PROJET 4 DATA ANALYST

Réalisez une étude de santé publique avec R ou Python

OBJECTIF DE CE NOTEBOOK

Bienvenue dans l'outil plébiscité par les analystes de données Jupyter.

Il s'agit d'un outil permettant de mixer et d'alterner codes, textes et graphique.

Cet outil est formidable pour plusieurs raisons:

- il permet de tester des lignes de codes au fur et à mesure de votre rédaction, de constater immédiatement le résultat d'un instruction, de la corriger si nécessaire.
- De rédiger du texte pour expliquer l'approche suivie ou les résultats d'une analyse et de le mettre en forme grâce à du code html ou plus simple avec **Markdown**
- d'agrémenter de graphiques

Pour vous aider dans vos premiers pas à l'usage de Jupyter et de Python, nous avons rédigé ce notebook en vous indiquant les instructions à suivre.

Il vous suffit pour cela de saisir le code Python répondant à l'instruction donnée.

Vous verrez de temps à autre le code Python répondant à une instruction donnée mais cela est fait pour vous aider à comprendre la nature du travail qui vous est demandée.

Et garder à l'esprit, qu'il n'y a pas de solution unique pour résoudre un problème et qu'il y a autant de résolutions de problèmes que de développeurs ;)...

Note jeremy Est ce qu'il faut faire le calcul de la sous nutrition sur les pays qu'on a ? Est ce qu'il faut faire des graphiques ? Rajouter le soja La liste des céréales est difficile a trouver ...

Etape 1 - Importation des librairies et chargement des fichiers

1.1 - Importation des librairies

In [ ]:
```
#Importation de la librairie Pandas
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import warnings
warnings.filterwarnings('ignore')
```

1.2 - Chargement des fichiers Excel

In [ ]:
```
#Importation du fichier population.csv
population = pd.read_csv('population.csv')

#Importation du fichier dispo_alimentaire.csv
dispo_alimentaire = pd.read_csv('dispo_alimentaire.csv')

#Importation du fichier aide_alimentaire.csv
aide_alimentaire = pd.read_csv('aide_alimentaire.csv')

#Importation du fichier sous_nutrition.csv
sous_nutrition = pd.read_csv('sous_nutrition.csv')
```

Etape 2 - Analyse exploratoire des fichiers

2.1 - Analyse exploratoire du fichier population

In [ ]:
```
#Afficher les dimensions du dataset
print("Le tableau comporte {} observation(s) ou article(s)".format(population.shape
print("Le tableau comporte {} colonne(s)".format(population.shape[1]))
```

```
Le tableau comporte 1416 observation(s) ou article(s)
Le tableau comporte 3 colonne(s)
```

In [ ]:
```
#Consulter le nombre de colonnes
#La nature des données dans chacune des colonnes
#Le nombre de valeurs présentes dans chacune des colonnes
```

In [ ]:
```
#Affichage les 5 premières lignes de la table
print(population.head())
```

```
          Zone  Année     Valeur
0  Afghanistan   2013  32269.589
1  Afghanistan   2014  33370.794
2  Afghanistan   2015  34413.603
3  Afghanistan   2016  35383.032
4  Afghanistan   2017  36296.113
```

In [ ]:
```
#Nous allons harmoniser les unités. Pour cela, nous avons décidé de multiplier la p
#Multiplication de la colonne valeur par 1000
population['Valeur'] = population['Valeur'] * 1000
```

In [ ]:
```
#changement du nom de la colonne Valeur par Population
population = population.rename(columns={'Valeur': 'Population'})
```

In [ ]:
```
#Affichage les 5 premières lignes de la table pour voir les modifications
print(population.head())
```

```
          Zone  Année  Population
0  Afghanistan   2013  32269589.0
1  Afghanistan   2014  33370794.0
2  Afghanistan   2015  34413603.0
3  Afghanistan   2016  35383032.0
4  Afghanistan   2017  36296113.0
```

2.2 - Analyse exploratoire du fichier disponibilité alimentaire

In [ ]:
```
#Afficher les dimensions du dataset
dimensions = dispo_alimentaire.shape
print("Dimensions de dispo_alimentaire = {}".format(dimensions))
```

```
Dimensions de dispo_alimentaire = (15605, 18)
```

In [ ]:
```
#Consulter le nombre de colonnes
nb_colonnes = len(dispo_alimentaire.columns)
print("Nombre de colonnes:", nb_colonnes)
```

```
Nombre de colonnes: 18
```

In [ ]:
```
#Affichage les 5 premières lignes de la table
print(dispo_alimentaire.head())
```

```
          Zone                Produit   Origine  Aliments pour animaux  \
0  Afghanistan       Abats Comestible   animale                    NaN
1  Afghanistan        Agrumes, Autres  vegetale                    NaN
2  Afghanistan  Aliments pour enfants  vegetale                    NaN
3  Afghanistan                 Ananas  vegetale                    NaN
4  Afghanistan                Bananes  vegetale                    NaN

   Autres Utilisations  Disponibilité alimentaire (Kcal/personne/jour)  \
0                  NaN                                             5.0
1                  NaN                                             1.0
2                  NaN                                             1.0
3                  NaN                                             0.0
4                  NaN                                             4.0

   Disponibilité alimentaire en quantité (kg/personne/an)  \
0                                               1.72
1                                               1.29
2                                               0.06
3                                               0.00
4                                               2.70

   Disponibilité de matière grasse en quantité (g/personne/jour)  \
0                                               0.20
1                                               0.01
2                                               0.01
3                                                NaN
4                                               0.02

   Disponibilité de protéines en quantité (g/personne/jour)  \
0                                               0.77
1                                               0.02
2                                               0.03
3                                                NaN
4                                               0.05

   Disponibilité intérieure  Exportations - Quantité  Importations - Quantité  \
0                      53.0                      NaN                      NaN
1                      41.0                      2.0                     40.0
2                       2.0                      NaN                      2.0
3                       0.0                      NaN                      0.0
4                      82.0                      NaN                     82.0

   Nourriture  Pertes  Production  Semences  Traitement  Variation de stock
0        53.0     NaN        53.0       NaN         NaN                 NaN
1        39.0     2.0         3.0       NaN         NaN                 NaN
2         2.0     NaN         NaN       NaN         NaN                 NaN
3         0.0     NaN         NaN       NaN         NaN                 NaN
4        82.0     NaN         NaN       NaN         NaN                 NaN
```

In [ ]:
```
#remplacement des NaN dans le dataset par des 0
dispo_alimentaire = dispo_alimentaire.fillna(0)
```

In [ ]:
```
#multiplication de toutes les lignes contenant des milliers de tonnes en Kg
col_a_multiplier = ['Aliments pour animaux', 'Autres Utilisations',
                        'Exportations - Quantité','Importations - Quantité',
                        'Nourriture', 'Pertes',
```

```
                        'Production', 'Semences',
                        'Traitement', 'Variation de stock', 'Disponibilité intérie

dispo_alimentaire[col_a_multiplier] *= 1000
```

In [ ]:
```
#Affichage les 5 premières lignes de la table
print(dispo_alimentaire.head())
```

```
          Zone                Produit   Origine  Aliments pour animaux  \
0  Afghanistan       Abats Comestible   animale                    0.0
1  Afghanistan        Agrumes, Autres  vegetale                    0.0
2  Afghanistan  Aliments pour enfants  vegetale                    0.0
3  Afghanistan                Ananas  vegetale                    0.0
4  Afghanistan               Bananes  vegetale                    0.0

   Autres Utilisations  Disponibilité alimentaire (Kcal/personne/jour)  \
0                  0.0                                             5.0
1                  0.0                                             1.0
2                  0.0                                             1.0
3                  0.0                                             0.0
4                  0.0                                             4.0

   Disponibilité alimentaire en quantité (kg/personne/an)  \
0                                               1.72
1                                               1.29
2                                               0.06
3                                               0.00
4                                               2.70

   Disponibilité de matière grasse en quantité (g/personne/jour)  \
0                                               0.20
1                                               0.01
2                                               0.01
3                                               0.00
4                                               0.02

   Disponibilité de protéines en quantité (g/personne/jour)  \
0                                               0.77
1                                               0.02
2                                               0.03
3                                               0.00
4                                               0.05

   Disponibilité intérieure  Exportations - Quantité  Importations - Quantité  \
0                   53000.0                     0.0                     0.0
1                   41000.0                  2000.0                  40000.0
2                    2000.0                     0.0                   2000.0
3                       0.0                     0.0                      0.0
4                   82000.0                     0.0                  82000.0

   Nourriture  Pertes  Production  Semences  Traitement  Variation de stock
0     53000.0     0.0     53000.0       0.0         0.0                 0.0
1     39000.0  2000.0      3000.0       0.0         0.0                 0.0
2      2000.0     0.0         0.0       0.0         0.0                 0.0
3         0.0     0.0         0.0       0.0         0.0                 0.0
4     82000.0     0.0         0.0       0.0         0.0                 0.0
```

2.3 - Analyse exploratoire du fichier aide alimentaire

In [ ]:
```
#Afficher les dimensions du dataset
dimensions = aide_alimentaire.shape
print("Dimensions de aide_alimentaire = {}".format(dimensions))
```

```
Dimensions de aide_alimentaire = (1475, 4)
```

In [ ]:
```
#Consulter le nombre de colonnes
nb_colonnes = len(aide_alimentaire.columns)
print("Nombre de colonnes:", nb_colonnes)
```

```
Nombre de colonnes: 4
```

In [ ]:
```
#Affichage les 5 premières lignes de la table
print(aide_alimentaire.head())
```

```
  Pays bénéficiaire  Année              Produit  Valeur
0       Afghanistan   2013  Autres non-céréales     682
1       Afghanistan   2014  Autres non-céréales     335
2       Afghanistan   2013         Blé et Farin   39224
3       Afghanistan   2014         Blé et Farin   15160
4       Afghanistan   2013             Céréales   40504
```

In [ ]:
```
#changement du nom de la colonne Pays bénéficiaire par Zone
aide_alimentaire = aide_alimentaire.rename(columns={'Pays bénéficiaire': 'Zone'})
```

In [ ]:
```
#Multiplication de la colonne Aide_alimentaire qui contient des tonnes par 1000 pou
aide_alimentaire['Valeur'] = aide_alimentaire['Valeur'] * 1000
```

In [ ]:
```
#Affichage les 5 premières lignes de la table
print(aide_alimentaire.head())
```

```
          Zone  Année              Produit    Valeur
0  Afghanistan   2013  Autres non-céréales    682000
1  Afghanistan   2014  Autres non-céréales    335000
2  Afghanistan   2013         Blé et Farin  39224000
3  Afghanistan   2014         Blé et Farin  15160000
4  Afghanistan   2013             Céréales  40504000
```

2.3 - Analyse exploratoire du fichier sous nutrition

In [ ]:
```
#Afficher les dimensions du dataset
dimensions_sn = sous_nutrition.shape
print("Dimensions du fichier de sous nutrition: = {}".format(dimensions_sn))
```

```
Dimensions du fichier de sous nutrition: = (1218, 3)
```

In [ ]:
```
#Consulter le nombre de colonnes
nb_colonnes_sn = len(aide_alimentaire.columns)
print("Nombre de colonnes du fichier sous nutritin:", nb_colonnes_sn)
```

```
Nombre de colonnes du fichier sous nutritin: 4
```

In [ ]:
```
#Afficher les 5 premières lignes de la table
print(sous_nutrition.head())
```

```
          Zone      Année Valeur
0  Afghanistan  2012-2014    8.6
1  Afghanistan  2013-2015    8.8
2  Afghanistan  2014-2016    8.9
3  Afghanistan  2015-2017    9.7
4  Afghanistan  2016-2018   10.5
```

In [ ]:
```
#Conversion de la colonne (avec l'argument errors=coerce qui permet de convertir au
#Puis remplacement des NaN en 0
sous_nutrition['Valeur'] = pd.to_numeric(sous_nutrition['Valeur'], errors='coerce')
sous_nutrition['Valeur'] = sous_nutrition['Valeur'].fillna(0)
```

In [ ]:
```
#changement du nom de la colonne Valeur par sous_nutrition
sous_nutrition = sous_nutrition.rename(columns={'Valeur': 'sous_nutrition'})
```

In [ ]:
```
#Multiplication de la colonne sous_nutrition par 1000000
sous_nutrition['sous_nutrition'] *= 1000000
```

In [ ]:
```
#Afficher les 5 premières lignes de la table
print(sous_nutrition.head())
```

```
          Zone      Année  sous_nutrition
0  Afghanistan  2012-2014       8600000.0
1  Afghanistan  2013-2015       8800000.0
2  Afghanistan  2014-2016       8900000.0
3  Afghanistan  2015-2017       9700000.0
4  Afghanistan  2016-2018      10500000.0
```

3.1 - Proportion de personnes en sous nutrition

In [ ]:
```
# Il faut tout d'abord faire une jointure entre la table population et la table sou
def calculer_moyenne(plage_annees):
    debut, fin = plage_annees.split('-')
    moyenne = (int(debut) + int(fin)) //2
    return moyenne
sous_nutrition['Année'] = sous_nutrition['Année'].apply(calculer_moyenne)

population_sous_nutrition = pd.merge(population, sous_nutrition, on=['Zone', 'Année
```

In [ ]:
```
#Affichage du dataset
population_sous_nutrition_2017 = population_sous_nutrition[population_sous_nutritio
print(population_sous_nutrition_2017)
```

```
                                        Zone  Année  Population  \
4                                  Afghanistan   2017  36296113.0
10                               Afrique du Sud   2017  57009756.0
16                                     Albanie   2017   2884169.0
22                                     Algérie   2017  41389189.0
28                                    Allemagne   2017  82658409.0
...                                         ...    ...         ...
1192  Venezuela (République bolivarienne du)   2017  29402484.0
1198                                  Viet Nam   2017  94600648.0
1204                                     Yémen   2017  27834819.0
1210                                    Zambie   2017  16853599.0
1216                                  Zimbabwe   2017  14236595.0

      sous_nutrition
4         10500000.0
10         3100000.0
16          100000.0
22         1300000.0
28               0.0
...              ...
1192       8000000.0
1198       6500000.0
1204             0.0
1210             0.0
1216             0.0

[203 rows x 4 columns]
```

In [ ]:
```
#Calcul et affichage du nombre de personnes en état de sous nutrition
total_sous_nutrition_2017 = population_sous_nutrition_2017[population_sous_nutritio
Total_pop = int(population_sous_nutrition_2017['Population'].sum())
print(Total_pop)
Total_2017 = int(total_sous_nutrition_2017['sous_nutrition'].sum())
print('Nombre de personnes en sous nutrition en 2017 :', Total_2017)
```

```
7543798779
Nombre de personnes en sous nutrition en 2017 : 535700000
```

3.2 - Nombre théorique de personne qui pourrait être nourries

In [ ]:
```
#Combien mange en moyenne un être humain ? Source => 2850 kcal / jour source FAO
kcal_minimum = 2850
```

In [ ]:
```
#On commence par faire une jointure entre le data frame population et Dispo_aliment
population_2017 = population[population['Année'] == 2017]
disponibilite_population =  pd.merge(population_2017, dispo_alimentaire, on='Zone')
```

In [ ]:
```
#Affichage du nouveau dataframe
print(disponibilite_population.head())
```

	Zone	Année	Population	Produit	Origine	Aliments pour animaux	Autres Utilisations	Disponibilité alimentaire (Kcal/personne/jour)	Disponibilité alimentaire en quantité (kg/personne/an)	Disponibilité de matière grasse en quantité (g/personne/jour)	Disponibilité de protéines en quantité (g/personne/jour)	Disponibilité intérieure	Exportations - Quantité	Importations - Quantité	Nourriture	Pertes	Production	Semences	Traitement	Variation de stock
0	Afghanistan	2017	36296113.0	Abats Comestible	animale	0.0	0.0	5.0	1.72	0.20	0.77	53000.0	0.0	0.0	53000.0	0.0	53000.0	0.0	0.0	0.0
1	Afghanistan	2017	36296113.0	Agrumes, Autres	vegetale	0.0	0.0	1.0	1.29	0.01	0.02	41000.0	2000.0	40000.0	39000.0	2000.0	3000.0	0.0	0.0	0.0
2	Afghanistan	2017	36296113.0	Aliments pour enfants	vegetale	0.0	0.0	1.0	0.06	0.01	0.03	2000.0	0.0	2000.0	2000.0	0.0	0.0	0.0	0.0	0.0
3	Afghanistan	2017	36296113.0	Ananas	vegetale	0.0	0.0	0.0	0.00	0.00	0.00	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
4	Afghanistan	2017	36296113.0	Bananes	vegetale	0.0	0.0	4.0	2.70	0.02	0.05	82000.0	0.0	82000.0	82000.0	0.0	0.0	0.0	0.0	0.0

In [ ]:
```
#Création de la colonne dispo_kcal avec calcul des kcal disponibles mondialement
disponibilite_population['dispo_kcal'] = disponibilite_population['Disponibilité al
```

In [ ]:
```
#Calcul du nombre d'humains pouvant être nourris
total_dispo_kcal = disponibilite_population['dispo_kcal'].sum()
humains_nourris = int(total_dispo_kcal / kcal_minimum)
print('Le nombre de personnes pouvant être nourries est de', humains_nourris,'.')
```

```
Le nombre de personnes pouvant être nourries est de 7339994606 .
```

3.3 - Nombre théorique de personne qui pourrait être nourrie avec les produits végétaux

In [ ]:
```
#Transfert des données avec les végétaux dans un nouveau dataframe
dispo_vegetaux = disponibilite_population[disponibilite_population['Origine'] == 'v
```

In [ ]:
```
#Calcul du nombre de kcal disponible pour les végétaux
Total_dispo_vegetaux_kcal = int(dispo_vegetaux['dispo_kcal'].sum())
print('Pour les végétaux, le nombre de kcal disponibles est de ',Total_dispo_vegeta
```

```
Pour les végétaux, le nombre de kcal disponibles est de  17260764211501
```

In [ ]:
```
#Calcul du nombre d'humains pouvant être nourris avec les végétaux
Total_humains_vegetaux = int(Total_dispo_vegetaux_kcal / 2850)
print('Grâce aux végétaux il est possible de nourrir', Total_humains_vegetaux,'pers
```

```
Grâce aux végétaux il est possible de nourrir 6056408495 personnes.
```

3.4 - Utilisation de la disponibilité intérieure

In [ ]:
```
#Calcul de la disponibilité totale
dispo_alimentaire['Dispo_Totale'] = dispo_alimentaire['Production'] + dispo_aliment
dispo_totale = dispo_alimentaire['Disponibilité intérieure'].sum()
print(f"La disponibilité totale est de",dispo_totale,"kg.")
```

```
La disponibilité totale est de 9848994000.0 kg.
```

In [ ]:
```
#création d'une boucle for pour afficher les différentes valeurs en fonction des co
utilisation = ['Aliments pour animaux', 'Semences', 'Traitement', 'Autres utilisati

for col in ['Aliments pour animaux', 'Semences', 'Traitement',  'Nourriture', 'Pert
    print("Le pourcentage de", col, ":", "{:.2f}".format(dispo_alimentaire[col].sum
```

```
Le pourcentage de Aliments pour animaux : 13.24 %
Le pourcentage de Semences : 1.57 %
Le pourcentage de Traitement : 22.38 %
Le pourcentage de Nourriture : 49.51 %
Le pourcentage de Pertes : 4.61 %
Le pourcentage de Autres Utilisations : 8.78 %
```

3.5 - Utilisation des céréales

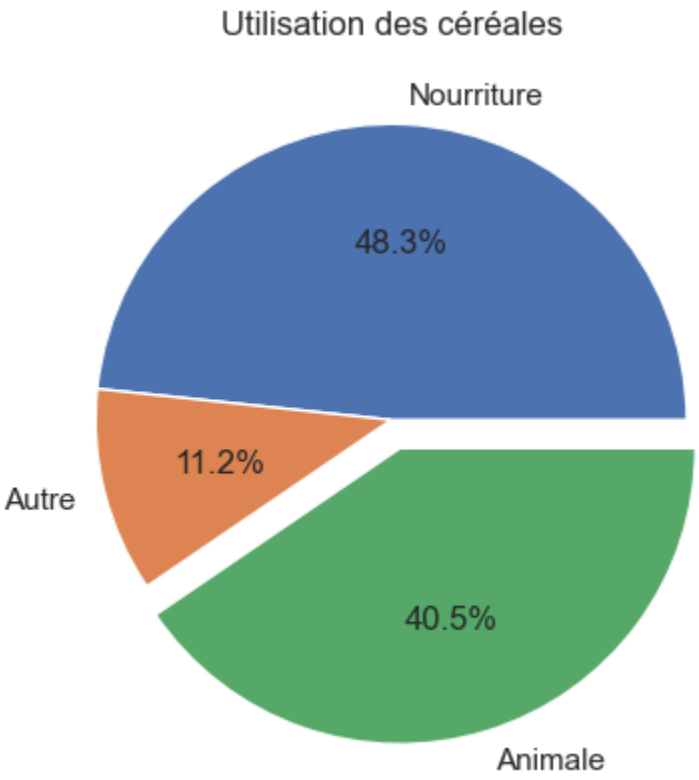
In [ ]:
```
#Création d'une liste avec toutes les variables
liste_céréales = {'Blé', 'Maïs', 'Orge', 'Avoine', 'Seigle', 'Millet', 'Riz (Eq Bla
```

In [ ]:
```
#Création d'un dataframe avec les informations uniquement pour ces céréales
disponibilite_cereales = disponibilite_population[disponibilite_population['Produit
print(disponibilite_cereales.head())
```

```
           Zone  Année  Population           Produit   Origine  \
7   Afghanistan   2017  36296113.0              Blé  vegetale
12  Afghanistan   2017  36296113.0  Céréales, Autres  vegetale
32  Afghanistan   2017  36296113.0              Maïs  vegetale
34  Afghanistan   2017  36296113.0            Millet  vegetale
40  Afghanistan   2017  36296113.0              Orge  vegetale

    Aliments pour animaux  Autres Utilisations  \
7                     0.0                  0.0
12                    0.0                  0.0
32               200000.0                  0.0
34                    0.0                  0.0
40               360000.0                  0.0

    Disponibilité alimentaire (Kcal/personne/jour)  \
7                                           1369.0
12                                             0.0
32                                            21.0
34                                             3.0
40                                            26.0

    Disponibilité alimentaire en quantité (kg/personne/an)  \
7                                              160.23
12                                               0.00
32                                               2.50
34                                               0.40
40                                               2.92

    Disponibilité de matière grasse en quantité (g/personne/jour)  ...  \
7                                                4.69              ...
12                                               0.00              ...
32                                               0.30              ...
34                                               0.02              ...
40                                               0.24              ...

    Disponibilité intérieure  Exportations - Quantité  \
7                  5992000.0                      0.0
12                       0.0                      0.0
32                  313000.0                      0.0
34                   13000.0                      0.0
40                  524000.0                      0.0

    Importations - Quantité  Nourriture    Pertes  Production  Semences  \
7                 1173000.0   4895000.0  775000.0   5169000.0  322000.0
12                      0.0         0.0       0.0         0.0       0.0
32                   1000.0     76000.0   31000.0    312000.0    5000.0
34                      0.0     12000.0    1000.0     13000.0       0.0
40                  10000.0     89000.0   52000.0    514000.0   22000.0

    Traitement  Variation de stock    dispo_kcal
7          0.0           -350000.0  4.968938e+10
12         0.0                 0.0  0.000000e+00
32         0.0                 0.0  7.622184e+08
34         0.0                 0.0  1.088883e+08
40         0.0                 0.0  9.436989e+08
```

```
[5 rows x 21 columns]
```

In [ ]:
```
#Affichage de la proportion d'alimentation animale
total_cereales_nourriture = disponibilite_cereales['Nourriture'].sum()
total_cereales_animaux = disponibilite_cereales['Aliments pour animaux'].sum()
total_cereales_autre = disponibilite_cereales['Autres Utilisations'].sum()


volume = ['Nourriture', 'Autre', 'Animale']
utilisation = [total_cereales_nourriture, total_cereales_autre, total_cereales_anim
explode = [0, 0, 0.1]
plt.pie(utilisation, labels = volume, explode = explode, autopct='%1.1f%%')
plt.title('Utilisation des céréales')
plt.show()
```

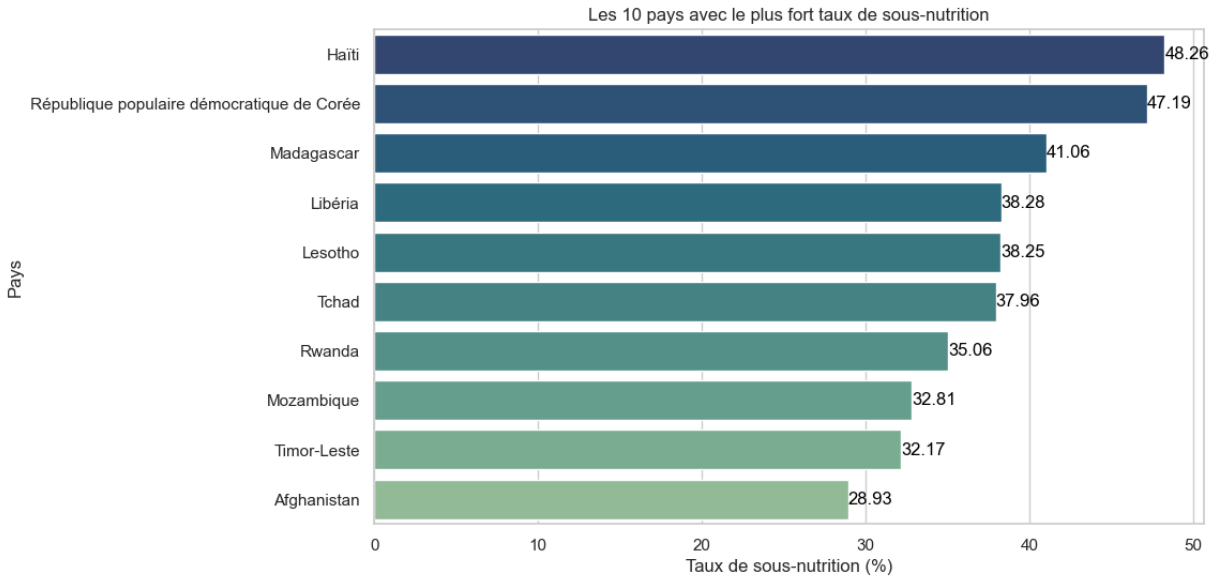


3.6 - Pays avec la proportion de personnes sous-alimentée la plus forte en 2017

In [ ]:
```
#Création de la colonne proportion par pays
population_sous_nutrition_2017['Proportion'] = (population_sous_nutrition_2017['sou
population_sous_nutrition_2017['Proportion'] = population_sous_nutrition_2017['Prop
```

In [ ]:
```
#affichage après trie des 10 pires pays
population_sous_nutrition_2017 = population_sous_nutrition_2017.sort_values('Propor
top_10 = population_sous_nutrition_2017.head(10)
```

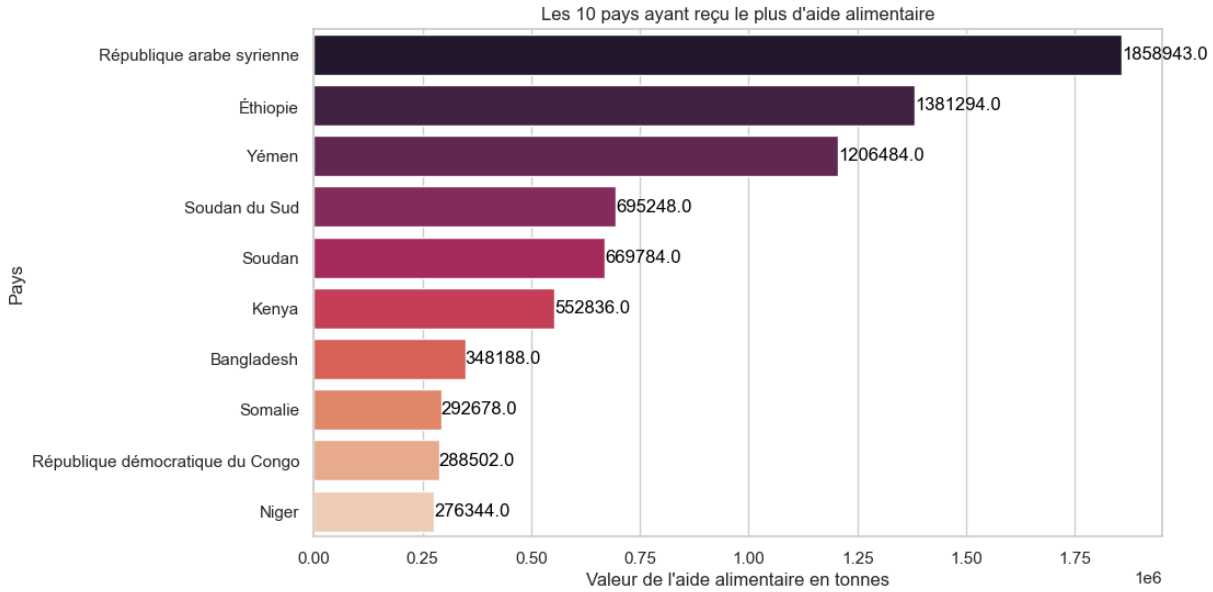
```
sns.set(style="whitegrid")
plt.figure(figsize=(10,6))
sns.barplot(x = 'Proportion', y = 'Zone', data = top_10, palette='crest_r')

for index, value in enumerate(top_10['Proportion']):
    plt.text(value, index, str(value), ha='left', va='center', color='black')
plt.title('Les 10 pays avec le plus fort taux de sous-nutrition')
plt.xlabel('Taux de sous-nutrition (%)')
plt.ylabel('Pays')
plt.show()
```



3.7 - Pays qui ont le plus bénéficié d'aide alimentaire depuis 2013

In [ ]:
```
#calcul du total de l'aide alimentaire par pays
Total_aide_alimentaire = aide_alimentaire.groupby('Zone')['Valeur'].sum().reset_ind
```

In [ ]:
```
#affichage après trie des 10 pays qui ont bénéficié le plus de l'aide alimentaire
Total_aide_alimentaire = Total_aide_alimentaire.sort_values('Valeur', ascending = F
Top_10_aide = Total_aide_alimentaire.head(10)
Top_10_aide['Valeur'] = Top_10_aide['Valeur'] / 1000


sns.set(style="whitegrid")
plt.figure(figsize=(10,6))
sns.barplot(x = 'Valeur', y = 'Zone', data = Top_10_aide, palette='rocket')

for index, value in enumerate(Top_10_aide['Valeur']):
    plt.text(value, index, str(value), ha='left', va='center', color='black')
plt.title("Les 10 pays ayant reçu le plus d'aide alimentaire")
plt.xlabel("Valeur de l'aide alimentaire en tonnes")
plt.ylabel('Pays')
plt.show()
```



3.8 - Evolution des 5 pays qui ont le plus bénéficiés de l'aide alimentaire entre 2013 et 2016

In [ ]:
```
#Création d'un dataframe avec la zone, l'année et l'aide alimentaire puis groupby s
df = aide_alimentaire.drop('Produit', axis=1)


df = df.groupby(['Zone', 'Année']).sum().reset_index()
```

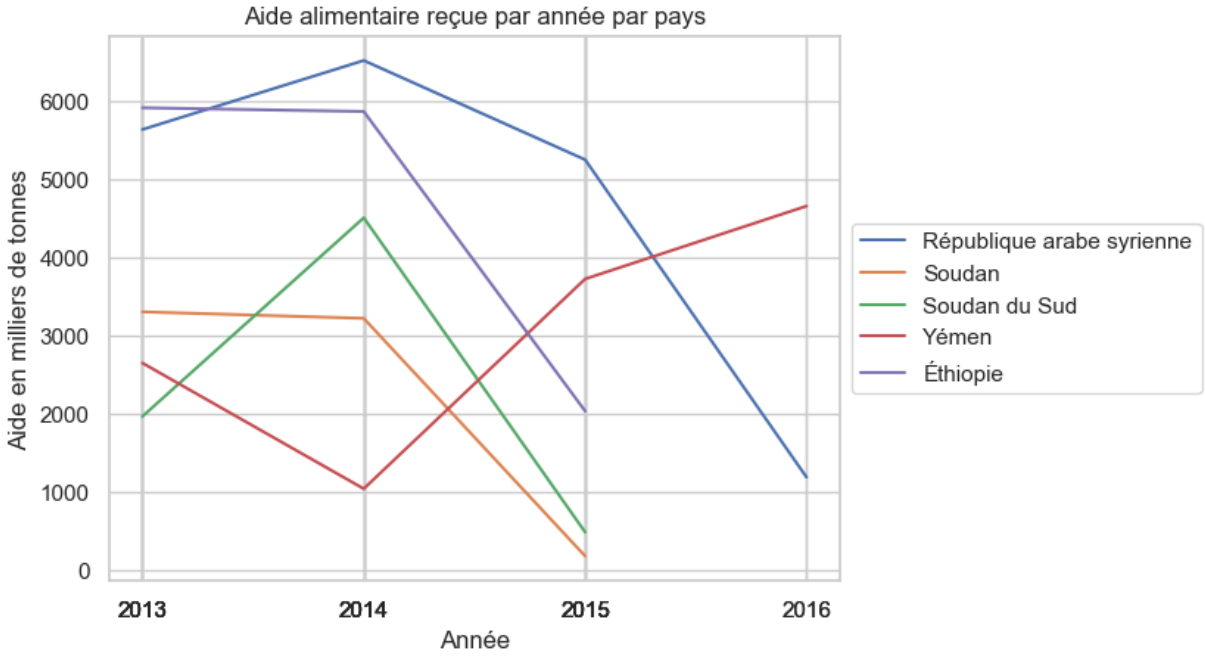
In [ ]:
```
#Création d'une liste contenant les 5 pays qui ont le plus bénéficiées de l'aide al
tri_pays = df.groupby('Zone')['Valeur'].sum().sort_values(ascending = False).reset_
top_pays = tri_pays[:5]

print(top_pays)
```

```
                        Zone      Valeur
0  République arabe syrienne  1858943000
1                   Éthiopie  1381294000
2                      Yémen  1206484000
3               Soudan du Sud   695248000
4                     Soudan   669784000
```

In [ ]:
```
#On filtre sur le dataframe avec notre liste
liste_pays = {'République arabe syrienne', 'Éthiopie', 'Yémen', 'Soudan du Sud', 'S

aide_trie = df[df['Zone'].isin(liste_pays)]
#On met ici la valeur en milliers de tonnes pour un meilleure lisibilité
aide_trie['Valeur'] = aide_trie['Valeur']/100000
print(aide_trie)
```

	Zone	Année	Valeur
157	République arabe syrienne	2013	5635.66
158	République arabe syrienne	2014	6518.70
159	République arabe syrienne	2015	5249.49
160	République arabe syrienne	2016	1185.58
189	Soudan	2013	3302.30
190	Soudan	2014	3219.04
191	Soudan	2015	176.50
192	Soudan du Sud	2013	1963.30
193	Soudan du Sud	2014	4506.10
194	Soudan du Sud	2015	483.08
214	Yémen	2013	2647.64
215	Yémen	2014	1038.40
216	Yémen	2015	3723.06
217	Yémen	2016	4655.74
225	Éthiopie	2013	5914.04
226	Éthiopie	2014	5866.24
227	Éthiopie	2015	2032.66

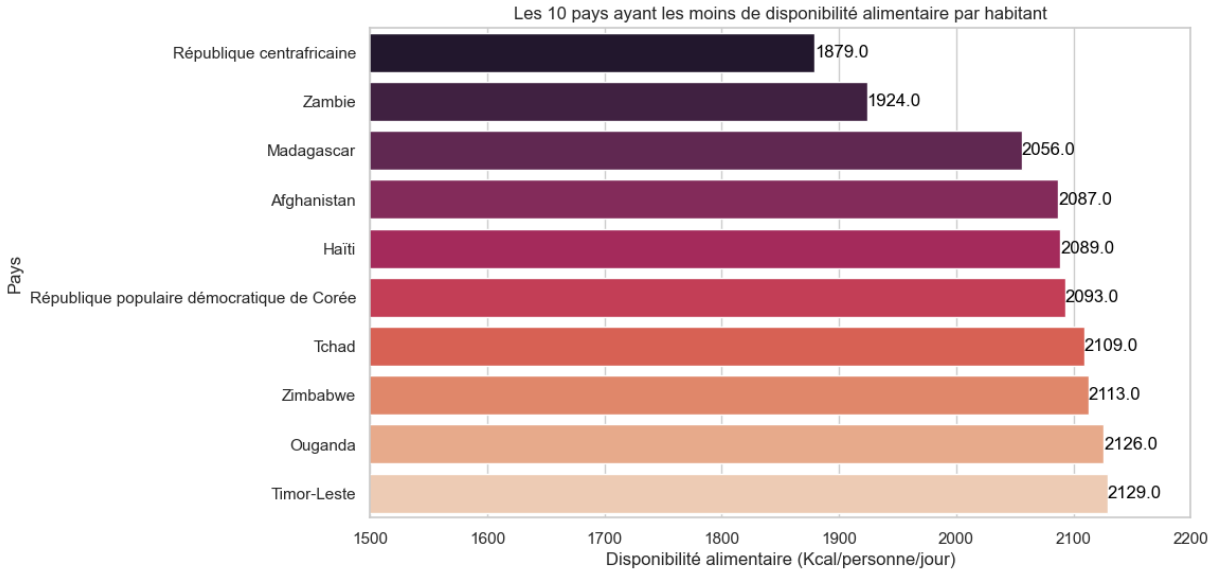
In [ ]:
```
# Affichage des pays avec l'aide alimentaire par année

fig, ax = plt.subplots()


for country, group in aide_trie.groupby('Zone'):
    ax.plot(group['Année'], group['Valeur'], label=country)


ax.legend()
ax.set_xlabel('Année')
ax.set_ylabel('Aide en milliers de tonnes')
ax.set_title('Aide alimentaire reçue par année par pays')
plt.xticks(aide_trie['Année'])
plt.legend(loc='center left', bbox_to_anchor=(1, 0.5))

plt.show()
```



3.9 - Pays avec le moins de disponibilité par habitant

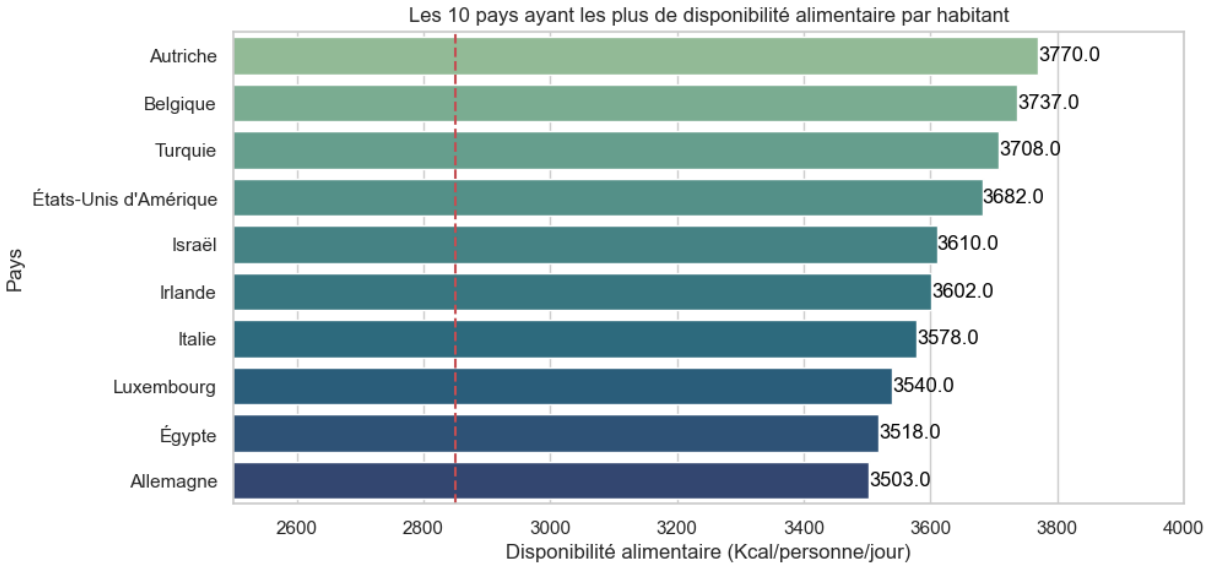
In [ ]:
```
#Calcul de la disponibilité en kcal par personne par jour par pays
Total_dispo_alimentaire = dispo_alimentaire.groupby('Zone')['Disponibilité alimenta
```

In [ ]:
```
#Affichage des 10 pays qui ont le moins de dispo alimentaire par personne
Total_dispo_alimentaire = Total_dispo_alimentaire.sort_values('Disponibilité alimen

Down_10_dispo = Total_dispo_alimentaire.head(10)

sns.set(style="whitegrid")
plt.figure(figsize=(10,6))
sns.barplot(x = 'Disponibilité alimentaire (Kcal/personne/jour)', y = 'Zone', data
for index, value in enumerate(Down_10_dispo['Disponibilité alimentaire (Kcal/person
    plt.text(value, index, str(value), ha='left', va='center', color='black')
plt.xlim(1500, 2200)

plt.title("Les 10 pays ayant les moins de disponibilité alimentaire par habitant")
plt.xlabel("Disponibilité alimentaire (Kcal/personne/jour)")
plt.ylabel('Pays')
plt.show()
```



3.10 - Pays avec le plus de disponibilité par habitant

In [ ]:
```
#Affichage des 10 pays qui ont le plus de dispo alimentaire par personne
Total_dispo_alimentaire = Total_dispo_alimentaire.sort_values('Disponibilité alimen

Top_10_dispo = Total_dispo_alimentaire.head(10)

sns.set(style="whitegrid")
plt.figure(figsize=(10,5))
sns.barplot(x = 'Disponibilité alimentaire (Kcal/personne/jour)', y = 'Zone', data
for index, value in enumerate(Top_10_dispo['Disponibilité alimentaire (Kcal/personn
    plt.text(value, index, str(value), ha='left', va='center', color='black')
plt.xlim(2500, 4000)
plt.axvline(2850, color="r", linestyle="--", label="2850 Calories")
plt.title("Les 10 pays ayant les plus de disponibilité alimentaire par habitant")
plt.xlabel("Disponibilité alimentaire (Kcal/personne/jour)")
plt.ylabel('Pays')
plt.show()
```



3.11 - Exemple de la Thaïlande pour le Manioc

In [ ]:
```
#création d'un dataframe avec uniquement la Thaïlande
sous_nutrition_thailande = population_sous_nutrition_2017[population_sous_nutrition
```

In [ ]:
```
#Calcul de la sous nutrition en Thaïlande
ss_nt_thai = sous_nutrition_thailande['Proportion'].sum()
print("La proportion de pesronnes en sous nutrition en Thaïlande est de ",ss_nt_tha
```

```
La proportion de pesronnes en sous nutrition en Thaïlande est de  8.96 %
```

In [ ]:
```
# On calcule la proportion exportée en fonction de la proportion

manioc_thailande =dispo_alimentaire.loc[(dispo_alimentaire["Produit"] == "Manioc")


manioc_thailande = manioc_thailande.reset_index()


proportion_manioc_thailande = round((manioc_thailande["Exportations - Quantité"][0]

print("La Thaïlande exporte",proportion_manioc_thailande,"% de sa production de Man
```

```
La Thaïlande exporte 83.41 % de sa production de Manioc.
```

Etape 6 - Analyse complémentaires

In [ ]:
```
#Rajouter en dessous toutes les analyses complémtaires suite à la demande de mélani
#"et toutes les infos que tu trouverais utiles pour mettre en relief les pays qui s
#le plus en difficulté au niveau alimentaire"


#On cherche à determiner l'evolution de l'aide alimentaire de 2013 à 2016
```

```
aide_alimentaire_globale = aide_alimentaire.groupby('Année')['Valeur'].sum().reset_
#On met ici la valeur en milliers de tonnes pour un meilleure lisibilité
aide_alimentaire_globale['Valeur'] = aide_alimentaire_globale['Valeur']/100000
print(aide_alimentaire_globale)

plt.figure(figsize=(10, 6))
plt.plot(aide_alimentaire_globale['Année'], aide_alimentaire_globale['Valeur'], mar


plt.title('Aide alimentaire distribuée par année')
plt.xlabel('Année')
plt.ylabel('Aide alimentaire en milliers de tonnes')
plt.xticks(aide_alimentaire_globale['Année'])

plt.grid(True)
plt.show()
```

```
   Année    Valeur
0   2013  41656.74
1   2014  39391.52
2   2015  21875.07
3   2016   7435.68
```