EdTech à l'échelle mondiale: opportunités de croissance

Présentation

EdTech Academy est l'un des principaux fournisseurs de contenu Web pour l'apprentissage en ligne. Fondée il y a de nombreuses années, elle est maintenant prête à étendre ses activités au-delà de son pays d'origine. Elle recherche des marchés prometteurs pour ses services. Pour résoudre cette question commerciale, les données statistiques de la Banque mondiale, qui comprennent plus de 3 600 indicateurs internationaux décrivant divers aspects de pays dans différentes régions du monde ont été analysées. Sur la base de cette recherche, des marchés prometteurs pour l'expansion des activités de l'entreprise ont été identifiés. La stratégie de choix d'un pays pour le développement des affaires est également présentée.

Sommaire

1.Traitement et Nettoyage

- Des données collectées pour leur exploitation
- Inspection des données

2. Analyse Exploratoire

- Des données traitées précédemment et statistiques
- Les informations pertinentes pour résoudre le problème
- Création des scores

3. Conclusion

- Observations, commentaires et pistes de réflexions

EdStatsData

Le fichier EdStatsData.csv est une base de données de statistiques sur l'éducation, recueillies et publiées par la Banque mondiale. Cette base de données contient des informations sur plus de 200 pays dans le monde, couvrant la période des années 1970 à 2020.

Pour chaque pays et pour différentes périodes de temps, la base de données comprend les données suivantes:

- Des indicateurs de qualité de l'éducation: la moyenne des scores des tests PISA et TIMSS, le taux d'alphabétisation, le taux d'abandon scolaire, l'indice d'égalité des sexes, etc.
- Des indicateurs d'accessibilité à l'éducation: le nombre d'élèves dans l'enseignement primaire, secondaire et supérieur, le nombre d'enseignants, le nombre d'écoles et d'autres établissements d'enseignement, le montant des dépenses publiques consacrées à l'éducation, etc.
- Des indicateurs d'efficacité économique de l'éducation: le montant des dépenses d'éducation par élève, le nombre de diplômés de l'enseignement supérieur, le revenu moyen des diplômés, etc.

Au total, la base de données EdStatsData.csv contient plus de 60 000 enregistrements. Ces données peuvent être utilisées pour des recherches dans le domaine de l'éducation, le développement de politiques éducatives et pour comparer le niveau d'éducation dans différents pays du monde.

Traitement et Nettoyage

Importation des Librairies

- import pandas as pd
- import numpy as np
- import matplotlib.pyplot as plt
- %matplotlib inline
- import seaborn as sns
- import scipy.stats as st
- import pycountry_convert as pc
- import plotly.express as px

Lecture et découverte des données

data = pd.read_csv('./Dataset_EdStats_csv/EdStatsData.csv', sep=',')

- df=data.copy()

Combien de lignes et colonnes sont contenues dans `df`

- dim=data.shape
- print(f"Rows:{dim[0]} Columns:{dim[1]}»)

Rows:886930, Columns:70

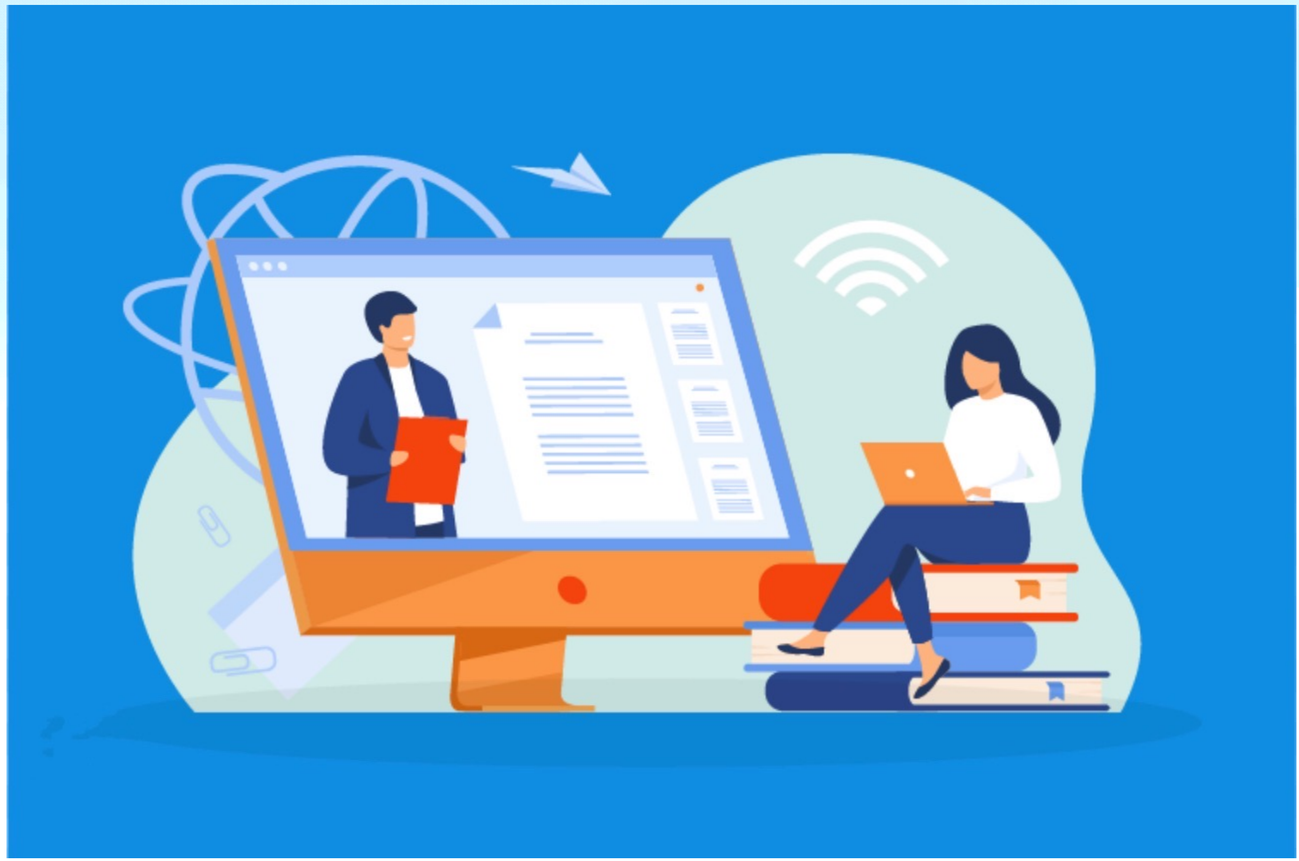
Inspection des données

Examiner les types de données dans chaque colonne pour déterminer comment les données sont stockées et comment elles peuvent être manipulées, la présence des valeurs manquantes

- df.info()

La base de données contient des données pour 242 pays et régions du monde pour la période de 1970 à 2100 concernant les valeurs de 3 665 indicateurs différents décrivant l'accès à l'éducation, l'obtention de diplômes et des informations relatives aux professeurs, aux dépenses liées à l'éducation.

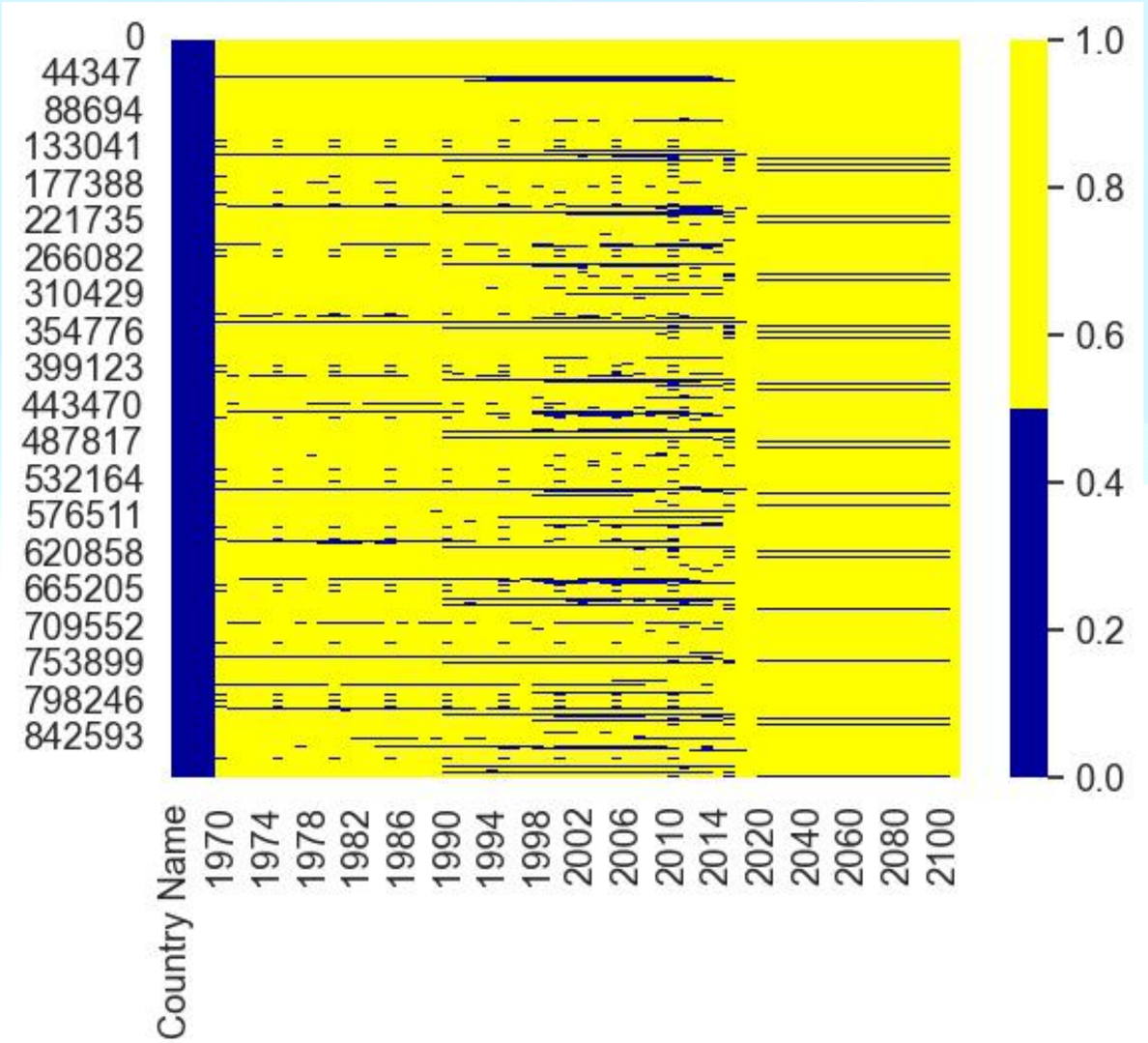
La quantité de mémoire utilisée par la base de données: 473.7+ MB,
dtypes: float64(66), object(4)
Les erreurs de type n'ont pas été détectées.



Affichage des valeurs manquantes

1) print(df.isnull().sum().sort_values()/df.shape[0]).

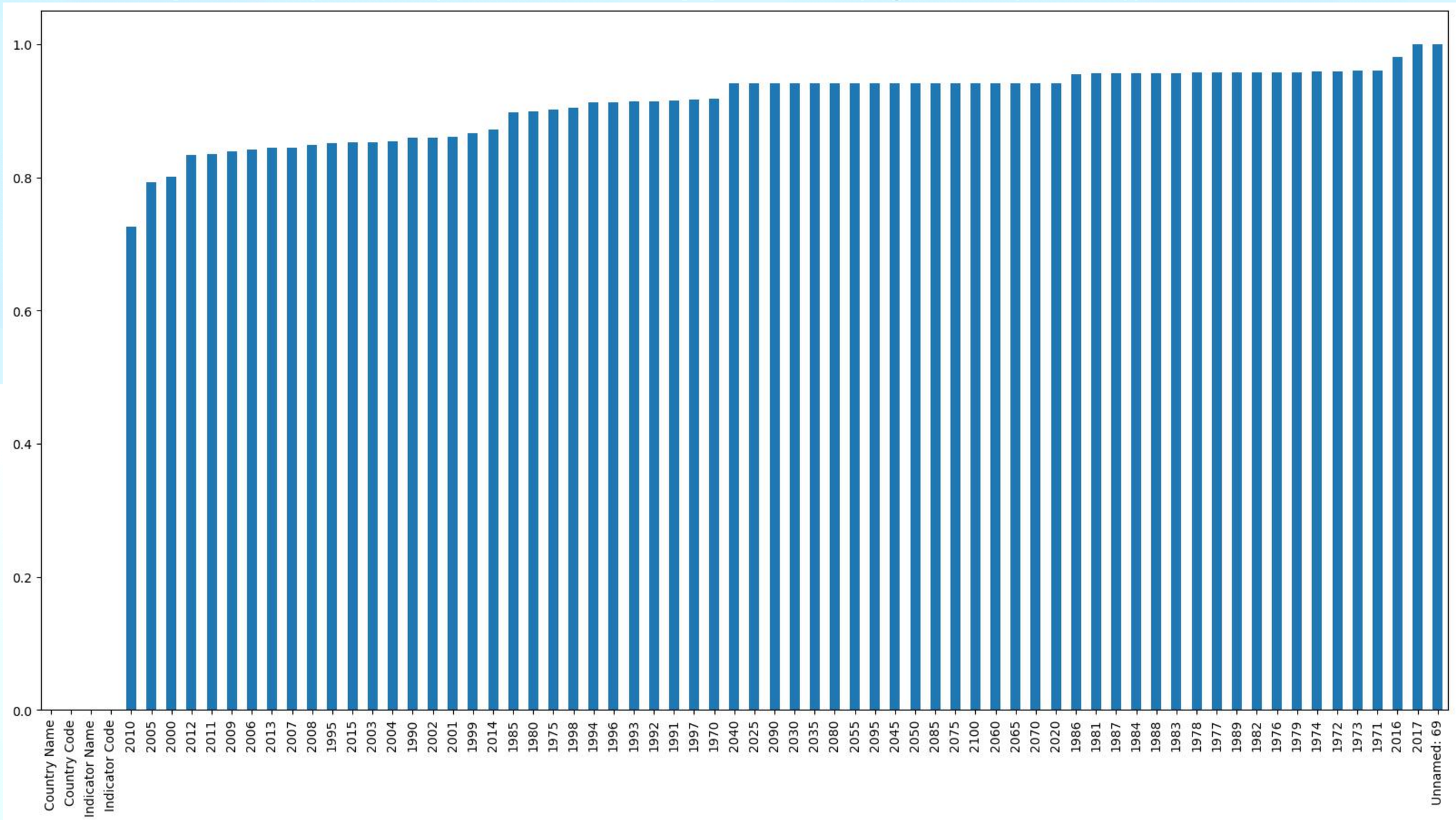
```
Country Name        0.000000
Country Code        0.000000
Indicator Name      0.000000
Indicator Code      0.000000
2010                0.726650
                       ...
1973                0.959924
1971                0.959933
2016                0.981442
2017                0.999839
Unnamed: 69         1.000000
Length: 70, dtype: float64
```



2) Visualiser les valeurs manquantes dans les 70 premières colonnes de la base de données "data" sous forme de carte de chaleur, ce qui peut aider à identifier les zones de la base de données qui nécessitent une attention particulière lors du traitement et de l'analyse des données..

```
cols = data.columns[:70]
colours = ['#000099', '#ffff00']
sns.heatmap(data[cols].isnull(), cmap=sns.color_palette(colours));
```

3) data.isna().mean(axis=0).sort_values().plot.bar(figsize=(20,10))



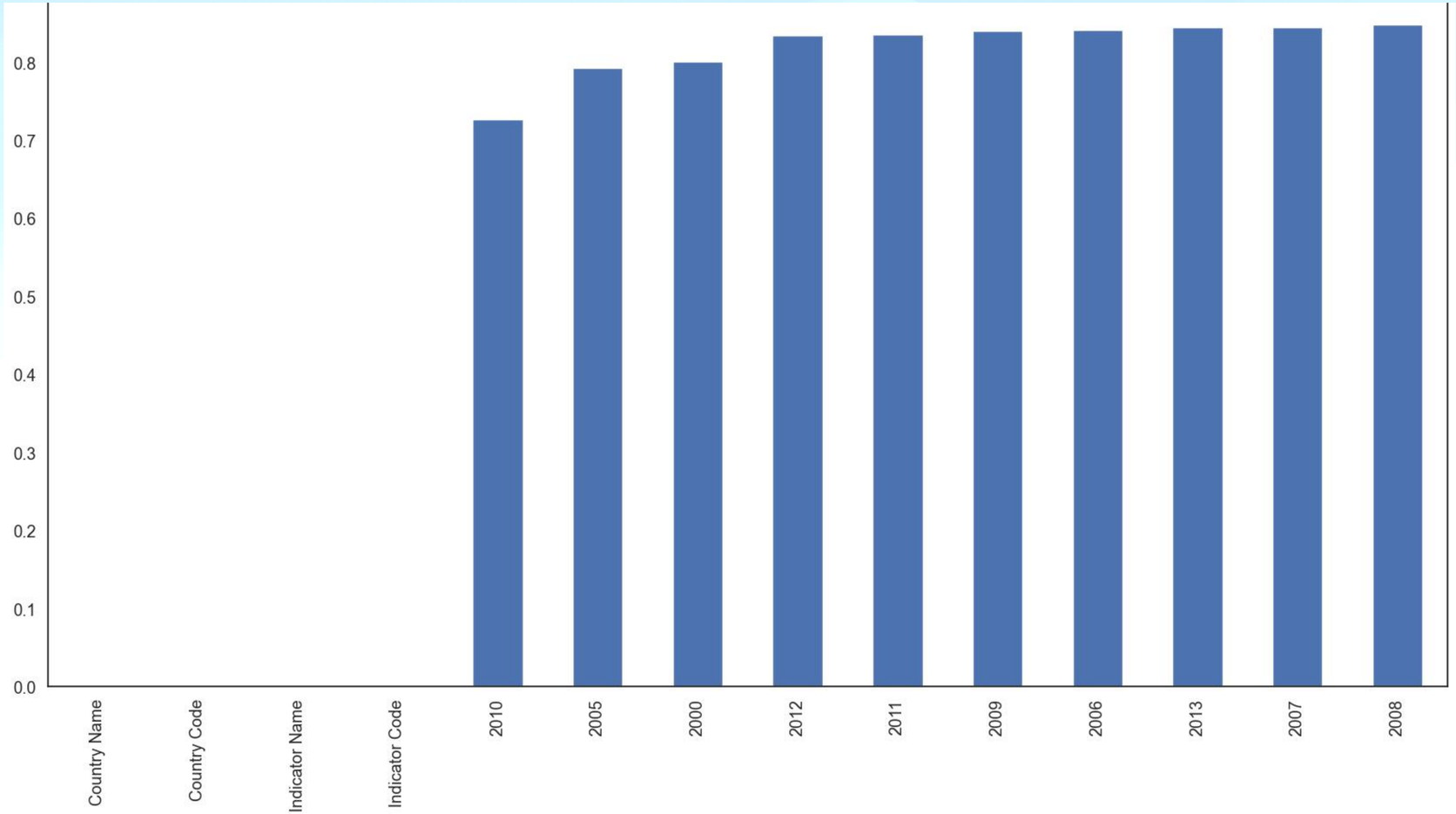
Beaucoup de NaN
Ne rien faire avec plus de 85% de valeurs manquantes

Colonnes avec des données manquantes pas plus de 85%

```
1)for col in df.columns:
    pct_missing = np.mean(df[col].isnull())
    if (pct_missing <= 0.85): print('{} - {}%'.format(col, round(pct_missing*100)))
```

```
2) testData = df.isna().mean(axis=0)
filteredByMed = testData[testData <= 0.85]
filteredByMed.sort_values().plot.bar(figsize=(20,10));
```

Country Name - 0%
Country Code - 0%
Indicator Name - 0%
Indicator Code - 0%
2000 - 80%
2005 - 79%
2006 - 84%
2007 - 85%
2008 - 85%
2009 - 84%
2010 - 73%
2011 - 84%
2012 - 83%
2013 - 84%



Traitement des valeurs manquantes

1) Éliminer les colonnes inutiles

```
df=df[df.columns[df.isna().sum()/df.shape[0]<= 0.85]]
```

`2) On compte le nombre de valeurs manquantes pour la ligne et on stocke dans une nouvelle colonne

```
df['NB_NaN']=df.isna().sum(axis=1)
df=df.sort_values(,NB_NaN‘)

df=df[df[,NB_NaN‘]<=9]
df=df[['Country Name’,’Indicator Name','2000','2005','2006','2007','2008','2009','2010','2011','2012','2013']]
```

3) Les valeurs manquantes dans les colonnes des années 2000 à 2013 sont remplacées par des valeurs moyennes pour les valeurs manquantes restantes.
Cela peut aider à améliorer la qualité des données et à éviter les erreurs lors de l'analyse des données.

```
years = df.loc[:, '2000':'2013']
years =years.apply(lambda row: row.fillna(row.mean()), axis=1)
df.loc[:, '2000':'2013'] = years
```

Vérifions s'il exist des doublons

Créer la variable id, qui sera un identifiant unique et vérifions s'il existe des doublons pour la variable id

```
df['id']=df['Country Code']+'+'+df['Indicator Code']
df.duplicated(,id').sum()
```

Il n'y a pas de doublons

FIN Du Nettoyage des données

Analyse Exploratoire

Choisissons des indicateurs dont nous avons besoin pour poursuivre nos recherches

Nous sommes intéressés par:

1)**L'accès des résidents à Internet (pour l'apprentissage à distance)**
Pour évaluer l'accès de la population de chaque pays à l'apprentissage en ligne, nous pouvons utiliser l'indicateur "Internet users (per 100 people)". Cet indicateur mesure la proportion de la population de chaque pays qui peut accéder à l'apprentissage en ligne.
*'Internet users (per 100 people)''

2)**Le nombre d'étudiants potentiels par pays**
La proportion de la population ayant fait des études supérieures peut également être marquée car elle permet d'évaluer le niveau de la population jeune et la demande potentielle.
*'Graduates from tertiary education, both sexes (number)'

3)**Croissance démographique**
Pour évaluer le potentiel des pays à vendre du contenu pour les étudiants à distance, l'indicateur "Population, total" peut également être utile. Il permet d'estimer dans quels pays la population croît le plus intensément, ce qui peut indiquer une demande potentiellement croissante de services éducatifs.
*'Population, total'

4)**Solvabilité de la population.**
Le PIB par habitant est un indicateur économique important qui peut indiquer le niveau de bien-être de la population et sa capacité de payer.
*'GDP per capita, PPP (current international $)'

Création des scores

Pour créer des coefficients de notation pour chacun des indicateurs sélectionnés, il est nécessaire de les normaliser. La normalisation nous permet d'amener toutes les valeurs des indicateurs dans la même plage de valeurs (de -1 à 1), ce qui nous permet de comparer objectivement différents indicateurs.

- **indicators** = ['Population, total', 'GDP per capita, PPP (current international $)','Internet users (per 100 people)', 'Graduates from tertiary education, both sexes (number)']
- **df** = df.loc[df['Indicator Name'].isin(indicators), ['Country Name', 'Indicator Name', '2000', '2005', '2006', '2007', '2008','2009', '2010', '2011', ,2012‘,'2013']]

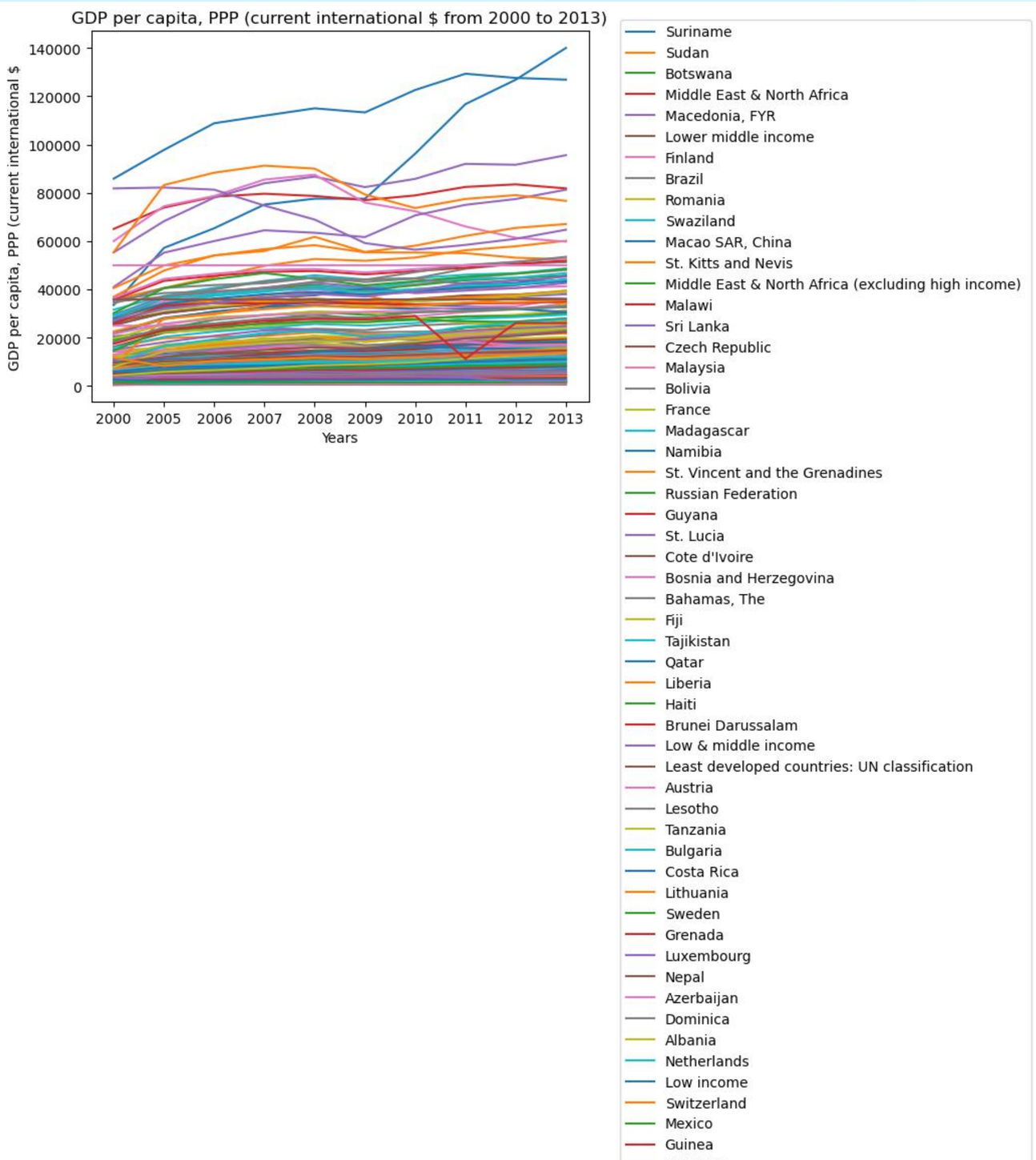
GDP per capita, PPP (current international $)

- gdp_pc = df[df['Indicator Name'] == 'GDP per capita, PPP (current international $)']

- fig, ax = plt.subplots()

```
for country in gdp_pc['Country Name']:
    ax.plot(gdp_pc.columns[2:], gdp_pc.loc[gdp_pc['Country Name'] ==
country].iloc[:,2:].values[0], label=country)
plt.title('GDP per capita, PPP (current international $ from 2000 to 2013)')
ax.set_xlabel('Years')
ax.set_ylabel('GDP per capita, PPP (current international $')
ax.legend(bbox_to_anchor=(1.05, 1.05), loc='upper left');
```

```
gdp_pc['delta_gdp_pc']=(gdp_pc['2013'] - gdp_pc[,2000'])
gdp_pc['score_gdp_pc']=gdp_pc['delta_gdp_pc']/gdp_pc['delta_gdp_pc'].abs().max()
countries=gdp_pc.sort_values('delta_gdp_pc', ascending=False).iloc[:10]
```



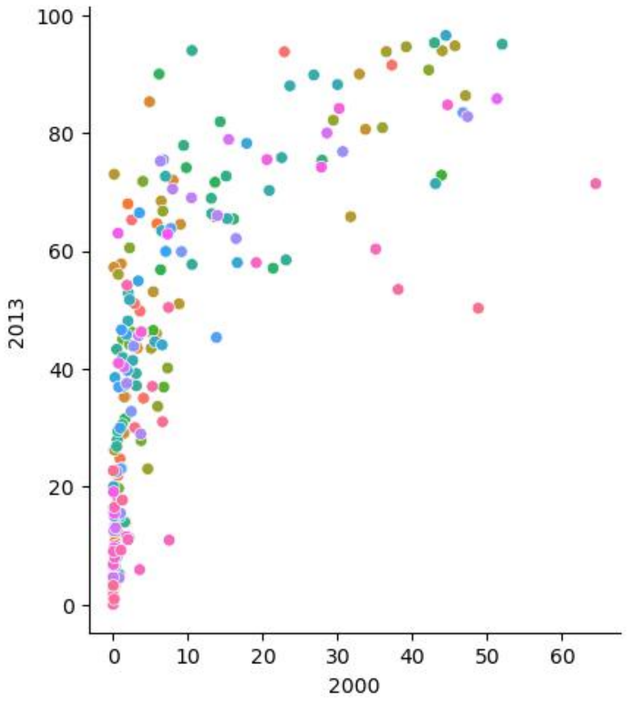
Internet users (per 100 people)

- inter_user=df[df['Indicator Name']=='Internet users (per 100 people)’]
- inter_user=inter_user.set_index("Country Name»)
- inter_user.drop(['Indicator Name'], axis=1, inplace=True)
- sns.relplot(data=inter_user, x="2000", y="2013", hue="Country Name", kind=«scatter");

- inter_user['delta_inter_user']=inter_user['2013']-inter_user['2000']

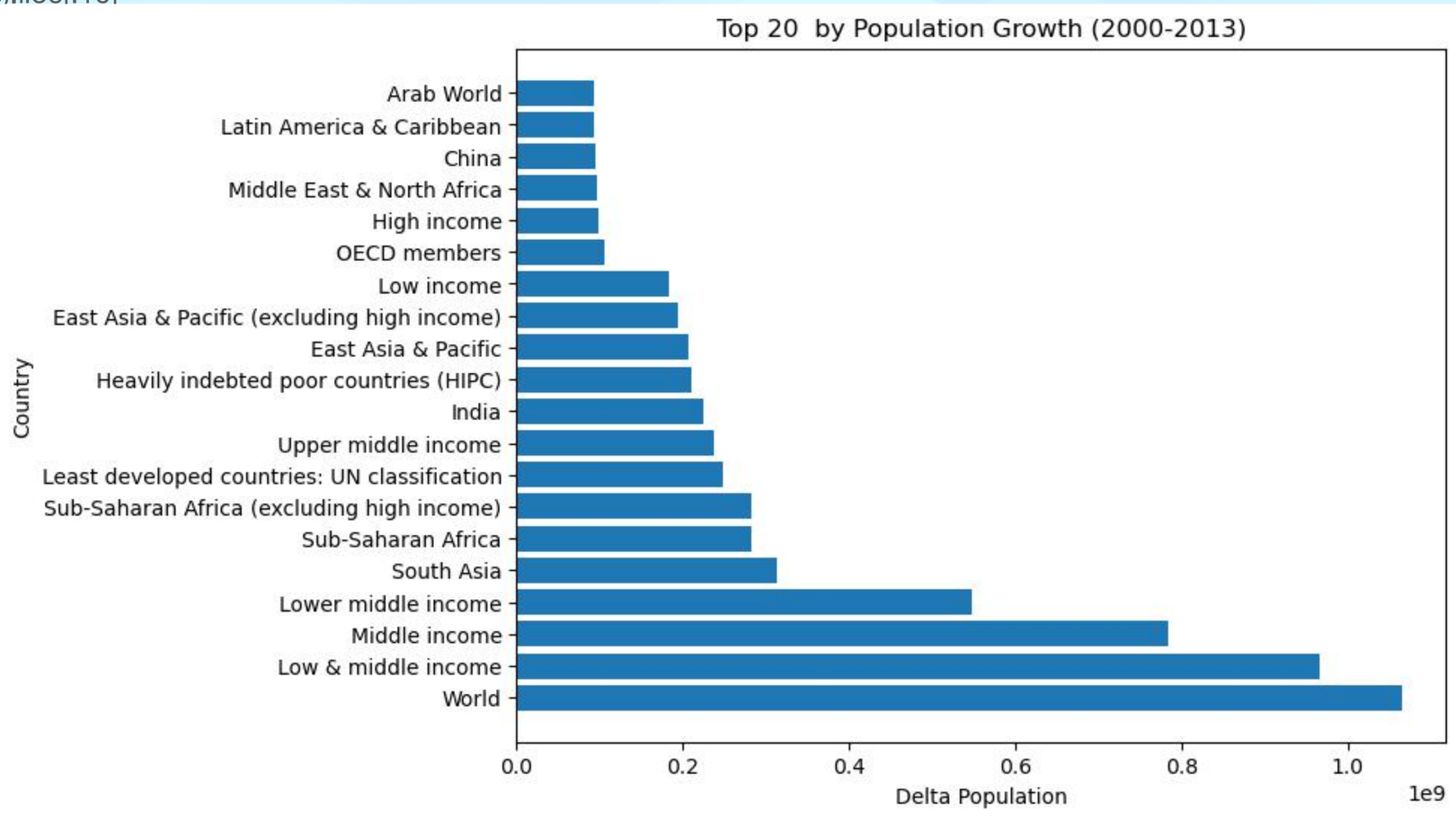
inter_user[,score_inter_user']=inter_user['delta_inter_user']/ inter_user['delta_inter_user'].abs().max()

inter_user.sort_values('score_inter_user', ascending=False).iloc[:20]



Population, total

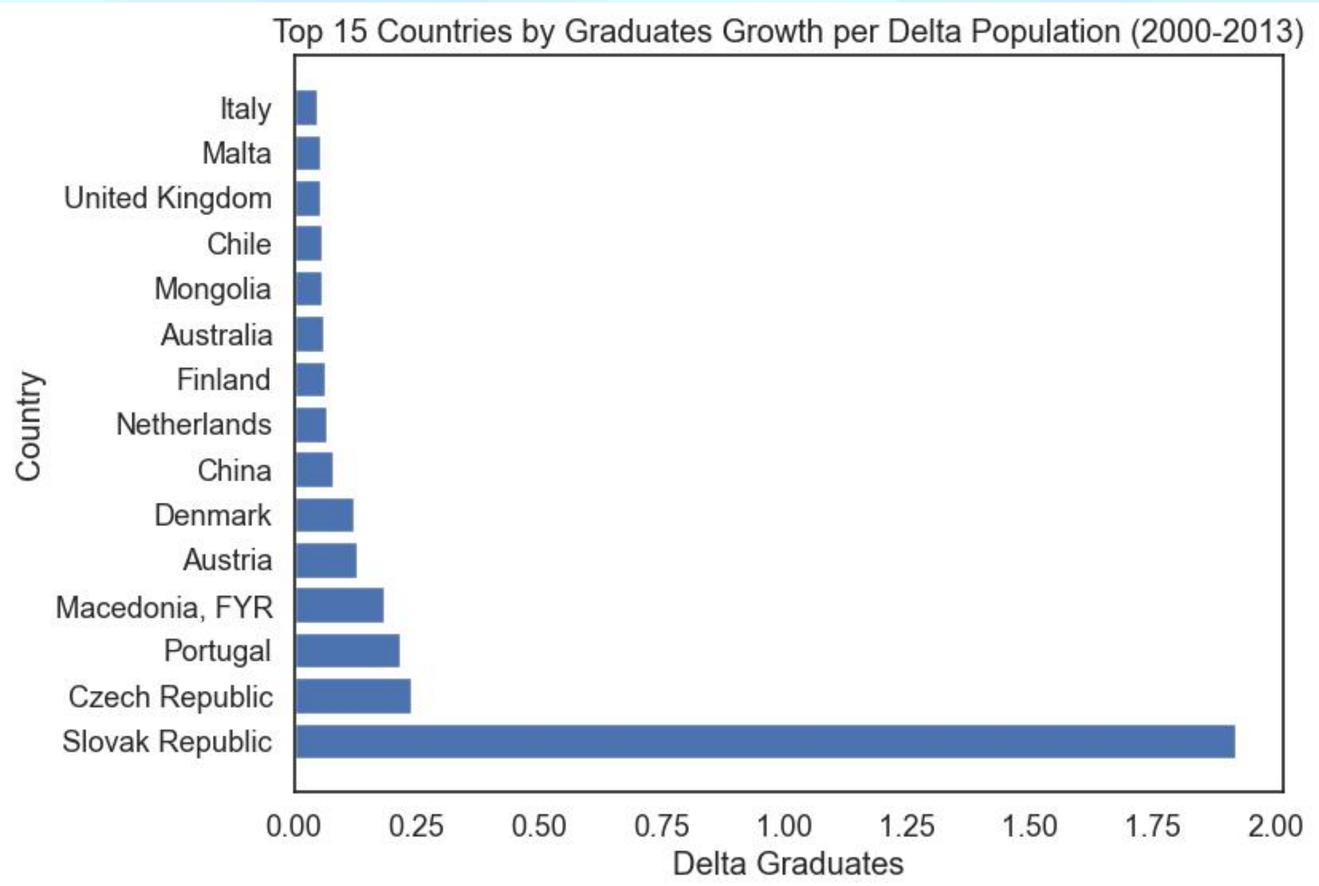
- pop=df[df['Indicator Name']=='Population, total’]
- pop=pop.set_index("Country Name")
- pop.drop(['Indicator Name'], axis=1, inplace=True)
- pop['delta_population']=pop['2013']-pop['2000']
- **pop['score_population']=pop['delta_population']/ pop['delta_population'].abs().max()**
- pop.sort_values('score_population', ascending=False).iloc[:10]



Graduates from tertiary education, both sexes (number)

- Graduat=df[df['Indicator Name']=='Graduates from tertiary education, both sexes (number)']
- Graduat=Graduat.set_index("Country Name")
- Graduat.drop(['Indicator Name'], axis=1, inplace=True)

- if (pop['delta_population'] == 0).any():
  print("delta_population value is 0")
else:
  print("delta_population value is not 0")
- Graduat['delta_Graduates']=(Graduat['2013']-Graduat['2000'])/pop['delta_population']
- **Graduat['score_Graduates']=Graduat['delta_Graduates']/Graduat['delta_Graduates'].abs().max()**
- Graduat.sort_values('score_Graduates', ascending=False).iloc[:15]



Donnerons la liste des 20 pays les mieux classés par rapport à `score_synt`

- **score_synt=(gdp_pc['score_gdp_pc']+pop['score_population']+Graduat['score_Graduates']+inter_user['score_inter_user'])/4**

score_synt.sort_values(ascending=False).iloc[:20]

Country Name

Slovak Republic	0.492941
Macao SAR, China	0.409280
Qatar	0.336600
Luxembourg	0.306977
Bahrain	0.270200
Czech Republic	0.256381
Kuwait	0.255131
Azerbaijan	0.249827
France	0.239781
Latvia	0.237618
United Kingdom	0.226985
Macedonia, FYR	0.226321
Ireland	0.226085
Lithuania	0.224289
Kazakhstan	0.223622
Denmark	0.223601
Hungary	0.219564
Saudi Arabia	0.217340
Barbados	0.213276
Trinidad and Tobago	0.210660

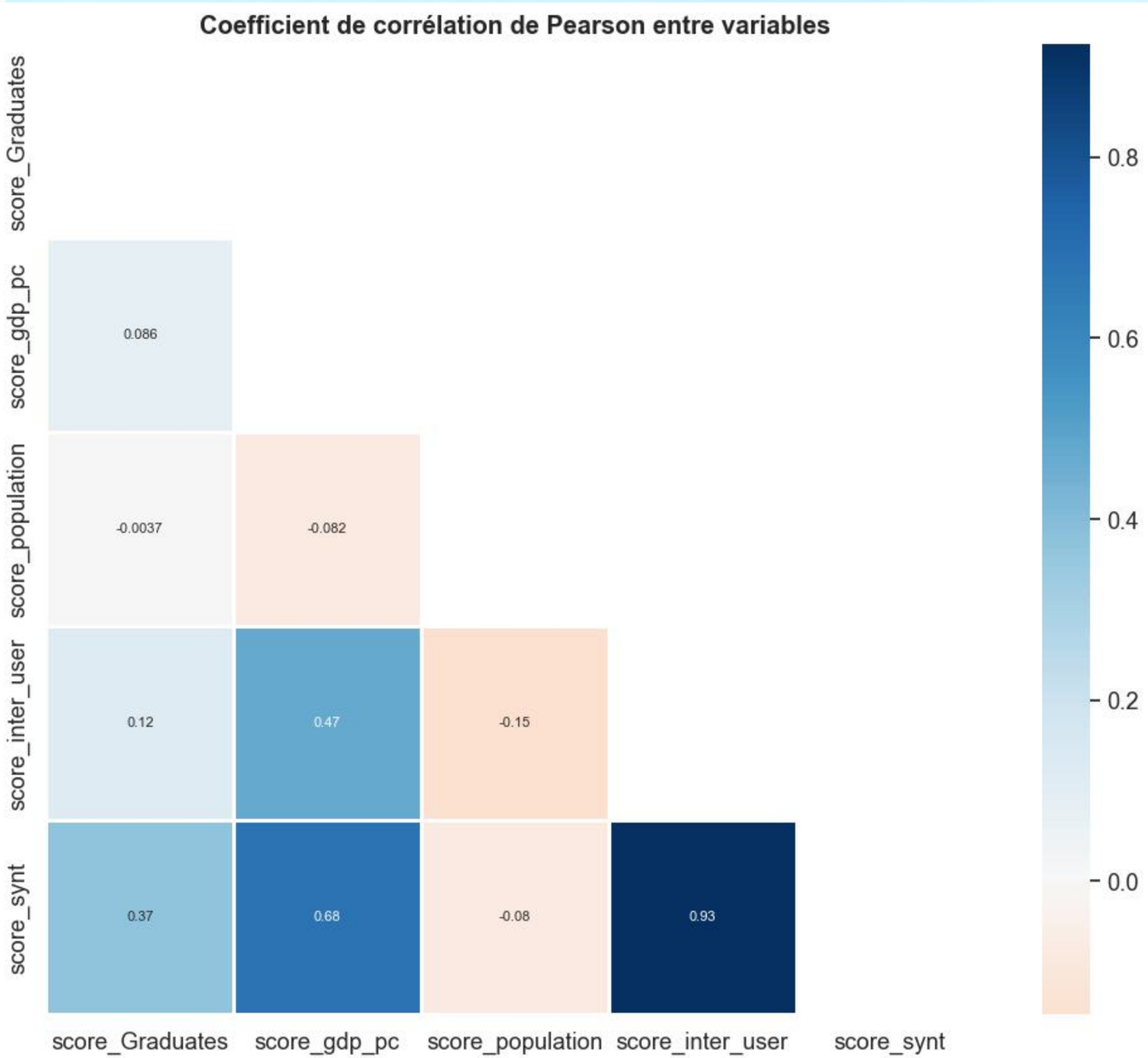
Name: score_synt, dtype: float64

Une visualisation par heatmap de la matrice de corrélation pour identifier les modèles et les relations entre les scores.

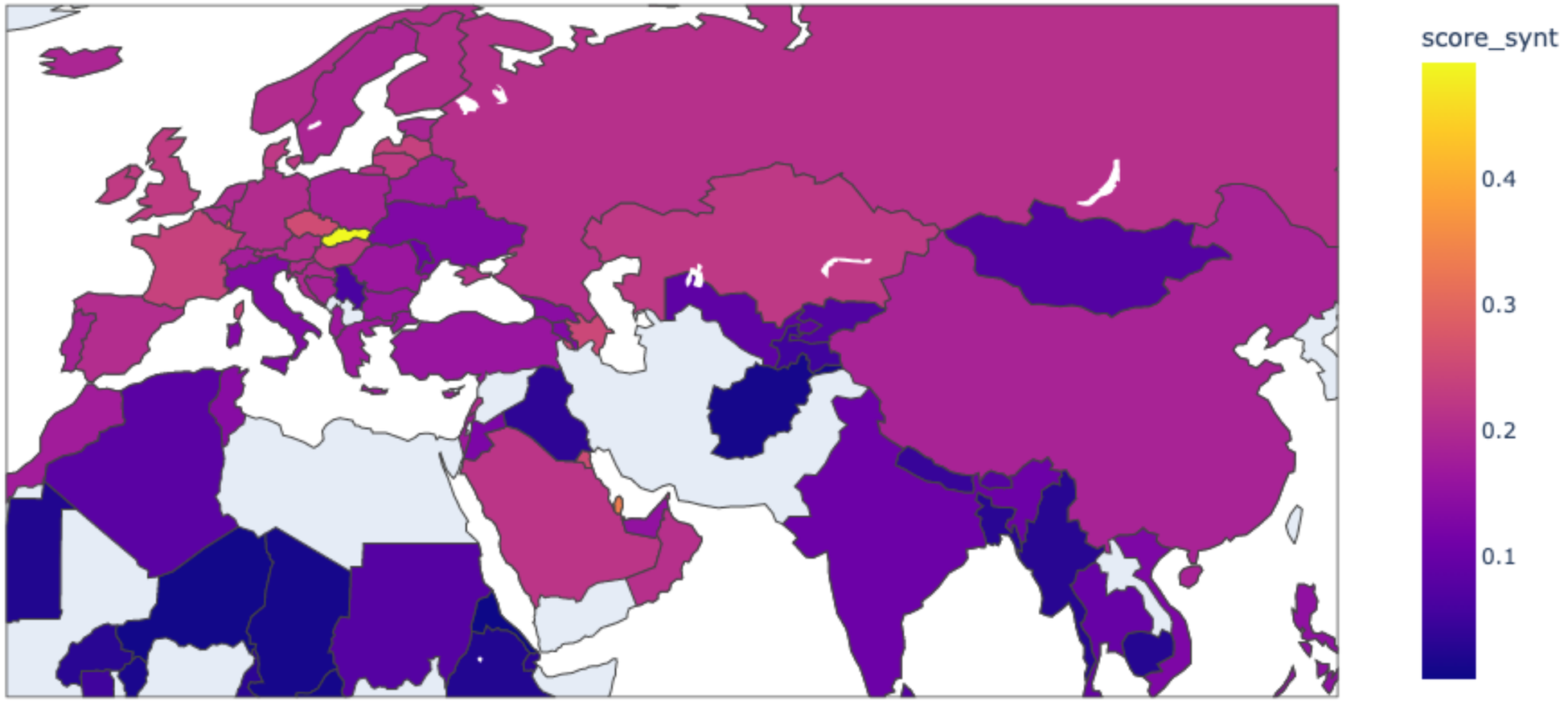
```
• corr_matrix = data_Test.corr()
plt.figure(figsize=(12, 10))
mask = np.triu(np.ones_like(corr_matrix, dtype=bool))
sns.heatmap(corr_matrix, mask=mask, center=0, cmap='RdBu',
linewidths=1, annot=True, annot_kws={'size':8})
plt.title('Coefficient de corrélation de Pearson entre
variables', fontweight='bold')
```

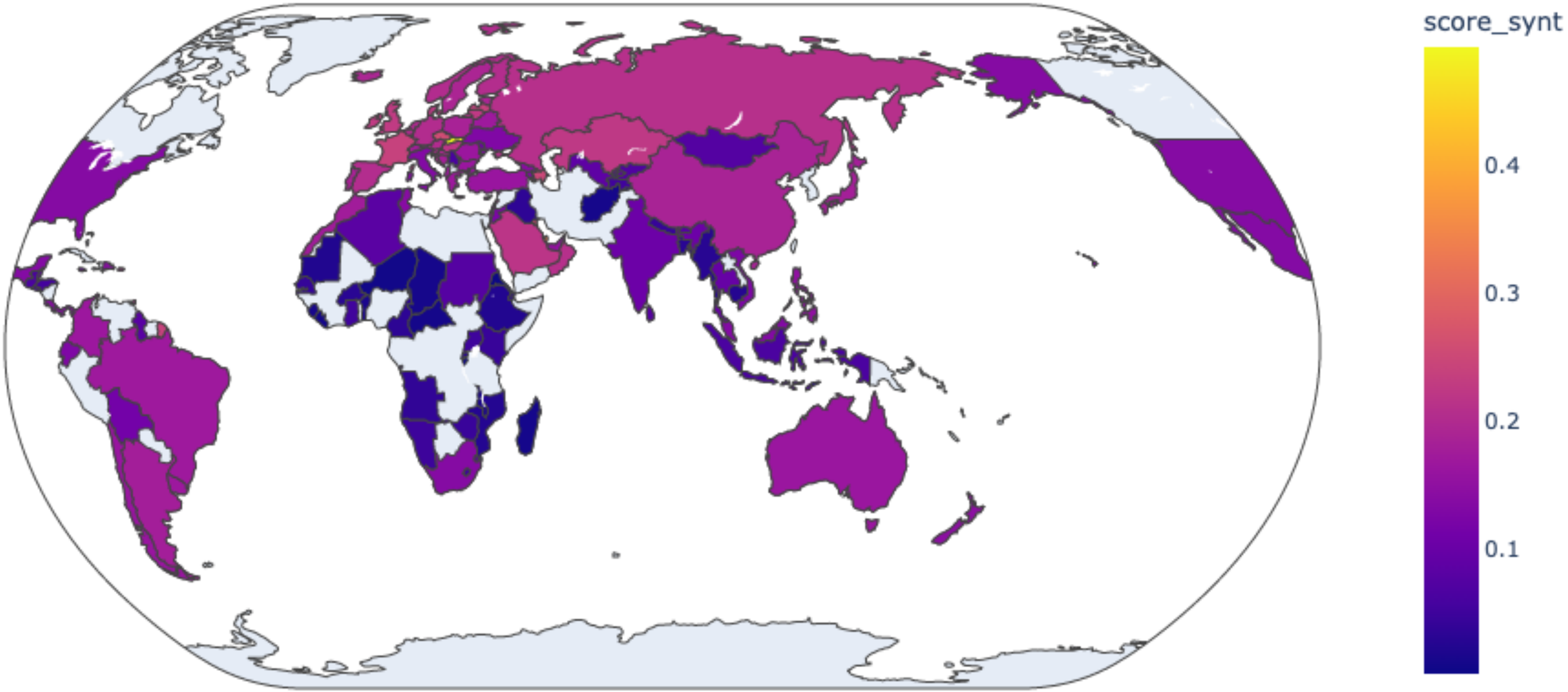
Construisons une carte interactive:

```
• iso3_codes = {}
for country_name in data_Test.index:
    try:
        country_code =
pc.country_name_to_country_alpha3(country_name)
        iso3_codes[country_name] = country_code
    except:
        iso3_codes[country_name] = None
data_Test['iso_alpha'] = [iso3_codes[country_name] for
country_name in data_Test.index]
import plotly.express as px
fig = px.choropleth(data_Test,
                    locations="iso_alpha",
                    color="score_synt",
                    hover_name=data_Test.index,
                    projection="natural earth",
                    )
fig.show();
```



Une carte interactive




score_synt
0.4
0.3
0.2
0.1

MERCI