Projet 7_Prédiction de revenus

Parcours Data Analyst

Xuefei ZHANG_03/2022

Sommaire

- M1_Description de données
- M2_Analyse descriptive
- M3_Génération des données
- M4_ANOVA, Multiple linear regression

M1_Description de données

- Récupération des jeux de données appropriés
- Nettoyage de jeux de données
- Traitement de jeux de données (filtrage, jointure)
- Réponses aux questions

M1 - jeu de données1: transformation

In [2]:
```
data = pd.read_csv("data-projet7.csv")
print(data.info())
data.head()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 11599 entries, 0 to 11598
Data columns (total 6 columns):
 #  Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0  country      11599 non-null  object
 1  year_survey  11599 non-null  int64
 2  quantile     11599 non-null  int64
 3  nb_quantiles 11599 non-null  int64
 4  income       11599 non-null  object
 5  gdpppp       11399 non-null  object
dtypes: int64(3), object(3)
memory usage: 543.8+ KB
None
```

Type: object

Out[2]:

	country	year_survey	quantile	nb_quantiles	income	gdpppp
0	ALB	2008	1	100	728,89795	7297
1	ALB	2008	2	100	916,66235	7297
2	ALB	2008	3	100	1010,916	7297
3	ALB	2008	4	100	1086,9078	7297

```
 #  Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0  country      11499 non-null  object
 1  year_survey  11499 non-null  int64
 2  quantile     11499 non-null  int64
 3  nb_quantiles 11499 non-null  int64
 4  income       11499 non-null  float64
 5  gdpppp       11299 non-null  float64
```

Type: float

	country	year_survey	quantile	nb_quantiles	income	gdpppp
11598	COD	2008	100	100	2243.1226	303.19305
11595	COD	2008	97	100	911.7834	303.19305
11594	COD	2008	96	100	810.6233	303.19305
11593	COD	2008	95	100	743.3720	303.19305

M1 - jeu de données1: imputation de données manquantes

XKX, PSE ont leur 0 nombre de valeur pour colonne **gdpppp**,
soit **200 valeurs null** dans gdpppp

```
lack200.groupby('country').mean('income')
```

country	year_survey	quantile	nb_quantiles	income	gdpppp
PSE	2009.0	50.5	100.0	1114.098514	NaN
XKX	2008.0	50.5	100.0	2176.269035	NaN

imputation de data gdpppp

```
# PSE 2009 GDPPPP = 5250
data.loc[(data['country']=='PSE'),'gdpppp']=5250
```

un pays a seulement **99 individus**

40	116	116	116	116	114
41	115	115	115	115	113
42	116	116	116	116	114

```
LTU41 = data.loc[(data['quantile'].isin([40,42]))&(data['country']=='LTU')
LTU41
```

	country	year_survey	quantile	nb_quantiles	income	gdpppp
6239	LTU	2008	40	100	4868.4507	17571.0
6240	LTU	2008	42	100	4895.8306	17571.0

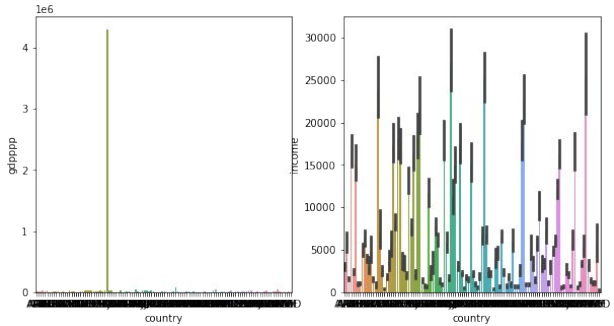
```
# append the row of quantile 41 for LTU onto the data

data = data.append(
   {'country' : 'LTU' ,
    'year_survey' : 2008,
    'quantile': 41,
    'nb_quantiles': 100,
    'income': LTU41['income'].mean(),
    'gdpppp': 17571.0,
   } , ignore_index=True)
```

M1 - jeu de données1: correction de data aberrant

il existe un individu aberrant sur gdpppp

	year_survey	quantile	nb_quantiles	income	gdpppp
count	11600.000000	11600.000000	11600.0	11600.000000	1.160000e+04
mean	2007.982759	50.500000	100.0	6069.121925	4.944408e+04
std	0.909593	28.867314	0.0	9413.786596	3.966304e+05
min	2004.000000	1.000000	100.0	16.719418	3.031931e+02
25%	2008.000000	25.750000	100.0	900.768507	2.577500e+03
50%	2008.000000	50.500000	100.0	2403.492950	7.532500e+03
75%	2008.000000	75.250000	100.0	7515.313700	1.819625e+04
max	2011.000000	100.000000	100.0	176928.550000	4.300332e+06



rectification de data gdpppp pour pays FJI

```
data.loc[(data['country']=='FJI'),'gdpppp']= 4300
data.describe()
```

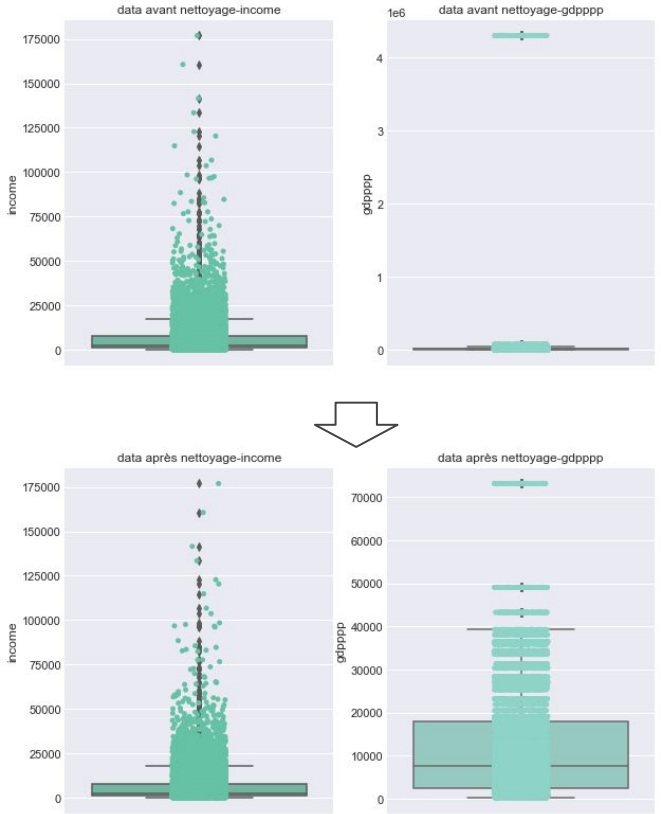
	year_survey	quantile	nb_quantiles	income	gdpppp
count	11600.000000	11600.000000	11600.0	11600.000000	11600.000000
mean	2007.982759	50.500000	100.0	6069.121925	12409.323437
std	0.909593	28.867314	0.0	9413.786596	13108.901817
min	2004.000000	1.000000	100.0	16.719418	303.193050
25%	2008.000000	25.750000	100.0	900.768507	2577.500000
50%	2008.000000	50.500000	100.0	2403.492950	7488.500000
75%	2008.000000	75.250000	100.0	7515.313700	17679.250000
max	2011.000000	100.000000	100.0	176928.550000	73127.000000

M1 - rendu du jeu de données1

```
countries = data.groupby(by=
'country').mean('income').sort_values('income')
```

```
 0   country       116 non-null    object
 1   year_survey   116 non-null    float64
 2   quantile      116 non-null    float64
 3   nb_quantiles  116 non-null    float64
 4   income        116 non-null    float64
 5   gdpppp        116 non-null    float64
dtypes: float64(5), object(1)
memory usage: 5.6+ KB
None
```

	country	year_survey	quantile	nb_quantiles	income	gdpppp
0	COD	2008.0	50.5	100.0	276.016044	303.193050
1	MDG	2010.0	50.5	100.0	345.237074	950.000000
2	CIV	2008.0	50.5	100.0	399.835204	1526.000000



M1 - nettoyage et traitement du jeu de données2

```
Worldbank = pd.read_csv("worldbankdata5.csv")          # .dropna(how= "any", axis=0)
Worldbank= Worldbank.drop(columns= ["Series Code"])
print(Worldbank.info())
Worldbank.head()
```

```
 #  Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0  Country Name    800 non-null    object
 1  Country Code    798 non-null    object
 2  Series Name     798 non-null    object
 3  2008 [YR2008]   798 non-null    object
dtypes: object(4)
memory usage: 25.2+ KB
None
```

	Country Name	Country Code	Series Name	2008 [YR2008]
0	Afghanistan	AFG	GDP (constant 2015 US$)	11060389403.7128
1	Afghanistan	AFG	GDP per capita (constant 2015 US$)	415.086995378652
2	Afghanistan	AFG	Population, total	27722281
3	Albania	ALB	GDP (constant 2015 US$)	9721650518.69391

- row 798- 802 have no sense, to be removed
- the 651-797 rows is subtotal for preceeding rows, to be removed

```
Worldbank  = Worldbank.drop(Worldbank.index[651:802], axis=0)
Worldbank['2008'] = pd.to_numeric(Worldbank['2008'])        #convert ['2008'] data to numeric
Worldbank.info()
Worldbank.tail(100)
```

```
 0  Country Name    652 non-null    object
 1  Country Code    651 non-null    object
 2  Series Name     651 non-null    object
 3  2008 [YR2008]   651 non-null    object
 4  2008            621 non-null    float64
dtypes: float64(1), object(4)
memory usage: 30.6+ KB
```

	Country Name	Country Code	Series Name	2008 [YR2008]	2008
552	St. Vincent and the Grenadines	VCT	GDP (constant 2015 US$)	757175251.215559	7.571753e+08
553	St. Vincent and the Grenadines	VCT	GDP per capita (constant 2015 US$)	6984.98753753832	6.984000e+03
554	St. Vincent and the Grenadines	VCT	Population, total	108401	1.084010e+05

M1 - rendu du jeu de données 2

```
pop= Worldbank[Worldbank["Series Name"] == "Population, total"][['Country Code', '2008']]
pop['2008'] = pop['2008'].astype(int)
pop.info()
pop.head()
#653-797 rows are subtotal data which includes already data by country
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 217 entries, 2 to 650
Data columns (total 2 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   Country Code  217 non-null    object
 1   2008          217 non-null    int64
dtypes: int64(1), object(1)
memory usage: 5.1+ KB
```

	Country Code	2008
2	AFG	27722281
5	ALB	2947314
8	DZA	34730604
11	ASM	57490
14	AND	83860

M1 - jointure de 2 dataframes

```
dfmerged = pd.merge(countries[['country', 'income', 'gdpppp']], pop[['Country Code','2008']],
                    left_on='country', right_on = 'Country Code')
dfmerged.drop('Country Code', axis=1)
dfmerged.dropna(how= 'all', axis=1)
print(dfmerged.info())
dfmerged.head()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 115 entries, 0 to 114
Data columns (total 5 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   country       115 non-null    object
 1   income        115 non-null    float64
 2   gdpppp        115 non-null    float64
 3   Country Code  115 non-null    object
 4   2008          115 non-null    int64
dtypes: float64(2), int64(1), object(2)
memory usage: 5.4+ KB
None
```

	country	income	gdpppp	Country Code	2008
0	COD	276.016044	303.19305	COD	60411195
1	MDG	345.237074	950.00000	MDG	19996476
2	CIV	399.835204	1526.00000	CIV	19605568

- Les années des données utilisées sont: 2004, 2006, 2007, 2008, 2009, 2010, 2011
- Le nombre de pays couverts par cette banque: **116**
- La population totale des pays où habitent les clients actuels est: 6180824944
- Ces pays clientèles occupent % de la population mondiale de l'année 2008: **91.79 %**

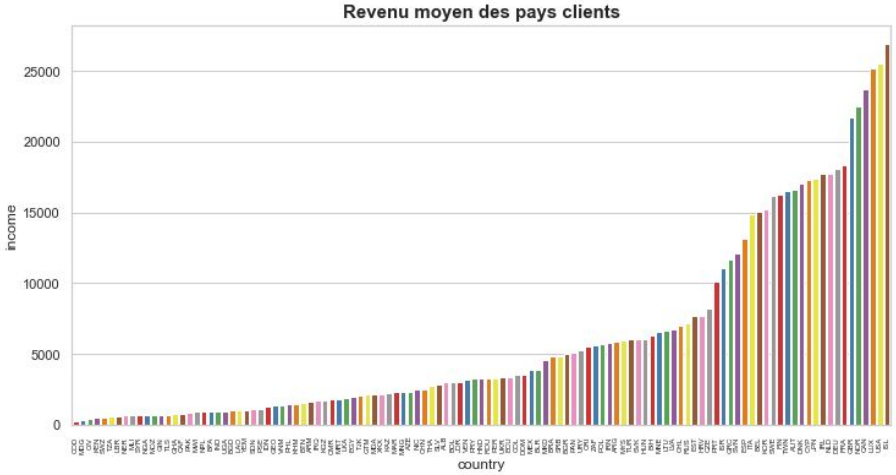
- **De quel type de quantiles s'agit-il ?** centile
- **Échantillonner une population en utilisant des quantiles est-il une bonne méthode ? Pourquoi ?**

oui et non. ça dépend du nombre de quantiles et l'approche employée pour le clivage, autrement dit on découpe la population en combien de morceaux et comment on définit les méthodes ou criteria de la découpe. Litérallement dit, le plus le nombre de quantiles pour une population, le plus on est précis et les échantillons sont représentatifs.

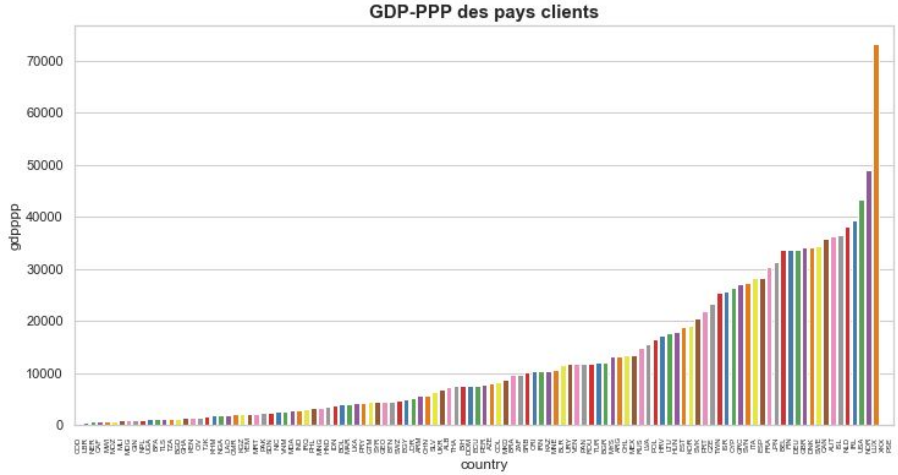
M2_Analyse descriptive

- Montrez la diversité des pays en termes de distribution de revenus à l'aide d'un graphique.
- Représentez la courbe de Lorenz de chacun des pays choisis.
- Pour chacun de ces pays choisis, représentez l'évolution de l'indice de Gini *au fil des ans*.
- Classez ces pays clientèle par indice de Gini de l'année 2008.

M2_diversité en revenu

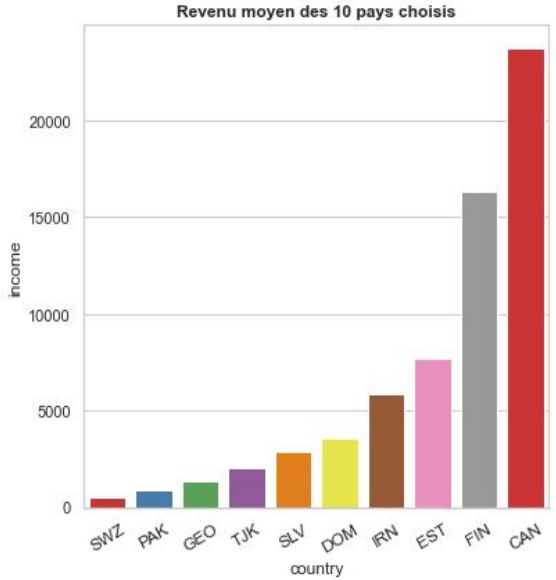


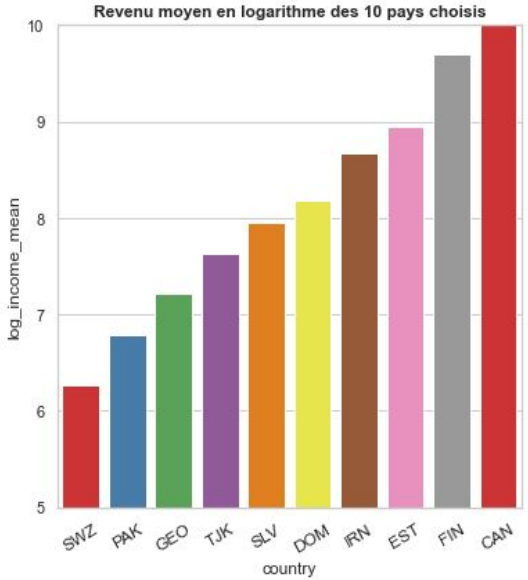
Revenu moyen:
Il existe de grands écarts entre les pays de revenu bas et de revenu élevé.

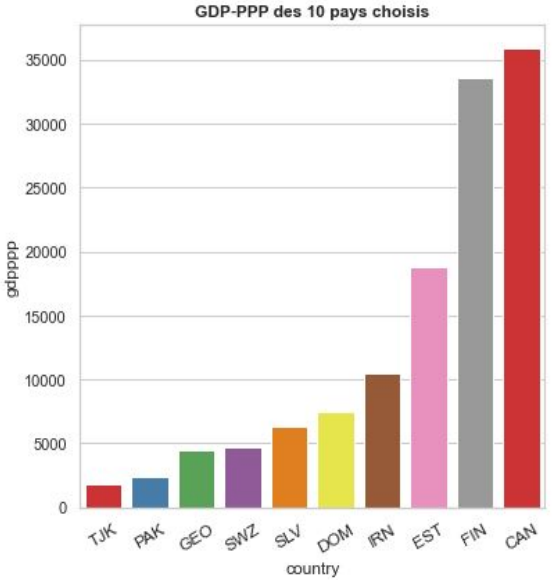


GDP-PPP:
Par rapport au revenu moyen, il n'existe pas autant d'écart entre les pays de gdpppp bas et de gdpppp élevé.

M2_diversité en revenu_10 pays







En comparant graphique 1 & 2, on constate que pour ces 10 pays, la différence entre les logarithmes de revenu moyen est plus claire et facile à mesurer.

Car logarithme transforme les valeurs **exponentielle** en **additionnelle.**

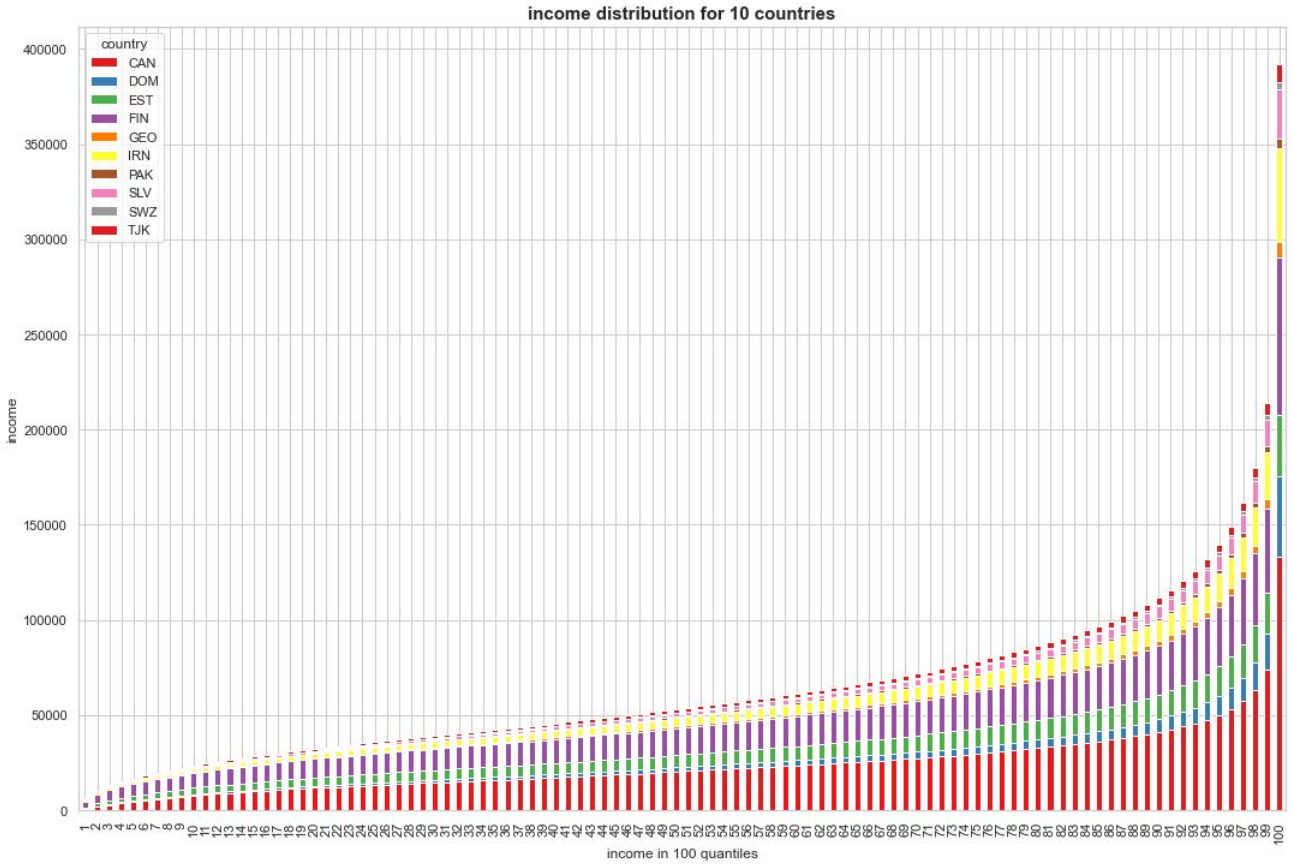
GDP-PPP n'accroît pas d'une même manière que le revenu moyen, mais plutôt proche d'**additionnelle**.

Autrement dit **malgré la différence exponentielle entre les revenus moyens, les GDP-PPP des pays ne se différencient pas à la même échelle.**

M2_distribution de revenu quantile par quantile_10 pays

EST, FIN:
il n'y a pas grand écart parmi les classes de revenu

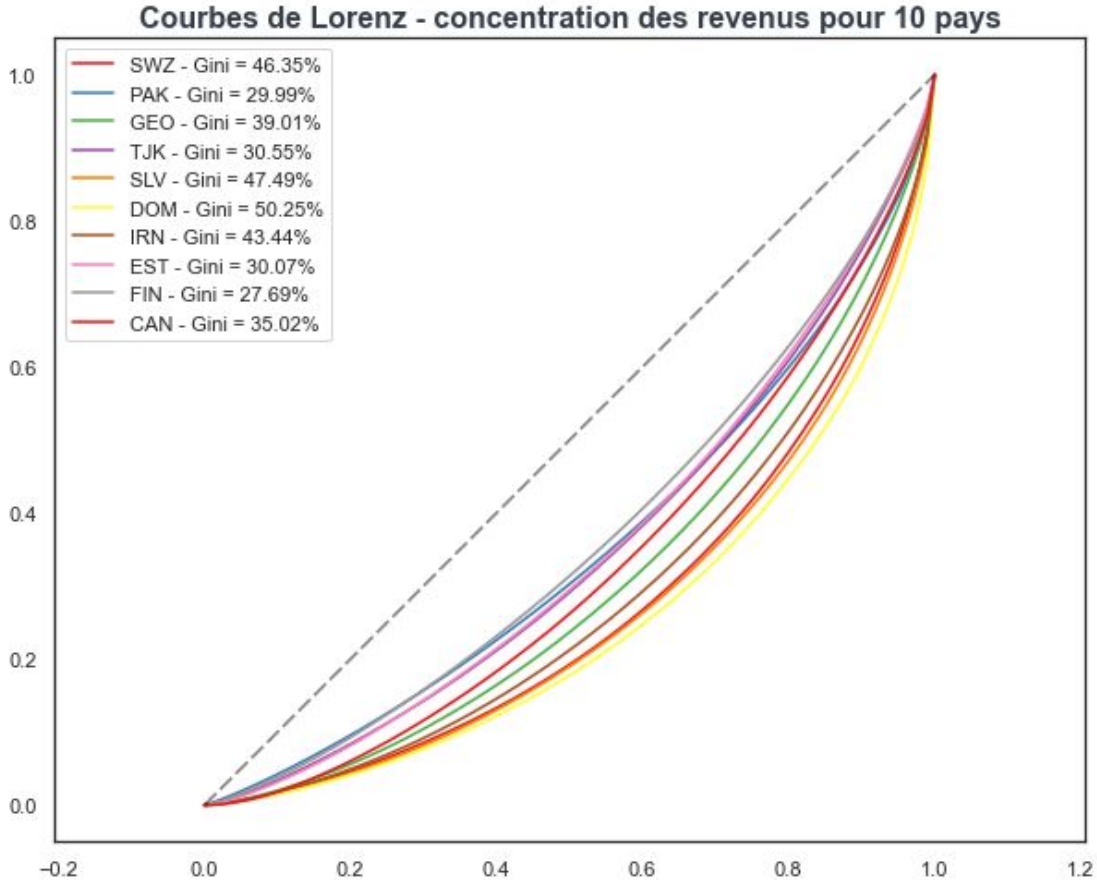
DOM, IRN, SLV:
grand écart parmi les classes de revenu => polarisation entre les plus riches et les plus pauvres



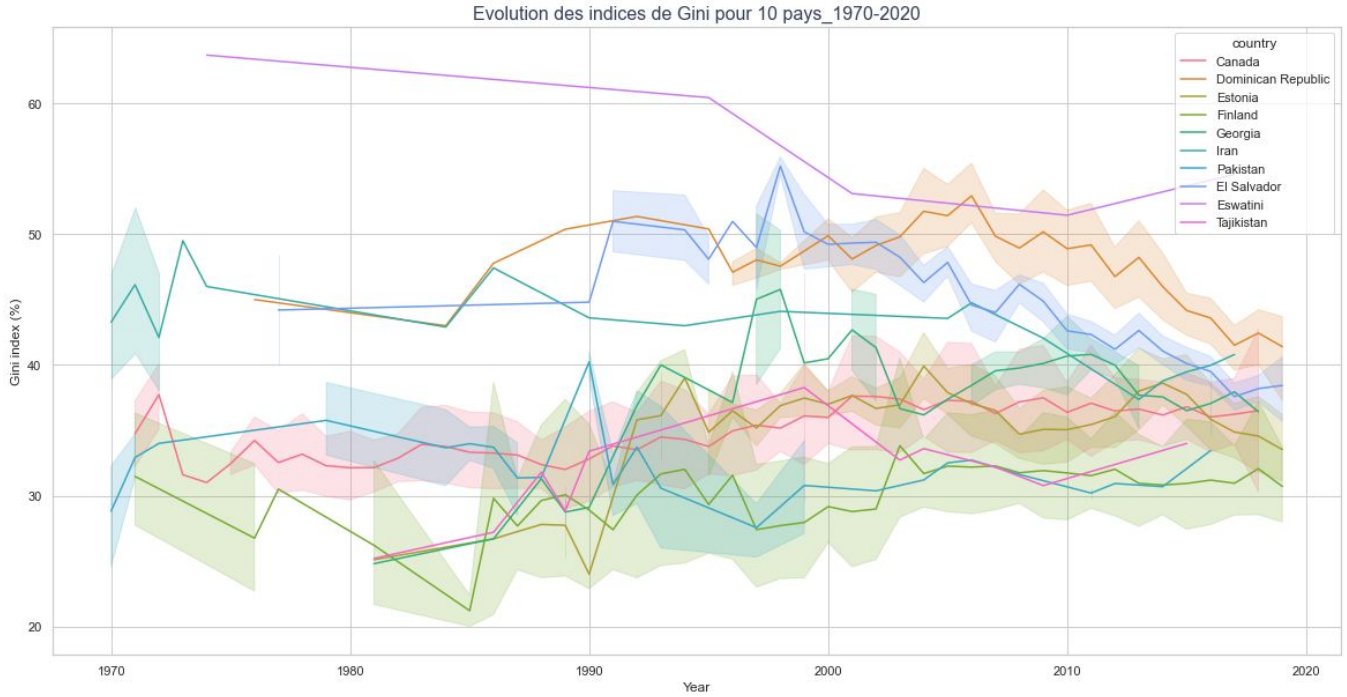
M2_Courbes de Lorenz_10 pays

DOM, IRN, SLV:
ont indices de Gini élévé => inégalité remarquable sur les revenus

FIN, EST:
Ont leur indices Gini les plus bas => sociétés relativement égalitaires

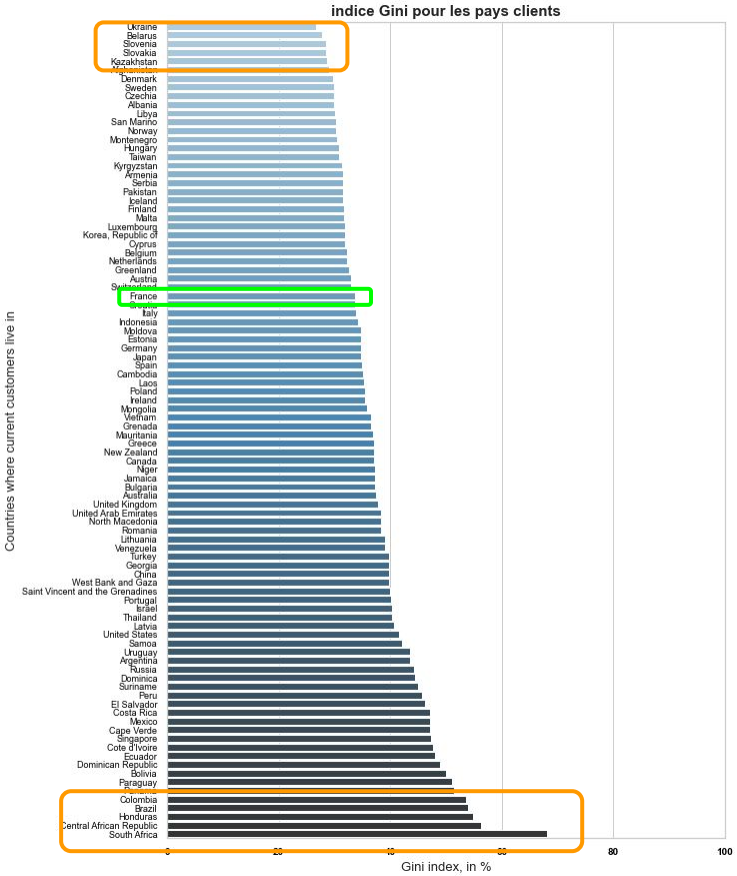


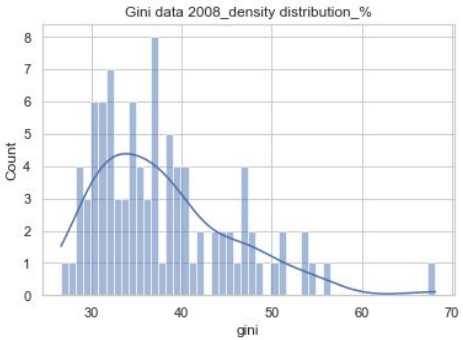
M2_Evolution des indices de Gini_1970-2020

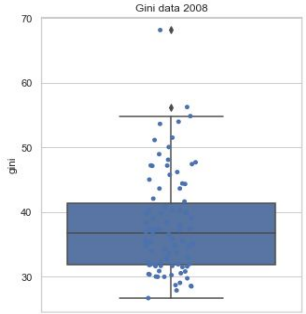


- Eswatini: Gini a diminué fortement en 1990-2000 et atteint son niveau bas en 2010
- El Salvador: Gini connait une décroissance depuis 2000 (50%) vers 40% en 2020
- Dominica: Gini connait une forte décroissance depuis 2005 (52%) vers 42% en 2020
- Canada: Gini reste stable autour de 35%
- Finland: stable autour de 30%

M2_indice Gini en 2008







les indices Gini des pays clients se répartissent majoritairement entre **30% et 40%**, avec médiane **36.7%**.

Où se trouve la France?

Gini France = **33.66%**

- entre 1er et 2ème quantile des indices Gini pour ces pays clients
- plus petit que la moyenne (37.98%)

=> La France a sa distribution de richesse relativement égalitaire

M3_Génération des données d’individus via fonctions

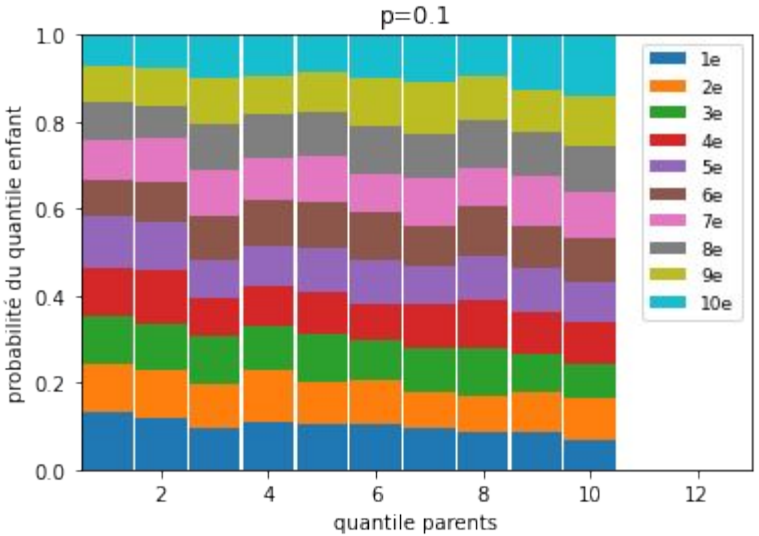
Maintenant qu'on a à disposition pour chaque pays, data sur le revenu et l’indice de Gini (2 variables explicatives) , donc il nous manque **la classe de revenu des parents** (3ème variable explicative).

Enjeu de cette mission :

À travers les fonctions, sortir la **distribution de probabilité** de chaque **classe revenu des parents** correspondante basé sur chaque classe de revenu des enfants et coefficient d'élasticité (pj)

M3_tests

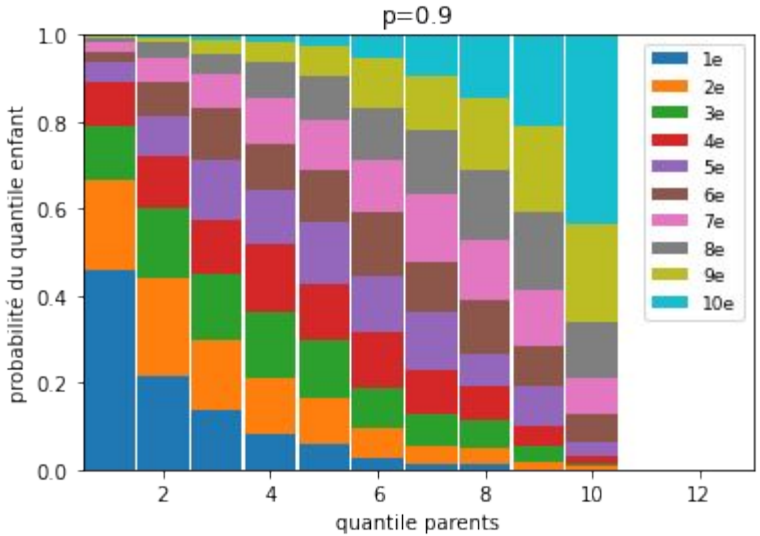
forte mobilité intergénérationnelle



Coeff d'élasticité =0.1:
faible corrélation entre revenus enfant & revenus parents.

Le status-quo des revenus parentaux a **très peu d'influence** sur les revenus de leur enfant,s.

faible mobilité intergénérationnelle



Coeff d'élasticité =0.9:
Forte corrélation entre revenus enfant & revenus parents.

Le status-quo des revenus parentaux **a de grands poids** sur les revenus de leur enfant,s.

M3_test_Matrice de distribution conditionnelle des probabilités

NB: pj=0.1

classe revenu parent

quantile_enfant	1	2	3	4	5	6	7	8	9	10
1	0.132	0.117	0.094	0.110	0.104	0.105	0.095	0.087	0.087	0.069
2	0.113	0.113	0.101	0.119	0.097	0.102	0.084	0.081	0.093	0.097
3	0.107	0.105	0.110	0.099	0.110	0.093	0.102	0.110	0.088	0.076
4	0.111	0.125	0.088	0.095	0.098	0.083	0.098	0.110	0.096	0.096
5	0.120	0.110	0.091	0.092	0.099	0.101	0.091	0.103	0.098	0.095
6	0.085	0.091	0.098	0.107	0.108	0.110	0.092	0.113	0.098	0.098
7	0.089	0.100	0.107	0.094	0.107	0.084	0.109	0.091	0.113	0.106
8	0.086	0.076	0.106	0.103	0.101	0.113	0.100	0.108	0.102	0.105
9	0.086	0.086	0.107	0.084	0.092	0.111	0.118	0.101	0.098	0.117
10	0.071	0.077	0.098	0.097	0.084	0.098	0.111	0.096	0.127	0.141

classe revenu enfant

ici l'échantillon est divisée en 10 classes de revenu, donc on a **10*10** estimations de ces probabilités conditionnelles, pour chaque pays

M3_constitution de dataframe_les 3 dataframes

```
#Q8 : dataframe dataWID
dataWID = pd.read_csv('data_cleaned_WID.csv')
dataWID= dataWID.drop(columns=['Unnamed: 0'])
dataWID.describe()
print(dataWID.info())
dataWID.head()
# data_cleaned_WID is the cleaned dataframe 'world_income_distribution.csv' for
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 11499 entries, 0 to 11498
Data columns (total 6 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   country      11499 non-null  object
 1   year_survey  11499 non-null  int64
 2   quantile     11499 non-null  int64
 3   nb_quantiles 11499 non-null  int64
 4   income       11499 non-null  float64
 5   gdpppp       11299 non-null  float64
dtypes: float64(2), int64(3), object(1)
memory usage: 539.1+ KB
None
```

	country	year_survey	quantile	nb_quantiles	income	gdpppp
0	COD	2008	100	100	2243.1226	303.19305
1	COD	2008	97	100	911.7834	303.19305
2	COD	2008	96	100	810.6233	303.19305
3	COD	2008	95	100	743.3720	303.19305
4	COD	2008	94	100	684.7071	303.19305

```
# Q8_dataframe Coeff_coeffcient d'électricité
coeff= pd.read_csv('coeff_pj.csv')
Coeff= coeff.drop(columns=['Unnamed: 0','year'])
print(Coeff.info())
Coeff
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 75 entries, 0 to 74
Data columns (total 2 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   wbcode     75 non-null     object
 1   IGEincome  75 non-null     float64
dtypes: float64(1), object(1)
memory usage: 1.3+ KB
None
```

	wbcode	IGEincome
0	ALB	0.815874
1	AUS	0.275000
2	AUT	0.245267
3	BEL	0.183176
4	BEN	0.855116

IGEincome: le coeff d'élasticité, pj

```
# Q8_dataframe ginidata
ginidata= pd.read_csv('ginidata.csv', index_col=None
print(ginidata.describe())
ginidata_mean=ginidata[['c3','gini']].groupby('c3').mea
ginidata_mean.head()
```

	id	year	gini
count	20592.000000	20592.000000	20592.000000
mean	10630.988782	2001.319542	37.281548
std	6145.293483	14.517767	9.416913
min	1.000000	1867.000000	12.100000
25%	5293.750000	1995.000000	30.300000
50%	10639.500000	2005.000000	35.570000
75%	15971.250000	2012.000000	43.462500
max	21253.000000	2020.000000	78.600000

	c3	gini
0	AFG	31.000000
1	AGO	47.095000
2	ALB	31.408889
3	AND	27.097500
4	ARE	35.405000

M3_constitution de dataframe & cloner 500 fois

```
0  gini          6300 non-null  float64
1  IGEincome     6300 non-null  float64
2  country       11600 non-null  object
3  quantile      11600 non-null  int64
4  nb_quantiles  11600 non-null  int64
5  income        11600 non-null  float64
6  gdpppp        11600 non-null  float64
dtypes: float64(4), int64(2), object(1)
memory usage: 725.0+ KB
None
```

pj

Classe revenu enfant

	gini	IGEincome	country	quantile	nb_quantiles	income	gdpppp
0	31.408889	0.815874	ALB	1	100	728.89795	7297.0
1	31.408889	0.815874	ALB	2	100	916.66235	7297.0
2	31.408889	0.815874	ALB	3	100	1010.91600	7297.0
3	31.408889	0.815874	ALB	4	100	1086.90780	7297.0

revenu enfant

Classe revenu parent (probabilité)

Cloner tous les individus par 500 fois:

Co500 = pd.concat([Co]*500, ignore_index=True)

Co500=Co500.rename({'quantile': 'quantile_child'}, axis=1)

gini	IGEincome	country	quantile_child	nb_quantiles	income	gdpppp
31.408889	0.815874	ALB	1	100	728.89795	7297.0
31.408889	0.815874	ALB	1	100	728.89795	7297.0
31.408889	0.815874	ALB	1	100	728.89795	7297.0

Maintenant qu'on a 500 individus pour chaque quantile_child de chaque pays,
donc on a **100*500 = 50000** individus pour chaque pays

M3_calculus de probabilité de distribution_exemple Albania

```
ALB = Co500[Co500['country']=='ALB']
print(ALB.info())
ALB.head()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 50000 entries, 2273 to 3145921
Data columns (total 7 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   gini            50000 non-null  float64
 1   IGEincome       50000 non-null  float64
 2   country         50000 non-null  object
 3   quantile_child  50000 non-null  int64
 4   nb_quantiles    50000 non-null  int64
 5   income          50000 non-null  float64
 6   gdpppp          50000 non-null  float64
dtypes: float64(4), int64(2), object(1)
memory usage: 3.1+ MB
None
```

	gini	IGEincome	country	quantile_child	nb_quantiles	income	gdpppp
2273	31.408889	0.815874	ALB	1	100	728.89795	7297.0
8573	31.408889	0.815874	ALB	1	100	728.89795	7297.0
14873	31.408889	0.815874	ALB	1	100	728.89795	7297.0

```
ALB_proba=[]

pj=Co500.loc[Co500['country']=='ALB','IGEincome'].iloc[0]
nb_quantiles = 100
n = 500*nb_quantiles

y_child, y_parents = generate_incomes(n, pj)
sample = compute_quantiles(y_child, y_parents, nb_quantiles)
cd = conditional_distributions(sample, nb_quantiles)
for c_i_child in range(100):
    for c_i_parent in range(100):
        p = proba_cond(c_i_parent, c_i_child, cd)
        ALB_proba.extend([c_i_parent+1]*(int(p*500)))
```

```
len(ALB_proba)
```

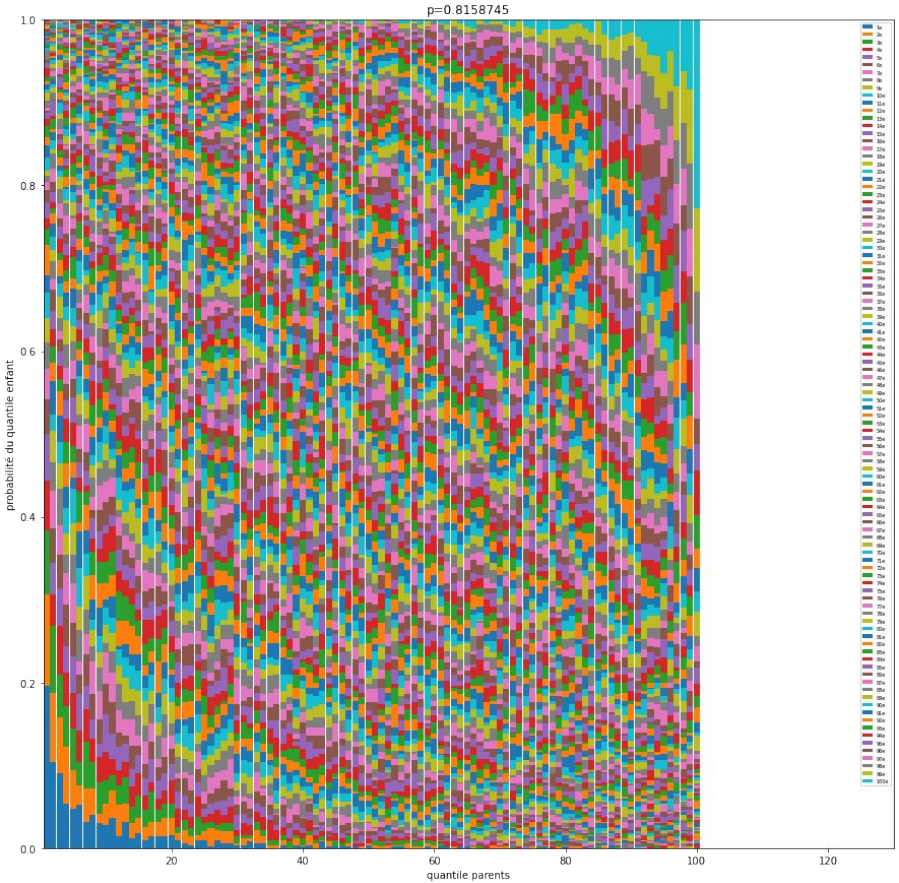
```
50000
```

```
ALB['proba_quantile_parent']=ALB_proba
```

country	quantile_child	nb_quantiles	income	gdpppp	proba_quantile_parent
ALB	1	100	728.89795	7297.0	1
ALB	1	100	728.89795	7297.0	1
ALB	1	100	728.89795	7297.0	1

Vu le volume énorme de jeu de données Co500, afin de faciliter la vitesse de calcul, on prend ALB pour exemple illustratif

M3_illustration de probabilité de distribution_exemple Albania



Albania connaît une **faible** mobilité intergénérationnelle en terme de revenu.

M4_ANOVA, Multiple Linear Regression

- **One-way ANOVA**
- **Multiple-linear regression (MLR):**
  - 2 variables explicatives
  - 2 variables explicatives en logarithme
  - 3 variables explicatives
  - 3 variables explicatives en logarithme

Terminologie

1. ANOVA: Analysis of variance
2. One-way ANOVA: ANOVA à un facteur
3. Multiple-linear regression: Régression linéaire avec de multiples variables explicatives

M4_One-way ANOVA

```
Q1_ANOVA=Co500.groupby(['country','quantile_child']).mean().reset_index()
Q1_ANOVA.head()
```

	country	quantile_child	gini	IGEincome	nb_quantiles	income	gdpppp
0	ALB	1	31.408889	0.815875	100.0	728.89795	7297.0
1	ALB	2	31.408889	0.815875	100.0	916.66235	7297.0
2	ALB	3	31.408889	0.815875	100.0	1010.91600	7297.0
3	ALB	4	31.408889	0.815875	100.0	1086.90780	7297.0
4	ALB	5	31.408889	0.815875	100.0	1132.69970	7297.0

```
anova_Q1 = smf.ols('income~country', data=Q1_ANOVA).fit()
anova_Q1.summary()

# why income~country ?  income is the dependent variable Y, country is the X
```

OLS Regression Results

Dep. Variable:	income	**R-squared:**	0.456
Model:	OLS	**Adj. R-squared:**	0.451
Method:	Least Squares	**F-statistic:**	84.42
Date:	Fri, 25 Feb 2022	**Prob (F-statistic):**	0.00
Time:	13:25:26	**Log-Likelihood:**	-65714.
No. Observations:	6300	**AIC:**	1.316e+05
Df Residuals:	6237	**BIC:**	1.320e+05
Df Model:	62		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	2994.8299	824.141	3.634	0.000	1379.229	4610.431

Step2: R-squared

R-squared = 0.45
=> la variance expliquée par ce modèle (variable explicative: country) est **45%**

Step1: test de validité d'hypothèse

H0: le pays (country) n'a pas d'impact sur les revenus (income).

F-statistic est petite et **la Prob(F-statistic)** est 0.00, inférieure à 0.5%.
=> Rejet de l'hypothèse H0 à niveau de test 0.5%,
ctd.le pays a de l'impact sur les revenus.

M4_MLR_traitement de dataframe

Income_normal:
revenu quantile par quantile

Income_mean:
revenu de tous les quantiles de chaque pays

	gini	IGEincome	country	quantile_child	nb_quantiles	income_normal	gdpppp	log_gdpppp	proba_quantile_parent	income_mean
0	31.408889	0.815874	ALB	1	100	728.89795	7297.0	8.895219	1	2994.829902
1	31.408889	0.815874	ALB	1	100	728.89795	7297.0	8.895219	1	2994.829902
2	31.408889	0.815874	ALB	1	100	728.89795	7297.0	8.895219	1	2994.829902
3	31.408889	0.815874	ALB	1	100	728.89795	7297.0	8.895219	1	2994.829902
4	31.408889	0.815874	ALB	1	100	728.89795	7297.0	8.895219	1	2994.829902

Créer les colonnes de revenu en logarithme

```
Linear_reg_log=Linear_reg.copy()
Linear_reg_log['log_income']=np.log(Linear_reg_log['income_normal'])
Linear_reg_log['log_income_mean']=np.log(Linear_reg_log['income_mean'])
Linear_reg_log.head()
```

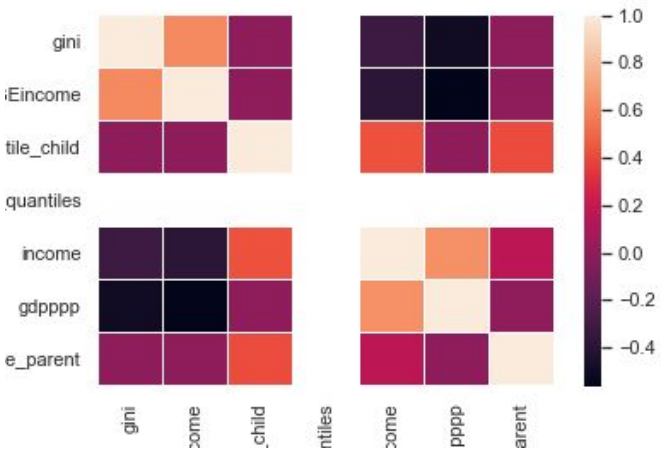
ini	IGEincome	country	quantile_child	nb_quantiles	income_normal	gdpppp	log_gdpppp	proba_quantile_parent	income_mean	log_income	log_income_mean
89	0.815874	ALB	1	100	728.89795	7297.0	8.895219	1	2994.829902	6.591534	8.004643
89	0.815874	ALB	1	100	728.89795	7297.0	8.895219	1	2994.829902	6.591534	8.004643
89	0.815874	ALB	1	100	728.89795	7297.0	8.895219	1	2994.829902	6.591534	8.004643
89	0.815874	ALB	1	100	728.89795	7297.0	8.895219	1	2994.829902	6.591534	8.004643
89	0.815874	ALB	1	100	728.89795	7297.0	8.895219	1	2994.829902	6.591534	8.004643

M4_vérification de conditions de validité pour régression linéaire

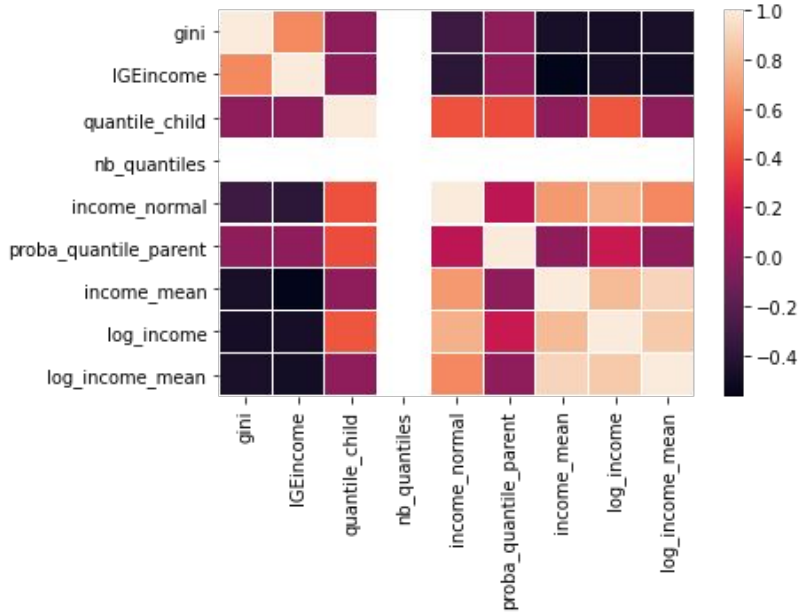
- il n'existe pas de grande **colinéarité** parmi les variables explicatives

- **normalité** des résidus
- **homosétasticité** des résidus: ctd. les résidus ont la même variance quel que soit le groupe (pays) considéré, ou quelle que soit la valeur de la variable explicative considérée

M4_vérification de colinéarité



forte colinéarité entre gdpppp et gini, gdpppp et income,
ainsi on enlève gdpppp



il n'y a pas de grande colinéarité entre les variables explicatives
qu'on va utiliser dans les modèles, soit:
gini, income_mean, log_income_mean, proba_quantile_parent

M4_MLR_2 variables

```
reg_multi = smf.ols('income_normal~income_mean+gini',
data=Linear_reg).fit()
```

Dep. Variable:	income_normal	**R-squared:**	0.456
Model:	OLS	**Adj. R-squared:**	0.456
Method:	Least Squares	**F-statistic:**	1.322e+06
Date:	Thu, 24 Feb 2022	**Prob (F-statistic):**	0.00
Time:	17:41:54	**Log-Likelihood:**	-3.2857e+07
No. Observations:	3150000	**AIC:**	6.571e+07
Df Residuals:	3149997	**BIC:**	6.571e+07
Df Model:	2		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	-9.976e-09	27.758	-3.59e-10	1.000	-54.406	54.406
income_mean	1.0000	0.001	1439.091	0.000	0.999	1.001
gini	7.773e-11	0.631	1.23e-10	1.000	-1.237	1.237

Omnibus:	3495810.153	**Durbin-Watson:**	0.001
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	499939554.028
Skew:	5.574	**Prob(JB):**	0.00
Kurtosis:	63.703	**Cond. No.**	6.75e+04

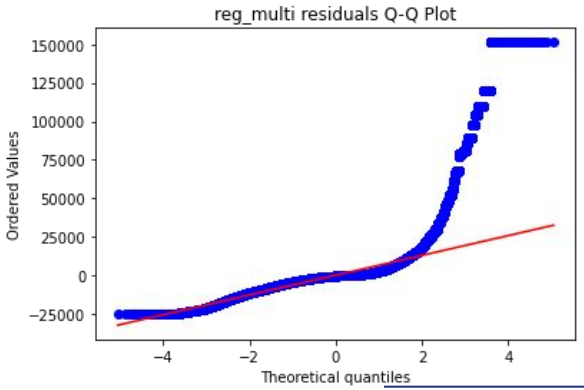
Variables explicatives:

- Income_mean
- gini

Le modèle arrive à expliquer **45.6%** de variance sur income_normal

- P-value income_mean = 0 : **rejet** de H0, income_mean joue un impact sur income_normal (revenu quantile par quantile)
- P-value gini = 1 : **Non-rejet** de H0, Gini n'a pas d'impact sur income_normal

M4_2 variables_normalité et homosétasticité des résidus



Grande superposition de **bleue** avec ligne rouge
=> condition normalité est valide.

```
analyse1=sm.stats.anova_lm(reg_multi, typ=1)
```

	df	sum_sq	mean_sq	F	PR(>F)
income_mean	1.0	1.777465e+14	1.777465e+14	2.643394e+06	0.0
gini	1.0	5.456975e-13	5.456975e-13	8.115455e-21	1.0
Residual	3149997.0	2.118114e+14	6.724177e+07	NaN	NaN

Décomposition

- la variance expliquée par le **revenu moyen** est environ **45.6%**
- **quasi zéro** par indice de **Gini**
- par les autres facteurs non-incluses dans ce modèle est **54.4%**

```
name = ['Lagrange multiplier statistic', 'p-value', 'f-value', 'f p-value']

test = sms.het_breuschpagan(reg_multi.resid, reg_multi.model.exog)
lzip(name, test)
```

```
[('Lagrange multiplier statistic', 81352.84092300001),
 ('p-value', 0.0),
 ('f-value', 41754.75243071066),
 ('f p-value', 0.0)]
```

H0: homoscétasticité pour les résidus

P-value = 0.0, <0.5%, rejet de H0 au niveau de test 0.5%, présence de l'hétérosétasticité pour les résidus de ce modèle reg_multi

M4_2 variables_en log

```
reg_multi = smf.ols('income_normal~log_income_mean+gini',
data=Linear_reg_log).fit()
```

Dep. Variable:	log_income	**R-squared:**	0.754
Model:	OLS	**Adj. R-squared:**	0.754
Method:	Least Squares	**F-statistic:**	4.834e+06
Date:	Thu, 24 Feb 2022	**Prob (F-statistic):**	0.00
Time:	17:42:13	**Log-Likelihood:**	-3.4190e+06
No. Observations:	3150000	**AIC:**	6.838e+06
Df Residuals:	3149997	**BIC:**	6.838e+06
Df Model:	2		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	0.6782	0.005	149.201	0.000	0.669	0.687
log_income_mean	0.9707	0.000	2597.477	0.000	0.970	0.971
gini	-0.0180	5.49e-05	-326.925	0.000	-0.018	-0.018

Omnibus:	320818.887	**Durbin-Watson:**	0.001
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	1914508.355
Skew:	-0.308	**Prob(JB):**	0.00
Kurtosis:	6.769	**Cond. No.**	452.

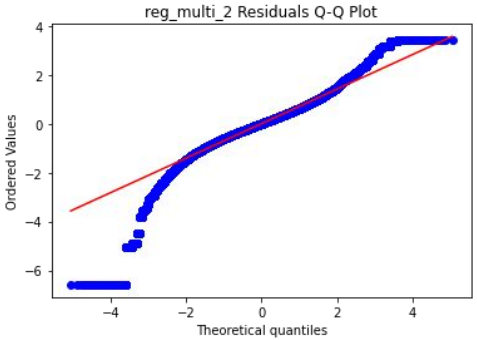
Variables explicatives:

- log_income_mean
- gini

Le modèle arrive à expliquer **75.4%** de variance sur income_normal

- P-value log_income_mean = 0 : **rejet** de H0, log_income_mean arrive à expliquer partiellement income_normal (revenu quantile par quantile)
- P-value gini = 0 : **rejet** de H0, Gini a l'impact sur income_normal

M4_2 variables log_normalité et homosétasticité des résidus



Grande superposition de **bleue** avec ligne rouge => condition normalité est valide.

```
analyse_2=sm.stats.anova_lm(reg_multi_2, typ=1)
analyse_2
```

	df	sum_sq	mean_sq	F	PR(>F)
log_income_mean	1.0	4.907014e+06	4.907014e+06	9.561381e+06	0.0
gini	1.0	5.485212e+04	5.485212e+04	1.068801e+05	0.0
Residual	3149997.0	1.616616e+06	5.132119e-01	NaN	NaN

Décomposition

- la variance expliquée par le revenu moyen en log est environ **74.6%**;
- **0.8 %** par indice de Gini;
- par les autres facteurs non-incluses dans ce modèle est **24.6%**

```
name = ['Lagrange multiplier statistic', 'p-value', 'f-value', 'f p-value']
test = sms.het_breuschpagan(reg_multi_2.resid, reg_multi_2.model.exog)
lzip(name, test)
```

```
[('Lagrange multiplier statistic', 145979.22385892624),
 ('p-value', 0.0),
 ('f-value', 76536.44090448711),
 ('f p-value', 0.0)]
```

P-value = 0.0, <0.5%, rejet de H0 au niveau de test 0.5%,
il existe de l'hétérosétasticité pour les résidus de ce modèle reg_multi_2

M4_MLR_3 variables

```
reg_multi_3 = smf.ols('income_normal~income_mean+gini+proba_quantile_parent', data=Linear_reg).fit()
```

Variables explicatives:

- Income_mean
- Gini
- proba_quantile_parent

Dep. Variable:	income_normal	**R-squared:**	0.484
Model:	OLS	**Adj. R-squared:**	0.484
Method:	Least Squares	**F-statistic:**	9.860e+05
Date:	Thu, 24 Feb 2022	**Prob (F-statistic):**	0.00
Time:	17:42:20	**Log-Likelihood:**	-3.2774e+07
No. Observations:	3150000	**AIC:**	6.555e+07
Df Residuals:	3149996	**BIC:**	6.555e+07
Df Model:	3		
Covariance Type:	nonrobust		

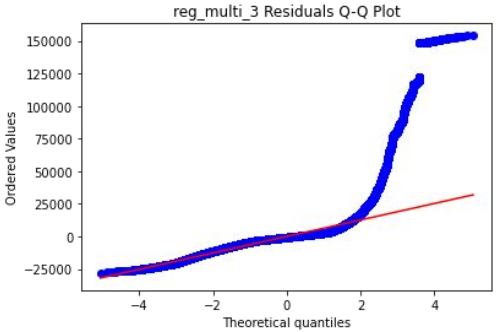
Le modèle arrive à expliquer **48.4%** de variance sur income_normal

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	-3256.1472	28.157	-115.643	0.000	-3311.334	-3200.961
income_mean	1.0000	0.001	1477.657	0.000	0.999	1.001
gini	7.816e-11	0.615	1.27e-10	1.000	-1.204	1.204
proba_quantile_parent	64.4782	0.156	413.639	0.000	64.173	64.784

Omnibus:	3536698.366	**Durbin-Watson:**	0.003
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	539955478.631
Skew:	5.667	**Prob(JB):**	0.00
Kurtosis:	66.131	**Cond. No.**	7.03e+04

- P-value income_mean = 0 : **rejet** de H0, income_mean joue un impact sur income_normal (revenu quantile par quantile)
- P-value gini = 1 : **Non-rejet** de H0, Gini n'a pas d'impact sur income_normal
- P-value proba_quantile_parent = 0: **rejet** de H0, proba_quantile_parent joue un impact sur income_normal (revenu quantile par quantile)

M4_3 variables_normalité et homosétasticité des résidus



Grande superposition de **bleue** avec ligne rouge
=> condition normalité est valide.

```
analyse3=sm.stats.anova_lm(reg_multi_3, typ=1)
analyse3
```

	df	sum_sq	mean_sq	F	PR(>F)
income_mean	1.0	1.777465e+14	1.777465e+14	2.787283e+06	0.0
gini	1.0	5.456975e-13	5.456975e-13	8.557208e-21	1.0
proba_quantile_parent	1.0	1.093450e+13	1.093450e+13	1.714664e+05	0.0
Residual	3149996.0	2.008769e+14	6.377051e+07	NaN	NaN

```
name = ['Lagrange multiplier statistic', 'p-value', 'f-value', 'f p-value']
test = sms.het_breuschpagan(reg_multi_3.resid, reg_multi_3.model.exog)
lzip(name, test)
```

```
[('Lagrange multiplier statistic', 90355.6983633005),
 ('p-value', 0.0),
 ('f-value', 31007.971337207444),
 ('f p-value', 0.0)]
```

P-value = 0.0, <0.5%, rejet de H0 au niveau de test 0.5%,
donc présence de l'hétérosétasticité pour les résidus de ce modèle reg_multi_3

Décomposition

La variance expliquée par la classe de revenu des parents est de : **2.8 %**
La variance expliquée par le revenu moyen du pays est de : **45.6 %**
La variance expliquée par Gini du pays est presque **0**.
La variance expliquée par les autres facteurs est de : **51.6 %**

M4_MLR_3 variables_log

```
reg_multi_3_log = smf.ols('log_income~log_income_mean+gini+proba_quantile_parent', data=Linear_reg_log).fit()
```

variable dépendante:

- log_income

variables explicatives:

- log_income_mean
- Gini
- proba_quantile_parent

Dep. Variable:	log_income	**R-squared:**	0.798
Model:	OLS	**Adj. R-squared:**	0.798
Method:	Least Squares	**F-statistic:**	4.137e+06
Date:	Thu, 24 Feb 2022	**Prob (F-statistic):**	0.00
Time:	17:42:38	**Log-Likelihood:**	-3.1136e+06
No. Observations:	3150000	**AIC:**	6.227e+06
Df Residuals:	3149996	**BIC:**	6.227e+06
Df Model:	3		
Covariance Type:	nonrobust		

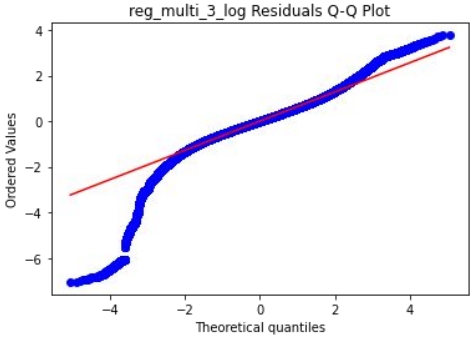
Le modèle arrive à expliquer **79.8%** de variance

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	0.1519	0.004	36.396	0.000	0.144	0.160
log_income_mean	0.9707	0.000	2861.985	0.000	0.970	0.971
gini	-0.0180	4.98e-05	-360.217	0.000	-0.018	-0.018
proba_quantile_parent	0.0104	1.27e-05	821.104	0.000	0.010	0.010

Omnibus:	347577.502	**Durbin-Watson:**	0.006
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	2087415.863
Skew:	-0.362	**Prob(JB):**	0.00
Kurtosis:	6.922	**Cond. No.**	781

- P-value log_income_mean = 0 : **rejet** de H0, income_mean joue un impact sur income_normal (revenu quantile par quantile)
- P-value gini = 0 : **rejet** de H0, Gini a l'impact sur income_normal
- P-value proba_quantile_parent = 0: **rejet** de H0, proba_quantile_parent joue un impact sur income_normal (revenu quantile par quantile)

M4_3 variables log_normalité et homosétasticité des résidus



Grande superposition de **bleue** avec ligne rouge
=> condition normalité est valide.

```
analyse4=sm.stats.anova_lm(reg_multi_3_log, typ=1)
analyse4
```

	df	sum_sq	mean_sq	F	PR(>F)
log_income_mean	1.0	4.907014e+06	4.907014e+06	1.161632e+07	0.0
gini	1.0	5.485212e+04	5.485212e+04	1.298508e+05	0.0
proba_quantile_parent	1.0	2.859814e+05	2.859814e+05	6.770005e+05	0.0
Residual	3149996.0	1.330634e+06	4.224242e-01	NaN	NaN

```
name = ['Lagrange multiplier statistic', 'p-value', 'f-value', 'f p-value']
test = sms.het_breuschpagan(reg_multi_3_log.resid, reg_multi_3_log.model.exog)
lzip(name, test)
```

```
[('Lagrange multiplier statistic', 101338.99782132792),
 ('p-value', 0.0),
 ('f-value', 34902.474403579006),
 ('f p-value', 0.0)]
```

P-value = 0.0, <0.5%, rejet de H0 au niveau de test 0.5%, présence de l'hétérosétasticité pour les résidus de ce modèle reg_multi_3

Décomposition

La variance expliquée par la classe de revenu des parents est de : **4.35 %**
La variance expliquée par le revenu moyen en log est de : **74.59 %**
La variance expliquée par lindice Gini est de : **0.83 %**
La variance expliquée par les autres facteurs est de : **20.23 %**

M4_MLR_3 variables_log + fit(cov_type='HC3')

```
reg_multi_3_log = smf.ols('log_income~log_income_mean+gini+proba_quantile_parent', data=Linear_reg_log).fit(cov_type='HC3')
```

Dep. Variable:	log_income	**R-squared:**	0.798
Model:	OLS	**Adj. R-squared:**	0.798
Method:	Least Squares	**F-statistic:**	4.733e+06
Date:	Wed, 16 Mar 2022	**Prob (F-statistic):**	0.00
Time:	11:05:30	**Log-Likelihood:**	-3.1124e+06
No. Observations:	3150000	**AIC:**	6.225e+06
Df Residuals:	3149996	**BIC:**	6.225e+06
Df Model:	3		
Covariance Type:	HC3		

	coef	std err	z	P>|z|	[0.025	0.975]
Intercept	0.1511	0.005	31.607	0.000	0.142	0.160
log_income_mean	0.9707	0.000	2777.395	0.000	0.970	0.971
gini	-0.0180	6.52e-05	-275.243	0.000	-0.018	-0.018
proba_quantile_parent	0.0104	1.36e-05	767.066	0.000	0.010	0.010

Omnibus:	348378.084	**Durbin-Watson:**	0.006
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	2099750.184
Skew:	-0.363	**Prob(JB):**	0.00

variable dépendante:

- log_income

Variables explicatives:

- log_income_mean
- Gini
- proba_quantile_parent

Le modèle arrive à expliquer **79.8%** de variance

- P-value log_income_mean = 0 : **rejet** de H0, income_mean joue un impact sur income_normal (revenu quantile par quantile)
- P-value gini = 0 : **rejet** de H0, Gini a l'impact sur income_normal
- P-value proba_quantile_parent = 0: **rejet** de H0, proba_quantile_parent joue un impact sur income_normal (revenu quantile par quantile)

M4_Conclusion

En comparant ces modèles ci-dessus, on conclut que:

Quand on transforme les variables en logarithme, on a un modèle **plus performant** qui arrive à expliquer plus de variance de notre variable dépendante.

Quand on ajoute la 3ème variable (le nombre de variables explicatives augmente), le % de variance expliquée par le nouveau modèle **augmente**.

=> le modèle avec 3 variables et en log explique mieux la variable dépendante.

méthode fit(cov_type='HC3') n'a pas servi grand-chose pour diminuer l'effet de l'hétéroscétasticité dans le cadre de régression pour tous ces pays clients.

M4_exemple ALB

```
reg_ALB = smf.ols('log_income~log_income_mean+gini+proba_quantile_parent', data=ALB_reg).fit(cov_type='HC3')
```

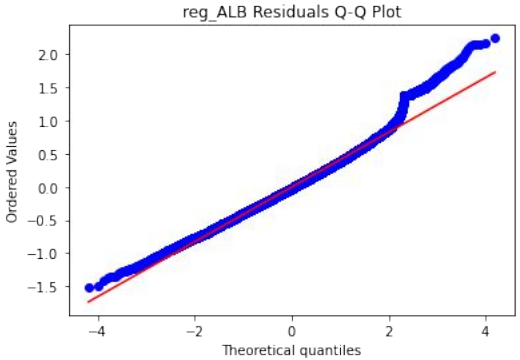
Dep. Variable:	log_income	**R-squared:**	0.370
Model:	OLS	**Adj. R-squared:**	0.370
Method:	Least Squares	**F-statistic:**	1.151e+15
Date:	Mon, 14 Mar 2022	**Prob (F-statistic):**	0.00
Time:	12:41:18	**Log-Likelihood:**	-27059.
No. Observations:	50000	**AIC:**	5.412e+04
Df Residuals:	49997	**BIC:**	5.415e+04
Df Model:	2		
Covariance Type:	HC3		

	coef	std err	z	P>|z|	[0.025	0.975]
Intercept	3.291e+08	9.703	3.39e+07	0.000	3.29e+08	3.29e+08
log_income_mean	5.637e+09	166.163	3.39e+07	0.000	5.64e+09	5.64e+09
gini	-1.447e+09	42.656	-3.39e+07	0.000	-1.45e+09	-1.45e+09
proba_quantile_parent	0.0110	2e-12	5.53e+09	0.000	0.011	0.011

Omnibus:	2778.436	**Durbin-Watson:**	0.014
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	4262.285
Skew:	0.479	**Prob(JB):**	0.00
Kurtosis:	4.062	**Cond. No.**	7.86e+19

Le modèle arrive à expliquer **37 %** de variance sur income_normal d'ALB

M4_ALB_normalité et homosétasticité des résidus



normalité quasi parfaite pour résidus du modèle reg_ALB

```
test = sms.het_breuschpagan(reg_ALB.resid, reg_ALB.model.exog)
lzip(name, test)
```

```
[('Lagrange multiplier statistic', 649.0151053177129),
 ('p-value', 2.38174700496322e-140),
 ('f-value', 328.7554269668309),
 ('f p-value', 1.4256401060724296e-142)]
```

Lagrange multiplier statistic est petit et sa P-value = 2.38 > 5%

=> **non-rejet de H0** au niveau de test 5%, il existe de l'**homosétasticité** pour les résidus de ce modèle reg_ALB

Décomposition

La variance expliquée par la classe de revenu des parents est de : **37%**
La variance expliquée par le revenu moyen en log est de : **0.01%**
La variance expliquée par lindice Gini est de : **0.01 %**
La variance expliquée par les autres facteurs est de : **62.9 %**

Q & A

Merci

P7_prédiction de revenus

Data Analyst_Xuefei ZHANG

12 mars 2022