# ANALYSEZ DES DONNÉES DE SYSTÈMES ÉDUCATIFS

- Projet d'expansion à l'international de l'entreprise EdTech

Zouheir HMIDI

academy

# SOMMAIRE

# 1- OBJECTIFS DE L'ÉTUDE

L'objectif de ce travail est d'obtenir les données, de les nettoyer et de faire une recommandation pour le projet d'expansion à l'international de la start-up « Academy » qui propose des contenus de formation en ligne pour un public de niveau lycée et université.

Besoins Métier :

- Quels sont les pays avec un fort potentiel de clients pour nos services ?
- Pour chacun de ces pays, quelle sera l'évolution de ce potentiel de clients ?
- Dans quels pays l'entreprise doit-elle opérer en priorité ?

# 2-DESCRIPTION DES JEUX DE DONNÉES

Les données sont issues de la Banque mondiale, celles-ci disposent de 5 fichiers .csv :

**EdStatsData**

Évolution des indicateurs par pays ou regroupe les États sur plusieurs années.

**EdStatsCountry**

Liste des pays avec leur région d'appartenance



THE WORLD BANK
IBRD • IDA

**EdStatsFootNote**

Informations concernant la source d'information

**EdStatsSeries**

Description détaillée de chaque indicateur avec son thème d'appartenance

**EdStatsCountry-Series**

Description de l'indicateur par pays

THE WORLD BANK
IBRD • IDA

# 3- EXPLOITATION ET NETTOYAGE DES DONNÉES

## 3-1 Exploitation des jeux de données

### a- Importer les librairies  et télécharger les données :

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import missingno as msno
```

**Chargement des dataset**

Les données de la Banque mondiale sont disponibles à l'adresse suivant

https://datacatalog.worldbank.org/dataset/education-statistics

qui contient 5 jeux de données

```
dataC = pd.read_csv('EdStatsCountry.csv')
dataS = pd.read_csv('EdStatsSeries.csv')
dataCS = pd.read_csv('EdStatsCountry-Series.csv')
dataFN = pd.read_csv('EdStatsFootNote.csv')
data = pd.read_csv('EdStatsData.csv')
```

b- On peut afficher les premières lignes du jeu de données avec **head()** :

```
data.head()
```

| | Country Name | Country Code | Indicator Name | Indicator Code | 1970 | 1971 | 1972 | 1973 | 1974 | 1975 | ... | 2060 | 2065 | 2070 | 2075 | 2080 | 2085 | 209 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | Arab World | ARB | Adjusted net enrolment rate, lower secondary, ... | UIS.NERA.2 | NaN | NaN | NaN | NaN | NaN | NaN | ... | NaN | NaN | NaN | NaN | NaN | NaN | Na |
| 1 | Arab World | ARB | Adjusted net enrolment rate, lower secondary, ... | UIS.NERA.2.F | NaN | NaN | NaN | NaN | NaN | NaN | ... | NaN | NaN | NaN | NaN | NaN | NaN | Na |

c- On peut afficher les dernières lignes du jeu de données avec **tail()** :

```
data.tail()
```

| | Country Name | Country Code | Indicator Name | Indicator Code | 1970 | 1971 | 1972 | 1973 | 1974 | 1975 | ... | 2060 | 2065 | 2070 | 2075 | 2080 | 2085 | 2090 | 2095 | 2100 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 886925 | Zimbabwe | ZWE | Youth illiterate population, 15-24 years, male... | UIS.LP.AG15T24.M | NaN | NaN | NaN | NaN | NaN | NaN | ... | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN |
| 886926 | Zimbabwe | ZWE | Youth literacy rate, population 15-24 | SE.ADT.1524.LT.ZS | NaN | NaN | NaN | NaN | NaN | NaN | ... | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN |

d- **Dimension** : On peut connaître le nombre de lignes et de colonnes avec la fonction **shape**

```
data.shape
```

```
(886930, 70)
```

e- **info()** renvoie une synthèse du dataframe

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 886930 entries, 0 to 886929
Data columns (total 70 columns):
 #   Column           Non-Null Count   Dtype
---  ------           --------------   -----
dtypes: float64(66), object(4)
memory usage: 473.7+ MB
```

**Concernant la « EdStatsData » :**

Nombre de colonnes = 70  dont 66 (Float64), 4 (object)

Nombre de lignes = 886930

f- **Enumération** des lignes et colonnes qui contiennent les valeurs ‘NaN’ .

```
data.isnull().sum()
```

```
Country Name            0
Country Code            0
Indicator Name          0
Indicator Code          0
1970               814642
                    ...
2085               835494
2090               835494
2095               835494
2100               835494
Unnamed: 69        886930
Length: 70, dtype: int64
```

# 3-2 Nettoyer les données

a- Nous allons créer une dataframe qui contient les informations à moins de 90 % de valeurs manquantes.

```
dataD=data.copy()
dataD=data.drop(data.columns[data.isnull().mean()*100>90], axis=1)
dataD
```

| | Country Name | Country Code | Indicator Name | Indicator Code | 1980 | 1985 | 1990 | 1995 | 1999 | 2000 | 2001 | 2002 | 2003 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | Arab World | ARB | Adjusted net enrolment rate, lower secondary, ... | UIS.NERA.2 | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN |
| 1 | Arab World | ARB | Adjusted net enrolment rate, lower secondary, ... | UIS.NERA.2.F | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN |
| 2 | Arab World | ARB | Adjusted net enrolment rate, lower secondary, ... | UIS.NERA.2.GPI | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN |

# a- Suppression des régions

```
regions=['ARB' , 'EAS' , 'EAP' , 'EMU' , 'ECS' , 'ECA' , 'EUU' , 'HPC' , 'HIC' , 'LCN' , 'LAC' , 'LDC' , 'LMY' , 'LIC' , 'LMC' ,
new_dataD=dataD.loc[~dataD['Country Code'].isin(regions)]
new_dataD
```

| | Country Name | Country Code | Indicator Name | Indicator Code | 1980 | 1985 | 1990 | 1995 | 1999 | 2000 | 2001 | 2002 | 2003 | 2004 | 2005 | 2006 | 2007 | 2008 | 200 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **91625** | Afghanistan | AFG | Adjusted net enrolment rate, lower secondary, ... | UIS.NERA.2 | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | 28.059870 | NaN | Na |
| **91626** | Afghanistan | AFG | Adjusted net enrolment rate, lower secondary, ... | UIS.NERA.2.F | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | 15.223100 | NaN | Na |

On obtient :

Nombre de colonnes = 25  au lieu de 70

Nombre de lignes = 795305 au lieu de 886930

# 4- ANALYSES

## a- **indicateurs**

1- les différents groupes d'indicateur à étudier sont :

**IT** : Infrastructure
**SE** : Social Education
**SP** : Social Population
**NY** : National Accounts, produits intérieurs et nationaux

2- les mots-clés à rechercher :

**15** : pour la cible de la population des 15-19 ans
**20** : pour la cible de la population des 20-24 ans
**SEC** : pour les regroupements par lycéens
**TER** : pour les regroupements par étudiants de l'enseignement supérieur
**IT** : pour l'accès aux infrastructures techniques
**TOT** : pour population totale

```
dataS[dataS['Series Code'].str.startswith('SP.') & dataS['Series Code'].str.contains('TER|SEC|15|20|TOT')]
```

| | Series Code | Topic | Indicator Name | Short definition | Long definition | Unit of measure | Periodicity | Base Period | Other notes | Aggregation method | ... | Notes from original source | General comments |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 2450 | SP.POP.1015.FE.UN | Population | Population, ages 10-15, female | Population, ages 10-15, female is the total nu... | Population, ages 10-15, female is the total nu... | NaN | NaN | NaN | NaN | NaN | ... | NaN | NaN |
| 2451 | SP.POP.1015.MA.UN | Population | Population, ages 10-15, male | Population, ages 10-15, male is the total numb... | Population, ages 10-15, male is the total numb... | NaN | NaN | NaN | NaN | NaN | ... | NaN | NaN |
| 2452 | SP.POP.1015.TO.UN | Population | Population, ages 10-15, total | Population, ages 10-15, total is the total pop... | Population, ages 10-15, total is the total pop... | NaN | NaN | NaN | NaN | NaN | ... | NaN | NaN |

**startswith()** est utilisée pour tester si le début de chaque élément de chaîne correspond par exemple à 'SP'  et je l'ai combiné avec **str.contains()** utilisée pour tester si par exemple '15' est contenu dans une chaîne d'une série ou d'un index.

**Bilan** : indicateurs retenus

numérique

**IT.NET.USER.P2**

économique

**NY.GNP.PCAP.PP.CD**

démographique

**SP.POP.1524.TO.UN**

**SP.POP.TOTL**

éducatif

**SE.SEC.ENRR**

**SE.TER.ENRR**

Création d'un nouveau Dataframe qui ne contient que les indicateurs choisis :

```
dataDInd = new_dataD.loc[new_dataD["Indicator Code"].isin(['SP.POP.TOTL','NY.GDP.PCAP.PP.CD', 'IT.NET.USER.P2', 'SP.POP.1524.TO.
dataDInd.head()
```

| | Country Name | Country Code | Indicator Name | Indicator Code | 1980 | 1985 | 1990 | 1995 | 1999 | 2000 | 2001 | 2 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **92872** | Afghanistan | AFG | GDP per capita, PPP (current international $) | NY.GDP.PCAP.PP.CD | NaN | NaN | NaN | NaN | NaN | NaN | NaN | 8.755176e |
| **92960** | Afghanistan | AFG | Gross enrolment ratio, secondary, both sexes (%) | SE.SEC.ENRR | 16.942141 | 13.46313 | 1.122283e+01 | 2.256602e+01 | NaN | NaN | 1.304874e+01 | |
| **92964** | Afghanistan | AFG | Gross enrolment ratio, tertiary, both sexes (%) | SE.TER.ENRR | NaN | NaN | 2.273170e+00 | NaN | NaN | NaN | NaN | |
| **93000** | Afghanistan | AFG | Internet users (per 100 people) | IT.NET.USER.P2 | NaN | NaN | 0.000000e+00 | NaN | NaN | NaN | 4.722568e-03 | 4.561395e |

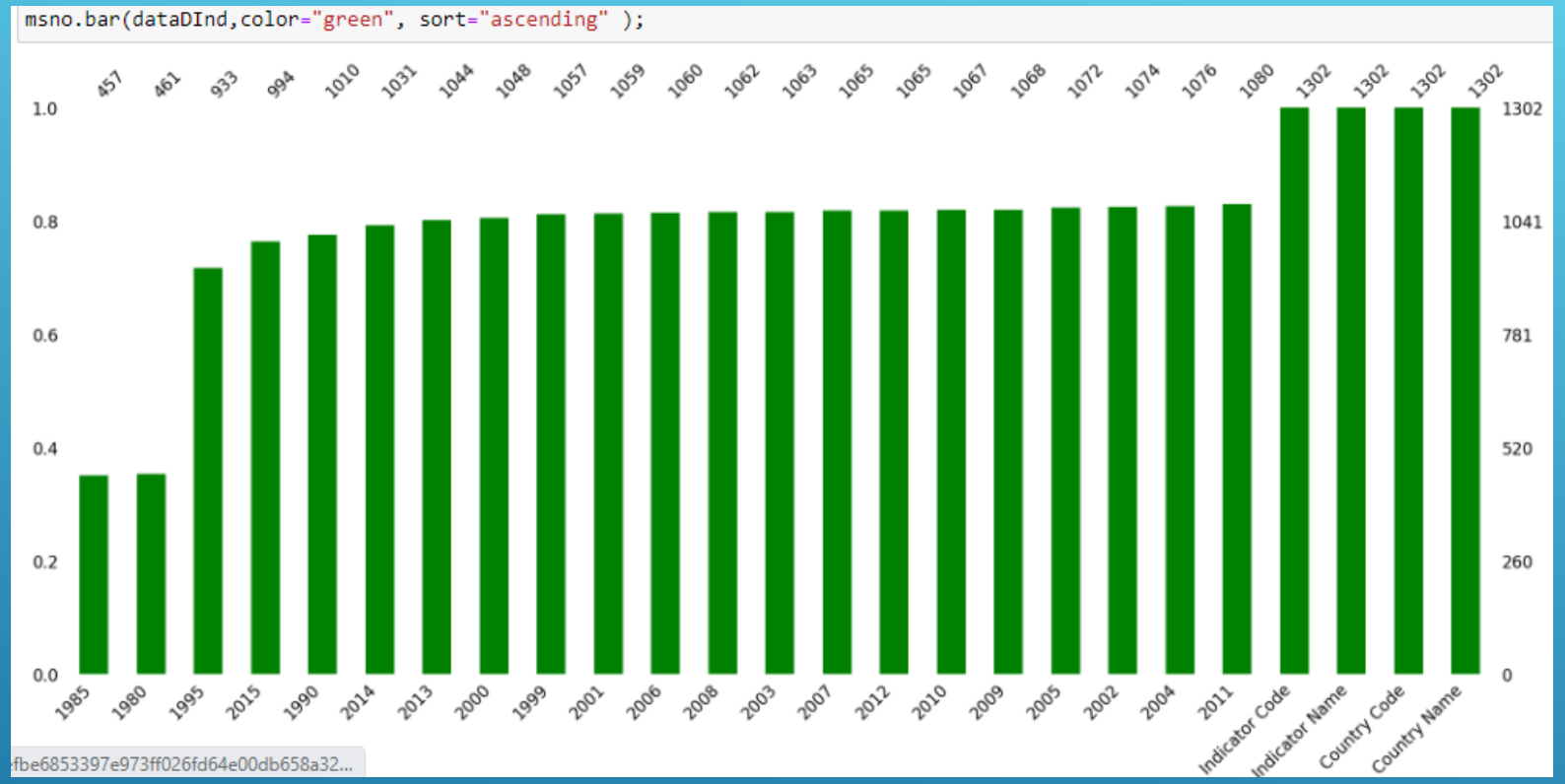
Le nombre de lignes et de colonnes avec la fonction **shape**

```
dataDInd.shape
```

```
(1302, 25)
```

# Visualisation des valeurs manquantes :

```
msno.bar(dataDInd,color="green", sort="ascending" );
```



# Création d'une nouvelle colonne avec valeur : la dernière année où la valeur est non nulle (NaN)

```
dataDInd['Nouveau']=dataDInd.loc[:, '2010': '2015'].fillna(method='ffill',axis=1)['2015']
dataDInd.head(5)
```

```
C:\Users\User_01\AppData\Local\Temp\ipykernel_10736\2881690812.py:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  dataDInd['Nouveau']=dataDInd.loc[:, '2010': '2015'].fillna(method='ffill',axis=1)['2015']
```

| )5 | 2006 | 2007 | 2008 | 2009 | 2010 | 2011 | 2012 | 2013 | 2014 | 2015 | Nouveau |
|---|---|---|---|---|---|---|---|---|---|---|---|
| )3 | 1.065620e+03 | 1.210479e+03 | 1.247066e+03 | 1.482099e+03 | 1.581601e+03 | 1.660740e+03 | 1.873154e+03 | 1.877412e+03 | 1.875447e+03 | 1.864974e+03 | 1.864974e+03 |
| )1 | 2.993046e+01 | 3.008316e+01 | 4.022338e+01 | 4.673276e+01 | 5.324683e+01 | 5.461618e+01 | 5.667734e+01 | 5.668866e+01 | 5.565616e+01 | 5.564441e+01 | 5.564441e+01 |

Création d'un tableau pivot : ce tableau permet d'avoir les indicateurs en colonnes et de faciliter la comparaison.
Ce tableau contient 207 lignes et 6 colonnes.

```
dataDIndP=dataDInd.pivot_table(index= 'Country Name', columns='Indicator Code', values='Nouveau')
dataDIndP
```

| Indicator Code / Country Name | IT.NET.USER.P2 | NY.GDP.PCAP.PP.CD | SE.SEC.ENRR | SE.TER.ENRR | SP.POP.1524.TO.UN | SP.POP.TOTL |
|---|---|---|---|---|---|---|
| Afghanistan | 8.260000 | 1864.973641 | 55.644409 | 8.662800 | 7252785.0 | 3.373649e+07 |
| Albania | 63.252933 | 11449.094589 | 95.765488 | 58.109951 | 556269.0 | 2.880703e+06 |
| Algeria | 38.200000 | 14643.343064 | 99.860191 | 36.922279 | 6467818.0 | 3.987153e+07 |
| American Samoa | NaN | NaN | NaN | NaN | NaN | 5.553700e+04 |
| Andorra | 96.910000 | NaN | NaN | NaN | NaN | 7.801400e+04 |
| Angola | 12.400000 | 6648.124016 | 28.898720 | 9.308020 | 4259352.0 | 2.785930e+07 |
| Antigua and Barbuda | 70.000000 | 21503.952551 | 102.705460 | 23.486240 | NaN | 9.992300e+04 |
| Argentina | 68.043064 | 20379.779854 | 106.777901 | 82.917389 | 6886530.0 | 4.341776e+07 |
| Armenia | 59.102378 | 8744.501489 | 88.502357 | 44.309502 | 446958.0 | 2.916950e+06 |

## b- **Score**

Le calcule du score nous permettra de comparer les indicateurs par rapport aux pays

1- Calcul du score :

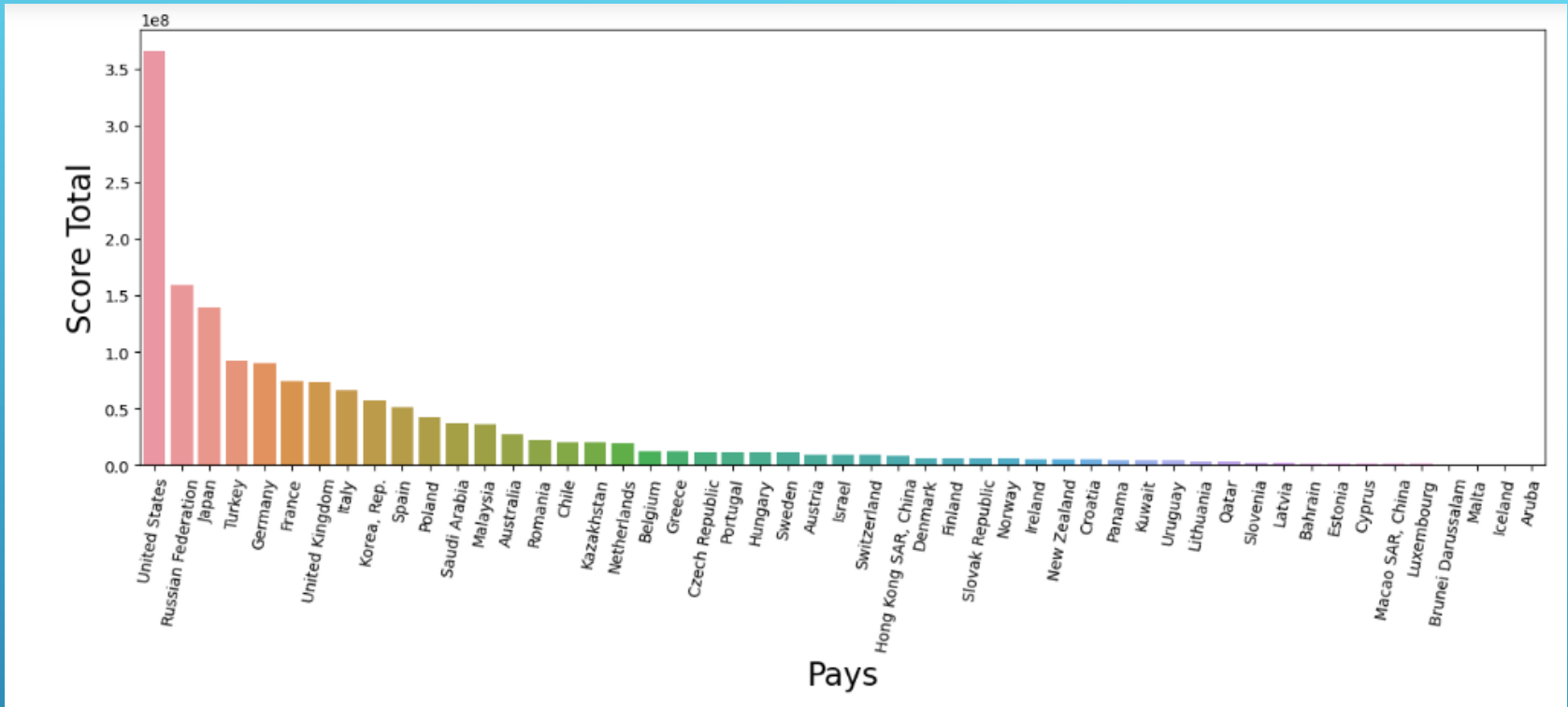
J'utilise l'équation essentielle du score z pour un exemple : $z = (x - \mu)/\sigma$
$\mu$ est la moyenne , $\sigma$ est l'écart-type des pays

```python
# Score: z = (x - μ)/σ => μ=la moyenne, σ= l'écart-type des pays
def scoreN(x):
    return (dataDIndPF[x]-dataDIndPF[x].mean())/dataDIndPF[x].std()

dataDIndPF['Score IT.NET.USER.P2']=scoreN('IT.NET.USER.P2')
dataDIndPF['Score NY.GDP.PCAP.PP.CD']=scoreN('NY.GDP.PCAP.PP.CD')
dataDIndPF['Score SE.SEC.ENRR']=scoreN('SE.SEC.ENRR')
dataDIndPF['Score SE.TER.ENRR']=scoreN('SE.TER.ENRR')
dataDIndPF['Score SP.POP.1524.TO.UN']=scoreN('SP.POP.1524.TO.UN')
dataDIndPF['Score SP.POP.TOTL']=scoreN('SP.POP.TOTL')
dataDIndPF['Score TOT']= dataDIndPF.sum(axis=1)
dataDIndPF
```

| R.ENRR | SP.POP.1524.TO.UN | SP.POP.TOTL | Score IT.NET.USER.P2 | Score NY.GDP.PCAP.PP.CD | Score SE.SEC.ENRR | Score SE.TER.ENRR | Score SP.POP.1524.TO.UN | Score SP.POP.TOTL | Score TOT |
|---|---|---|---|---|---|---|---|---|---|
| 662800 | 7252785.0 | 3.373649e+07 | -1.395456 | -0.827170 | -0.941558 | -1.074883 | -0.024998 | -0.089553 | 4.099121e+07 |
| 109951 | 556269.0 | 2.880703e+06 | 0.512176 | -0.401961 | 0.394427 | 0.633086 | -0.273285 | -0.279057 | 3.448639e+06 |
| 922279 | 6467818.0 | 3.987153e+07 | -0.356877 | -0.260246 | 0.530776 | -0.098764 | -0.054102 | -0.051874 | 4.635416e+07 |
| 308020 | 4259352.0 | 2.785930e+07 | -1.251845 | -0.614961 | -1.832159 | -1.052597 | -0.135986 | -0.125648 | 3.212535e+07 |
| 017289 | 6986530.0 | 4.241776e+07 | 0.679239 | 0.005742 | 0.761139 | 1.489068 | 0.038578 | 0.039095 | 5.023494e+07 |

```python
plt.figure(figsize=(16,5))
sns.barplot(x=dataDIndPF_liste2.index, y=dataDIndPF_liste2['Score TOT'])
plt.rcParams['font.size'] = '5'
plt.xticks(rotation=80)
plt.xlabel('Pays',fontsize=20 )
plt.ylabel('Score Total', fontsize=20);
```



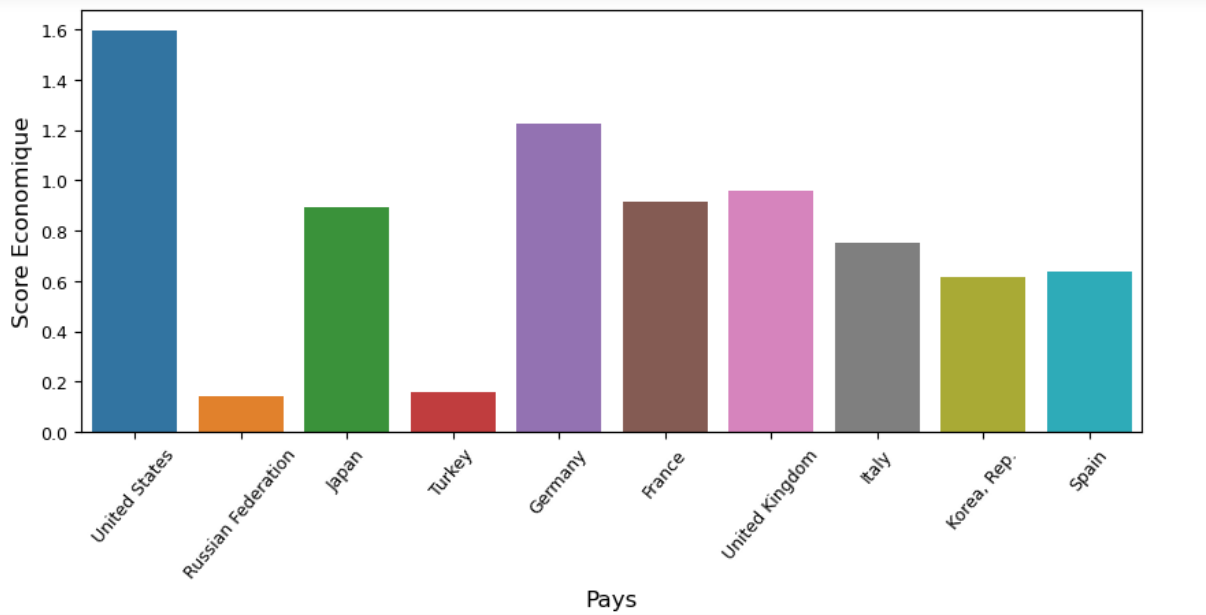

La start-up de la EdTech propose des contenus de formation en ligne pour un public de niveau lycée et université, je supprime les pays qui ont comme score négatif les indicateurs :

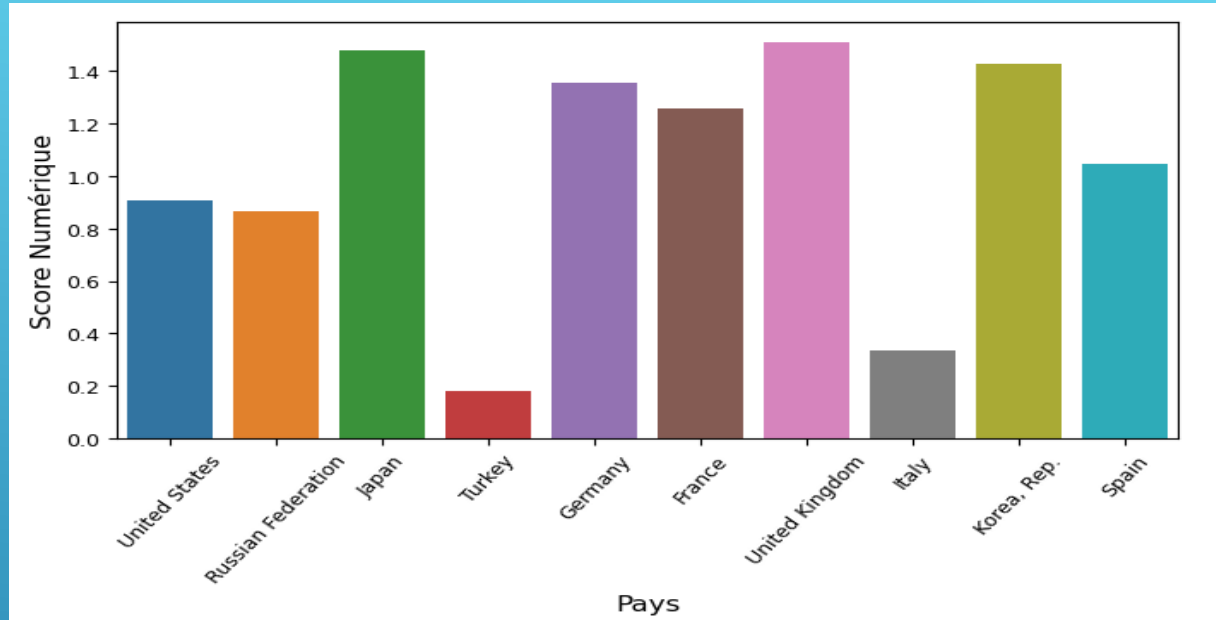
**‘IT.NET.USER.P2’**

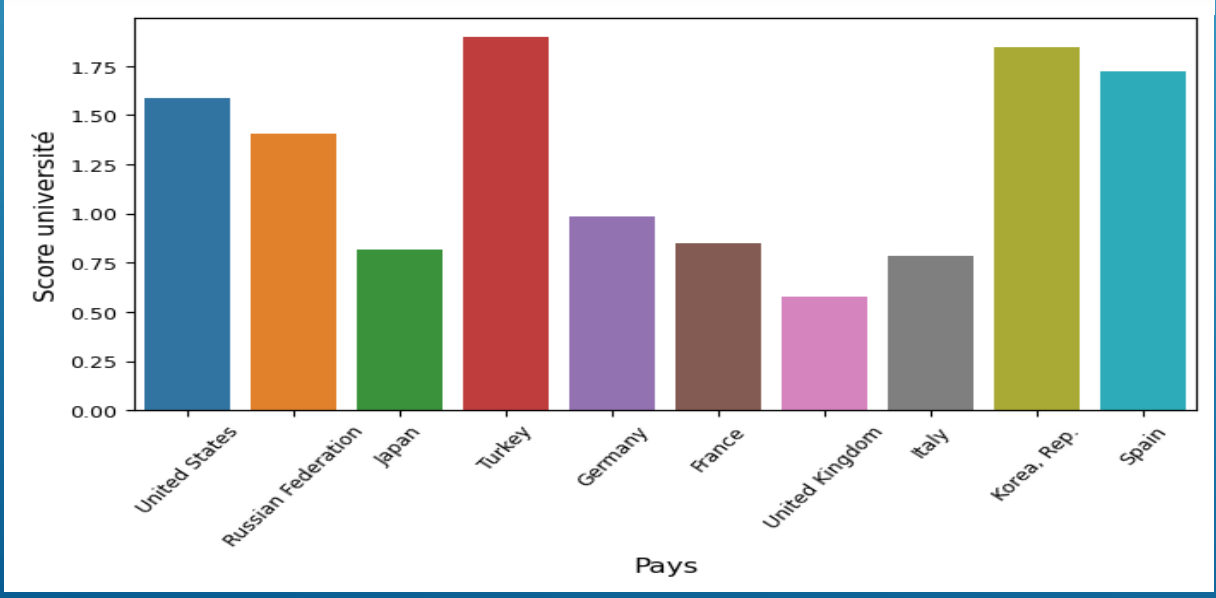
et

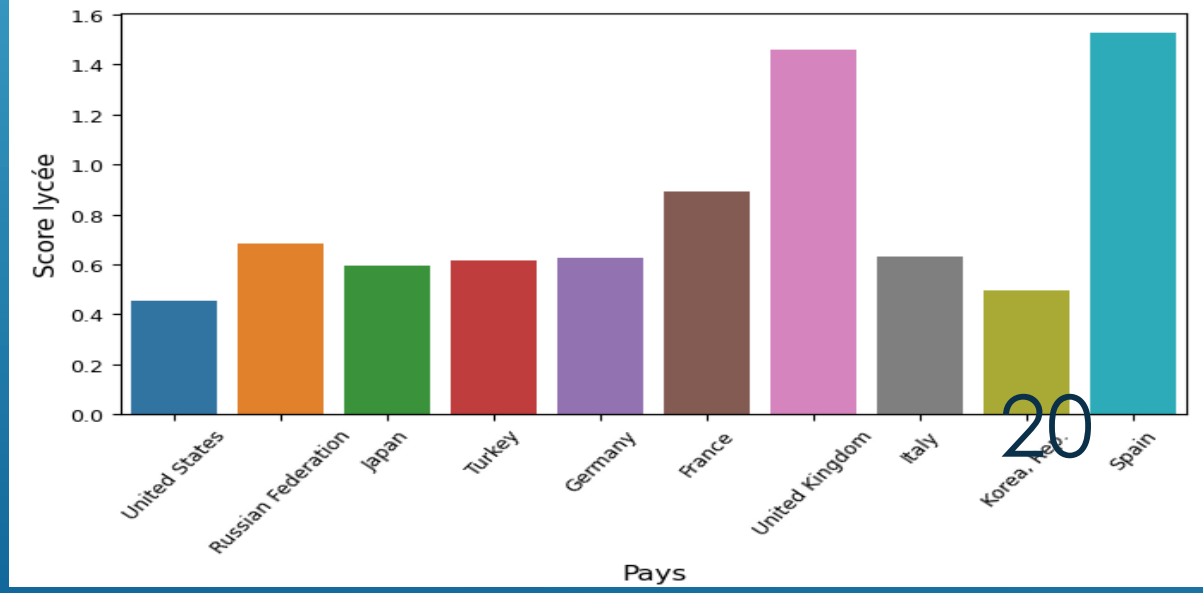
**‘NY.GDP.PCAP.PP.CD’**

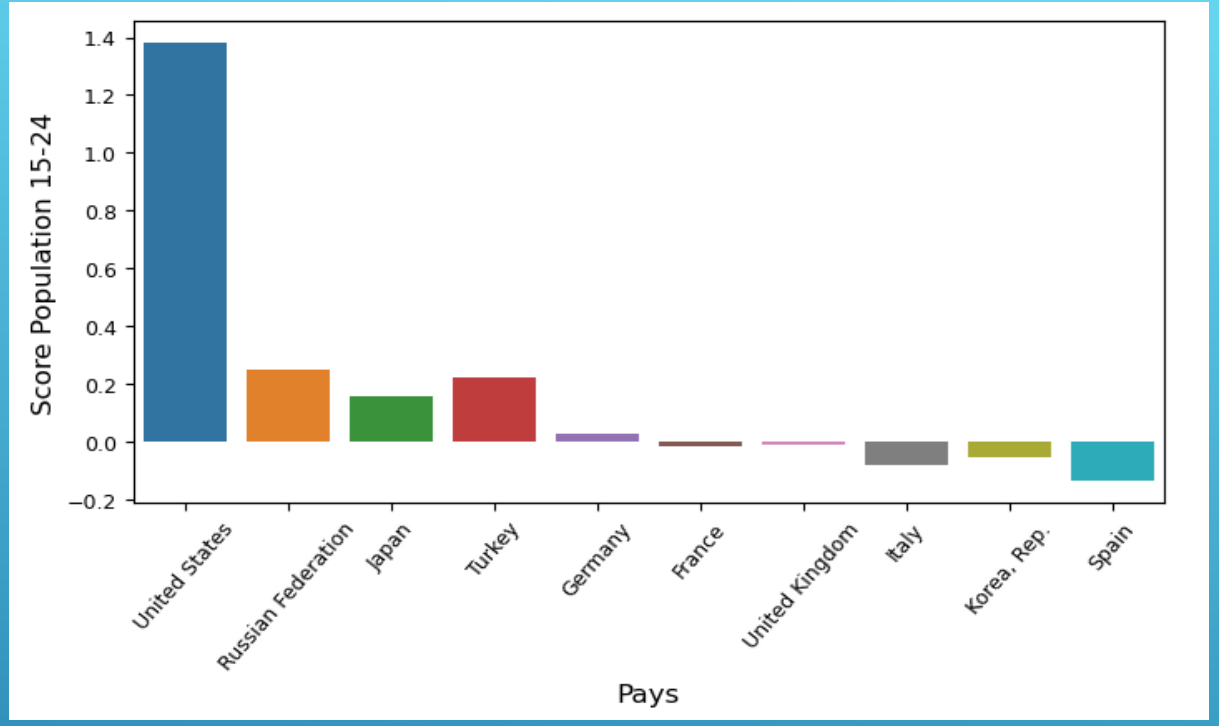
# 2- Les 10 pays attractifs par score d'indicateur :

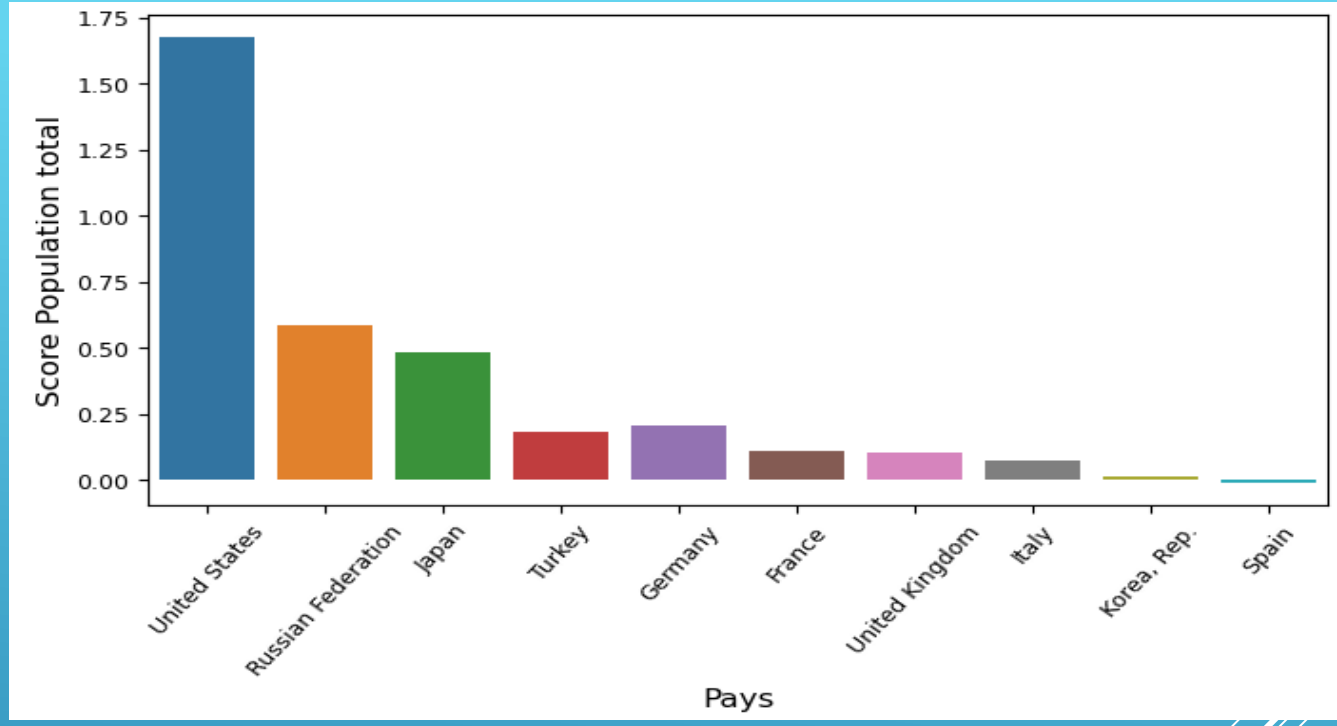


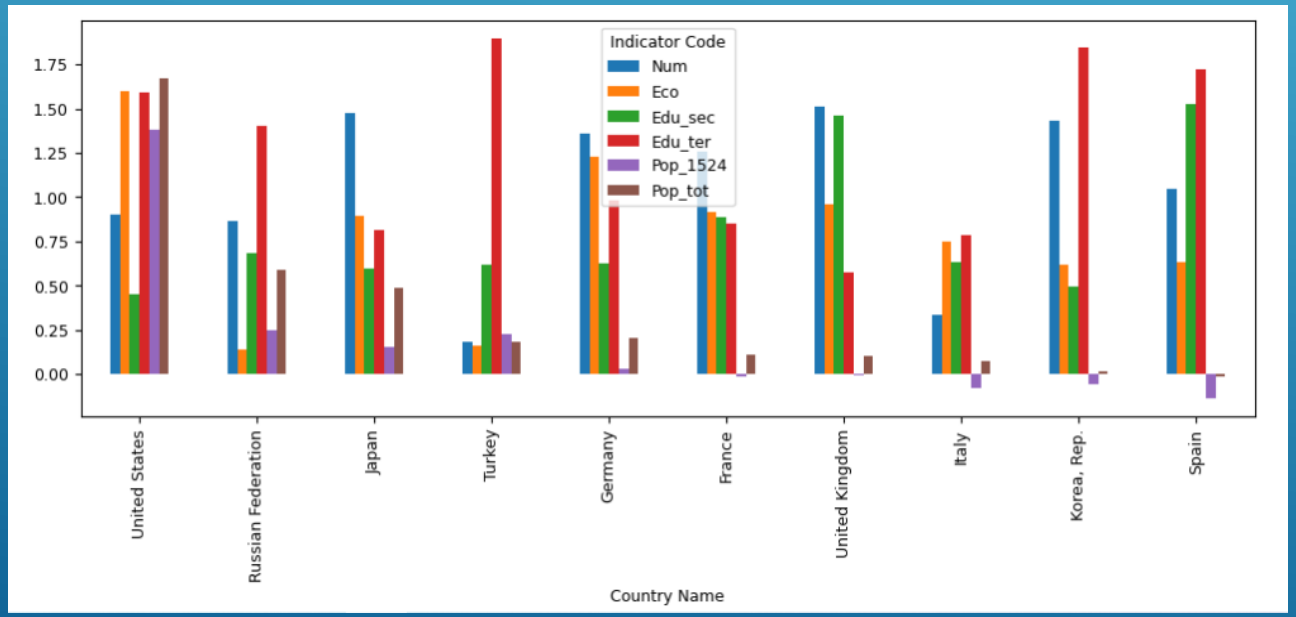




Score Population 15-24
1.4
1.2
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
United States
Russian Federation
Japan
Turkey
Germany
France
United Kingdom
Italy
Korea, Rep.
Spain
Pays

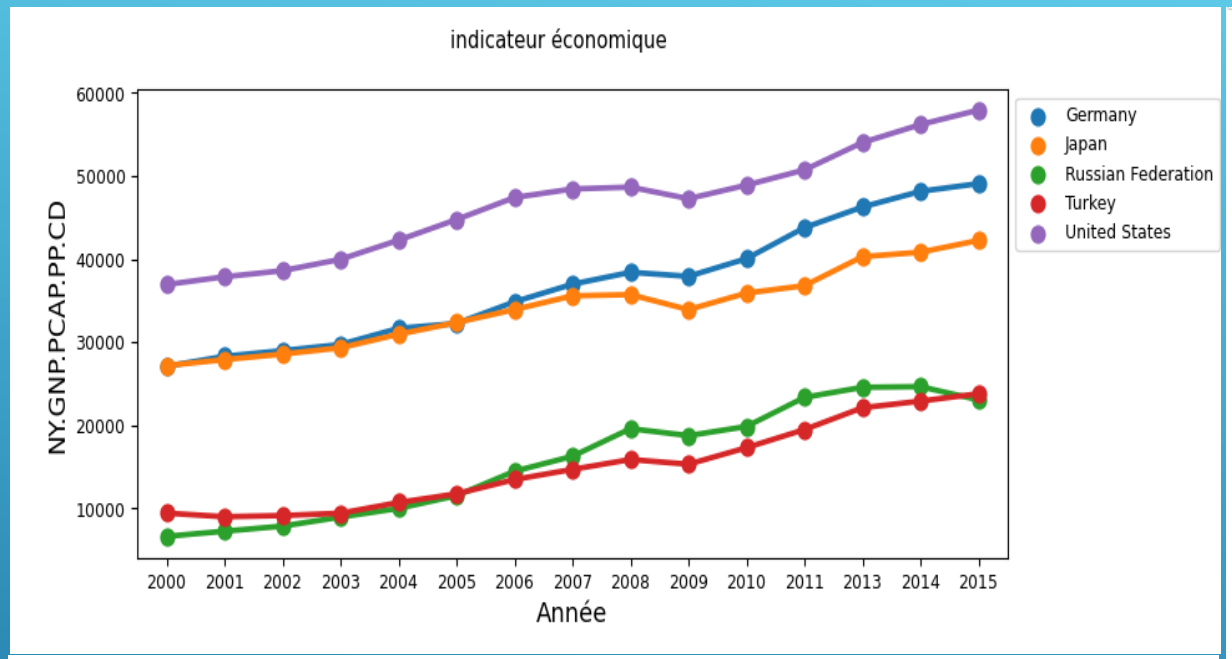

Score Population total
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
United States
Russian Federation
Japan
Turkey
Germany
France
United Kingdom
Italy
Korea, Rep.
Spain
Pays

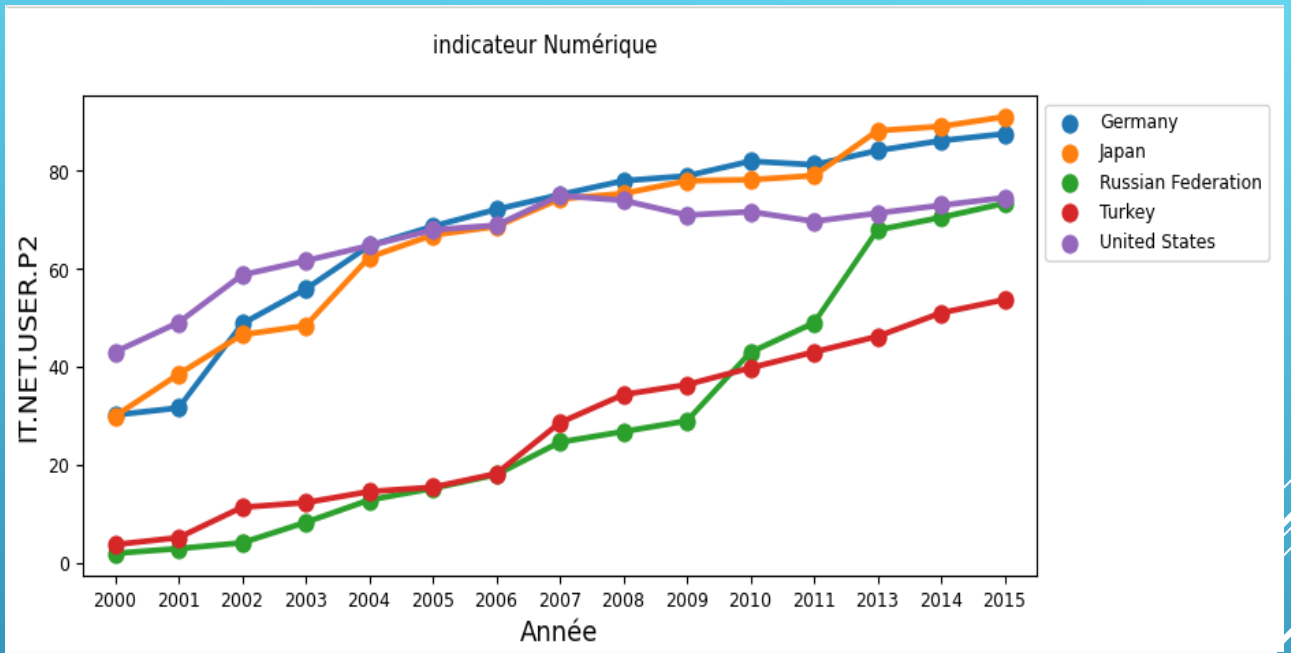

Indicator Code
Num
Eco
Edu_sec
Edu_ter
Pop_1524
Pop_tot
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
United States
Russian Federation
Japan
Turkey
Germany
France
United Kingdom
Italy
Korea, Rep.
Spain
Country Name

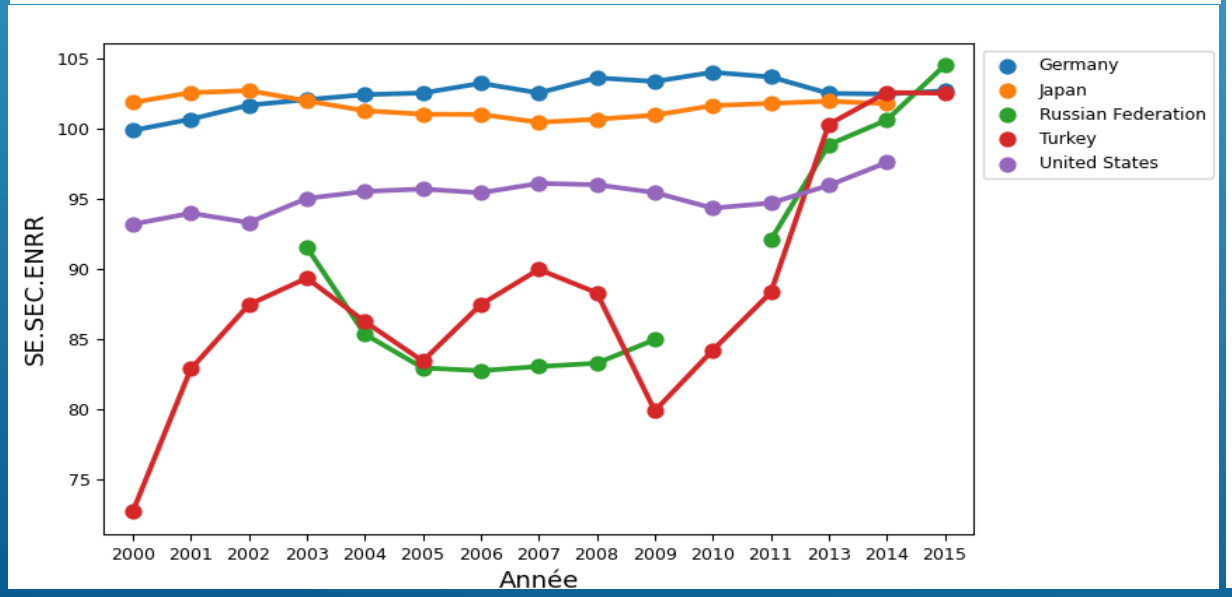
L'analyse des 6 indicateurs (Économique, numérique, Éducation secondaire, Éducation universitaire, population 14-25 ans et la population totale) nous permet de considérer que les pays avec un fort potentiel de clients pour ces services sont :

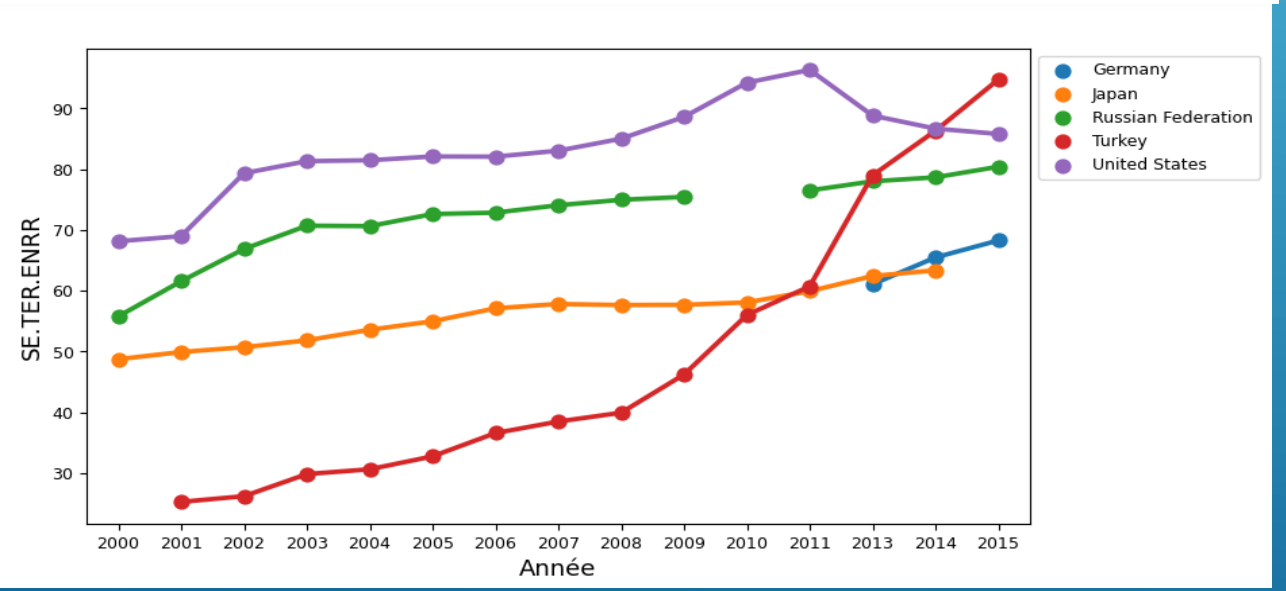
United States, Russia,Japan,Turkey et Germany.

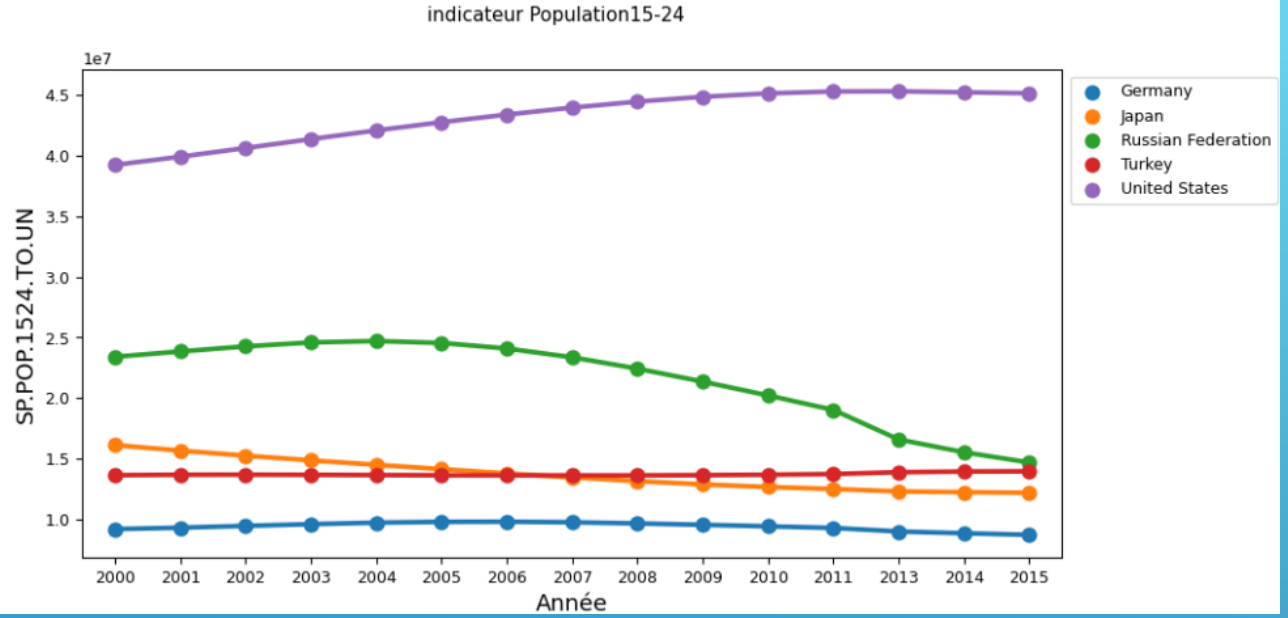
# 5- ÉVOLUTION DU POTENTIEL DES PAYS

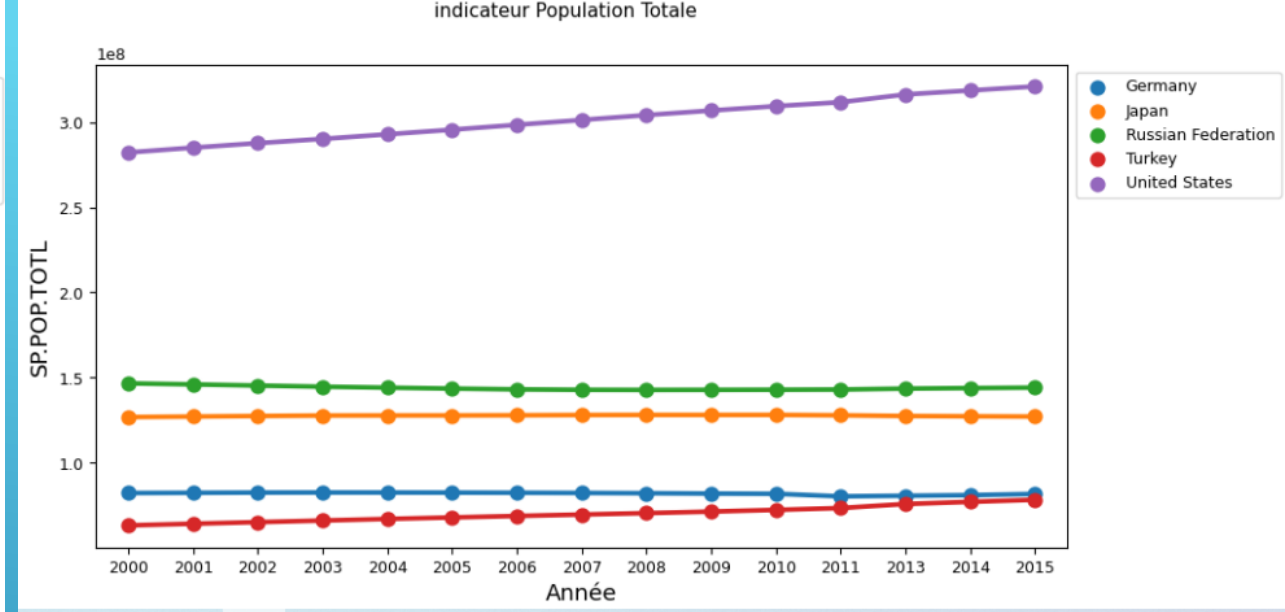



indicateur Population15-24
1e7
SP.POP.1524.TO.UN
Année
Germany
Japan
Russian Federation
Turkey
United States


indicateur Population Totale
1e8
SP.POP.TOTL
Année
Germany
Japan
Russian Federation
Turkey
United States

# 6- CONCLUSION

L'analyse des 6 indicateurs (Économique, numérique, Éducation secondaire, Éducation universitaire, population 14-25 ans et la population totale) nous permet de considérer que les pays avec un fort potentiel de clients pour ces services sont :

**United States.**
**Germany,**
**Japan,**
**Turkey**

# BOÎTE À OUTILS

academy

## Environnement



## Librairies de base



## Outil de bureau

## Visualisation



Missingno