Projet 7 : Effectuez une prédiction de revenus

Mission 0 :
Nettoyage des données

Import du DataFrame et contrôle des NaN

```
data = pd.read_csv("C:/Users/KyRun69/Desktop/Cours_Openclassrooms/Projets/Projet_7/data_projet_7.csv",sep=',',decimal=b',')
df=pd.DataFrame(data)
print(df.head())
```

```
  country  year_survey  quantile  nb_quantiles      income  gdpppp
0     ALB         2008         1           100   728.89795  7297.0
1     ALB         2008         2           100   916.66235  7297.0
2     ALB         2008         3           100  1010.91600  7297.0
3     ALB         2008         4           100  1086.90780  7297.0
4     ALB         2008         5           100  1132.69970  7297.0
```

```
print(df[df.isna().any(axis=1)].country.unique())
#Le Kosovo et le territoire Palestinien n'ont pas de valeur pour le gdpppp, il faudra les retirer pour certaines analyses
```

```
['XKX' 'PSE']
```

Mission 0 :
Nettoyage des données

Vérification des doublons et describe

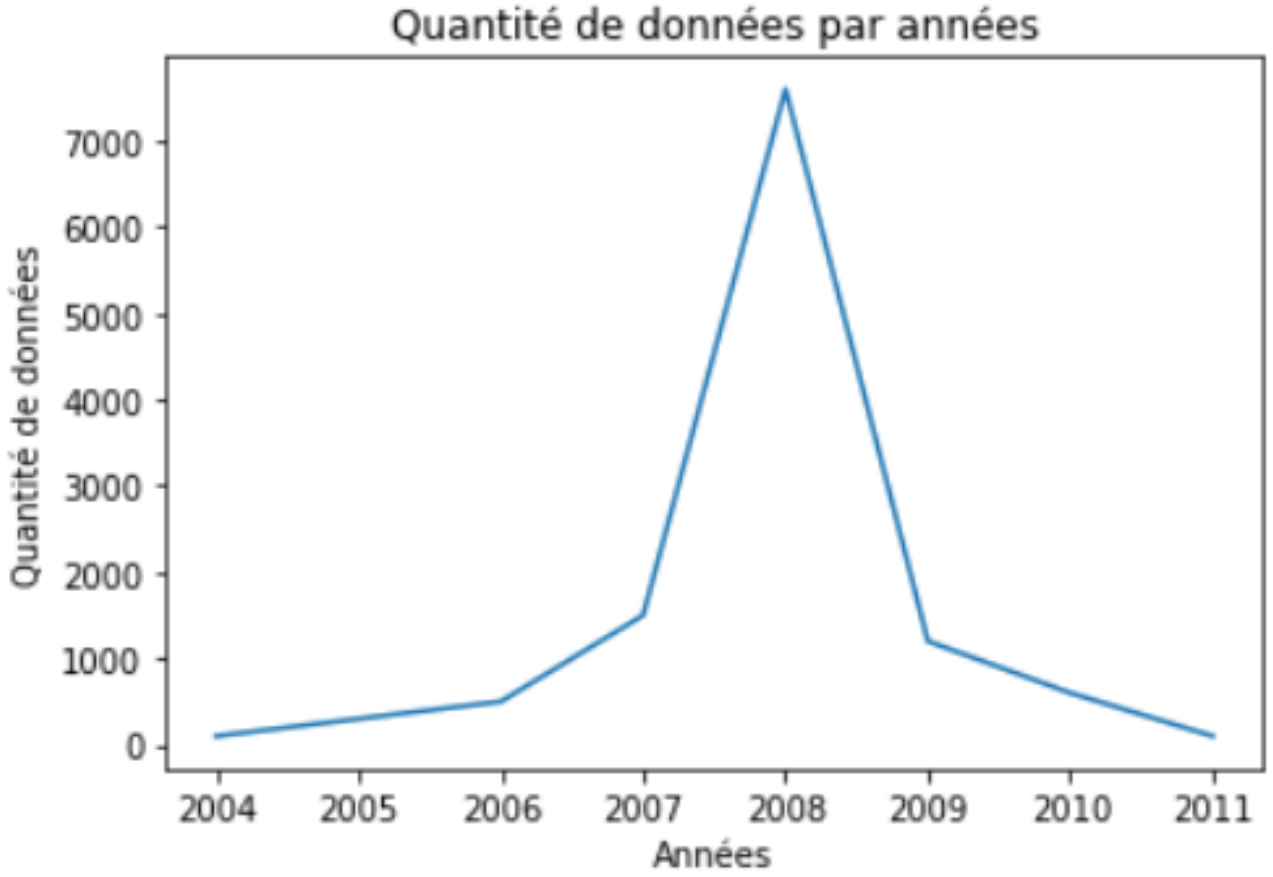
```
df.drop(columns=["country","year_survey","quantile","nb_quantiles"]).duplicated().sum() #Il n'y a aucun doublon
```

```
0
```

```
df.describe()
```

	year_survey	quantile	nb_quantiles	income	gdpppp
count	11599.000000	11599.000000	11599.0	11599.000000	1.139900e+04
mean	2007.982757	50.500819	100.0	6069.224260	5.022128e+04
std	0.909633	28.868424	0.0	9414.185972	4.000688e+05
min	2004.000000	1.000000	100.0	16.719418	3.031931e+02
25%	2008.000000	25.500000	100.0	900.685515	2.576000e+03
50%	2008.000000	51.000000	100.0	2403.244900	7.560000e+03
75%	2008.000000	75.500000	100.0	7515.420900	1.877300e+04
max	2011.000000	100.000000	100.0	176928.550000	4.300332e+06

Mission 0 :
Nettoyage des données



year_survey	
2004	100
2006	500
2007	1500
2008	7599
2009	1200
2010	600
2011	100

Mission 0 :
Nettoyage des données

```
# On regarde si tous les pays ont bien le bon nombre de quantiles

print(df[df.year_survey==2008].groupby("country").sum()[df[df.year_survey==2008].groupby("country").sum()["quantile"]<5050]) #LTU
```

```
         year_survey  quantile  nb_quantiles       income     gdpppp
country
LTU           198792      5009          9900  657483.5158  1739529.0
```

```
# Il manque le quantile 41 à la Lituanie
# Je décide alors d'y entrer la moyenne entre le quantile précédent et le suivant.

a=(4868.4507+4895.8306)/2

part1=df.iloc[:6240]
part2=df.iloc[6240:]

part2.index+=1
colonne={'country':"LTU",'year_survey':2008,'quantile':41,'nb_quantiles':100,'income':a,'gdpppp':17571.0}

liste=pd.DataFrame(index=[6140],data=colonne)

tout=part1.append(liste)

df=tout.append(part2)
```

Mission 0 :
Nettoyage des données

```
Les données contiennent 116 pays
```

Supposition : Chaque pays n'est représenté qu'une seule année

```
liste=[]
for i in df.country.unique():
    liste.append(i)
print(pd.DataFrame(liste).count())

liste2=[]
for i in df[df["quantile"]==1].country:
    liste2.append(i)
print(pd.DataFrame(liste2).count())  # Effectivement, aucun pays n'est représenté deux années différentes
```

Mission 0 :
Nettoyage des données

```
#Recherche d'outlier pour l'income, la Bolivie, la Chine et la Corée du Sud sont suspects

print(df.sort_values(by='income',ascending=True).head(10))
print("\n",df.sort_values(by='income',ascending=True).tail(10))
```

	country	year_survey	quantile	nb_quantiles	income	gdpppp
1700	CHN	2007	1	100	16.719418	5712.00000
5700	KOR	2008	1	100	17.317732	19162.00000
1100	BOL	2008	1	100	20.584948	3950.00000
11500	COD	2008	1	100	29.364283	303.19305
6700	MDG	2010	1	100	29.411367	950.00000
1800	CIV	2008	1	100	34.555264	1526.00000
9800	SWZ	2009	1	100	35.930096	4748.00000
6000	LBR	2007	1	100	37.782673	372.00000
3900	GTM	2011	1	100	38.463615	4367.00000
1400	CAF	2008	1	100	40.928130	685.00000

	country	year_survey	quantile	nb_quantiles	income	gdpppp
2299	CYP	2008	100	100	98480.560	26273.0
2499	DEU	2008	100	100	103661.980	33758.0
10998	USA	2008	99	100	106765.260	43261.0
6399	LUX	2008	100	100	114823.680	73127.0
8199	NOR	2008	100	100	120354.220	49070.0
3399	FRA	2008	100	100	122775.164	30357.0
1599	CAN	2008	100	100	133454.840	35895.0
3499	GBR	2008	100	100	141565.230	34048.0
4899	ISL	2008	100	100	160645.270	36527.0
10999	USA	2008	100	100	176928.550	43261.0

Mission 0 :
Nettoyage des données

Bolivie

quantile	income
1	20.584948
2	57.159256
3	85.552185
4	112.422820
5	143.889390
6	177.537310
7	213.239530
8	251.587940
9	284.519800
10	321.951020

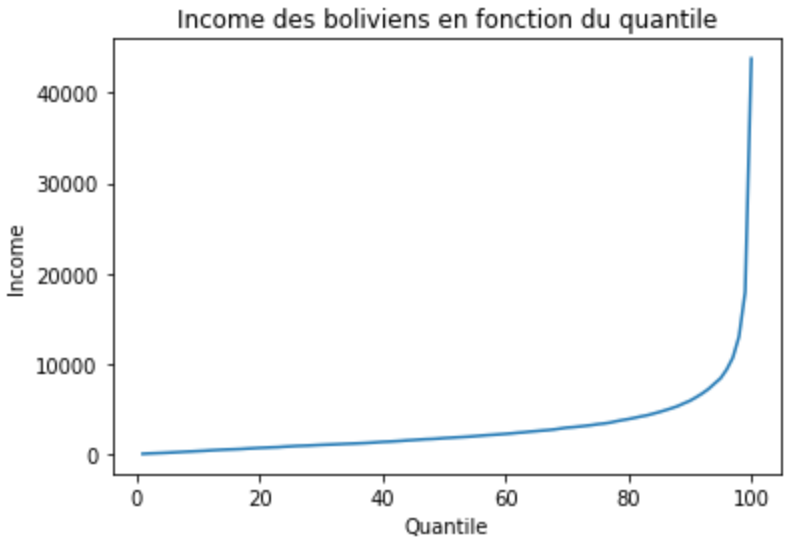
Chine

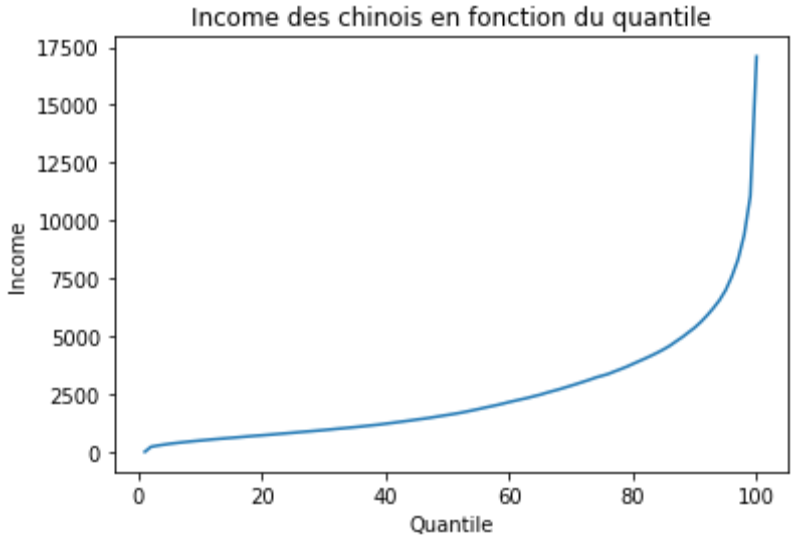
quantile	income
1	16.719418
2	234.150730
3	282.894900
4	323.826500
5	362.260600
6	395.279660
7	425.010070
8	452.844970
9	480.534400
10	505.896820

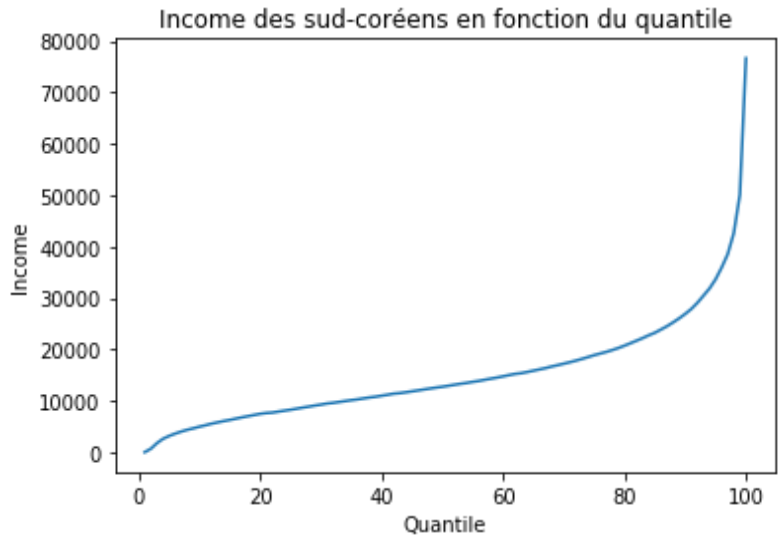
Corée du Sud

quantile	income
1	17.317732
2	679.077600
3	1763.440800
4	2609.038300
5	3166.106200
6	3620.534000
7	4023.435300
8	4369.286600
9	4655.849000
10	4972.012000

Mission 0 :
Nettoyage des données







Mission 0 :
Nettoyage des données

```
#Recherche d'outliers pour le gdpppp

print(df[df['quantile']==1].sort_values(by='gdpppp',ascending=True).head(10))
print("\n",df[df['quantile']==1].sort_values(by='gdpppp',ascending=True).tail(10))

#Les îles Figji sont un outlier au niveau du gdpppp
```

	country	year_survey	quantile	nb_quantiles	income	gdpppp
11500	COD	2008	1	100	29.364283	303.19305
6000	LBR	2007	1	100	37.782673	372.00000
7700	NER	2007	1	100	137.395310	631.00000
1400	CAF	2008	1	100	40.928130	685.00000
7500	MWI	2010	1	100	115.632480	728.81158
7300	MOZ	2008	1	100	42.583840	773.00000
7000	MLI	2010	1	100	122.693474	929.52966
6700	MDG	2010	1	100	29.411367	950.00000
3700	GIN	2007	1	100	93.354190	977.00000
8200	NPL	2010	1	100	192.238780	1048.18080

	country	year_survey	quantile	nb_quantiles	income	gdpppp
300	AUT	2008	1	100	2958.07640	36193.0
4800	ISL	2008	1	100	5191.96830	36527.0
8000	NLD	2008	1	100	2558.82080	38065.0
4500	IRL	2008	1	100	2819.13500	39268.0
10900	USA	2008	1	100	663.47610	43261.0
8100	NOR	2008	1	100	3520.26150	49070.0
6300	LUX	2008	1	100	5780.83740	73127.0
3200	FJI	2008	1	100	308.17334	4300332.0
5800	XKX	2008	1	100	437.89370	NaN
11200	PSE	2009	1	100	195.28990	NaN

Mission 0 :
Nettoyage des données

```
#Valeur trouvée sur internet pour les fidji : 7066.72

df.loc[3200:3299,"gdpppp"]=7066.72
```

```
a=0
b=0
for i in df.country.unique() :
    if not df[df.country==i].income.is_monotonic_increasing :
        print(i)
#Les quantiles sont bien croissants
```

Mission 0 :
Nettoyage des données

```
#Import et merge d'une table contenant les noms des pays ainsi que les régions

data2 = pd.read_csv("C:/Users/KyRun69/Desktop/Cours_Openclassrooms/Projets/Projet_7/Indice_gini/Country.csv")
df_country=pd.DataFrame(data2)
df_country=df_country.rename(index=str, columns={'Country Code':'country'})
df=pd.merge(df,df_country,how='left')
df=df.drop(columns=['IncomeGroup', 'SpecialNotes','Unnamed: 5'])

# Recherche de NaN

df.TableName=df.TableName.dropna(axis=0)
print(df.loc[10401]) # country TWN mais TableName et Region NaN
df.loc[10400:10499,"TableName"]="Taiwan"
df.loc[10400:10499,"Region"]='South Asia'
```

Mission 0 :
Nettoyage des données

```
# Import et merge d'une table contenant la population de chaque pays

pop=pd.read_csv("C:/Users/KyRun69/Desktop/Cours_Openclassrooms/Projets/Projet_7/Pop/pop1.csv",skiprows=3)
df_pop=pd.DataFrame({'country':pop["Country Code"],'Population 2017':pop["2017"]})
df=pd.merge(df,df_pop,how="left")
df.loc[10400:10499,"Population 2017"]=23415126

#La population mondiale est estimée à 7,55 milliards au 1er juillet 2017 selon l'ONU

print("L'analyse couvre ", round(df.groupby(df.country)["Population 2017"].mean().sum()/7550000000*100,2), "% de la population")
```

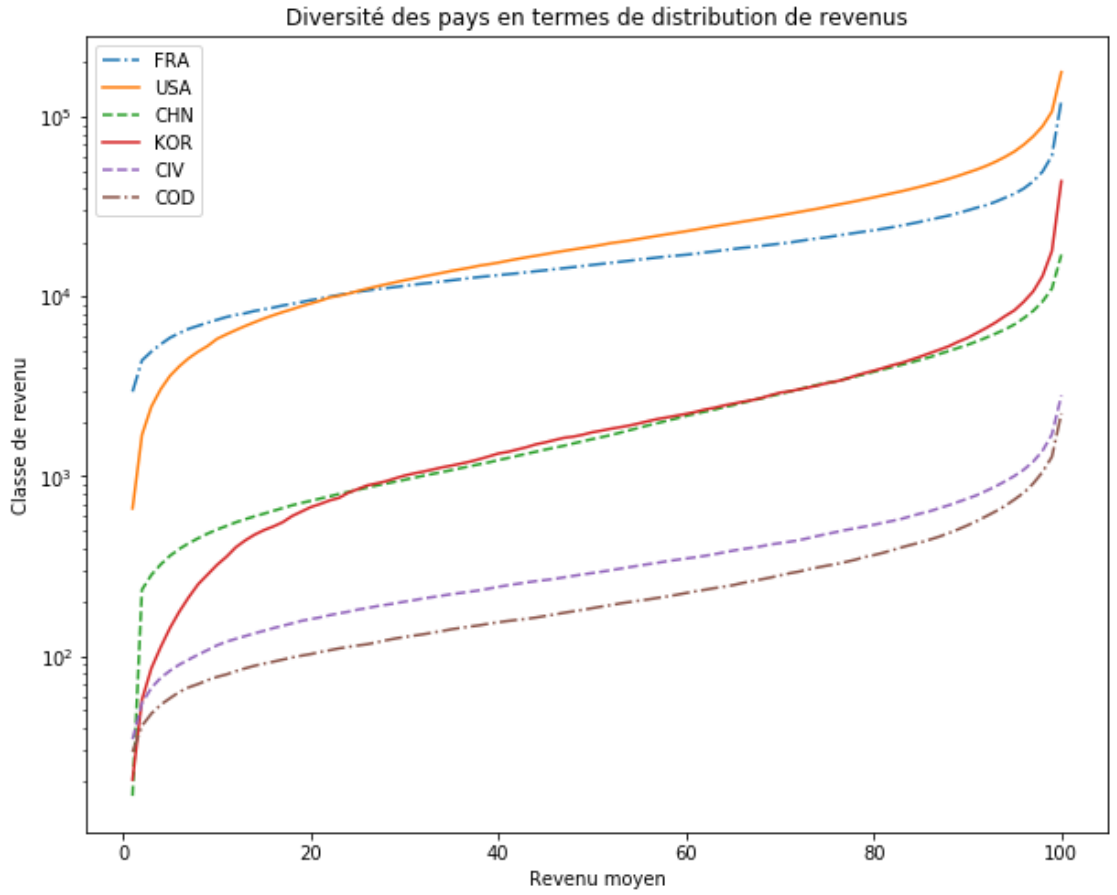
```
L'analyse couvre  90.79 % de la population
```

Mission 2 :
Inégalités entre différents pays

```
#Import d'une table contenant l'indice de gini

df_gini=pd.read_csv("C:/Users/KyRun69/Desktop/API_SI.POV.GINI_DS2_en_csv_v2_10134274/API_SI.POV.GINI_DS2_en_csv_v2_10134274.csv"
df_gini=df_gini.drop(columns='Unnamed: 62')
#print(gini.fillna(gini.drop(columns=['Indicator Code','Indicator Name','Country Code','Country Name']).mean()))
df_gini=df_gini.dropna(axis=1,how="all")
print(df_gini) #19 colonnes ne contenaient que des nan
```

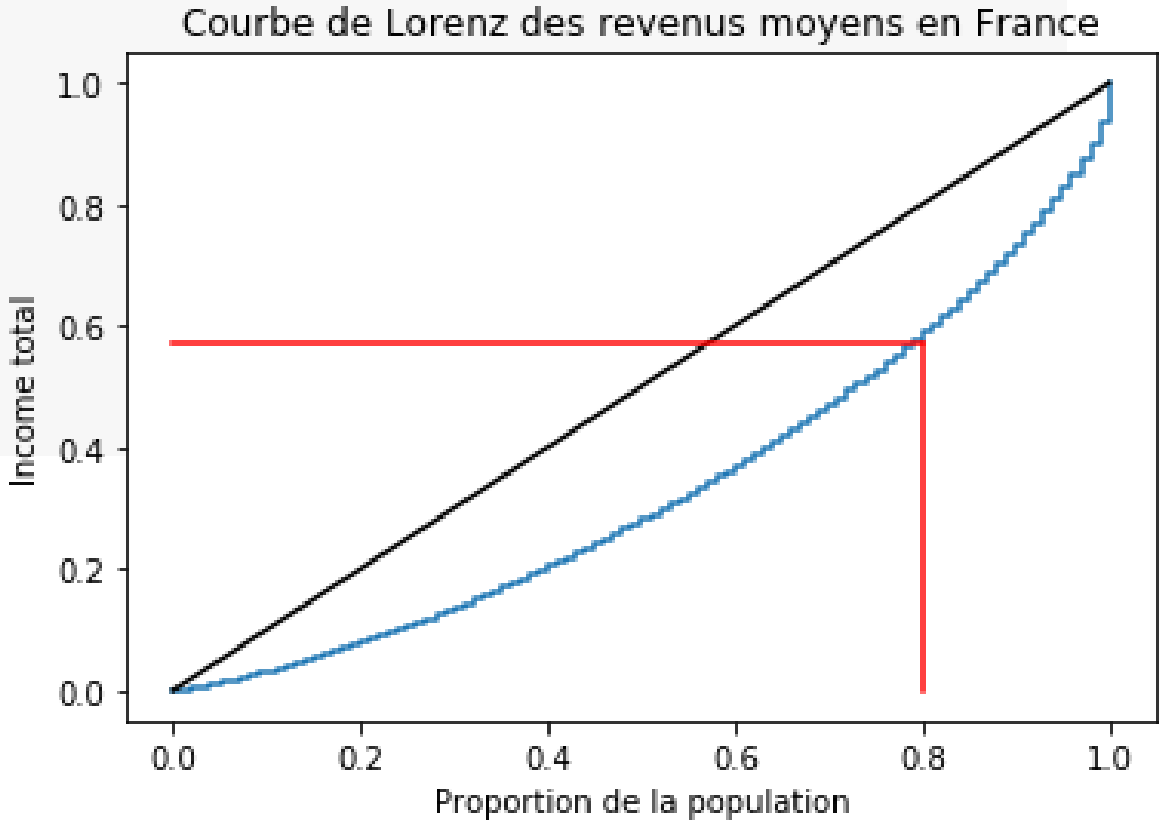
Mission 2 :
Inégalités entre différents pays

```
fig = plt.figure(figsize=[10,8])
plt.plot(df[df.country=="FRA"]["quantile"],df[df.country=="FRA"].income,label='FRA',linestyle='-.')
plt.plot(df[df.country=="USA"]["quantile"],df[df.country=="USA"].income,label='USA')
plt.plot(df[df.country=="CHN"]["quantile"],df[df.country=="CHN"].income,label='CHN',linestyle='--')
plt.plot(df[df.country=="BOL"]["quantile"],df[df.country=="BOL"].income,label='KOR')
plt.plot(df[df.country=="CIV"]["quantile"],df[df.country=="CIV"].income,label='CIV',linestyle='--')
plt.plot(df[df.country=="COD"]["quantile"],df[df.country=="COD"].income,label='COD',linestyle='-.')
plt.yscale('log')
plt.xlabel("Revenu moyen")
plt.ylabel("Classe de revenu")
plt.legend()
plt.title("Diversité des pays en termes de distribution de revenus")

plt.show()
```

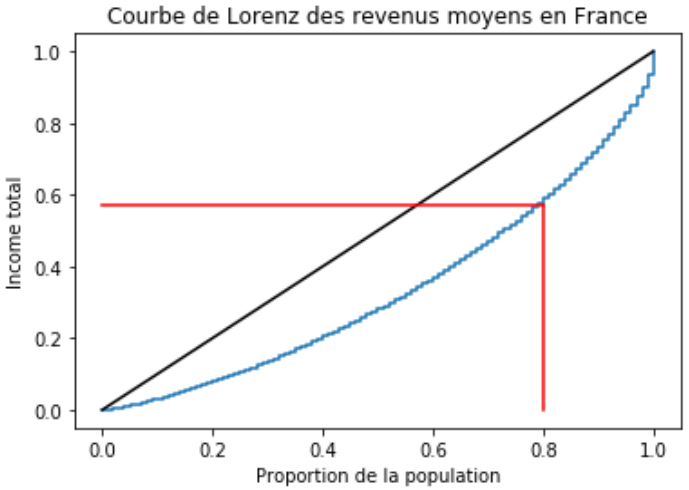


Mission 2 :
Inégalités entre différents pays

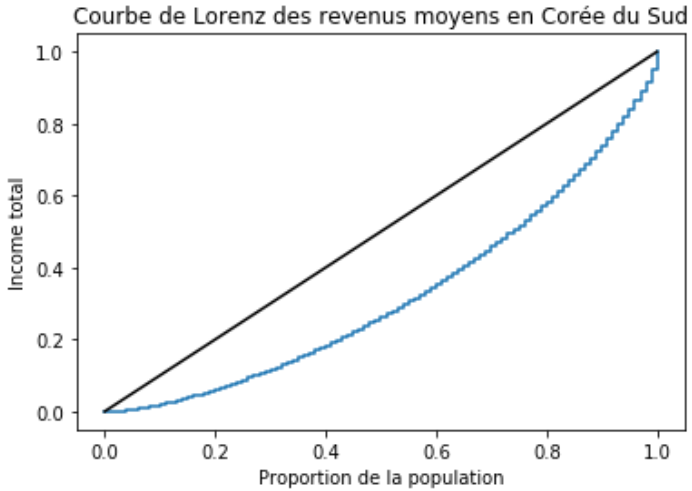
```
lorenz = np.cumsum(df[df.year_survey==2008][df.country=="FRA"].income / df[df.year_survey==2008][df.country=="FRA"].income.sum())
lorenz = np.append([0],lorenz)

plt.plot(np.linspace(0,1,len(lorenz)),lorenz,drawstyle='steps-post')
plt.plot([0,1],"k")
plt.plot([0.8,0.0],[0.57,0.57],"r")
plt.plot([0.8,0.8],[0.57,0.0],"r")
plt.title('Courbe de Lorenz des revenus moyens en France')
plt.xlabel("Proportion de la population")
plt.ylabel("Income total")
plt.show()

aire_ss_courbe = lorenz[:-1].sum()/len(lorenz)
S = 0.5 - aire_ss_courbe
gini = 2*S
print(gini)
```



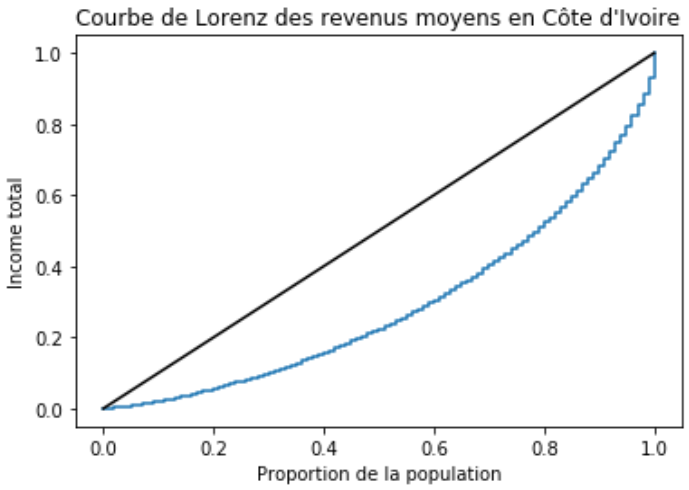
Mission 2 :
Inégalités entre différents pays



Indice de Gini : 0.35

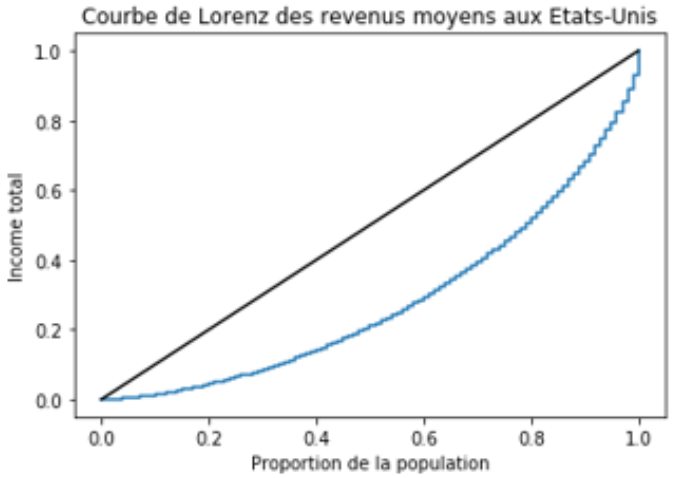


Indice de Gini : 0.37

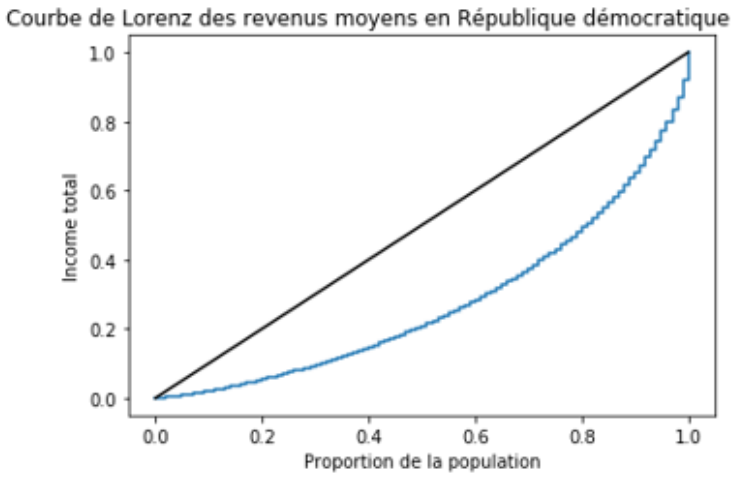


Indice de Gini : 0.43

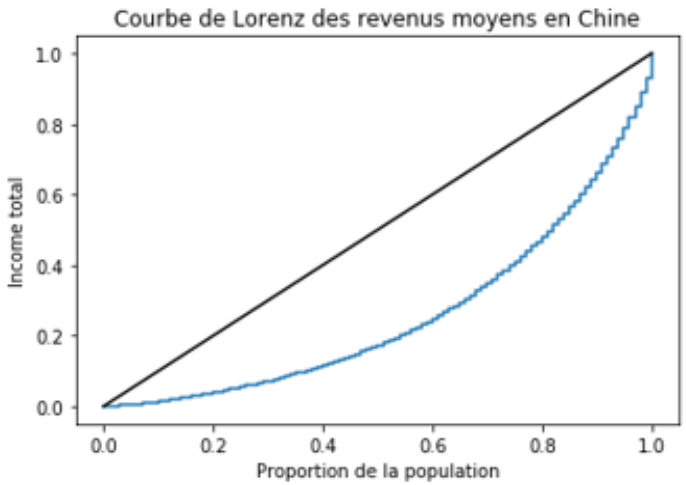
Mission 2 :
Inégalités entre différents pays



Indice de Gini : 0.45



Indice de Gini : 0.46



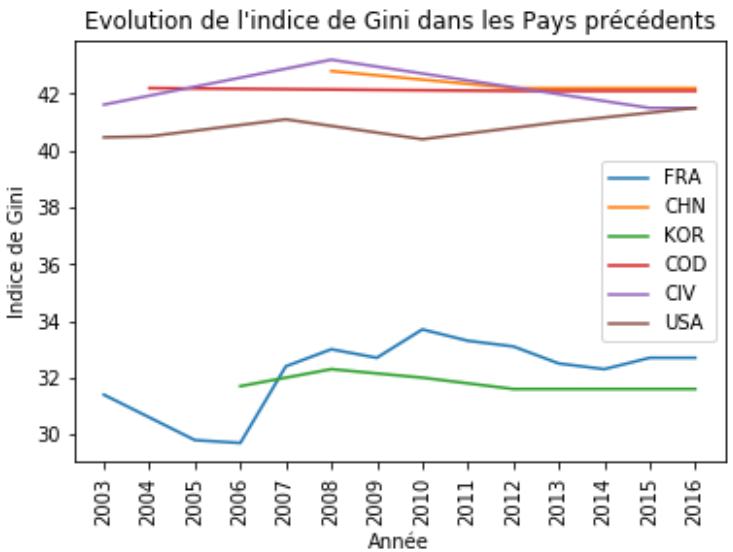
Indice de Gini : 0.49

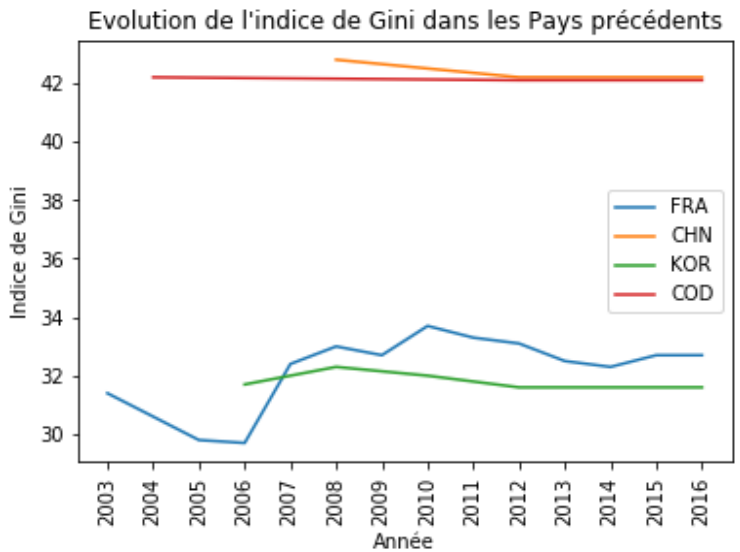
Mission 2 :
Inégalités entre différents pays

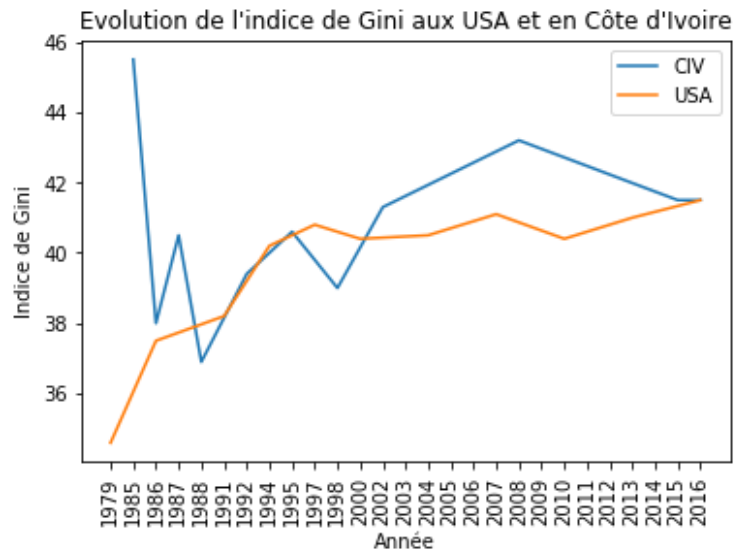
```
# Création d'une petit DataFrame pour analyser l'évolution des indices de Gini dans les pays précédents

df_tempo=pd.merge(((df_gini[df_gini["Country Code"]=='USA'].dropna(axis=1,how="all")).drop(columns=['Country Name','Indicator Name
df_tempo=pd.merge(df_tempo,((df_gini[df_gini["Country Code"]=='CHN'].dropna(axis=1,how="all")).drop(columns=['Country Name','Indic
df_tempo=pd.merge(df_tempo,((df_gini[df_gini["Country Code"]=='KOR'].dropna(axis=1,how="all")).drop(columns=['Country Name','Indic
df_tempo=pd.merge(df_tempo,((df_gini[df_gini["Country Code"]=='CIV'].dropna(axis=1,how="all")).drop(columns=['Country Name','Indic
df_tempo=pd.merge(df_tempo,((df_gini[df_gini["Country Code"]=='COD'].dropna(axis=1,how="all")).drop(columns=['Country Name','Indic

df_tempo=df_tempo.reindex(sorted(df_tempo.columns), axis=1)
df_tempo=df_tempo.set_index("Country Code",drop=True).transpose()
```

```
df_tempo=df_tempo.interpolate()


plt.plot(df_tempo.loc['2003':].FRA,label="FRA")
plt.plot(df_tempo.loc['2003':].CHN,label="CHN")
plt.plot(df_tempo.loc['2003':].KOR,label="KOR")
plt.plot(df_tempo.loc['2003':].COD,label="COD")
plt.plot(df_tempo.loc['2003':].CIV,label="CIV")
plt.plot(df_tempo.loc['2003':].USA,label="USA")
plt.legend()
plt.xticks(rotation=90)
plt.title("Evolution de l'indice de Gini dans les Pays précédents")
plt.xlabel("Année")
plt.ylabel("Indice de Gini")
plt.show()
```

Mission 2 :
Inégalités entre différents pays







Mission 2 :
Inégalités entre différents pays

```
#Création d'une colonne nommée recent contenant les indices de gini les plus recents pour tous les pays

df_gini=df_gini.drop(columns=["Country Code","Indicator Name", "Indicator Code"]).set_index("Country Name").dropna(how="all")
df_gini=df_gini.reset_index(drop=False)
df_gini_test=df_gini.reset_index(drop=True).drop(columns="Country Name")
print(float(df_gini_test.loc[0].dropna().tail(1)))
i=0
liste=[]
for i in df_gini_test.index :
    if i == 161 :
        break
    else :
        liste.append(float(df_gini_test.loc[i].dropna().tail(1)))
df_gini_test=df_gini_test.add(pd.DataFrame(liste),fill_value=0)
df_gini=pd.merge(df_gini,df_gini_test,how="left")
df_gini=df_gini.rename(columns={0:"recent"}).set_index("Country Name")
print(df_gini)
```

Mission 2 :
Inégalités entre différents pays

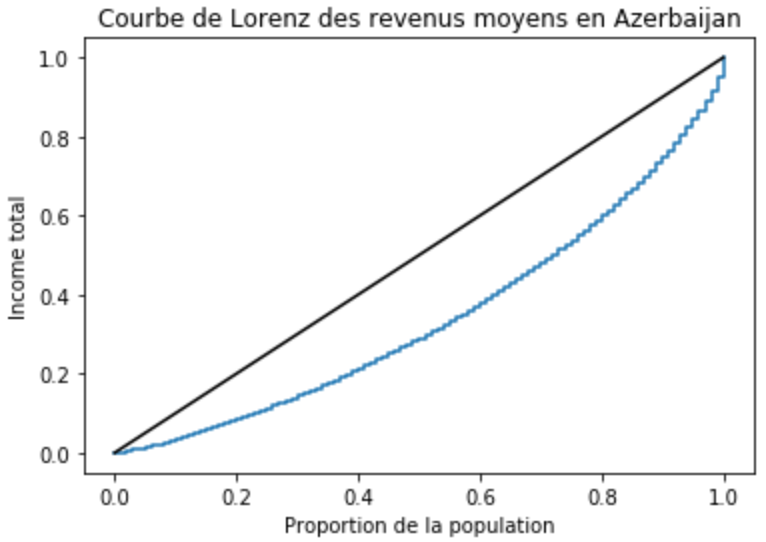
```
#Recherche d'outliers dans les indices de gini

print("La moyenne des indices de Gini est de :",df_gini["recent"].mean())
print("\n",df_gini.sort_values(by="recent",ascending=False)["recent"].head(5))
print("\n",df_gini.sort_values(by="recent",ascending=True)["recent"].head(5))
print("\n La France est en position",df_gini.sort_values(by="recent",ascending=True)["recent"]

#L'Azerbaijan semble être un outlier
```

```
La moyenne des indices de Gini est de : 38.4385093167702

 Country Name
South Africa    63.0
Botswana        60.5
Namibia         59.1
Suriname        57.6
Zambia          57.1
Name: recent, dtype: float64

 Country Name
Azerbaijan        16.6
Ukraine           25.0
Slovenia          25.4
Czech Republic    25.9
Moldova           26.3
Name: recent, dtype: float64

 La France est en position 39 en allant de l'indice le moins élevé au plus élevé
```

Mission 2 :
Inégalités entre différents pays

```
print(df_gini.loc["Azerbaijan"].dropna())

lorenz = np.cumsum(df[df.country=="AZE"].income / df[df.country=="AZE"].income.sum())
lorenz = np.append([0],lorenz)

plt.plot(np.linspace(0,1,len(lorenz)),lorenz,drawstyle='steps-post')
plt.plot([0,1],"k")
plt.title('Courbe de Lorenz des revenus moyens en Azerbaijan')
plt.xlabel("Proportion de la population")
plt.ylabel("Income total")
plt.show()

aire_ss_courbe = lorenz[:-1].sum()/len(lorenz)
S = 0.5 - aire_ss_courbe
gini = 2*S
print(gini)

df_gini.loc["Azerbaijan"][39]=32.9
```

```
1995      34.7
2001      36.5
2002      17.4
2003      18.8
2004      16.2
2005      16.6
recent    16.6
Name: Azerbaijan, dtype: float64
```



Indice de Gini : 0.329

Mission 2 :
Inégalités entre différents pays

```
print("l\'indice gini de la France est de : {}".format(df_gini.loc["France"][39]))
print("l\'indice gini de la Corée du Sud est de : {}".format(df_gini.loc["Korea, Rep."][39]))
print("l\'indice gini de la Cote d'Ivoire est de : {}".format(df_gini.loc["Cote d'Ivoire"][39]))
print("l\'indice gini des USA est de : {}".format(df_gini.loc["United States"][39]))
print("l\'indice gini de la République Démocratique du Congo est de : {}".format(df_gini.loc["Congo, Dem. Rep."][39]))
print("l\'indice gini de la Chine est de : {}".format(df_gini.loc["China"][39]))
```

```
l'indice gini de la France est de : 32.7
l'indice gini de la Corée du Sud est de : 31.6
l'indice gini de la Cote d'Ivoire est de : 41.5
l'indice gini des USA est de : 41.5
l'indice gini de la République Démocratique du Congo est de : 42.1
l'indice gini de la Chine est de : 42.2
```

Mission 2 :
Inégalités entre différents pays

```
#Import d'un DataFrame contenant le coefficient d'elasticité de chaque pays

df_elastici=pd.read_csv("C:/Users/KyRun69/Desktop/Cours_Openclassrooms/Projets/Projet_7/GDIMMay2018.csv")
IGE=df_elastici[['region','IGEincome']]
print(IGE)
```

```
                region  IGEincome
0           South Asia        NaN
1           South Asia        NaN
2           South Asia        NaN
3           South Asia        NaN
4           South Asia        NaN
5           South Asia        NaN
6           South Asia        NaN
7           South Asia        NaN
8           South Asia        NaN
```

Mission 2 :
Inégalités entre différents pays

```
#Import d'un txt contenant des informations nous permettant de combler les coefficients d'elasticité manquants

df_elastici3=pd.read_table("C:/Users/KyRun69/Desktop/Cours_Openclassrooms/Projets/Projet_7/elasticity.txt")
print(df_elastici3)
print(df.Region.unique())
```

```
    Coefficients of intergenerational elasticity between parents' and
0                                    children's income
1                             | Base case  | Optim...
2                             |            | (high...
3    ------------------------------------------------...
4    Nordic European countries |    0.2     |      ...
5    and Canada                |            |      ...
6    ------------------------------------------------...
7    Europe (except nordic     |    0.4     |      ...
8    countries)                |            |      ...
9    ------------------------------------------------...
10   Australia/New Zealand/USA |    0.4     |      ...
11   ------------------------------------------------...
12   Asia                      |    0.5     |      ...
13   ------------------------------------------------...
14   Latin America/Africa      |    0.66    |      ...
15   ------------------------------------------------...
16                 Extrapolations from these sources :
17   Lam, David and Robert Schoeni (1993), "Effects...
18   earnings and returns to schooling: Evidence fr...
19                               vol. 110, pp. 710-740.
20   Grawe, Nathan (2001), "Intergenerational mobil...
21   Quantile and mean regression measures", Ph.D. ...
```

Mission 2 :
Inégalités entre différents pays

```
#On comble les valeurs manquantes

data3={"region":['South Asia', 'Sub-Saharan Africa', 'Europe & Central Asia',
 'Latin America & Caribbean', 'East Asia & Pacific','Middle East & North Africa'],
 "elasticity":[0.5,0.66,0.4,0.66,0.4,0.66]}
region=pd.DataFrame(data=data3)
df_elastici=pd.merge(df_elastici,region,how="left")

df_elastici.IGEincome=df_elastici.IGEincome.fillna(df_elastici.elasticity)

#Pas de valeur pour les pays ['Estonia','Hungary','Iceland','Israel','Lithuania','Poland','Uruguay']

df_elastici=df_elastici[['countryname','region','IGEincome']].groupby(['countryname','region']).mean()
df_elastici=df_elastici.reset_index()
df_elastici.loc[42,"IGEincome"]=0.2
df_elastici.loc[56,"IGEincome"]=0.4
df_elastici.loc[57,"IGEincome"]=0.2
df_elastici.loc[63,"IGEincome"]=0.66
df_elastici.loc[78,"IGEincome"]=0.2
df_elastici.loc[108,"IGEincome"]=0.4
df_elastici.loc[142,"IGEincome"]=0.66
```

Mission 2 :
Inégalités entre différents pays

```
print(df[df.isnull().any(axis=1)].TableName.unique())
#print(df[df.TableName=="Taiwan"])
df=pd.merge(df,df_gini.reset_index()[["Country Name","recent"]].rename(columns={"Country Name":"TableName","recent":"gini"}),how='
#print(df)
print(df[df.isnull().any(axis=1)].TableName.unique())
print(df[df.TableName=="Côte d'Ivoire"].head())
```

```
['Kosovo' 'West Bank and Gaza']
["Côte d'Ivoire" 'Cambodia' 'Kosovo' 'Philippines' 'Poland' 'Serbia'
 'Taiwan' 'West Bank and Gaza']
     country  year_survey  quantile  nb_quantiles     income  gdpppp  \
1800     CIV         2008         1           100  34.555264  1526.0
1801     CIV         2008         2           100  54.766040  1526.0
1802     CIV         2008         3           100  66.659250  1526.0
1803     CIV         2008         4           100  75.798510  1526.0
1804     CIV         2008         5           100  83.163830  1526.0

                  Region      TableName  Population 2017  gini
1800  Sub-Saharan Africa  Côte d'Ivoire       24294750.0   NaN
1801  Sub-Saharan Africa  Côte d'Ivoire       24294750.0   NaN
1802  Sub-Saharan Africa  Côte d'Ivoire       24294750.0   NaN
1803  Sub-Saharan Africa  Côte d'Ivoire       24294750.0   NaN
1804  Sub-Saharan Africa  Côte d'Ivoire       24294750.0   NaN
```

Mission 2 :
Inégalités entre différents pays

```
#On comble les valeurs manquantes

lorenz = np.cumsum(df[df.country=="CIV"].income / df[df.country=="CIV"].income.sum())
lorenz = np.append([0],lorenz)
aire_ss_courbe = lorenz[:-1].sum()/len(lorenz)
S = 0.5 - aire_ss_courbe
gini = 2*S
df.loc[1800:1899,"gini"]=gini

lorenz = np.cumsum(df[df.country=="KHM"].income / df[df.country=="KHM"].income.sum())
lorenz = np.append([0],lorenz)
aire_ss_courbe = lorenz[:-1].sum()/len(lorenz)
S = 0.5 - aire_ss_courbe
gini = 2*S
df.loc[5600:5699,"gini"]=gini
```

Mission 2 :
Inégalités entre différents pays

```
df=pd.merge(df,df_elastici.rename(columns={"countryname":"TableName"}).drop(columns="region"),how="left")
```

```
#On comble les valeurs manquantes

df.loc[1800:1899,"IGEincome"]=0.66
df.loc[9800:9899,"IGEincome"]=0.66
df.loc[9900:9999,"IGEincome"]=0.66
df.loc[10400:10499,"IGEincome"]=0.5

print(df[df.isnull().any(axis=1)].country.unique())
```

```
['XKX' 'PSE']
```

Mission 3 :
Création d'un nouvel échantillon

```
pj = 0.9                  # coefficient d'élasticité du pays j
nb_quantiles = 100        # nombre de quantiles (nombre de classes de revenu)
n  = 1000*nb_quantiles    # taille de l'échantillon

y_child, y_parents = generate_incomes(n, pj)
sample = compute_quantiles(y_child, y_parents, nb_quantiles)
cd = conditional_distributions(sample, nb_quantiles)
print(cd)

c_i_child = 5
c_i_parent = 8
p = proba_cond(c_i_parent, c_i_child, cd)
print("\nP(c_i_parent = {} | c_i_child = {}, pj = {}) = {}".format(c_i_parent, c_i_child, pj, p))
print(sample)
pj=0
nb_quantiles=0
n=0
sample=0
y_child=0
y_parents=0
c_i_child=0
c_i_parent=0
#Je ne garde que la distribution conditionnelle ( cd)
```

Mission 3 :
Création d'un nouvel échantillon

```python
# Création de df2 1000 fois plus grand que df

df2=pd.concat([df]*1000, ignore_index=True)
df2=df2.sort_values(by=['country', 'quantile'], ascending=True)
df2=df2.reset_index(drop=True)

print(df2)
```

	country	year_survey	quantile	nb_quantiles	income	gdpppp
0	ALB	2008	1	100	728.89795	7297.0
1	ALB	2008	1	100	728.89795	7297.0
2	ALB	2008	1	100	728.89795	7297.0
3	ALB	2008	1	100	728.89795	7297.0
4	ALB	2008	1	100	728.89795	7297.0
5	ALB	2008	1	100	728.89795	7297.0
6	ALB	2008	1	100	728.89795	7297.0
7	ALB	2008	1	100	728.89795	7297.0
8	ALB	2008	1	100	728.89795	7297.0

Mission 3 :
Création d'un nouvel échantillon

```
# Création de deux listes. La seconde, liste4, contient le numéro du quantile. La première, liste3, contient le nombre de fois
# où il faut insérer le quantile correspondant. Par exemple à mon loc[0] les deux valeurs sont 1 et 238.
# Par la suite il faudra donc insérer 238 fois les quantile 1

liste3=[]
liste4=[]
i=0
j=0
k=0
l=0
#for k in range(100) :
#    print(k)


for i in df.country.unique() :
    for j in range(1,101,1) :
        for k in range(0,100,1) :
            liste4.append(k+1)
            p=proba_cond(k,(j-1),cd)
            liste3.append(int(1000*p))

print(pd.DataFrame(liste3))
print(pd.DataFrame(liste4))
```

Mission 3 :
Création d'un nouvel échantillon

```python
# Création de liste5 comme décrite précédement

enfants=pd.DataFrame({"nombre":liste3,"classe":liste4})
liste5=[]
i=0
j=0
for i in enfants.index :
    for j in range(0,enfants.loc[i,'nombre'],1) :
        liste5.append(enfants.loc[i,'classe'])
print(pd.DataFrame(liste5))    #pas super niveau temps d'execution. Optimisable
```

```
    0
0   1
1   1
2   1
3   1
4   1
5   1
6   1
7   1
8   1
9   1
10  1
11  1
12  1
13  1
```

Mission 3 :
Création d'un nouvel échantillon

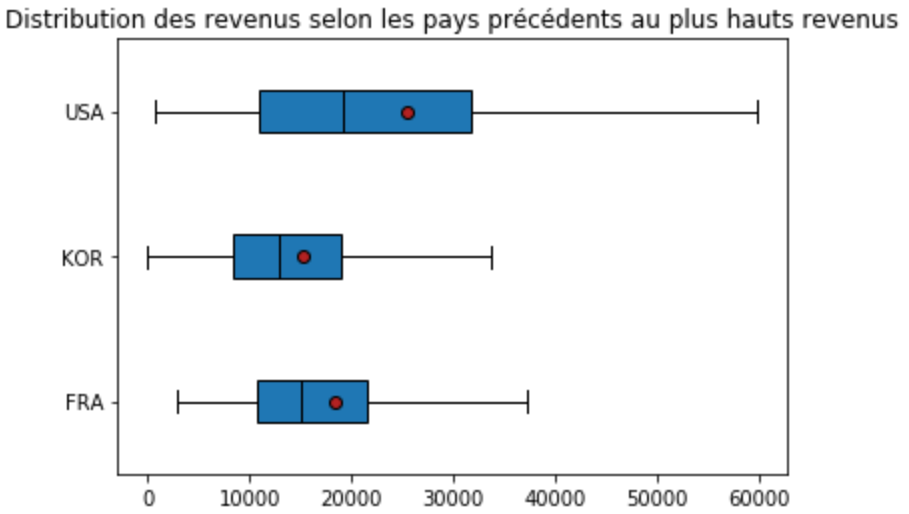
```
df2["quantile_parents"]=pd.DataFrame(liste5)
```

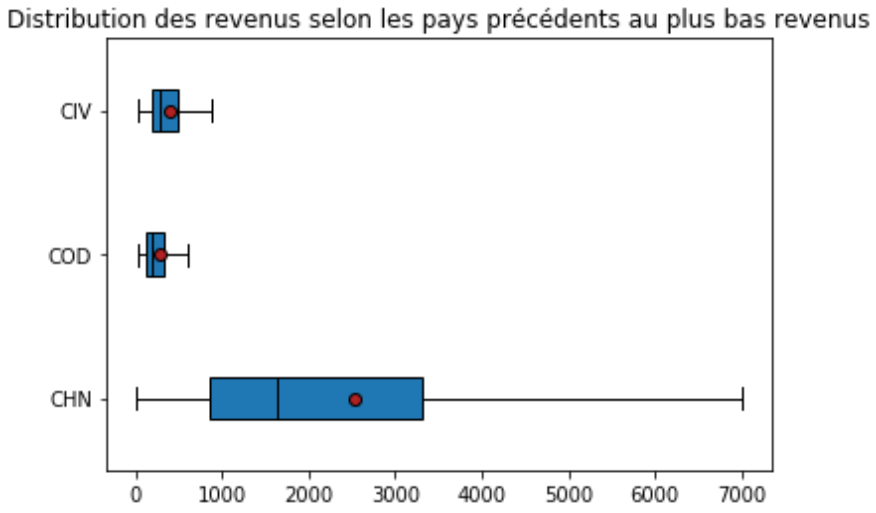
```
#Création de df_sans qui exclue les pays où il manque des valeurs

df_sans = df2[df2.country!="XKX"]
df_sans = df_sans[df_sans.country!="PSE"]
df_sans=pd.merge(df_sans,df_sans.groupby("country")["income"].mean().reset_index(drop=False).rename(columns={"income":"mean_income
```

Mission 4 :
Régression linéaire

```
# ANOVA

X = 'country'
Y = 'income'

df_FRA=df2[df2.country=="FRA"]
df_CHN=df2[df2.country=="CHN"]
df_KOR=df2[df2.country=="KOR"]
df_COD=df2[df2.country=="COD"]
df_CIV=df2[df2.country=="CIV"]
df_USA=df2[df2.country=="USA"]

df_6=pd.concat([df_FRA,df_KOR,df_USA],ignore_index=True)
df_7=pd.concat([df_CHN,df_COD,df_CIV],ignore_index=True)


modalites = df_6[X].unique()
groupes = []
for m in modalites:
    groupes.append(df_6[df_6[X]==m][Y])


medianprops = {'color':"black"}
meanprops = {'marker':'o', 'markeredgecolor':'black','markerfacecolor':'firebrick'}

plt.boxplot(groupes, labels=modalites, showfliers=False, medianprops=medianprops,vert=False, patch_artist=True, showmeans=True,
plt.title('Distribution des revenus selon les pays précédents au plus hauts revenus')
plt.show()


modalites = df_7[X].unique()
groupes = []
for m in modalites:
    groupes.append(df_7[df_7[X]==m][Y])
```

Mission 4 :
Régression linéaire





Mission 4 :
Régression linéaire

```python
def calc_anova(x,y,data):

    k = len(pd.unique(data[x]))  # nombre de groupes
    N = len(data.values)  # taille de l'échantillon
    n = data.groupby(x).size() # nb de valeurs par groupes

    # DF = Degré de liberté

    DFbetween = k - 1
    DFwithin = N - k
    DFtotal = N - 1

    moyenne_y = data[y].mean()

    classes = []

    for classe in data[x].unique():

        yi_classe = data[y][data[x]==classe]

        classes.append({'ni': len(yi_classe),
                        'moyenne_classe': yi_classe.mean(),
                       'variance_classe': yi_classe.var(ddof=0)})

    SCT = sum([(yj-moyenne_y)**2 for yj in data[y]])

    SCE = sum([c['ni']*(c['moyenne_classe']-moyenne_y)**2 for c in classes])

    SCR = sum([c['ni']*(c['variance_classe']) for c in classes])




    MSbetween = SCE/DFbetween
    MSwithin = SCR/DFwithin
    F_value = MSbetween/MSwithin
    p_value = st.f.sf(F_value, DFbetween, DFwithin)

    resultat = dict({'SCE':round(SCE,3),'SCT':round(SCT,3),'SCR':round(SCR,3),'eta_squared':round(SCE/SCT,3),


    return resultat
```

Mission 4 :
Régression linéaire

```
#Hypothèse H0 : les deux variables sont indépendantes

calc_anova(X,Y,df2)

#P-value < seuil de 5%  on rejette donc l'hypothèse. Les deux variables sont donc corrélées
```

```
{'SCE': 510237524303841.25,
 'SCT': 1027896166368465.9,
 'SCR': 517658642063213.56,
 'eta_squared': 0.496,
 'Valeur_F': 99422.51201414193,
 'P-valeur': 0.0}
```

Mission 4 :
Régression linéaire

```
# Regression linéaire 1 v2

X = df_sans[['gdpppp','gini']]
Y = df_sans['income']
X= sm.add_constant(X)
```

```
results=sm.OLS(Y,X).fit()
results.params
```

```
const     225.110399
gdpppp      0.479678
gini       -2.910243
dtype: float64
```

```
regr = linear_model.LinearRegression()
regr.fit(X, Y)
Y_predits = regr.predict(X)

# mean squared error : Elle montre à quel point la droite est proche du nuage de points
print("Mean squared error: %.2f"
      % mean_squared_error(Y, Y_predits))

# Pourcentage de variance expliquée : Si 1 le modèle prédit parfaitement
print('Variance score: %.2f' % r2_score(Y, Y_predits))

var_expl_model_2 = round(results.rsquared,2)
```

```
Mean squared error: 49766199.59
Variance score: 0.45
```

Mission 4 :
Régression linéaire

```
# Regression linéaire 2 v2
df_sans["gdpppp_log"] = np.log(df_sans["gdpppp"])

X = df_sans[['gdpppp_log','gini']]
Y = df_sans['income_log']
X= sm.add_constant(X)
```

```
results=sm.OLS(Y,X).fit()
results.params
```

```
const         0.327250
gdpppp_log    0.885492
gini         -0.007733
dtype: float64
```

```
regr = linear_model.LinearRegression()
regr.fit(X, Y)
Y_predits = regr.predict(X)

# mean squared error : Elle montre à quel point la droite est proche du nuage de points
print("Mean squared error: %.2f"
      % mean_squared_error(Y, Y_predits))

# Pourcentage de variance expliquée : Si 1 le modèle prédit parfaitement
print('Variance score: %.2f' % r2_score(Y, Y_predits))

var_expl_model_2 = round(results.rsquared,2)
```

```
Mean squared error: 0.67
Variance score: 0.65
```

Mission 4 :
Régression linéaire

```
# Regression linéaire 3 v2

X = df_sans[['gdpppp','gini','quantile_parents']]
Y = df_sans['income']
```

```
X=sm.add_constant(X)
results=sm.OLS(Y,X).fit()
results.params
```

```
const              -4217.694492
gdpppp                 0.479678
gini                  -2.910243
quantile_parents      87.976334
dtype: float64
```

```
Y_predits = results.predict(X)
X_pred = X.copy()
X_pred['revenus_predits'] = Y_predits
```

```
regr = linear_model.LinearRegression()
regr.fit(X, Y)
Y_predits = regr.predict(X)

# mean squared error : Elle montre à quel point la droite est proche du nuage de points
print("Mean squared error: %.2f"
      % mean_squared_error(Y, Y_predits))

# Pourcentage de variance expliquée : Si 1 le modèle prédit parfaitement
print('Variance score: %.2f' % r2_score(Y, Y_predits))

var_expl_model_2 = round(results.rsquared,2)

print("le  modèle  explique  près de {}%  de  la variabilité de la variable cible".format(round(results.rsquared*100,2)))

var_expl_autres = round((1-round(results.rsquared,2)),2)
print("\n Variance expliquée par d'autres facteurs :",var_expl_autres)
```

```
Mean squared error: 43316981.72
Variance score: 0.52
le  modèle  explique  près de 51.76%  de  la variabilité de la variable cible

 Variance expliquée par d'autres facteurs : 0.48
```

Mission 4 :
Régression linéaire

```
# Regression linéaire 4 v2

X = df_sans[['gdpppp_log','gini','quantile_parents']]
Y = df_sans['income_log']
X= sm.add_constant(X)
```

```
results=sm.OLS(Y,X).fit()
results.params
```

```
const              -0.456268
gdpppp_log          0.885492
gini               -0.007733
quantile_parents    0.015515
dtype: float64
```

```
regr = linear_model.LinearRegression()
regr.fit(X, Y)
Y_predits = regr.predict(X)

# mean squared error : Elle montre à quel point la droite est proche du nuage de points
print("Mean squared error: %.2f"
      % mean_squared_error(Y, Y_predits))

# Pourcentage de variance expliquée : Si 1 le modèle prédit parfaitement
print('Variance score: %.2f' % r2_score(Y, Y_predits))

var_expl_model_2 = round(results.rsquared,2)

print("le  modèle  explique  près de {}%  de  la variabilité de la variable cible".format(round(results.rsquared*100,2)))

var_expl_autres_2 = round((1-round(results.rsquared,2)),2)
print("\n Variance expliquée par d'autres facteurs :",var_expl_autres_2)  # Variance expliquée par les autres facteurs (efforts,
```

```
Mean squared error: 0.47
Variance score: 0.75
le  modèle  explique  près de 75.38%  de  la variabilité de la variable cible

 Variance expliquée par d'autres facteurs : 0.25
```

Mission 4 :
Régression linéaire

```
results = sm.OLS(Y, X).fit()
print(results.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:             income_log   R-squared:                       0.754
Model:                            OLS   Adj. R-squared:                  0.754
Method:                 Least Squares   F-statistic:                 1.164e+07
Date:                Mon, 03 Dec 2018   Prob (F-statistic):               0.00
Time:                        14:30:01   Log-Likelihood:            -1.1917e+07
No. Observations:            11400000   AIC:                         2.383e+07
Df Residuals:                11399996   BIC:                         2.383e+07
Df Model:                           3
Covariance Type:            nonrobust
==============================================================================
                     coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const             -0.4563      0.002   -261.205      0.000      -0.460      -0.453
gdpppp_log         0.8855      0.000   5333.821      0.000       0.885       0.886
gini              -0.0077   1.88e-05   -410.928      0.000      -0.008      -0.008
quantile_parents   0.0155   7.06e-06   2197.079      0.000       0.016       0.016
==============================================================================
Omnibus:                   558795.548   Durbin-Watson:                   0.005
Prob(Omnibus):                  0.000   Jarque-Bera (JB):          1244118.442
Skew:                          -0.323   Prob(JB):                         0.00
Kurtosis:                       4.484   Cond. No.                         576.
==============================================================================
```

Mission 4 :
Régression linéaire

En observant le coefficient de régression associé à l’indice de Gini, peut-on affirmer que le fait de vivre dans un pays plus inégalitaire favorise plus de personnes qu’il n’en défavorise ?

Coefficient de régression
associé à l'indice de Gini :
`-0.0077`

Merci de votre écoute