Projet4 Étude sur l'alimentation dans le monde

October 30, 2023

```
<h1>Étude sur l'alimentation dans le monde</h1>
```

```
<h2 style="font-weight: bold; color: #001F3F;">Etape 1 - Importation des librairies et chargeme
```

1.1 - Importation des librairies

[632]:
```
#Importation de la librairie Pandas
import pandas as pd
```

1.2 - Chargement des fichiers Excel

[633]:
```
#Importation du fichier population.csv
population = pd.read_csv('/Users/Bouboule/Documents/population.csv')

#Importation du fichier dispo_alimentaire.csv
dispoAlimentaire = pd.read_csv('/Users/Bouboule/Documents/dispo_alimentaire.
 ↪csv')

#Importation du fichier aide_alimentaire.csv
aideAlimentaire = pd.read_csv('/Users/Bouboule/Documents/aide_alimentaire.csv')

#Importation du fichier sous_nutrition.csv
sousNutrition = pd.read_csv('/Users/Bouboule/Documents/sous_nutrition.csv')
```

```
<h2 style="font-weight: bold; color: #001F3F;">Etape 2 - Analyse exploratoire des fichiers</h2
```

2.1 - Analyse exploratoire du fichier population

[634]:
```
#Afficher les dimensions du dataset
print("Le tableau comporte {} observation(s) ou article(s)".format(population.
 ↪shape[0]))
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)
```

[635]:
```
#Consulter le nombre de colonnes
population.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1416 entries, 0 to 1415
Data columns (total 3 columns):
```

```
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   Zone    1416 non-null   object
 1   Année   1416 non-null   int64
 2   Valeur  1416 non-null   float64
dtypes: float64(1), int64(1), object(1)
memory usage: 33.3+ KB
```

[636]:
```
#Affichage des 5 premières lignes de la table
population.head()
```

[636]:

	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

[637]:
```
#Multiplication de la colonne valeur par 1000
population['Valeur']*=1000
```

[638]:
```
#changement du nom de la colonne Valeur par Population
population.rename(columns={'Valeur': 'Population'}, inplace=True)
```

[639]:
```
#Affichage les 5 premières lignes de la table pour voir les modifications
population.head()
```

[639]:

	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

[640]:
```
#Afficher les dimensions du dataset
print("Le tableau comporte {} observation(s) ou article(s)".
  ↪format(dispoAlimentaire.shape[0]))
print("Le tableau comporte {} colonne(s)".format(dispoAlimentaire.shape[1]))
```

```
Le tableau comporte 15605 observation(s) ou article(s)
Le tableau comporte 18 colonne(s)
```

[641]:
```
#Consulter le nombre de colonnes
dispoAlimentaire.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 15605 entries, 0 to 15604
Data columns (total 18 columns):
```

```
 #   Column                                                Non-Null
Count  Dtype
---  ------
--------------  -----
 0   Zone                                                  15605 non-
null  object
 1   Produit                                               15605 non-
null  object
 2   Origine                                               15605 non-
null  object
 3   Aliments pour animaux                                 2720 non-
null   float64
 4   Autres Utilisations                                   5496 non-
null   float64
 5   Disponibilité alimentaire (Kcal/personne/jour)        14241 non-
null  float64
 6   Disponibilité alimentaire en quantité (kg/personne/an)  14015 non-
null  float64
 7   Disponibilité de matière grasse en quantité (g/personne/jour)  11794 non-
null  float64
 8   Disponibilité de protéines en quantité (g/personne/jour)  11561 non-
null  float64
 9   Disponibilité intérieure                              15382 non-
null  float64
 10  Exportations - Quantité                               12226 non-
null  float64
 11  Importations - Quantité                               14852 non-
null  float64
 12  Nourriture                                            14015 non-
null  float64
 13  Pertes                                                4278 non-
null   float64
 14  Production                                            9180 non-
null   float64
 15  Semences                                              2091 non-
null   float64
 16  Traitement                                            2292 non-
null   float64
 17  Variation de stock                                    6776 non-
null   float64
dtypes: float64(15), object(3)
memory usage: 2.1+ MB
```

[642]:
```
# Remplacement des NaN par 0
dispoAlimentaire.fillna(0, inplace=True)
```

[643]:
```
dispoAlimentaire.info
```

[643]: <bound method DataFrame.info of 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
… … … … …
15600 Îles Salomon Viande de Suides animale 0.0
15601 Îles Salomon Viande de Volailles animale 0.0
15602 Îles Salomon Viande, Autre animale 0.0
15603 Îles Salomon Vin vegetale 0.0
15604 Îles Salomon Épices, Autres 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
… … …
15600 0.0 45.0
15601 0.0 11.0
15602 0.0 0.0
15603 0.0 0.0
15604 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
… …
15600 4.70
15601 3.34
15602 0.06
15603 0.07
15604 0.48

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

```
15600                                                4.28
15601                                                0.69
15602                                                0.00
15603                                                0.00
15604                                                0.21

       Disponibilité de protéines en quantité (g/personne/jour)  \
0                                                    0.77
1                                                    0.02
2                                                    0.03
3                                                    0.00
4                                                    0.05
…                                                     …
15600                                                1.41
15601                                                1.14
15602                                                0.04
15603                                                0.00
15604                                                0.15

       Disponibilité intérieure  Exportations - Quantité  \
0                          53.0                       0.0
1                          41.0                       2.0
2                           2.0                       0.0
3                           0.0                       0.0
4                          82.0                       0.0
…                           …                         …
15600                       3.0                       0.0
15601                       2.0                       0.0
15602                       0.0                       0.0
15603                       0.0                       0.0
15604                       0.0                       0.0

       Importations - Quantité  Nourriture  Pertes  Production  Semences  \
0                          0.0        53.0     0.0        53.0       0.0
1                         40.0        39.0     2.0         3.0       0.0
2                          2.0         2.0     0.0         0.0       0.0
3                          0.0         0.0     0.0         0.0       0.0
4                         82.0        82.0     0.0         0.0       0.0
…                          …          …        …           …         …
15600                      0.0         3.0     0.0         2.0       0.0
15601                      2.0         2.0     0.0         0.0       0.0
15602                      0.0         0.0     0.0         0.0       0.0
15603                      0.0         0.0     0.0         0.0       0.0
15604                      0.0         0.0     0.0         0.0       0.0

       Traitement  Variation de stock
0             0.0                 0.0
```

```
1              0.0                  0.0
2              0.0                  0.0
3              0.0                  0.0
4              0.0                  0.0
…              …                    …
15600          0.0                  0.0
15601          0.0                  0.0
15602          0.0                  0.0
15603          0.0                  0.0
15604          0.0                  0.0

[15605 rows x 18 columns]>
```

[644]:
```
# Multiplication des colonnes par 1000000
colonne_tonne = ['Aliments pour animaux', 'Autres Utilisations', 'Disponibilité␣
 ↪intérieure', 'Exportations - Quantité', 'Importations - Quantité',␣
 ↪'Nourriture', 'Pertes', 'Production', 'Semences', 'Traitement', 'Variation␣
 ↪de stock']
for i in colonne_tonne:
    dispoAlimentaire[i]*= 1000000
```

[645]:
```
#Affichage les 5 premières lignes de la table
dispoAlimentaire.head()
```

[645]:

	Zone	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)
0	Afghanistan	Abats Comestible	animale	0.0	0.0	5.0	1.72	0.20
1	Afghanistan	Agrumes, Autres	vegetale	0.0	0.0	1.0	1.29	
2	Afghanistan	Aliments pour enfants	vegetale	0.0	0.0	1.0	0.06	
3	Afghanistan	Ananas	vegetale	0.0	0.0	0.0	0.00	
4	Afghanistan	Bananes	vegetale	0.0	0.0	4.0	2.70	

```
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                53000000.0                      0.0                      0.0
1                41000000.0                2000000.0                40000000.0
2                 2000000.0                      0.0                 2000000.0
3                       0.0                      0.0                      0.0
4                82000000.0                      0.0                82000000.0

   Nourriture     Pertes  Production  Semences  Traitement  Variation de stock
0  53000000.0        0.0  53000000.0       0.0         0.0                 0.0
1  39000000.0  2000000.0   3000000.0       0.0         0.0                 0.0
2   2000000.0        0.0         0.0       0.0         0.0                 0.0
3         0.0        0.0         0.0       0.0         0.0                 0.0
4  82000000.0        0.0         0.0       0.0         0.0                 0.0
```

2.3 - Analyse exploratoire du fichier aide alimentaire

[646]:
```
#Afficher les dimensions du dataset
print("Le tableau comporte {} observation(s) ou article(s)".
  ↪format(aideAlimentaire.shape[0]))
print("Le tableau comporte {} colonne(s)".format(aideAlimentaire.shape[1]))
```

```
Le tableau comporte 1475 observation(s) ou article(s)
Le tableau comporte 4 colonne(s)
```

[647]:
```
#Consulter le nombre de colonnes
aideAlimentaire.info
```

[647]:
```
<bound method DataFrame.info of      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
…                  …    …                          …      …
1470          Zimbabwe   2015  Mélanges et préparations      96
```

```
1471          Zimbabwe   2013              Non-céréales    5022
1472          Zimbabwe   2014              Non-céréales    2310
1473          Zimbabwe   2015              Non-céréales     306
1474          Zimbabwe   2013                Riz, total      64

[1475 rows x 4 columns]>
```

[648]:
```
#Affichage les 5 premières lignes de la table
aideAlimentaire.head()
```

[648]:

	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

[649]:
```
#changement du nom de la colonne Pays bénéficiaire par Zone
aideAlimentaire.rename(columns={'Pays bénéficiaire':'Zone'}, inplace=True)
```

[650]:
```
#changement du nom de la colonne Valeur bénéficiaire par Aide_alimentaire
aideAlimentaire.rename(columns={'Valeur':'Aide_alimentaire'}, inplace=True)
#Multiplication de la colonne Aide_alimentaire qui contient des tonnes par 1000␣
 ↪pour avoir des kg
aideAlimentaire['Aide_alimentaire'] *=1000
```

[651]:
```
#Affichage les 5 premières lignes de la table
aideAlimentaire.head()
```

[651]:

	Zone	Année	Produit	Aide_alimentaire
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

2.4 - Analyse exploratoire du fichier sous nutrition

[652]:
```
#Afficher les dimensions du dataset
print("Le tableau comporte {} observation(s) ou article(s)".
 ↪format(sousNutrition.shape[0]))
print("Le tableau comporte {} colonne(s)".format(sousNutrition.shape[1]))
```

```
Le tableau comporte 1218 observation(s) ou article(s)
Le tableau comporte 3 colonne(s)
```

[653]:
```
#Consulter le nombre de colonnes
sousNutrition.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1218 entries, 0 to 1217
Data columns (total 3 columns):
 #   Column  Non-Null Count  Dtype 
---  ------  --------------  ----- 
 0   Zone    1218 non-null   object
 1   Année   1218 non-null   object
 2   Valeur  624 non-null    object
dtypes: object(3)
memory usage: 28.7+ KB
```

[654]:
```
#Afficher les 5 premières lignes de la table
sousNutrition.head()
```

[654]:

	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

[655]:
```
#Conversion de la colonne
sousNutrition['Valeur'] = pd.to_numeric(sousNutrition['Valeur'],␣
 ↪errors='coerce')
#Puis remplacement des NaN en 0
sousNutrition = sousNutrition.fillna(0)
```

[656]:
```
#changement du nom de la colonne Valeur par sous_nutrition
sousNutrition.rename(columns={'Valeur':'sous_nutrition'}, inplace=True)
```

[657]:
```
#Multiplication de la colonne sous_nutrition par 1000000
sousNutrition['sous_nutrition'] *=1000000
```

[658]:
```
#Afficher les 5 premières lignes de la table
sousNutrition.head()
```

[658]:

	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

```
<h2 style="font-weight: bold; color: #001F3F;">Etape 2 - Analyse</h2>
```

3.1 - Proportion de personnes en sous nutrition

[659]:
```
#jointure & filtrage
```

```
nombre_sous_nutrition = pd.merge(population.loc[population['Année'] ==␣
↪2017,["Zone", "Population"]],
                                 sousNutrition.loc[sousNutrition['Année'] ==␣
↪'2016-2018',["Zone", "sous_nutrition"]],
                                 on='Zone')
print(nombre_sous_nutrition)
```

```
                                      Zone  Population  sous_nutrition
0                                 Afghanistan  36296113.0      10500000.0
1                              Afrique du Sud  57009756.0       3100000.0
2                                     Albanie   2884169.0        100000.0
3                                     Algérie  41389189.0       1300000.0
4                                   Allemagne  82658409.0             0.0
..                                        …         …              …
198  Venezuela (République bolivarienne du)  29402484.0       8000000.0
199                                Viet Nam  94600648.0       6500000.0
200                                   Yémen  27834819.0             0.0
201                                  Zambie  16853599.0             0.0
202                                Zimbabwe  14236595.0             0.0

[203 rows x 3 columns]
```

[660]:
```
# Calculer la proportion de personnes en sous-nutrition par rapport à la␣
↪population totale
proportion_sous_nutrition = (nombre_sous_nutrition['sous_nutrition'].sum()) /␣
↪(nombre_sous_nutrition['Population'].sum()) * 100
total_sous_nutrition = nombre_sous_nutrition['sous_nutrition'].sum()

print("Nombre total de personnes en sous-nutrition : {:.0f} millions".
↪format(total_sous_nutrition))
print("Proportion de personnes en sous-nutrition par rapport à la population␣
↪totale : {:.2f}%".format(proportion_sous_nutrition))
```

```
Nombre total de personnes en sous-nutrition : 535700000 millions
Proportion de personnes en sous-nutrition par rapport à la population totale :
7.10%
```

3.2 - Nombre théorique de personne qui pourrait être nourries

[661]:
```
#moyenne calorique que doit consommer un être humain par jour
moyenneKcal = 2250
```

[662]:
```
#Jointure de la table population filtré par l'année 2017 et agregation de␣
↪dispoAlimentaire
dispo_alimentaire2017 = pd.merge(
    population.loc[population['Année'] == 2017, ["Zone", "Population"]],
    dispoAlimentaire.groupby('Zone')['Disponibilité alimentaire (Kcal/personne/
↪jour)'].sum().reset_index(),
```

```
    on='Zone'
)
```

[663]:
```
#Création de la colonne dispo_kcal avec calcul des kcal disponibles mondialement
dispo_alimentaire2017['dispo_kcal'] = dispo_alimentaire2017['Disponibilité␣
 ↪alimentaire (Kcal/personne/jour)'] * dispo_alimentaire2017['Population']*365
```

[664]:
```
#Calcul du nombre d'humains pouvant être nourris
total_humain = round(dispo_alimentaire2017['dispo_kcal'].sum() / (365*␣
 ↪moyenneKcal))
# Proportion d'êtres humains pouvant être nourris avec la disponibilité␣
 ↪alimentaire totale
proportion_nourris = total_humain *100 / (population.loc[population['Année'] ==␣
 ↪2017,'Population'].sum())

print("Le total de disponibilité alimentaire disponible au total est de {:.2f}␣
 ↪Kcal.".format(dispo_alimentaire2017['dispo_kcal'].sum()))
print("Cela pourrait nourrir environ", round(total_humain), "êtres humains.")
print("La proportion d'êtres humains pouvant être nourris avec la disponibilité␣
 ↪alimentaire totale est d'environ", round(proportion_nourris, 2), "% de la␣
 ↪population mondiale.")
```

```
Le total de disponibilité alimentaire disponible au total est de
7635429388975815.00 Kcal.
Cela pourrait nourrir environ 9297326501 êtres humains.
La proportion d'êtres humains pouvant être nourris avec la disponibilité
alimentaire totale est d'environ 123.17 % de la population mondiale.
```

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

[665]:
```
#Transfert des données avec les végétaux dans un nouveau dataframe
dispoAlimentaire_vegetaux = pd.merge(
    population.loc[population['Année'] == 2017, ["Zone", "Population"]],
    dispoAlimentaire[dispoAlimentaire['Origine'] == 'vegetale'].
 ↪groupby('Zone')['Disponibilité alimentaire (Kcal/personne/jour)'].sum().
 ↪reset_index(),
    on='Zone'
)
#Ajout de la colonne dispo_kcal
dispoAlimentaire_vegetaux['dispo_kcal'] =␣
 ↪dispoAlimentaire_vegetaux['Disponibilité alimentaire (Kcal/personne/jour)']␣
 ↪* dispoAlimentaire_vegetaux['Population']*365
print(dispoAlimentaire_vegetaux)
```

```
                                  Zone  Population  \
0                           Afghanistan  36296113.0
1                         Afrique du Sud  57009756.0
2                                Albanie   2884169.0
```

```
3                                    Algérie  41389189.0
4                                   Allemagne  82658409.0
..                                        …         …
167  Venezuela (République bolivarienne du)  29402484.0
168                                Viet Nam  94600648.0
169                                   Yémen  27834819.0
170                                  Zambie  16853599.0
171                                Zimbabwe  14236595.0

     Disponibilité alimentaire (Kcal/personne/jour)    dispo_kcal
0                                            1871.0  2.478716e+13
1                                            2533.0  5.270808e+13
2                                            2203.0  2.319146e+12
3                                            2915.0  4.403706e+13
4                                            2461.0  7.424916e+13
..                                              …           …
167                                          2157.0  2.314872e+13
168                                          2169.0  7.489391e+13
169                                          2028.0  2.060389e+13
170                                          1818.0  1.118354e+13
171                                          1935.0  1.005495e+13

[172 rows x 4 columns]
```

[666]:
```
#Calcul du nombre de kcal disponible pour les végétaux
total_kcal = dispoAlimentaire_vegetaux['dispo_kcal'].sum()
print("Le total de disponibilité alimentaire pour les végétaux disponible est␣
 ↪de", total_kcal, "Kcal.")
```

```
Le total de disponibilité alimentaire pour les végétaux disponible est de
6300178937197865.0 Kcal.
```

[667]:
```
#Calcul du nombre d'humains pouvant être nourris
total_humain = round(dispoAlimentaire_vegetaux['dispo_kcal'].sum() /␣
 ↪(365*moyenneKcal))
# Proportion d'êtres humains pouvant être nourris avec la disponibilité␣
 ↪alimentaire totale
proportion_nourris = total_humain *100 / (population.loc[population['Année'] ==␣
 ↪2017,'Population'].sum())

print("Cela pourrait nourrir environ", round(total_humain), "êtres humains.")
print("La proportion d'êtres humains pouvant être nourris avec la disponibilité␣
 ↪alimentaire végétale est d'environ", round(proportion_nourris, 2), "% de la␣
 ↪population mondiale.")
```

```
Cela pourrait nourrir environ 7671450761 êtres humains.
La proportion d'êtres humains pouvant être nourris avec la disponibilité
alimentaire végétale est d'environ 101.63 % de la population mondiale.
```

3.4 - Utilisation de la disponibilité intérieure

[668]:
```
#Calcul de la disponibilité totale
total_dispo_interieure = dispoAlimentaire['Disponibilité intérieure'].sum()
print("Le total de disponibilité alimentaire disponible au total est de",␣
 ↪total_dispo_interieure, "Kg")
```

```
Le total de disponibilité alimentaire disponible au total est de 9848994000000.0
Kg
```

[669]:
```
# Boucle for pour parcourir les colonnes
for colonne in ['Aliments pour animaux', 'Pertes', 'Nourriture', 'Semences',␣
 ↪'Traitement', 'Autres Utilisations']:
    somme_dispo_interieure = round(dispoAlimentaire[colonne].sum()*100 /␣
 ↪total_dispo_interieure,2)
    print(f"Somme totale de la disponibilité intérieure pour ':{colonne}':
 ↪{somme_dispo_interieure} kcal", "%")
```

```
Somme totale de la disponibilité intérieure pour ':Aliments pour animaux':13.24
kcal %
Somme totale de la disponibilité intérieure pour ':Pertes':4.61 kcal %
Somme totale de la disponibilité intérieure pour ':Nourriture':49.51 kcal %
Somme totale de la disponibilité intérieure pour ':Semences':1.57 kcal %
Somme totale de la disponibilité intérieure pour ':Traitement':22.38 kcal %
Somme totale de la disponibilité intérieure pour ':Autres Utilisations':8.78
kcal %
```

3.5 - Utilisation des céréales

[670]:
```
# Création d'une liste avec toutes les variables
liste_cereale = ["Blé", "Riz (Eq Blanchi)", "Orge", "Maïs", "Seigle","Avoine",␣
 ↪"Millet", "Sorgho", "Céréales, Autres"]
# Création d'un DataFrame avec les informations uniquement pour ces céréales
cereale = dispoAlimentaire[dispoAlimentaire['Produit'].isin(liste_cereale)]
print(cereale)
```

```
                Zone           Produit   Origine  Aliments pour animaux  \
7        Afghanistan              Blé  vegetale                    0.0
12       Afghanistan  Céréales, Autres  vegetale                    0.0
32       Afghanistan             Maïs  vegetale            200000000.0
34       Afghanistan            Millet  vegetale                    0.0
40       Afghanistan             Orge  vegetale            360000000.0
…                …                …       …                      …
15545  Îles Salomon  Céréales, Autres  vegetale                    0.0
15568  Îles Salomon             Maïs  vegetale                    0.0
15575  Îles Salomon             Orge  vegetale                    0.0
15591  Îles Salomon  Riz (Eq Blanchi)  vegetale                    0.0
15593  Îles Salomon           Sorgho  vegetale                    0.0
```

```
       Autres Utilisations  Disponibilité alimentaire (Kcal/personne/jour)  \
7                      0.0                                          1369.0
12                     0.0                                             0.0
32                     0.0                                            21.0
34                     0.0                                             3.0
40                     0.0                                            26.0
…                        …                                             …
15545                  0.0                                             0.0
15568                  0.0                                             1.0
15575                  0.0                                             0.0
15591           12000000.0                                           623.0
15593                  0.0                                             0.0

       Disponibilité alimentaire en quantité (kg/personne/an)  \
7                                                     160.23
12                                                      0.00
32                                                      2.50
34                                                      0.40
40                                                      2.92
…                                                          …
15545                                                   0.00
15568                                                   0.15
15575                                                   0.07
15591                                                  63.76
15593                                                   0.00

       Disponibilité de matière grasse en quantité (g/personne/jour)  \
7                                                       4.69
12                                                      0.00
32                                                      0.30
34                                                      0.02
40                                                      0.24
…                                                          …
15545                                                   0.00
15568                                                   0.01
15575                                                   0.00
15591                                                   1.36
15593                                                   0.00

       Disponibilité de protéines en quantité (g/personne/jour)  \
7                                                      36.91
12                                                      0.00
32                                                      0.56
34                                                      0.08
40                                                      0.79
…                                                          …
15545                                                   0.00
15568                                                   0.03
```

```
15575                                                0.01
15591                                               10.90
15593                                                0.00

       Disponibilité intérieure  Exportations - Quantité  \
7                  5.992000e+09                      0.0
12                 0.000000e+00                      0.0
32                 3.130000e+08                      0.0
34                 1.300000e+07                      0.0
40                 5.240000e+08                      0.0
…                           …                        …
15545              0.000000e+00                      0.0
15568              0.000000e+00                      0.0
15575              1.000000e+06                      0.0
15591              4.900000e+07                      0.0
15593              0.000000e+00                      0.0

       Importations - Quantité    Nourriture       Pertes    Production  \
7                  1.173000e+09  4.895000e+09  775000000.0  5.169000e+09
12                 0.000000e+00  0.000000e+00          0.0  0.000000e+00
32                 1.000000e+06  7.600000e+07   31000000.0  3.120000e+08
34                 0.000000e+00  1.200000e+07    1000000.0  1.300000e+07
40                 1.000000e+07  8.900000e+07   52000000.0  5.140000e+08
…                           …           …           …           …
15545              0.000000e+00  0.000000e+00          0.0  0.000000e+00
15568              0.000000e+00  0.000000e+00          0.0  0.000000e+00
15575              1.000000e+06  0.000000e+00          0.0  0.000000e+00
15591              4.700000e+07  3.600000e+07    1000000.0  3.000000e+06
15593              0.000000e+00  0.000000e+00          0.0  0.000000e+00

          Semences  Traitement  Variation de stock
7      322000000.0         0.0       -350000000.0
12             0.0         0.0                0.0
32       5000000.0         0.0                0.0
34             0.0         0.0                0.0
40      22000000.0         0.0                0.0
…              …           …                  …
15545          0.0         0.0                0.0
15568          0.0         0.0                0.0
15575          0.0   1000000.0                0.0
15591          0.0         0.0                0.0
15593          0.0         0.0                0.0

[1497 rows x 18 columns]
```

[671]:
```
#Affichage de la proportion d'alimentation animale
```

```
proportion = cereale["Aliments pour animaux"].sum() / cereale["Disponibilité␣
 ↪intérieure"].sum() * 100
print("Proportion d'alimentation animale : {:.2f}%".format(proportion))
# Calcul de la proportion d'alimentation humaine
proportion = cereale["Nourriture"].sum() * 100 / cereale["Disponibilité␣
 ↪intérieure"].sum()
print("Proportion d'alimentation humaine : {:.2f}%".format(proportion))
```

```
Proportion d'alimentation animale : 36.29%
Proportion d'alimentation humaine : 42.75%
```

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

[672]:
```
proportion_sousAlimente = pd.merge(
    sousNutrition.loc[sousNutrition['Année'] == '2016-2018', ["Zone",␣
 ↪"sous_nutrition"]],
    population.loc[population['Année'] == 2017, ["Zone", "Population"]],
    on='Zone'
)
#Création de la colonne proportion par pays
proportion_sousAlimente['proportion_par_pays'] =␣
 ↪(proportion_sousAlimente['sous_nutrition'] /␣
 ↪proportion_sousAlimente['Population']).round(2)
```

[673]:
```
#affichage après trie des 10 pires pays
proportion_sousAlimente.sort_values('proportion_par_pays', ascending=False).
 ↪head(10)
```

[673]:

	Zone	sous_nutrition	Population	proportion_par_pays
78	Haïti	5300000.0	10982366.0	0.48
157	République populaire démocratique de Corée	12000000.0	25429825.0	0.47
108	Madagascar	10500000.0	25570512.0	0.41
103	Libéria	1800000.0	4702226.0	0.38
183	Tchad	5700000.0	15016753.0	0.38
100	Lesotho	800000.0	2091534.0	0.38
161	Rwanda	4200000.0	11980961.0	0.35
121	Mozambique	9400000.0	28649018.0	
186	Timor-Leste	400000.0	1243258.0	
0	Afghanistan	10500000.0	36296113.0	

```
121                 0.33
186                 0.32
0                   0.29
```

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

[674]:
```
pays_aideAlimentaire = aideAlimentaire[['Zone', 'Aide_alimentaire']].
 ↪groupby('Zone')['Aide_alimentaire'].sum().reset_index()
print(pays_aideAlimentaire)
```

```
           Zone  Aide_alimentaire
0   Afghanistan         185452000
1       Algérie          81114000
2        Angola           5014000
3    Bangladesh         348188000
4       Bhoutan           2666000
..          …                 …
71       Zambie           3026000
72     Zimbabwe          62570000
73       Égypte           1122000
74     Équateur           1362000
75     Éthiopie        1381294000

[76 rows x 2 columns]
```

[675]:
```
#affichage après trie des 10 pays qui ont bénéficié le plus de l'aide␣
 ↪alimentaire
pays_aideAlimentaire.sort_values('Aide_alimentaire', ascending=False).head(10)
```

[675]:
```
                              Zone  Aide_alimentaire
50         République arabe syrienne        1858943000
75                          Éthiopie        1381294000
70                             Yémen        1206484000
61                       Soudan du Sud         695248000
60                             Soudan         669784000
30                              Kenya         552836000
3                         Bangladesh         348188000
59                           Somalie         292678000
53  République démocratique du Congo         288502000
43                              Niger         276344000
```

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

[676]:
```
# Effectuer une opération de groupby sur les colonnes 'Zone' et 'Année' et␣
 ↪calculer la somme de l'aide alimentaire
aideAlimentaire_evolution = aideAlimentaire.groupby(['Zone',␣
 ↪'Année'])['Aide_alimentaire'].sum().reset_index()
print(aideAlimentaire_evolution)
```

```
            Zone  Année  Aide_alimentaire
0    Afghanistan   2013         128238000
1    Afghanistan   2014          57214000
2        Algérie   2013          35234000
3        Algérie   2014          18980000
4        Algérie   2015          17424000
..           …    …                …
223       Égypte   2013           1122000
224     Équateur   2013           1362000
225     Éthiopie   2013         591404000
226     Éthiopie   2014         586624000
227     Éthiopie   2015         203266000

[228 rows x 3 columns]
```

[677]:
```
#Création d'une liste contenant les 5 pays qui ont le plus bénéficiées de␣
 ↪l'aide alimentaire
paysAide = ["République arabe syrienne", "Éthiopie", "Yémen", "Soudan du Sud",␣
 ↪"Soudan"]
```

[678]:
```
#filtre sur le dataframe avec notre liste
top5_aideAlimentaire = aideAlimentaire_grouped[aideAlimentaire_grouped['Zone'].
 ↪isin(paysAide)]
```

[679]:
```
# Affichage des pays avec l'aide alimentaire par année
print(top5_aideAlimentaire)
```

```
                          Zone  Année  Aide_alimentaire
157  République arabe syrienne   2013         563566000
158  République arabe syrienne   2014         651870000
159  République arabe syrienne   2015         524949000
160  République arabe syrienne   2016         118558000
189                     Soudan   2013         330230000
190                     Soudan   2014         321904000
191                     Soudan   2015          17650000
192               Soudan du Sud   2013         196330000
193               Soudan du Sud   2014         450610000
194               Soudan du Sud   2015          48308000
214                       Yémen   2013         264764000
215                       Yémen   2014         103840000
216                       Yémen   2015         372306000
217                       Yémen   2016         465574000
225                    Éthiopie   2013         591404000
226                    Éthiopie   2014         586624000
227                    Éthiopie   2015         203266000
```

3.9 - Pays avec le moins de disponibilité par habitant

[680]:
```
habitant_dispoAlimentaire = dispoAlimentaire[['Zone', 'Disponibilité␣
↪alimentaire (Kcal/personne/jour)']].groupby('Zone')['Disponibilité␣
↪alimentaire (Kcal/personne/jour)'].sum().reset_index()
#Affichage des 10 pays qui ont le moins de dispo alimentaire par personne
habitant_dispoAlimentaire.sort_values('Disponibilité alimentaire (Kcal/personne/
↪jour)', ascending=True).head(10)
```

[680]:

	Zone	Disponibilité alimentaire (Kcal/personne/jour)
128	République centrafricaine	1879.0
166	Zambie	1924.0
91	Madagascar	2056.0
0	Afghanistan	2087.0
65	Haïti	2089.0
133	République populaire démocratique de Corée	2093.0
151	Tchad	2109.0
167	Zimbabwe	2113.0
114	Ouganda	2126.0
154	Timor-Leste	2129.0

3.10 - Pays avec le plus de disponibilité par habitant

[681]:
```
#Affichage des 10 pays qui ont le plus de dispo alimentaire par personne
habitant_dispoAlimentaire.sort_values('Disponibilité alimentaire (Kcal/personne/
↪jour)', ascending=False).head(10)
```

[681]:

	Zone	Disponibilité alimentaire (Kcal/personne/jour)
11	Autriche	3770.0
16	Belgique	3737.0
159	Turquie	3708.0
171	États-Unis d'Amérique	3682.0
74	Israël	3610.0
72	Irlande	3602.0
75	Italie	3578.0
89	Luxembourg	3540.0
168	Égypte	3518.0
4	Allemagne	3503.0

3.11 - Exemple de la Thaïlande pour le Manioc

[682]:
```
#Calcul de la sous nutrition en Thaïlande
thailande_sousNutrition = sousNutrition.loc[(sousNutrition['Zone'] ==␣
 ↪'Thaïlande'), ["Zone", "Année", "sous_nutrition"]]
# Calculer la moyenne de la sous-nutrition en Thaïlande
moyenne_sous_nutrition = thailande_sousNutrition['sous_nutrition'].mean()
print("Moyenne de la sous-nutrition en Thaïlande :", moyenne_sous_nutrition)
```

```
Moyenne de la sous-nutrition en Thaïlande : 6133333.333333333
```

[683]:
```
# On calcule la proportion exportée en fonction de la proportion
thailande_manioc = dispoAlimentaire.loc[(dispoAlimentaire['Zone'] ==␣
 ↪'Thaïlande')&(dispoAlimentaire['Produit'] == 'Manioc') , ["Zone", "Produit",␣
 ↪"Production", "Exportations - Quantité", "Nourriture"]]
thailande_manioc['Proportion Exportée'] = (thailande_manioc['Exportations -␣
 ↪Quantité'] / thailande_manioc['Production'] * 100).round(2)
print("Proportion de Manioc exportée : {:.2f}%".
 ↪format(thailande_manioc['Proportion Exportée'].values[0]))
```

```
Proportion de Manioc exportée : 83.41%
```

[684]:
```
# Calculez la proportion de manioc utilisée pour la nourriture en pourcentage
proportion_nourriture_manioc = round((thailande_manioc['Nourriture'] /␣
 ↪thailande_manioc['Production']) * 100, 1)

# Affichez la proportion de manioc utilisée pour nourrir la population en␣
 ↪Thaïlande
print(f"Proportion de Manioc utilisée pour nourrir la population en Thaïlande :␣
 ↪{proportion_nourriture_manioc.values[0]}%")
```

```
Proportion de Manioc utilisée pour nourrir la population en Thaïlande : 2.9%
```

```
<h2 style="font-weight: bold; color: #001F3F;">Etape 3 - Analyse complémentaires</h2>
```

Analyse du gaspillage alimentaire

[685]:
```
# Calcul du taux global de gaspillage alimentaire
dispoAlimentaire['Taux de Gaspillage'] = dispoAlimentaire['Pertes'] /␣
 ↪dispoAlimentaire['Disponibilité intérieure'] *100
# Afficher le taux global de gaspillage alimentaire
taux_global = dispoAlimentaire['Taux de Gaspillage'].mean()
print("Taux global de gaspillage alimentaire : {:.2f}%".format(taux_global))

# Calcul du taux global de gaspillage alimentaire par pays
taux_global_par_pays = round(dispoAlimentaire.groupby('Zone')['Pertes'].sum() /␣
 ↪dispoAlimentaire.groupby('Zone')['Disponibilité intérieure'].sum() * 100,2)

# Afficher les 10 pays avec le taux le plus élevé de gaspillage
```

```
top_10_gaspi = taux_global_par_pays.nlargest(10)
print("\nLes 10 pays avec les taux de gaspillage les plus élevés :")
print(top_10_gaspi)
top_10_gaspi.to_csv('top_10_gaspi.csv', index=False)
```

```
Taux global de gaspillage alimentaire : 2.20%

Les 10 pays avec les taux de gaspillage les plus élevés :
Zone
Sierra Leone    33.35
Ghana           19.08
Cameroun        15.18
Angola          14.98
Dominique       14.18
Guinée          12.78
Malawi          12.22
Togo            11.62
Bulgarie        11.52
Bénin           11.34
dtype: float64
```

[686]:
```
# Calcul du nombre de personnes qui pourraient être nourries avec les 2,22 % de␣
↪gaspillage
gaspillage_total_kcal = dispo_alimentaire2017['dispo_kcal'].sum() * 0.0222  #␣
↪2,22 % en kilocalories
personnes_nourries = round(gaspillage_total_kcal / (365 * moyenneKcal))

print("Le total de disponibilité alimentaire gaspillée est de {:.2f} Kcal.".
↪format(gaspillage_total_kcal))
print("Cela pourrait nourrir environ", personnes_nourries, "êtres humains.")
```

```
Le total de disponibilité alimentaire gaspillée est de 169506532435263.09 Kcal.
Cela pourrait nourrir environ 206400648 êtres humains.
```

[687]:
```
# Listes de produits par type de produit
cereales_et_grains = ["Blé", "Maïs", "Orge", "Riz (Eq Blanchi)", "Millet",␣
↪"Avoine", "Seigle", "Sorgho", "Céréales, Autres"]
fruits = ["Agrumes, Autres", "Ananas", "Bananes", "Dattes", "Fruits, Autres",␣
↪"Pommes", "Raisin", "Citrons & Limes", "Pamplemousse"]
legumes = ["Légumes, Autres", "Tomates", "Pommes de Terre", "Pois", "Ignames",␣
↪"Patates douces", "Oignons", "Racines nda", "Piments", "Soja"]
viandes_et_produits_carnes = ["Abats Comestible", "Viande d'Ovins/Caprins",␣
↪"Viande de Bovins", "Viande de Volailles", "Viande, Autre", "Viande de␣
↪Suides", "Cephalopodes", "Crustacés", "Mollusques, Autres", "Poissons␣
↪Marins, Autres", "Poissons Pelagiques", "Perciform", "Poissons Eau Douce",␣
↪"Animaux Aquatiques Autre"]
```

```
produits_laitiers_et_graisses = ["Beurre, Ghee", "Crème", "Graisses Animales␣
 ↪Crue", "Huile Graines de Coton", "Huile d'Arachide", "Huile d'Olive", "Huile␣
 ↪de Colza&Moutarde", "Huile de Palme", "Huile de Soja", "Huile de Sésame",␣
 ↪"Huile de Tournesol", "Lait - Excl Beurre", "Huile de Coco", "Huile de Germe␣
 ↪de Maïs", "Huile de Palmistes", "Huiles de Foie de Poisso", "Huiles de␣
 ↪Poissons", "Huile de Son de Riz"]
cultures_et_graines_oleagineuses = ["Graines de coton", "Graines de tournesol",␣
 ↪"Graines Colza/Moutarde", "Graines de Sésame"]
produits_sucres = ["Sucre Eq Brut", "Sucre, betterave", "Sucre, canne", "Sucre␣
 ↪non centrifugé"]
boissons = ["Bière", "Boissons Alcooliques", "Boissons Fermentés", "Vin",␣
 ↪"Thé", "Café"]
autres_produits = ["Alcool, non Comestible", "Miscellanees", "Épices, Autres",␣
 ↪"Aliments pour enfants", "Edulcorants Autres"]
produits_a_base_de_plantes_et_legumineuses = ["Coco (Incl Coprah)", "Graines de␣
 ↪coton", "Arachides Decortiquees", "Palmistes", "Miel", "Graines Colza/
 ↪Moutarde", "Haricots", "Poivre", "Poissons", "Palmistes", "Noix", "Olives",␣
 ↪"Oeufs", "Légumineuses Autres"]
produits_marins = ["Piments", "Plantes Aquatiques"]

# Liste des types de produits
types_de_produits = [
    ("Céréales et Grains", cereales_et_grains),
    ("Fruits", fruits),
    ("Légumes", legumes),
    ("Viandes et Produits Carnés", viandes_et_produits_carnes),
    ("Produits Laitiers et Graisses", produits_laitiers_et_graisses),
    ("Cultures et Graines Oléagineuses", cultures_et_graines_oleagineuses),
    ("Produits sucrés", produits_sucres),
    ("Boissons", boissons),
    ("Autres Produits", autres_produits),
    ("Produits à base de plantes et Légumineuses",␣
 ↪produits_a_base_de_plantes_et_legumineuses),
    ("Produits Marins", produits_marins)
]

# Créez une liste pour stocker les taux de gaspillage globaux par type de␣
 ↪produit
taux_globaux_par_type = []

# Créez une liste pour stocker les noms de type de produit
types_de_produits_noms = []

for nom_type, liste_de_produits_pour_ce_type in types_de_produits:
    # Sélectionnez les produits correspondant à ce type
```

```
    produits_du_type = dispoAlimentaire[dispoAlimentaire['Produit'].
 ↪isin(liste_de_produits_pour_ce_type)].copy()  # Créez une copie du DataFrame

    # Calcul du taux de gaspillage global pour ce type de produit
    taux_gaspillage_global = round(produits_du_type['Pertes'].sum() /␣
 ↪produits_du_type['Disponibilité intérieure'].sum() * 100, 2)

    # Ajoutez le taux global à la liste
    taux_globaux_par_type.append(taux_gaspillage_global)

    # Ajoutez le nom du type de produit à la liste
    types_de_produits_noms.append(nom_type)

# Créez un DataFrame avec les résultats
tableau_resultats = pd.DataFrame({'Type de Produit': types_de_produits_noms,␣
 ↪'Taux de Gaspillage Global': taux_globaux_par_type})
print(tableau_resultats)
```

	Type de Produit	Taux de Gaspillage Global
0	Céréales et Grains	4.45
1	Fruits	9.33
2	Légumes	7.44
3	Viandes et Produits Carnés	0.26
4	Produits Laitiers et Graisses	2.10
5	Cultures et Graines Oléagineuses	2.74
6	Produits sucrés	3.04
7	Boissons	0.59
8	Autres Produits	0.17
9	Produits à base de plantes et Légumineuses	3.67
10	Produits Marins	0.50

Part de l'utilisation des principales céréales entre l'alimentation humaine et animale

[688]:
```
# Création d'un nouveau DataFrame pour stocker les proportions
proportions_df = pd.DataFrame(columns=['Céréale', 'Proportion pour␣
 ↪l\'alimentation humaine (%)', 'Proportion pour l\'alimentation animale (%)'])

for cereale_name in liste_cereale:
    # Création d'un sous-DataFrame pour une céréale spécifique
    cereale = dispoAlimentaire[dispoAlimentaire['Produit'] == cereale_name]

    # Calcul de la proportion d'alimentation humaine et arrondi
    proportion_humaine = round((cereale['Nourriture'].sum() /␣
 ↪cereale['Disponibilité intérieure'].sum()) * 100)

    # Calcul de la proportion d'alimentation animale et arrondi
    proportion_animale = round((cereale['Aliments pour animaux'].sum() /␣
 ↪cereale['Disponibilité intérieure'].sum()) * 100)
```

```
    # Ajout des données dans le DataFrame des proportions en utilisant pandas.
 ↪concat
    proportions_df = pd.concat([proportions_df, pd.DataFrame({'Céréale':␣
 ↪[cereale_name], 'Proportion pour l\'alimentation humaine (%)':␣
 ↪[proportion_humaine], 'Proportion pour l\'alimentation animale (%)':␣
 ↪[proportion_animale]})], ignore_index=True)

# Affichage du tableau de proportions
print(proportions_df)
```

```
            Céréale Proportion pour l'alimentation humaine (%)  \
0              Blé                                          67
1  Riz (Eq Blanchi)                                         79
2              Orge                                          5
3              Maïs                                         13
4            Seigle                                         33
5            Avoine                                         17
6            Millet                                         77
7            Sorgho                                         41
8  Céréales, Autres                                         19

  Proportion pour l'alimentation animale (%)
0                                         19
1                                          7
2                                         66
3                                         57
4                                         49
5                                         69
6                                         11
7                                         43
8                                         69
```

[ ]: