de la meme population

#Interprétation
alpha = 0.05
if p > alpha:
    print('On ne rejette donc pas H0 car les var afiche la meme variance. Ils provient donc de la meme population')
else:
    print('H0 est rejetée car p < alpha. Les var ne provient pas de la meme population')
```

```
Statistics=3.024, p=0.082
On ne rejette donc pas H0 car les var afiche la meme variance. Ils provient donc de la meme population
```

In [12]:
```
#help(bartlett)
```

Je test egalement les moyennes de la meme var avec un t-test

In [15]:
```
#On teste ensuite l'égalité des moyennes à l'aide de la commande
from scipy.stats import ttest_ind
stat, p = ttest_ind(cluster1, cluster2, equal_var=True)
print('Statistics=%.3f, p=%.9f' % (stat, p))

#Les hypotheses:
#H0: La variable choisi affiche les memes moyennes dans les deux cluster
#H1: La variable choisi n'affiche pas la meme variance dans les deux cluster
#Si p > 0.05, le niveau de significance choisi, nous ne pouvons pas rejeter H0,
#La variable choisi affiche la meme moyenne dans les deux cluster, les var des deux cluster
#provient de la meme population

#Si la p < 0.05 le niveau de significance choisi, nous rejetons H0 et appliquant H1.
#La variable choisi n'affiche pas la meme moyene dans les deux cluster dans ce cas,
#les var des deux cluster ne provient pas de la meme population



#Interprétation
alpha = 0.05
if p > alpha:
    print('On ne rejette donc pas H0, l'égalité des moyennes de la var des deux cluster')
else:
    print('H0 l\'hypothèse d'égalité des moyennes est rejetée au niveau de test 5%')
```

```
Statistics=1.965, p=0.053849107
On ne rejette donc pas H0, l'égalité des moyennes de la var des deux cluster
```

In [16]:
```
#help(ttest_ind)
```

In [ ]: