toyage-analyse-exploratoire-112023

May 5, 2024

[1]:
```
import pandas as pd
import numpy as np
import matplotlib as plt
import seaborn as sns
import plotly_express as px
import plotly.graph_objs as go
```

[393]:
```
import matplotlib.ticker as ticker
```

[2]:
```
df_disponibilite = pd.read_csv('/Users/helmisaddem/Documents/DAN-P9-data 2/
↪DisponibiliteAlimentaire_2017.csv')
```

[4]:
```
df_population = pd.read_csv('/Users/helmisaddem/Documents/DAN-P9-data 2/
↪Population_2000_2018.csv')
```

[276]:
```
donnees_securite = pd.read_csv('/Users/helmisaddem/Documents/donnees_securite.
↪csv')
```

[275]:
```
donnees_economiques = pd.read_csv('/Users/helmisaddem/Documents/
↪donnees_economiques.csv')
```

[306]:
```
df_croissance = pd.read_csv('/Users/helmisaddem/Documents/croissance_2017.csv')
```

[277]:
```
# Crée la liste des DataFrames
dfs = ['df_disponibilite', 'df_population', 'donnees_securite',␣
↪'donnees_economiques', 'df_croissance']

# Affiche la taille des DataFrames
print(f'Dimensions des DataFrames :')
for df in dfs:
    print(f'- {df} : {eval(df).shape}')

del df
```

```
Dimensions des DataFrames :
- df_disponibilite : (176600, 14)
- df_population : (4411, 15)
```

```
- donnees_securite : (792, 15)
- donnees_economiques : (844, 15)
```

[15]:
```python
#fonction qui va enlever les caractère spéciaux des noms de colonnes
def snake_case (df):
    """Renomme les variables d'un DataFrame selon la convention snake case."""
    df.columns = (
        df.columns
        .str.lower()
        .str.replace('\'', '_')
        .str.replace(' - ', '_')
        .str.replace('-', '_')
        .str.replace(' ', '_')
        .str.normalize('NFKD')
        .str.encode('ascii', errors='ignore')
        .str.decode('utf-8')
    )
```

[278]:
```python
# Appliquer la fonction 'snake_case' aux DataFrames
for df in dfs:
    snake_case(eval(df))

del df
```

[30]:
```python
df_total = df_population.loc[
    df_population['annee'] == 2017, ['code_zone', 'zone', 'valeur']
].reset_index()
```

[32]:
```python
df_total.rename(
    columns={'valeur': 'population_milles_hab'}, inplace=True)
```

[33]:
```python
df_total.head()
```

[33]:

	index	code_zone	zone	population_milles_hab
0	17	2	Afghanistan	36296.113
1	36	202	Afrique du Sud	57009.756
2	55	3	Albanie	2884.169
3	74	4	Algérie	41389.189
4	93	79	Allemagne	82658.409

[34]:
```python
#temporary_table pour le calcul du pourcentage d'évolution de la population
temporary_table = df_population.pivot_table(
    index='annee', columns=['code_zone', 'zone'], values='valeur').
 ↪fillna(method='bfill', axis='rows').pct_change().mean().
 ↪to_frame(name='evolution_population_pct').reset_index()
```

```
/var/folders/r_/fd0gwkxn6n995_hk_5lc9f540000gn/T/ipykernel_93035/2345157236.py:3
: FutureWarning: DataFrame.fillna with 'method' is deprecated and will raise in
```

```
a future version. Use obj.ffill() or obj.bfill() instead.
  index='annee', columns=['code_zone', 'zone'],
values='valeur').fillna(method='bfill', axis='rows').pct_change().mean().to_fram
e(name='evolution_population_pct').reset_index()
/var/folders/r_/fd0gwkxn6n995_hk_5lc9f540000gn/T/ipykernel_93035/2345157236.py:3
: FutureWarning: The default fill_method='pad' in DataFrame.pct_change is
deprecated and will be removed in a future version. Either fill in any non-
leading NA values prior to calling pct_change or specify 'fill_method=None' to
not fill NA values.
  index='annee', columns=['code_zone', 'zone'],
values='valeur').fillna(method='bfill', axis='rows').pct_change().mean().to_fram
e(name='evolution_population_pct').reset_index()
```

[35]:
```
df_total = pd.merge(
    df_total,
    temporary_table[['code_zone', 'evolution_population_pct']],
    on='code_zone',
    how='left')
```

[37]:
```
df_total["evolution_population_pct"] =␣
 ↪round((df_total["evolution_population_pct"]*100),2)
```

[38]:
```
df_total.head()
```

[38]:

	index	code_zone	zone	population_milles_hab	\
0	17	2	Afghanistan	36296.113	
1	36	202	Afrique du Sud	57009.756	
2	55	3	Albanie	2884.169	
3	74	4	Algérie	41389.189	
4	93	79	Allemagne	82658.409	

	evolution_population_pct
0	3.29
1	1.40
2	-0.45
3	1.72
4	0.12

[53]:
```
df_temporary = df_disponibilite.loc[df_disponibilite['element'] ==␣
 ↪'Nourriture'].pivot_table(
    index='code_zone', columns='produit', values='valeur'
).reset_index().fillna(0)
```

[54]:
```
# quantité de viande de volailles consommées par chaque pays et pourcentage de␣
 ↪la quantité totale consommée
#pourcentage calculé à partir des valeurs en milliers de tonnes de viande de␣
 ↪volailles divisée par les quantités en milliers de tonnes
```

```
#d'autres denrées destinées a l'alimentation humaine (nourriture)
df_temporary['consommation_volaille_pct'] = round((
    df_temporary['Viande de Volailles'] * 100 / df_temporary.sum(axis='columns')
),2)
```

[55]:
```
df_temporary.head()
```

[55]:

produit	code_zone	Abats Comestible	Agrumes, Autres	Alcool, non Comestible	\
0	1	18.0	0.0	0.0	
1	2	53.0	48.0	0.0	
2	3	17.0	0.0	0.0	
3	4	57.0	1.0	0.0	
4	7	53.0	252.0	0.0	

produit	Aliments pour enfants	Ananas et produits	Animaux Aquatiques Autre	\
0	0.0	1.0	0.0	
1	4.0	0.0	0.0	
2	0.0	0.0	0.0	
3	18.0	8.0	0.0	
4	5.0	657.0	0.0	

produit	Arachides Decortiquees	Avoine	Bananes	…	Tomates et produits	\
0	2.0	2.0	12.0	…	184.0	
1	1.0	0.0	133.0	…	0.0	
2	1.0	0.0	28.0	…	212.0	
3	4.0	0.0	86.0	…	1262.0	
4	121.0	1.0	1039.0	…	95.0	

produit	Viande d'Ovins/Caprins	Viande de Anim Aquatiq	Viande de Bovins	\
0	9.0	0.0	52.0	
1	151.0	0.0	95.0	
2	26.0	0.0	42.0	
3	277.0	0.0	192.0	
4	23.0	0.0	129.0	

produit	Viande de Suides	Viande de Volailles	Viande, Autre	Vin	\
0	26.0	47.0	0.0	5.0	
1	0.0	55.0	11.0	0.0	
2	21.0	47.0	0.0	29.0	
3	0.0	264.0	15.0	0.0	
4	224.0	315.0	10.0	34.0	

produit	Épices, Autres	consommation_volaille_pct
0	0.0	1.59
1	15.0	0.42
2	0.0	1.29
3	15.0	0.80

```
4                   0.0                          1.85

[5 rows x 100 columns]
```

[56]:
```
df_total = pd.merge(
    df_total,
    df_temporary[['code_zone', 'consommation_volaille_pct']],
    how='left',
    on='code_zone'
)
del df_temporary
```

[148]:
```
#A partir de la table disponibilité viandes de volailles je garde les colonnes␣
 ↪qui m'interesse
#df_disponibilite_volailles.pivot_table(index='code_zone', columns='element',␣
 ↪values='valeur').reset_index()
df_reduit = df_disponibilite.loc[df_disponibilite["produit"] == "Viande de␣
 ↪Volailles"].pivot_table(index='code_zone', columns='element',␣
 ↪values='valeur').reset_index()[["code_zone",
                                  "Disponibilité alimentaire (Kcal/
 ↪personne/jour)",
                                  "Disponibilité alimentaire en quantité␣
 ↪(kg/personne/an)",
                                  "Disponibilité intérieure",
                                  "Exportations - Quantité",
                                  "Importations - Quantité",
                                  "Production"]]
```

[149]:
```
df_reduit.head(10)
```

[149]:
```
element  code_zone  Disponibilité alimentaire (Kcal/personne/jour)  \
0                1                                            54.0
1                2                                             5.0
2                3                                            85.0
3                4                                            22.0
4                7                                            35.0
5                8                                           233.0
6                9                                           182.0
7               10                                           192.0
8               11                                            65.0
9               12                                           182.0

element  Disponibilité alimentaire en quantité (kg/personne/an)  \
0                                                        16.06
1                                                         1.53
2                                                        16.36
3                                                         6.38
```

```
4                                                    10.56
5                                                    54.10
6                                                    42.24
7                                                    47.65
8                                                    18.20
9                                                    43.17

element  Disponibilité intérieure  Exportations - Quantité  \
0                            47.0                      0.0
1                            57.0                      NaN
2                            47.0                      0.0
3                           277.0                      0.0
4                           319.0                      0.0
5                             7.0                      0.0
6                          1962.0                    207.0
7                          1171.0                     42.0
8                           173.0                     78.0
9                            26.0                      NaN

element  Importations - Quantité  Production
0                           35.0        11.0
1                           29.0        28.0
2                           38.0        13.0
3                            2.0       275.0
4                          277.0        42.0
5                            7.0         0.0
6                            8.0      2161.0
7                           16.0      1269.0
8                          110.0       148.0
9                           24.0         6.0
```

[150]:
```
snake_case(df_reduit)
```

[152]:
```
df_reduit["TDI"] = round((df_reduit["importations_quantite"]*100/(
    ␣
 ↪df_reduit["importations_quantite"]+df_reduit["production"]-df_reduit["exportations_quantite
```

[158]:
```
df_reduit.head()
```

[158]:
```
element  code_zone  disponibilite_alimentaire_(kcal/personne/jour)  \
0                1                                            54.0
1                2                                             5.0
2                3                                            85.0
3                4                                            22.0
4                7                                            35.0

element  disponibilite_alimentaire_en_quantite_(kg/personne/an)  \
```

```
0                                                  16.06
1                                                   1.53
2                                                  16.36
3                                                   6.38
4                                                  10.56

element  disponibilite_interieure  exportations_quantite  \
0                            47.0                    0.0
1                            57.0                    NaN
2                            47.0                    0.0
3                           277.0                    0.0
4                           319.0                    0.0

element  importations_quantite  production    TDI
0                         35.0        11.0  76.09
1                         29.0        28.0    NaN
2                         38.0        13.0  74.51
3                          2.0       275.0   0.72
4                        277.0        42.0  86.83
```

[153]:
```
df_total = pd.merge(df_total, df_reduit[["code_zone", "TDI",␣
↪"disponibilite_alimentaire_en_quantite_(kg/personne/an)"]], on="code_zone",␣
↪how="left")
```

[280]:
```
donnees_economiques.rename(columns={"code_zone_(fao)":"code_zone"},␣
↪inplace=True)
```

[440]:
```
donnees_economiques.head(3)
```

[440]:
```
  code_domaine            domaine  code_zone         zone  code_element  \
0           MK  Indicateurs macro          2  Afghanistan          6110
1           MK  Indicateurs macro          2  Afghanistan          6119
2           MK  Indicateurs macro          2  Afghanistan          6110

                   element  code_produit                produit  code_annee  \
0             Valeur US $         22008  Produit Intérieur Brut        2017
1  Valeur US $ par habitant         22008  Produit Intérieur Brut        2017
2             Valeur US $         22011    Revenu national brut        2017

   annee           unite        valeur symbole  \
0   2017  Millions d'USD  18896.352021       X
1   2017            US$    530.149831       X
2   2017  Millions d'USD  19145.017023       X

            description_du_symbole  note
0  Ciffre de sources internationales   NaN
1  Ciffre de sources internationales   NaN
```

```
2  Ciffre de sources internationales   NaN
```

[287]:
```
df_total = pd.merge(df_total, donnees_economiques.
↪loc[(donnees_economiques['element'] == 'Valeur US $ par habitant')
      & (donnees_economiques['produit'] == 'Produit Intérieur␣
↪Brut')][["code_zone", "valeur"]], on="code_zone", how="left")
```

[288]:
```
df_total.head()
```

[288]:

	index	code_zone	zone	population_milles_hab	\
0	17	2	Afghanistan	36296.113	
1	36	202	Afrique du Sud	57009.756	
2	55	3	Albanie	2884.169	
3	74	4	Algérie	41389.189	
4	93	79	Allemagne	82658.409	

	evolution_population_pct	consommation_volaille_pct	TDI	\
0	3.29	0.42	NaN	
1	1.40	6.37	24.27	
2	-0.45	1.29	74.51	
3	1.72	0.80	0.72	
4	0.12	2.06	49.24	

	disponibilite_alimentaire_en_quantite_(kg/personne/an)	PIB_par_habitant	\
0	1.53	2096.1	
1	35.69	13950.5	
2	16.36	12771	
3	6.38	11809.5	
4	19.47	53071.5	

	revenu_brut_habitant	valeur_x	valeur_y
0	537.126294	530.149831	1.889635e+04
1	6537.504869	6723.928582	3.808514e+05
2	4532.791977	4521.752217	1.301973e+04
3	4081.775086	4134.936055	1.700970e+05
4	45734.557501	44670.222282	3.690849e+06

[289]:
```
df_total = df_total.drop(columns=["PIB_par_habitant", "revenu_brut_habitant",␣
↪"valeur_y"])
```

[290]:
```
df_total.rename(columns={"valeur_x":"PIB_par_habitant"}, inplace=True)
```

[291]:
```
df_total.head()
```

[291]:

	index	code_zone	zone	population_milles_hab	\
0	17	2	Afghanistan	36296.113	
1	36	202	Afrique du Sud	57009.756	

```
2     55          3         Albanie               2884.169
3     74          4         Algérie              41389.189
4     93         79       Allemagne              82658.409

   evolution_population_pct  consommation_volaille_pct    TDI  \
0                      3.29                       0.42    NaN
1                      1.40                       6.37  24.27
2                     -0.45                       1.29  74.51
3                      1.72                       0.80   0.72
4                      0.12                       2.06  49.24

   disponibilite_alimentaire_en_quantite_(kg/personne/an)  PIB_par_habitant
0                                                  1.53            530.149831
1                                                 35.69           6723.928582
2                                                 16.36           4521.752217
3                                                  6.38           4134.936055
4                                                 19.47          44670.222282
```

[293]:
```
donnees_securite.rename(columns={"code_zone_(fao)":"code_zone"}, inplace=True)
```

[297]:
```
df_total = pd.merge(df_total,
                    donnees_securite.loc[donnees_securite["code_produit"] ==␣
 ↪21032][["code_zone", "valeur"]],
                    on = "code_zone",
                    how="left")
```

[299]:
```
df_total.rename(columns={"valeur":"indice_stabilite_politique"}, inplace=True)
```

[300]:
```
df_total.head()
```

[300]:
```
   index  code_zone            zone  population_milles_hab  \
0     17          2     Afghanistan               36296.113
1     36        202  Afrique du Sud               57009.756
2     55          3         Albanie                2884.169
3     74          4         Algérie               41389.189
4     93         79       Allemagne               82658.409

   evolution_population_pct  consommation_volaille_pct    TDI  \
0                      3.29                       0.42    NaN
1                      1.40                       6.37  24.27
2                     -0.45                       1.29  74.51
3                      1.72                       0.80   0.72
4                      0.12                       2.06  49.24

   disponibilite_alimentaire_en_quantite_(kg/personne/an)  PIB_par_habitant  \
0                                                  1.53            530.149831
1                                                 35.69           6723.928582
```

```
2                                                 16.36            4521.752217
3                                                  6.38            4134.936055
4                                                 19.47           44670.222282

  indice_stabilite_politique
0                       -2.8
1                      -0.28
2                       0.38
3                      -0.92
4                       0.59
```

[307]:
```
snake_case(df_croissance)
```

[310]:
```
df_croissance.rename(columns={"code_zone_(fao)":"code_zone"}, inplace=True)
```

[312]:
```
df_total = pd.merge(df_total,
                    df_croissance.loc[df_croissance["code_element"] ==␣
↪6129][["code_zone", "valeur"]],
                    on="code_zone",
                    how="left")
```

[313]:
```
df_total.rename(columns={"valeur":"pib_pct_croissance"}, inplace=True)
```

[316]:
```
df_total.head()
```

[316]:
```
   index  code_zone            zone  population_milles_hab  \
0     17          2     Afghanistan               36296.113
1     36        202  Afrique du Sud               57009.756
2     55          3         Albanie                2884.169
3     74          4         Algérie               41389.189
4     93         79       Allemagne               82658.409

   evolution_population_pct  consommation_volaille_pct    TDI  \
0                      3.29                       0.42    NaN
1                      1.40                       6.37  24.27
2                     -0.45                       1.29  74.51
3                      1.72                       0.80   0.72
4                      0.12                       2.06  49.24

   disponibilite_alimentaire_en_quantite_(kg/personne/an)  PIB_par_habitant  \
0                                                  1.53            530.149831
1                                                 35.69           6723.928582
2                                                 16.36           4521.752217
3                                                  6.38           4134.936055
4                                                 19.47          44670.222282

  indice_stabilite_politique  pib_pct_croissance
```

```
0                      -2.8           4.865789
1                     -0.28          17.703799
2                      0.38           9.765943
3                     -0.92           6.287892
4                      0.59           6.441276
```

[441]:
```
df_total = pd.merge(df_total, donnees_economiques.
↪loc[(donnees_economiques['element'] == 'Valeur US $ par habitant')
      & (donnees_economiques['produit'] == 'Revenu national␣
↪brut')][["code_zone", "valeur"]], on="code_zone", how="left")
```

[443]:
```
df_total.rename(columns={"valeur":"RNB_par_habitant"}, inplace=True)
```

[444]:
```
df_total.head(3)
```

[444]:
```
   index  code_zone            zone  population_milles_hab  \
0     17          2     Afghanistan              36296.113
1     36        202  Afrique du Sud              57009.756
2     55          3         Albanie               2884.169

   evolution_population_pct  consommation_volaille_pct    TDI  \
0                      3.29                       0.42    NaN
1                      1.40                       6.37  24.27
2                     -0.45                       1.29  74.51

   disponibilite_alimentaire_en_quantite_(kg/personne/an)  PIB_par_habitant  \
0                                               1.53            530.149831
1                                              35.69           6723.928582
2                                              16.36           4521.752217

  indice_stabilite_politique  pib_pct_croissance  RNB_par_habitant
0                       -2.8            4.865789        537.126294
1                      -0.28           17.703799       6537.504869
2                       0.38            9.765943       4532.791977
```

[445]:
```
print(f'Nombre de doublons dans le DataFrame : {df_total.duplicated().sum()}')
```

```
Nombre de doublons dans le DataFrame : 0
```

[446]:
```
print(f'Dimensions du DataFrame : {df_total.shape}')
```

```
Dimensions du DataFrame : (183, 12)
```

[447]:
```
df_total.isna().sum()
```

[447]:
```
index                                                 0
code_zone                                             0
```

```
zone                                                  0
population_milles_hab                                 0
evolution_population_pct                              0
consommation_volaille_pct                             9
TDI                                                  48
disponibilite_alimentaire_en_quantite_(kg/personne/an)   11
PIB_par_habitant                                      2
indice_stabilite_politique                            7
pib_pct_croissance                                    2
RNB_par_habitant                                      2
dtype: int64
```

[448]:
```
#supprimer les pays dont on ne dispose pas d'informations importantes tel que
 ↪le pourcentage de
#consommation de viande de volaille, taux de dépendance des importations et la
 ↪disponibilité alimentaire
df_final = df_total.loc[
    (~df_total['consommation_volaille_pct'].isna())
    & (~df_total['TDI'].isna())
    & (~df_total['disponibilite_alimentaire_en_quantite_(kg/personne/an)'].
 ↪isna())
]
```

[449]:
```
# examiner le reste des informations manquantes :
df_final[df_final.isna().any(axis='columns')]
```

[449]:
```
     index  code_zone                     zone  population_milles_hab  \
36     823         41        Chine, continentale              1421021.791
37     842        214  Chine, Taiwan Province de                23674.546
134   3219         70        Polynésie française                  276.102

     evolution_population_pct  consommation_volaille_pct    TDI  \
36                       0.56                       1.27   2.50
37                       0.43                       5.11  20.00
134                      0.80                       5.08  93.75

     disponibilite_alimentaire_en_quantite_(kg/personne/an)  PIB_par_habitant  \
36                                                   12.33         8729.136930
37                                                   33.17                  NaN
134                                                  47.40        19743.958993

    indice_stabilite_politique  pib_pct_croissance  RNB_par_habitant
36                         NaN            9.589146       8717.698269
37                         NaN                 NaN               NaN
134                        NaN            6.190222      19743.958993
```

[ ]:
```
# En regardant le TDI de la chine continentale et de Taiwan on se rend compte␣
 ↪qu'il est faible et que ses
# pays comptent pratiquement sur leur resssources internes (+ argument␣
 ↪géographique donc cout du transport ) donc on peut les omettre
#reste la polynésie française qui manque une seule valeur qui est le TDI et␣
 ↪qu'on peut lui affecter celui de la France
```

[450]:
```
df_final.loc[
    df_final['zone'] == 'France', 'indice_stabilite_politique'].values
```

[450]:
```
array(['0.28'], dtype=object)
```

[451]:
```
df_final.loc[df_final['zone'] == 'Polynésie française',
    'indice_stabilite_politique'] = df_final.loc[
    df_final['zone'] == 'France', 'indice_stabilite_politique'].values
```

[452]:
```
df_final[df_final.isna().any(axis='columns')]
```

[452]:

	index	code_zone	zone	population_milles_hab	evolution_population_pct	consommation_volaille_pct	TDI	disponibilite_alimentaire_en_quantite_(kg/personne/an)	PIB_par_habitant	indice_stabilite_politique	pib_pct_croissance	RNB_par_habitant
36	823	41	Chine, continentale	1421021.791	0.56	1.27	2.5	12.33	8729.13693	NaN	9.589146	8717.698269
37	842	214	Chine, Taiwan Province de	23674.546	0.43	5.11	20.0	33.17	NaN	NaN	NaN	NaN

[453]:
```
df_final = df_final.loc[
    (~df_final['indice_stabilite_politique'].isna())]
```

[454]:
```
df_final.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 133 entries, 1 to 181
Data columns (total 12 columns):
 #   Column                                                Non-Null Count
Dtype
---  ------                                                --------------
-----
 0   index                                                 133 non-null
```

```
int64
 1   code_zone                                                 133 non-null
int64
 2   zone                                                      133 non-null
object
 3   population_milles_hab                                     133 non-null
float64
 4   evolution_population_pct                                  133 non-null
float64
 5   consommation_volaille_pct                                 133 non-null
float64
 6   TDI                                                       133 non-null
float64
 7   disponibilite_alimentaire_en_quantite_(kg/personne/an)    133 non-null
float64
 8   PIB_par_habitant                                          133 non-null
float64
 9   indice_stabilite_politique                                133 non-null
object
 10  pib_pct_croissance                                        133 non-null
float64
 11  RNB_par_habitant                                          133 non-null
float64
dtypes: float64(8), int64(2), object(2)
memory usage: 13.5+ KB
```

[455]:
```
liste_initiale_zones = set(df_total['zone'].unique())
liste_finale_zones = set(df_final['zone'].unique())
# Crée la liste des zones supprimées
liste_zones_supprimees = liste_initiale_zones - liste_finale_zones

# Affiche la liste des zones supprimées
print(len(liste_zones_supprimees))
```

```
50
```

[ ]:
```
# Au total j'ai supprimé 50 pays soit par manque d'info pour les indicateurs␣
 ↪considérés les plus importants soit pour
#un TDI faible après premier filtrage
```

[456]:
```
df_final.head()
```

[456]:

	index	code_zone	zone	population_milles_hab	\
1	36	202	Afrique du Sud	57009.756	
2	55	3	Albanie	2884.169	
3	74	4	Algérie	41389.189	
4	93	79	Allemagne	82658.409	

```
5    131          7          Angola               29816.766

   evolution_population_pct  consommation_volaille_pct    TDI  \
1                      1.40                       6.37  24.27
2                     -0.45                       1.29  74.51
3                      1.72                       0.80   0.72
4                      0.12                       2.06  49.24
5                      3.57                       1.85  86.83

   disponibilite_alimentaire_en_quantite_(kg/personne/an)  PIB_par_habitant  \
1                                                35.69           6723.928582
2                                                16.36           4521.752217
3                                                 6.38           4134.936055
4                                                19.47          44670.222282
5                                                10.56           4042.681403

  indice_stabilite_politique  pib_pct_croissance  RNB_par_habitant
1                      -0.28           17.703799       6537.504869
2                       0.38            9.765943       4532.791977
3                      -0.92            6.287892       4081.775086
4                       0.59            6.441276      45734.557501
5                      -0.38           20.766646       3791.443851
```

[462]:
```
#Si on veut supprimer PIB_par_habitant, on le supprime à ce niveau
df_final_new = df_final.drop(columns=["code_zone", "index"], axis=1).
↪reset_index().drop(columns=["index"], axis=1)
```

[463]:
```
df_final_new.head(3)
```

[463]:
```
            zone  population_milles_hab  evolution_population_pct  \
0  Afrique du Sud               57009.756                      1.40
1         Albanie                2884.169                     -0.45
2         Algérie               41389.189                      1.72

   consommation_volaille_pct    TDI  \
0                       6.37  24.27
1                       1.29  74.51
2                       0.80   0.72

   disponibilite_alimentaire_en_quantite_(kg/personne/an)  PIB_par_habitant  \
0                                                35.69           6723.928582
1                                                16.36           4521.752217
2                                                 6.38           4134.936055

  indice_stabilite_politique  pib_pct_croissance  RNB_par_habitant
0                      -0.28           17.703799       6537.504869
1                       0.38            9.765943       4532.791977
```

```
2                        -0.92            6.287892       4081.775086
```

[464]:
```
#je constate que l'indice de stabilité politique n'est pas de type float :␣
 ↪problème avec les graph
df_final_new["indice_stabilite_politique"] =␣
 ↪df_final_new["indice_stabilite_politique"].astype('float64')
```

[458]:
```
# voir si la population formée par nos pays représente plus de 60% de la␣
 ↪population mondiale en 2017
(df_final_new["population_milles_hab"]*1000).sum()
# population mondiale en 2017 : 7369316352.32
```

[458]:
```
5249893313.0
```

[459]:
```
(((df_final_new["population_milles_hab"]*1000).sum()*100)/7369316352.32).
 ↪round(2)
```

[459]:
```
71.24
```

[465]:
```
indicateurs = (list(df_final_new.columns))
indicateurs.remove('zone')
print(indicateurs)
```

```
['population_milles_hab', 'evolution_population_pct',
'consommation_volaille_pct', 'TDI',
'disponibilite_alimentaire_en_quantite_(kg/personne/an)', 'PIB_par_habitant',
'indice_stabilite_politique', 'pib_pct_croissance', 'RNB_par_habitant']
```

[466]:
```
df_final_new.describe().T.style.format('{:.2f}')
```

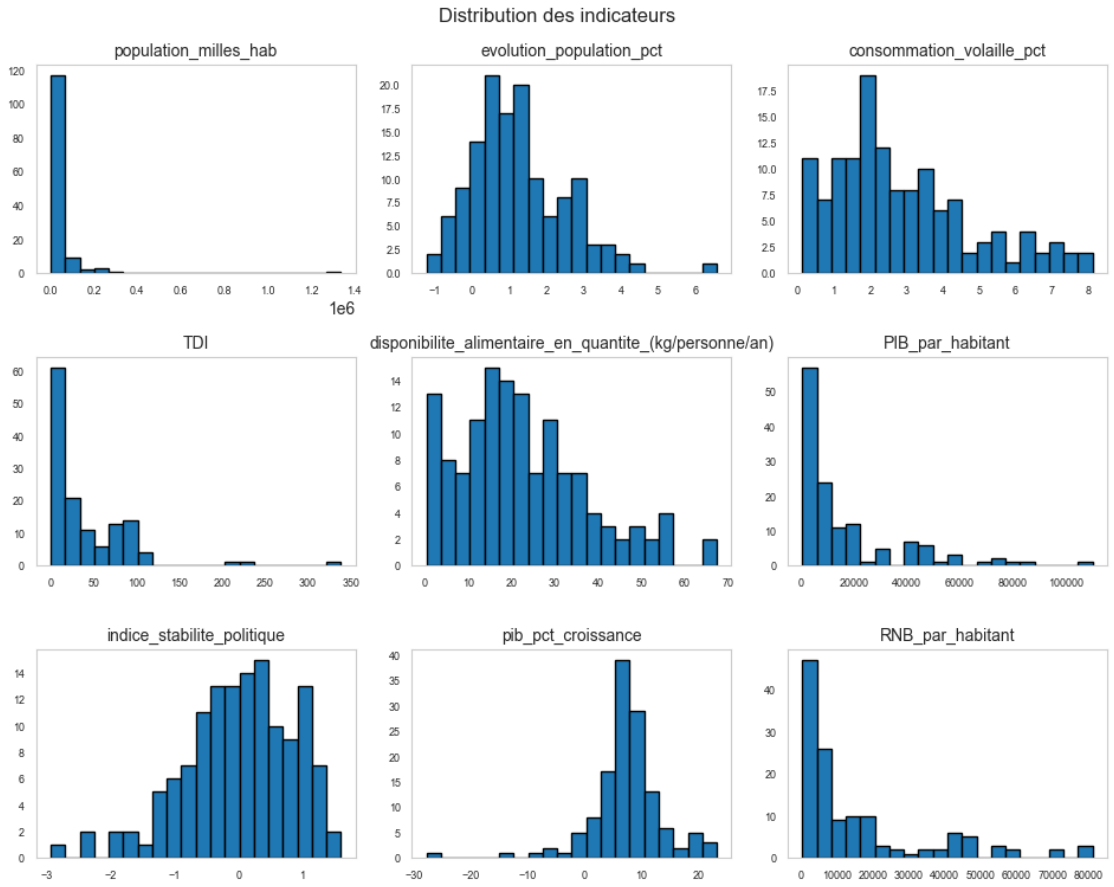
[466]:
```
<pandas.io.formats.style.Styler at 0x12cd78850>
```

[467]:
```
df_final_new.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 133 entries, 0 to 132
Data columns (total 10 columns):
 #   Column                                                Non-Null Count
Dtype
---  ------                                                --------------
-----
 0   zone                                                  133 non-null
object
 1   population_milles_hab                                 133 non-null
float64
 2   evolution_population_pct                              133 non-null
float64
```

```
 3   consommation_volaille_pct                                133 non-null
float64
 4   TDI                                                      133 non-null
float64
 5   disponibilite_alimentaire_en_quantite_(kg/personne/an)   133 non-null
float64
 6   PIB_par_habitant                                         133 non-null
float64
 7   indice_stabilite_politique                               133 non-null
float64
 8   pib_pct_croissance                                       133 non-null
float64
 9   RNB_par_habitant                                         133 non-null
float64
dtypes: float64(9), object(1)
memory usage: 10.5+ KB
```

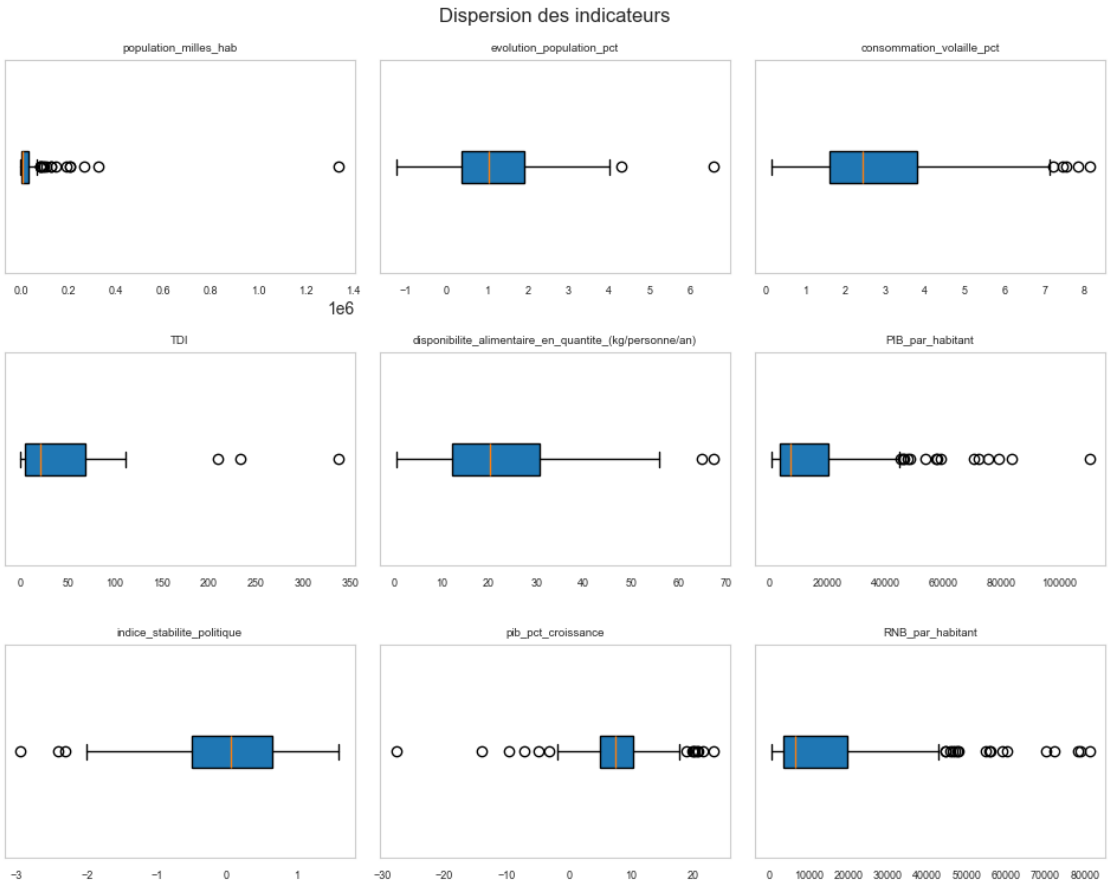
[468]:
```
#distribution des indicateurs pour chaque variable

fig, axs = plt.pyplot.subplots(3, 3, figsize=(9.6, 7.68))

for i, indicateur in enumerate(indicateurs):
    r = i // 3
    c = i % 3
    axs[r, c].hist(indicateur, data=df_final_new, edgecolor='k', bins=20)
    axs[r, c].set_title(indicateur, size=10)
    axs[r, c].tick_params(axis='both', which='both', labelsize=6.5)

# Supprime les graphiques vides
[fig.delaxes(ax) for ax in axs.flatten() if not ax.has_data()]

fig.suptitle('Distribution des indicateurs')

plt.pyplot.tight_layout()

plt.pyplot.show()

del fig, axs, i, r, c, indicateur
```



[ ]:
```
#quand j'ai changé le type de la colonne (object --> float64) indice de stabilité, l'axe des abscisses a changé
```

[469]:
```
# Dispersion des indicateurs pour chaque variable

fig, axs = plt.pyplot.subplots(3, 3, figsize=(9.6, 7.68))

for i, indicateur in enumerate(indicateurs):
    r = i // 3
    c = i % 3
    axs[r, c].boxplot(indicateur, data=df_final_new, patch_artist=True, vert=False)
    axs[r, c].set_title(indicateur, size=7)
    axs[r, c].yaxis.set_major_locator(plt.pyplot.NullLocator())
    axs[r, c].tick_params(axis='x', which='both', labelsize=6.5)


[fig.delaxes(ax) for ax in axs.flatten() if not ax.has_data()]
```

```
fig.suptitle('Dispersion des indicateurs')

plt.pyplot.tight_layout()

plt.pyplot.show()

del fig, axs, i, indicateur, r, c
```



[ ]:
```
# données non centrée donc distribution non normale, plusieurs outliers : vraies valeurs aberrantes ou non ?
#je crée une fontion outliers_values qui m'affiche tous les outliers pour bien les étudier et trancher si c'est des valeurs à garder ou à
#supprimer
```

[425]:
```
def outliers_values(indicateur):



    Q1 = df_final_new[indicateur].quantile(0.25)
```

```
    Q3 = df_final_new[indicateur].quantile(0.75)
    IQR = Q3 - Q1
    max_boxplot = 1.5 * IQR + Q3
    min_boxplot = Q1 - 1.5 * IQR



    return (df_final_new.loc[
            (df_final_new[indicateur] >= max_boxplot) |␣
↪(df_final_new[indicateur] <= min_boxplot), ['zone', indicateur]
        ].sort_values(by=indicateur, ascending=False))
```

[470]:
```
outliers_values(indicateurs[8])
```

[470]:

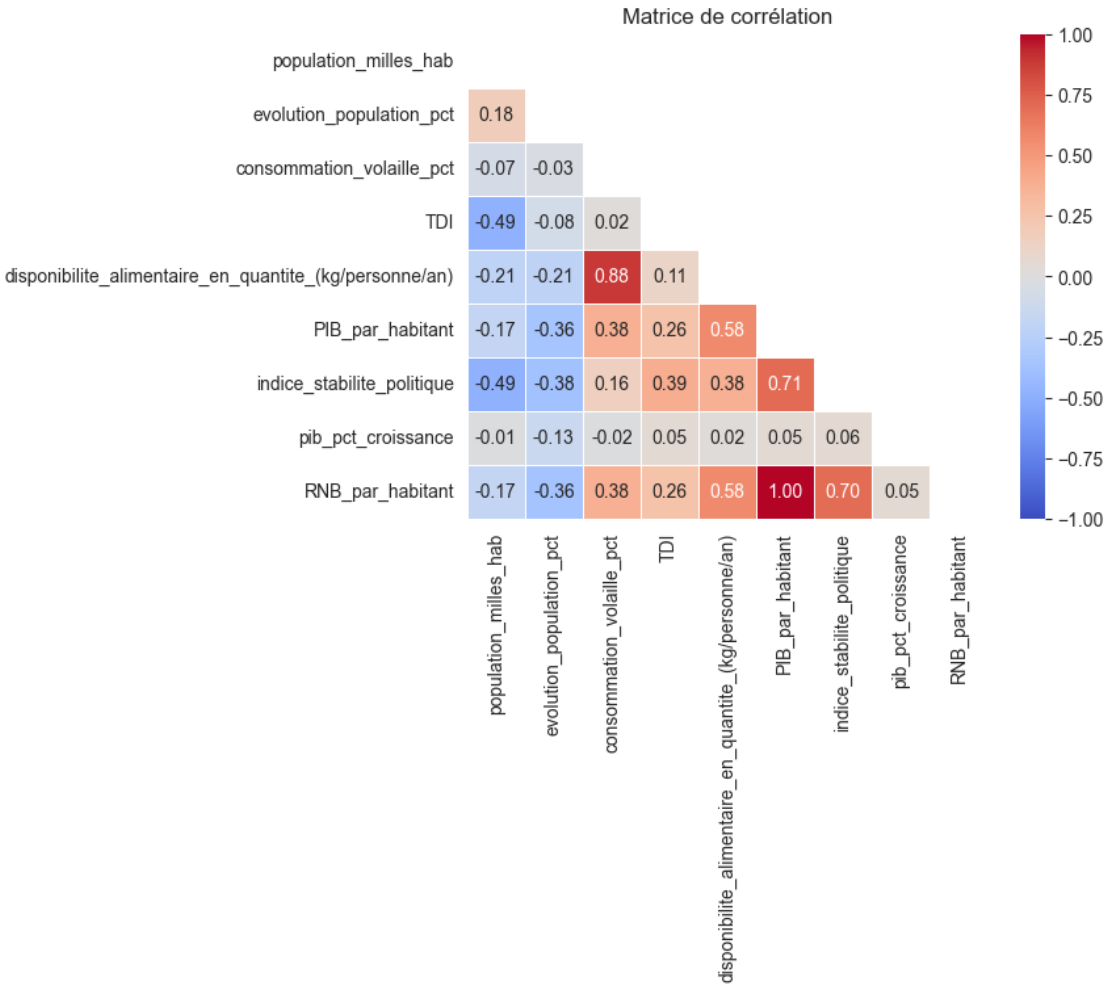
	zone	RNB_par_habitant
118	Suisse	81118.449929
76	Luxembourg	78873.811051
89	Norvège	78263.704694
61	Islande	72312.699769
26	Chine - RAS de Macao	70300.937274
42	États-Unis d'Amérique	60297.391594
33	Danemark	59155.352625
60	Irlande	56058.607923
9	Australie	55633.452624
117	Suède	54722.389209
96	Pays-Bas	47903.524245
25	Chine - RAS de Hong-Kong	47724.637596
10	Autriche	46984.806580
45	Finlande	46435.935523
3	Allemagne	45734.557501
14	Belgique	44567.104135
23	Canada	44551.243913

[471]:
```
matrice_corr = df_final_new.corr(numeric_only=True, method='spearman')
matrice_corr.style.format('{:.2f}')
```

[471]:
```
<pandas.io.formats.style.Styler at 0x12a9e4b50>
```

[472]:
```
fig, ax = plt.pyplot.subplots(figsize=(6.4, 4.8))
mask = np.triu(np.ones_like(matrice_corr, dtype=bool))
sns.heatmap(matrice_corr, annot=True, fmt='.2f', vmin=-1, vmax=1,
annot_kws=None, linewidths=0.6, cmap='coolwarm', ax=ax, mask=mask)

ax.set_title('Matrice de corrélation')

plt.pyplot.show()
```

```
del fig, ax
```

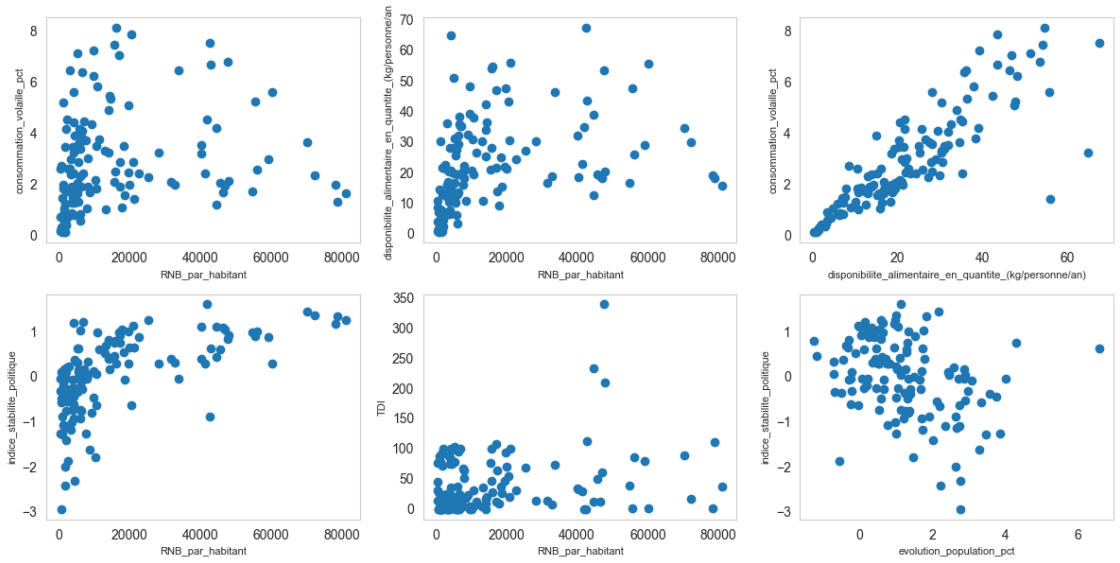


[ ]:
```
#Forte correlation positive entre :
##disponibilité alimentaire en Kg/personne/an et pourcentage de consommation de␣
 ↪volaille
##RNB par habitant et indice de stabilité politique : 0.7
##RNB par habitant et disponiblité alimentaire en Kg/personne/an
```

[ ]:
```
fig, ax = plt.pyplot.subplots(figsize=(6, 4))
mask = np.triu(np.ones_like(df_final_new[["RNB_par_habitant",␣
 ↪"consommation_volaille_pct", "disponibilite_alimentaire_en_quantite_(kg/
 ↪personne/an)",
             "indice_stabilite_politique", "TDI"]].corr(method='spearman'),␣
 ↪dtype=bool))
sns.heatmap(df_final_new[["RNB_par_habitant", "consommation_volaille_pct",␣
 ↪"disponibilite_alimentaire_en_quantite_(kg/personne/an)",
```

```
            "indice_stabilite_politique", "TDI"]].corr(method='spearman'),␣
 ↪annot=True, fmt='.2f', vmin=-1, vmax=1,
annot_kws=None, linewidths=0.6, cmap='coolwarm', ax=ax, mask=mask)

ax.set_title('Matrice de corrélation')

plt.pyplot.show()

del fig, ax
```

[498]:
```
fig, axs = plt.pyplot.subplots(nrows=2, ncols=3,
                               figsize=(12, 6))
i = 0
for r in range(2):
    for c in range(2):
        axs[r, c].scatter(df_final_new["RNB_par_habitant"],␣
 ↪df_final_new[list[i]])
        axs[r, c].set_xlabel("RNB_par_habitant", size=8)
        axs[r, c].set_ylabel(list[i], size=8)
        i += 1

axs[0, 2].scatter(df_final_new["disponibilite_alimentaire_en_quantite_(kg/
 ↪personne/an)"],
                  df_final_new["consommation_volaille_pct"])
axs[0, 2].set_xlabel("disponibilite_alimentaire_en_quantite_(kg/personne/an)",␣
 ↪size=8)
axs[0, 2].set_ylabel("consommation_volaille_pct", size=8)

axs[1, 2].scatter(df_final_new["evolution_population_pct"],
                  df_final_new["indice_stabilite_politique"])
axs[1, 2].set_xlabel("evolution_population_pct", size=8)
axs[1, 2].set_ylabel("indice_stabilite_politique", size=8)

fig.tight_layout()
```



[499]:
```
df_final_new.to_csv('/Users/helmisaddem/Downloads/FAOSTAT_data_fr_4-14-2024 (1).
↪csv', index=False)
```