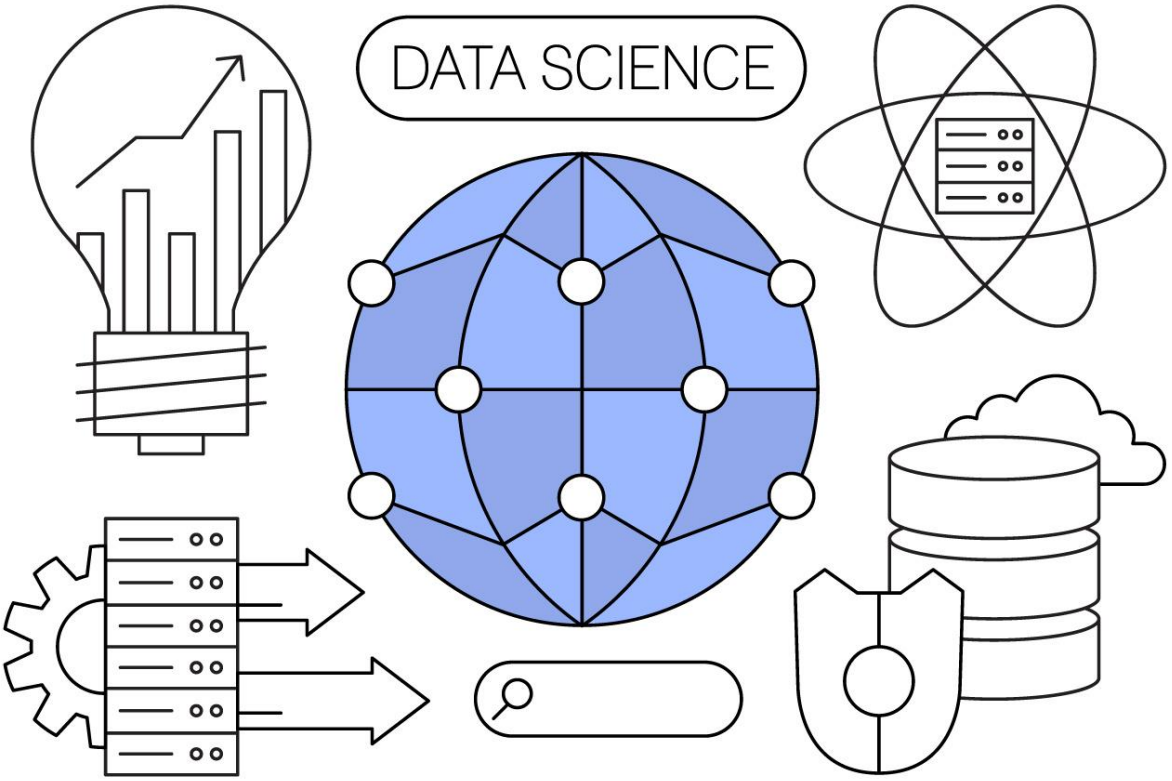
Rapport Projet de Science des données 3



Elise Commandré/Grace Palenfo

Département MIASHS, UFR 6, Informatique, Mathématiques et Statistiques
Université Paul Valéry Montpellier 3
Novembre 2023

Présentation du sujet

Dans le cadre de la concrétisation de notre projet, initialement axé sur le réchauffement climatique, nous avons choisi d'aborder le sujet du méthane.
En effet, le méthane est responsable d'environ 30% de la hausse des températures mondiales depuis la révolution industrielle; il est donc une des causes du réchauffement climatique.
Le secteur de l'énergie, comprenant le pétrole, le gaz naturel, le charbon et la bioénergie, représente près de 40% des émissions de méthane liées à l'activité humaine.
On a donc trouvé un jeu de données sur le site kaggle.com qui contient des informations sur la quantité d'émissions de méthane en fonction des continents et des pays, les types, les sous-secteurs et les raisons d'émissions ainsi que des années.
Source: https://www.kaggle.com/datasets/ashishraut64/global-methane-emissions

Nettoyage des données

```
import numpy as np
import seaborn as sns
import geopandas
import matplotlib.pyplot as plt
import pandas as pd
import matplotlib.colors as mplc
import plotly.express as px
!pip install geopandas
```

Nous avons commencé par importer tous les modules nécessaires au bon déroulement de notre code

```
data = pd.read_csv("Methane_final.csv", sep=",")
#On supprime les colonnes qui ne sont pas utiles
data=data.drop(columns=["baseYear"])
data=data.drop(columns=["notes"])
data=data.drop(columns=["Unnamed: 0"])
#On supprime les lignes qui ne sont pas utiles
data = data[(data['country'] != 'European Union') & (data['country'] !=
'Other EU7 countries')& (data['country'] != 'Other EU17 countries')&
(data['country'] != 'World')]
data1 = data[data['segment'] == 'Total']
#data1.to_csv("nouv.csv", index=False) # On enregistre le nouveau fichier
nettoyé
```

Nous avons lu notre fichier csv avec la fonction “read_csv” de Pandas et nous l’avons mis dans un dataframe appelé “data”.
Puis nous avons nettoyé les données: nous avons tout d’abord supprimé les colonnes qui ne nous intéressaient pas avec la méthode ‘drop’ de la librairie Pandas puis les lignes qui ne nous intéressaient pas.

Première figure réalisée par E.Commandré:

```
world =
geopandas.read_file(geopandas.datasets.get_path('naturalearth_lowres'))
#On remplace les noms qui ne correspondent pas entre notre jeu de données et
le GeoDataFrame de Geopandas
world['name'] = world['name'].replace('United States of America', 'United
States')
world['continent'] = world['continent'].replace('South America', 'Central
and South America')
world['continent'] = world['continent'].replace('Asia', 'Asia Pacific')

merged_data = world.merge(data1,  left_on='name', right_on='country')
# Liste des colonnes à supprimer
colonnes_a_supprimer = ['pop_est', 'continent',
'name','iso_a3','gdp_md_est']
# Supprimer les colonnes spécifiées
merged_data = merged_data.drop(colonnes_a_supprimer, axis=1)

# Créer une figure et un axe
fig, ax = plt.subplots(1, 1, figsize=(30, 20))

merged_data.plot(column='emissions',
                 cmap='viridis',
                 linewidth=0.8, ax=ax, edgecolor='0.8',
                 norm=mplc.LogNorm(vmin=merged_data['emissions'].min(),
vmax=merged_data['emissions'].max()),
                 legend=True, cax=plt.axes([0.9, 0.25, 0.02, 0.5]))
ax.axis(False)
ax.set_title('Emissions mondiales de méthane', fontdict={'fontsize': '25',
'fontweight' : '3'})
plt.show()
plt.savefig("carte.png")
```

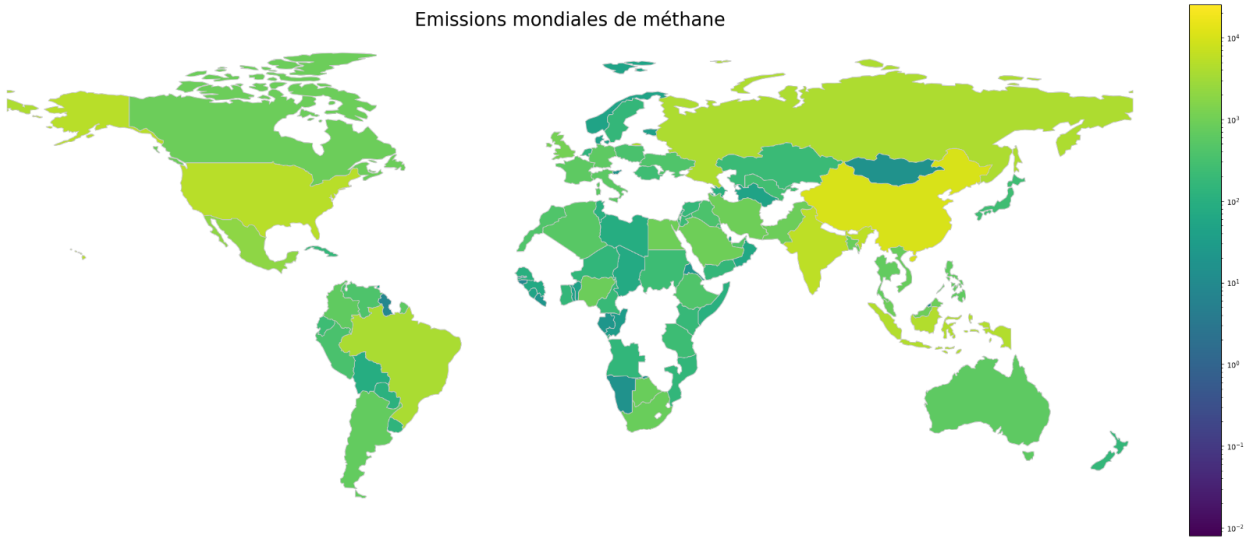
Dans ce code nous commençons par lire et mettre dans la variable world le GeoDataFrame créé à partir du jeu de données “naturalearth_lowres” se trouvant dans la bibliothèque Geopandas et qui contient des informations géographiques sur les pays du monde, telles que les limites géographiques, les noms des pays, etc.
La fonction “read_file” de Geopandas est utilisée pour lire des fichiers géographiques: dans ce cas, elle récupère le chemin du fichier du jeu de données "naturalearth_lowres" obtenu à partir de “get_path” et charge les données dans un objet GeoDataFrame appelé ici “world”.
Après avoir obtenu notre GeoDataFrame, nous remplaçons les noms des continents qui ne correspondent pas entre notre dataframe et le GeoDataFrame (différences d’appellations) avec la fonction “replace”.
Une fois cela fait, nous fusionnons notre dataframe avec notre GeoDataFrame à l’aide de la fonction “merge” en utilisant la colonne “pays” du dataframe “data1” et la colonne “name” du GeoDataFrame “world” comme clés de fusions dans “merged_data”.

Nous supprimons ensuite les colonnes inutiles avec la fonction “drop”.

Une fois ce traitement effectué, nous pouvons passer à la création de la carte choroplèthe des émissions mondiales de méthane en utilisant le GeoDataFrame “merged_data” avec la bibliothèque Geopandas et Matplotlib.
`fig, ax = plt.subplots(1, 1, figsize=(30, 20))` est utilisée pour créer une figure et un axe dans Matplotlib;

On utilise ensuite la fonction “plot” pour représenter graphiquement les données spatiales contenues dans le GeoDataFrame. Cette fonction prend en paramètre la colonne à utiliser pour déterminer la coloration de chaque zone sur la carte (ici c’est la colonne “emissions”), l’argument “cmap” qui spécifie la carte de couleurs à utiliser pour représenter les valeurs de la colonne 'emissions' (ici “viridis”), l’argument “linewidth” qui définit l'épaisseur des lignes de contour des zones géographiques, l’argument “ax=ax” qui indique l'axe sur lequel la carte sera tracée (ici, ax est l'axe créé précédemment avec plt.subplots), l’argument “edgecolor” qui définit la couleur des lignes des contours des zones géographiques, et enfin l’ajout de la légende avec une échelle logarithmique.

Enfin, on supprime les axes, et on ajoute un titre au graphique avec la fonction “set_title”.
On affiche la figure avec “plt.show()”.
Enfin, on utilise la fonction “plt.savefig” pour enregistrer notre figure en image.



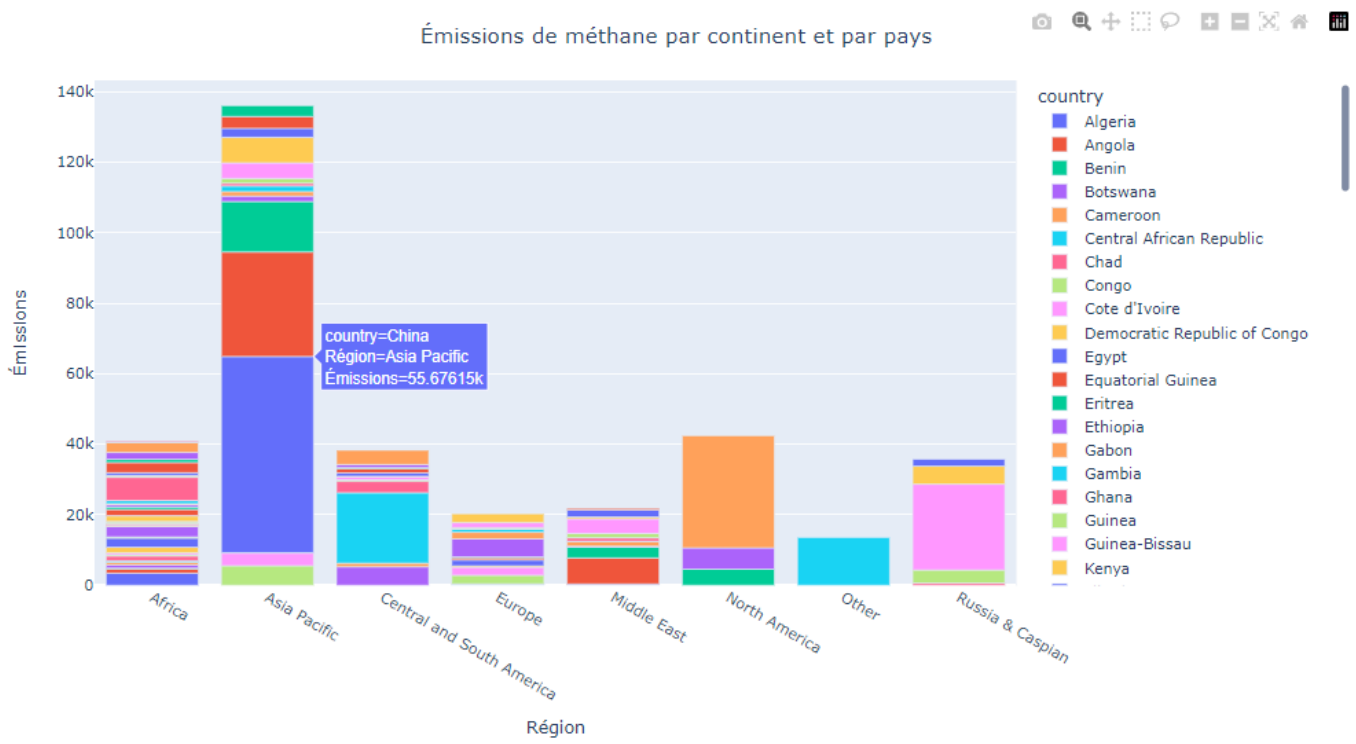
La carte choroplèthe permet d'avoir une visualisation globale des émissions de méthane par pays grâce à une échelle de couleur variant selon la quantité d'émissions. Elle permet ainsi de pouvoir comparer visuellement les émissions selon les pays afin d'avoir un bref aperçu des pays les plus émetteurs et à contrario des moins émetteurs.
On remarque ainsi que la Chine, les Etats-Unis, la Russie et le Brésil sont les pays qui émettent le plus de méthane.

Deuxième figure réalisée par E.Commandré:

```
#On fait la somme des émissions par pays
sum_by_country = data1.groupby(['region',
'country'])['emissions'].sum().reset_index()
# Créer un graphique à barres interactif avec Plotly Express
fig = px.bar(sum_by_country, x='region', y='emissions', color='country',
             labels={'emissions': 'Émissions', 'region': 'Région'},
             height=600, width=1100)

fig.update_layout(title_text='Émissions de méthane par continent et par
pays', title_x=0.5)

# Afficher le graphique interactif
fig.show()
```

Dans ce code, on commence par faire une agrégation des données dans le DataFrame “data1” en regroupant les lignes par les colonnes 'region' et 'country' et en calculant la somme des émissions pour chaque groupe à l’aide des fonctions “groupby” et “sum”.
Une fois la somme des émissions effectuée, on passe à la création du graphique en utilisant la fonction “px.bar()” de la bibliothèque Plotly Express. Cette fonction prend en argument la somme des émissions obtenue précédemment, la variable à utiliser pour l'axe des abscisses (ici x=”region”), la variable à utiliser pour l'axe des ordonnées (ici y=”emissions”), l’argument “color” qui permet de colorer les barres en fonction des valeurs de la colonne 'country' (chaque pays aura une couleur différente sur le graphique), l’argument “labels” qui permet de définir des étiquettes personnalisées pour les axes X et Y (dans ce cas, 'Emissions' est utilisé comme étiquette pour l'axe Y et 'Région' pour l'axe X) et enfin les dimensions de la figure générée en pixels.
On ajoute ensuite un titre au graphique avec la fonction “update_layout”, et on affiche la figure avec la fonction “fig.show().

Après avoir eu un aperçu global des émissions de méthane par pays grâce à la carte, nous réalisons un graphique barplot interactif qui permet d'observer plus en détail la quantité d'émissions par continent. Grâce à l'outil intéractif, on peut voir en détail en passant la souris sur les barplots la quantité d'émissions de chaque pays dans les continents et en double-cliquant sur les pays situés dans la légende, on peut avoir un zoom uniquement sur le pays cliqué, ce qui permet d’avoir une observation plus précise. On peut aussi en cliquant une seule fois sur un pays dans la légende, l’enlever des observations.



On observe donc que l'Asie Pacifique est le plus gros émetteur de méthane avec en particulier la Chine (55.67k d'émissions), l'Inde (29.67k d'émissions) et l'Indonésie (14.3k d'émissions). Les Etats-Unis apparaissent aussi en gros émetteurs dans la partie Amérique du Nord avec 31.8k d'émissions. Ces observations sont en adéquation avec les observations visuelles réalisées sur la carte précédente.

Troisième figure réalisée par G.Palenfo:

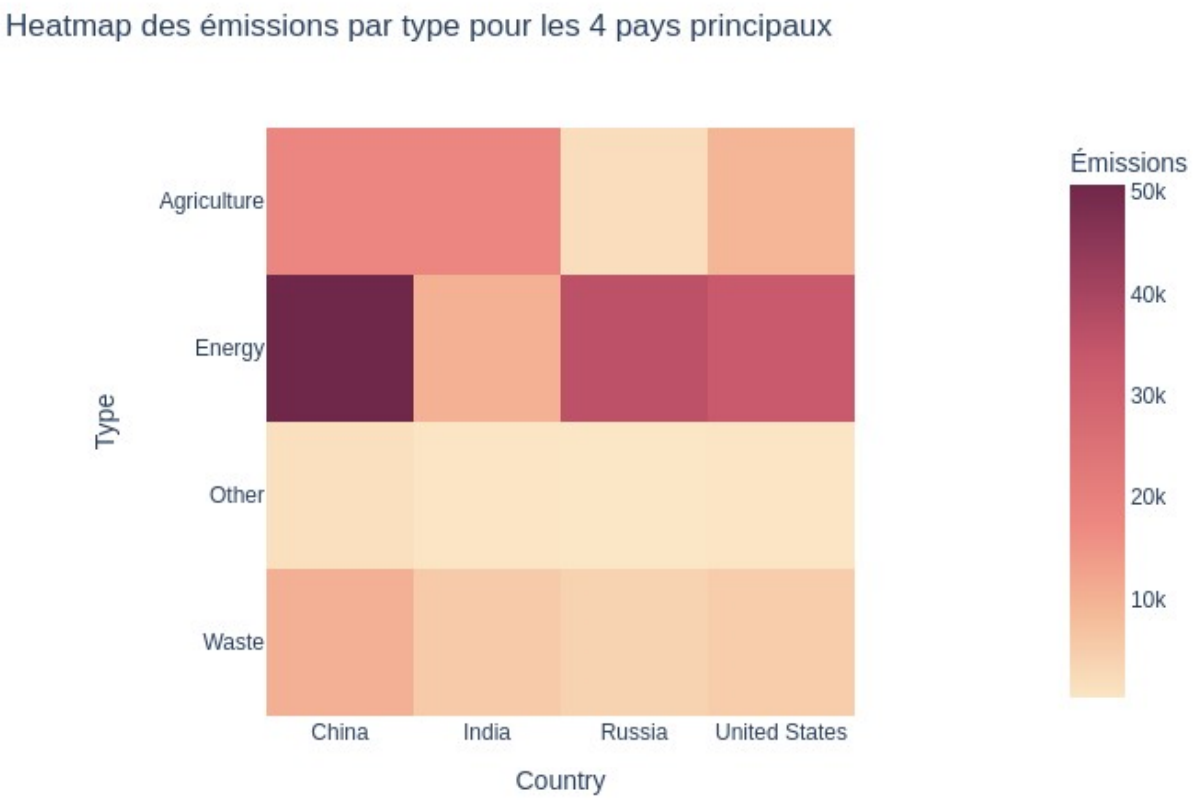
```
# 1.Refaire appel au jeu de données initial
data2= pd.read_csv("Methane_final.csv", sep=";")

# 2. Calculer les émissions totales par pays
emissions_par_pays =
data2.groupby('country')['emissions'].sum().reset_index()

# 3. Sélectionner les 4 pays ayant les émissions les plus élevées (en
excluant 'World')
pays_principaux = emissions_par_pays[emissions_par_pays['country'] !=
'World'].nlargest(4, 'emissions')['country']

# 4. Filtrer les données pour inclure uniquement les 4 pays principaux
donnees_pays_principaux = data2[data2['country'].isin(pays_principaux)]

# 5. Créer une heatmap interactive
fig_heatmap_type_pays_principaux =
px.imshow(donnees_pays_principaux.pivot_table(index='type',
```

```
columns='country', values='emissions', aggfunc='sum'),
labels=dict(x='Country', y='Type', color='Émissions'),
color_continuous_scale='Burgyl',
title='Heatmap des émissions par type pour les 4 pays principaux')
# 6. Afficher le graphique interactif
#fig_heatmap_type_pays_principaux.write_image("heatmap.png") # Permet
d'enregistrer l'image sous plotly.express
fig_heatmap_type_pays_principaux.show()
```

Nous avons d'abord lu encore une fois le fichier csv dans une variable nommée data2 car des modifications ont été faites dans data et data1 qui ne conviennent pas à la suite de notre projet.

Ensuite nous avons calculé dans *emissions_par_pays* une somme des émissions de méthane en les regroupant par pays et ainsi pouvoir garder que les 4 pays les plus émetteurs de méthane. Nous avons également dans *pays_principaux* exclus la variable 'World' dans 'country' pour que les résultats ne soient pas biaisés 'World ' étant le reste des pays du monde non évoqué. Cela dit nous filtrons dans *donnes_pays_principaux* les données du dataframe data2 pour inclure uniquement les lignes où la colonne 'country' a des valeurs présentes dans la liste *pays_principaux*.
Puis dans *fig_heatmap_pays_principaux*, nous utilisons Plotly Express pour créer une heatmap interactive à partir des émissions par type pour les 4 pays principaux, en utilisant les données du DataFrame *donnes_pays_principaux* . En effet, ***px.imshow()*** est une fonction de Plotly Express pour créer une heatmap. Elle prend en paramètre le dataframe pivoté qui est réalisé par la fonction ***.pivot_table()*,** ainsi que d'autres options comme les étiquettes avec ***labels***, la palette de couleurs avec ***color_continous_scale***, etc. Et enfin nous terminons le code par *fig_heatmap_pays_principaux.show()* pour afficher la heatmap.

Nous pouvons ainsi voir ci-après une heatmap interactive sur les plus quatre pays les plus émetteurs à savoir la Chine, l'Inde, la Russie et les Etats Unis en fonction des quatre types d'émissions à savoir l'agriculture , energy, Other, waste(les déchets). On en a conclu que c'est l'émission dû à l'énergie qui pollue le plus et contribue à augmenter les températures terrestres.



Quatrième figure réalisée par G.Palenfo:

```
# 1.Filtrer les données pour exclure la raison 'All'
data2_filtered = donnees_pays_principaux[donnees_pays_principaux['reason']
!= 'All']

# 2.Créer une figure avec 2 sous-graphiques en disposition 2x2
fig, axes = plt.subplots(2, 2, figsize=(12, 10))
fig.suptitle('Émissions par pays par raison', fontsize=16)

# 3.Définir les valeurs d'éclatement pour chaque pays
explode_values = (0.1, 0.1, 0.1, 0.1)  # À ajuster selon vos préférences

# 4.Itérer à travers les pays pour créer les camemberts éclatés
for (country, data), ax, explode_val in
zip(data2_filtered.groupby('country'), axes.flatten(), explode_values):
    data.groupby('reason')['emissions'].sum().plot.pie(ax=ax,
autopct='%1.1f%%', startangle=90,
explode=(explode_val,)*len(data['reason'].unique()))
    ax.set_title(country)
```

```
#plt.savefig('pie')
plt.show()
```

Pour ce code, nous commençons d'abord par récupérer dans ***data2_filtered,*** la data frame *donnes_pays_principaux* que nous filtrons de manière à exclure la variable *reason* = 'All'. Cela nous permet ainsi de focaliser nos camembert sur les quatre pays les plus émetteurs que nous avons vu plus haut et de garder que le type '*Energy'*.

Ensuite, nous utilisons *fig, axes* de Matplotlib pour créer une disposition de sous-graphiques (subplots) en format de grille 2x2. La fonction plt.subplots() crée une figure (fig) et une grille de sous-graphiques (axes). Ici, la grille est de dimension 2x2, ce qui signifie qu'il y aura 2 lignes et 2 colonnes de sous-graphiques. Puis, nous utilisons la fonction ***fig.suptitle()*** pour définir le titre principal de la figure et la variable ***explode_values*** dans le code est utilisée pour définir les valeurs d'éclatement (explode values) dans un diagramme en secteurs (pie chart).
Enfin, la boucle ***for*** itère à travers chaque pays, crée un camembert éclaté pour les émissions par raison de ce pays, et affiche le résultat sur une grille de sous-graphiques. Chaque camembert est étiqueté avec le nom du pays correspondant. Cette approche permet de visualiser les émissions par raison pour chaque pays de manière comparée. En effet, *data2_filtered*('country'): groupe les données par pays, créant ainsi des groupes distincts pour chaque pays; *axens_flatten*():"aplatit" la matrice de sous-graphiques (axes) en une liste, ce qui permet d'itérer à travers les sous-graphiques de manière linéaire. La fonction *plot.pie()* crée un diagramme en secteurs (pie chart) pour les émissions par raison, en utilisant les paramètres spécifiés, tels que l'axe (ax), le pourcentage automatique (autopct), l'angle de départ (starangle), et les valeurs d'éclatement.

Ainsi avec ***plt.show(),*** nous avons une représentation claire et non biaisée des camembert des 4 pays les plus émetteurs par rapport à la variable *reason*. Cela nous a permis de voir que la cause de pollution énergétique la plus courante est 'vented' (energy liberé sous forme de gaz), figure ci-après:

Émissions par pays par raison