[WELFARE AND INEQUALITY ANALYSIS]

Hye-Jin Cho-Drugeon

Choh9323@gmail.com

ROADMAP

1. Introduction
2. Data and data sources
3. Data collection
4. Data cleaning and Exploratory data analysis
5. Data base type selection
6. Entities. ERD
7. DATA Visualization

INTRO

Business Use Case

Goal: This project studies intergenerational transfers such as tax and social security in the field of welfare and inequality for Euro-currency using countries with the OECD distribution data.

It intends to explain the behavior of working age generation of 18-65 and old generation of above 65.

Domain: Economics and Statistics

Plan

1. Planning of my project in Jira
2. Code in Python for Data collection and cleaning
3. ER Diagram
4. Data source and Meta Data
5. Database Script
6. Report (10 pages)
7. Slides

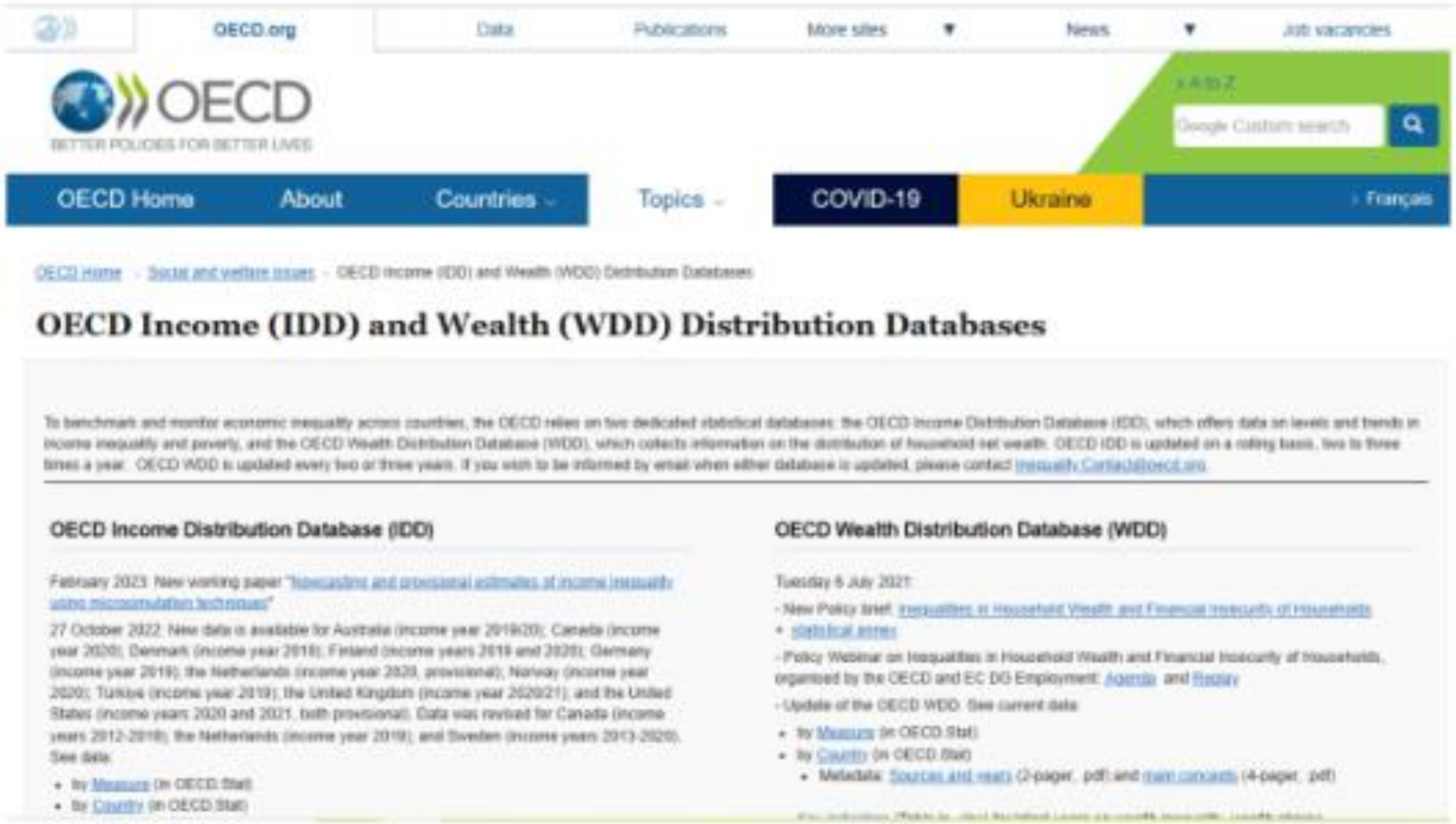
SOLUTION

Output

In Github [DAFT_0410/module5 at main · chatlapin/DAFT_0410 · GitHub]

#starter	#SQL, API, Python Toymodel (France), Python ML
Jira	OECD.sql
Datadescription.pdf	OECD.API31.ipynb
#ER, Data Cleaning, EDA	Project Toymodel.ipynb
Erdiagram_OECD.drawio.pdf	ProjectBig.ipynb
OECD Data Cleaning and EDA.ipynb	
OECDRawData.ipynb	#Euro-using country stationary test results
#Data, cleaned (by variables)	UnitRootTestOECD meandi.ipynb
Clean_data_pivotedCI.csv	UnitRootTestOECDCI.ipynb
Clean_data_pivotedCPI.csv	UnitRootTestOECDCPI.ipynb
Clean_data_pivotedMDI.csv	UnitRootTestOECDP90P10.ipynb
Clean_data_pivotedP90.csv	UnitRootTestOECDTE.ipynb
Clean_data_pivotedTE.csv	UnitRootTestOECDtaxsecu.ipynb
Clean_data_pivotedTaxsecu.csv	UnitRootTestOECD transfer.ipynb
Clean_data_pivotedtransfer.csv	#Rawdata (by world, France, Euro-using country
Clean_data_pivotedCPI.csv	group, each variable)
#Data, pivoted (by variables)	France.csv
data_pivotedCI.csv	Monde (1).csv
data_pivotedCPI.csv	Welfare(1).csv
data_pivotedMDI.csv	Welfare.csv
data_pivotedP90.csv	# Data Visualisation
data_pivotedTE.csv	Welfare_DataVisualisation.xlsx
data_pivotedTaxsecu.csv	
data_pivotedtransfer.csv	
data_pivotedCPI.csv	

Data source



RAW data from: https://www.oecd.org/social/income-distribution-database.htm#:~:text=The%20OECD%20Income%20Distribution%20Database%20provides%20information%20on%20the%20equivalised,households%20on%20a%20comparable%20basis.

```
df.sample(4)
```

	LOCATION	Country	MEASURE	Measure	AGE	Age group	DEFINITION	Definition	METHODO	Methodology	...	Year
19190	ESP	Spain	PVTAA5	Age group 51-65: Poverty rate after taxes and ...	TOT	Total population	CURRENT	Current definition	METH2012	New income definition since 2012	...	2011
18845	NOR	Norway	GINIG	Gini (gross income, before taxes)	OLD	Retirement age population: above 65	CURRENT	Current definition	METH2012	New income definition since 2012	...	2020
15389	POL	Poland	TRROTCTOTAL	Current transfers received from non-profit ins...	WA	Working age population: 18-65	CURRENT	Current definition	METH2012	New income definition since 2012	...	2017

SAMPLING

MACHINE LEARNING: SUPERVISED LEARNING

TOY MODEL: FRANCE 2012_2019
DESCRIPTIVE STATISTICS

	totalearning	capitalincome	transferrec	transferpaid	priceindex2015	meandi	saving
count	8.000000	8.000000	8.000000	8.00000	8.000000	8.000000	8.00000
mean	21618.750000	2343.750000	5708.750000	-5001.25000	100.843026	26693.750000	2343.06875
std	1254.232121	189.581306	49.117207	313.16073	1.909861	903.199669	186.82237
min	20240.000000	2070.000000	5640.000000	-5470.00000	98.605000	25850.000000	2156.60000
25%	20760.000000	2272.500000	5667.500000	-5115.00000	99.835420	26142.500000	2192.83250
50%	21250.000000	2345.000000	5715.000000	-4930.00000	100.090835	26280.000000	2284.43000
75%	22265.000000	2402.500000	5742.500000	-4822.50000	101.685025	27020.000000	2463.00500
max	23790.000000	2710.000000	5780.000000	-4590.00000	104.232500	28390.000000	2617.00000

MACHINE LEARNING: SUPERVISED LEARNING
TOY MODEL: FRANCE 2012_2019

```
model.coef_
```

```
array([32.25850999, 86.42024677])
```

```
model.intercept_
```

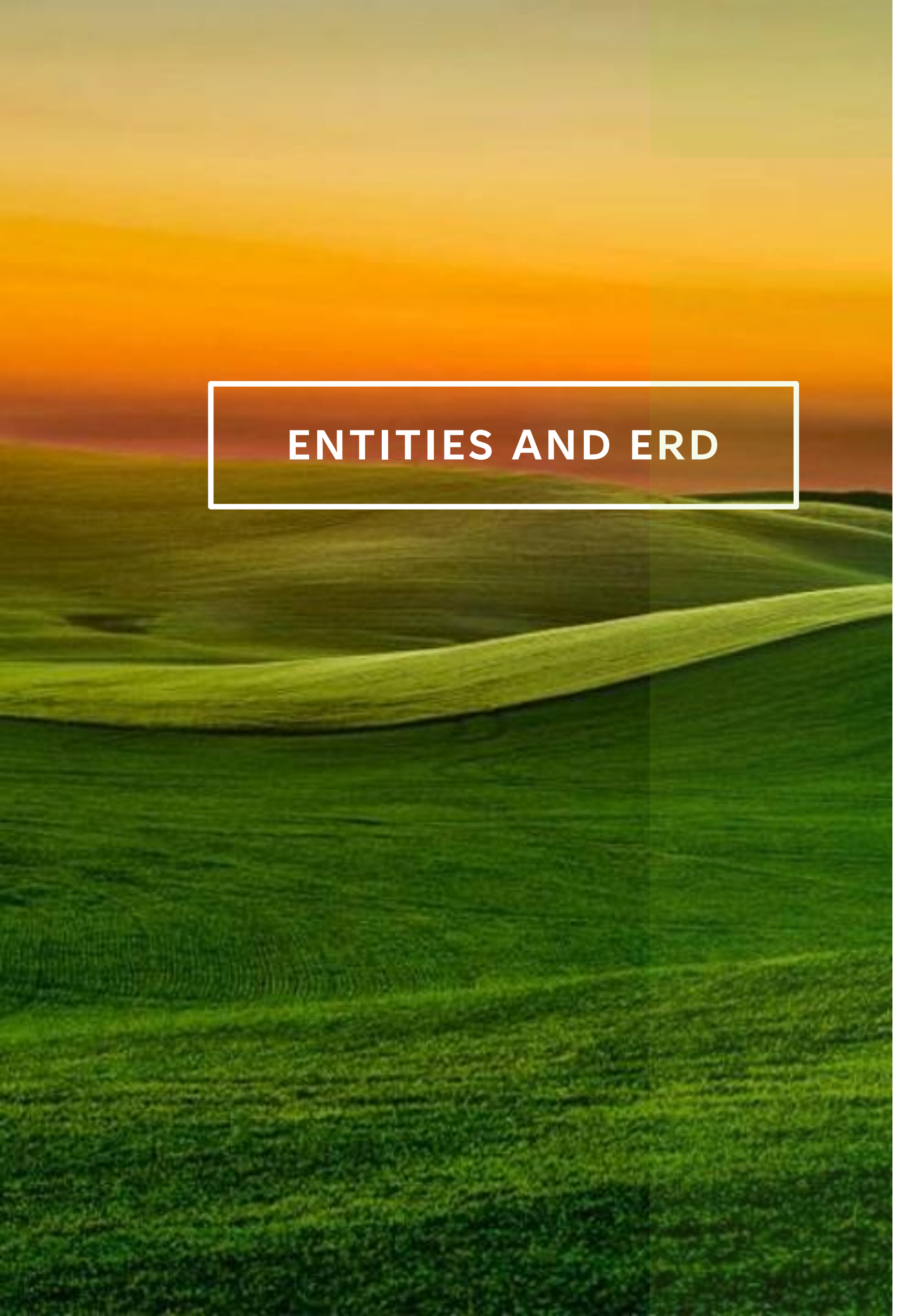
```
0.007303861711072557
```

5. Predict labels for unknown data

```
# new instances where we do not know the answer
Xfit, _ = make_regression(n_samples=3, n_features=2, noise=0.1, random_state=1)
# make a prediction
yfit = model.predict(Xfit)
# show the inputs and predicted outputs
for i in range(len(Xnew)):
 print("X=%s, Predicted=%s" % (Xfit[i], yfit[i]))
```

```
X=[-1.07296862 -0.52817175], Predicted=-80.2497983168563
X=[-0.61175641  1.62434536], Predicted=120.649280643451
X=[-2.3015387   0.86540763], Predicted=0.5518357031231957
```

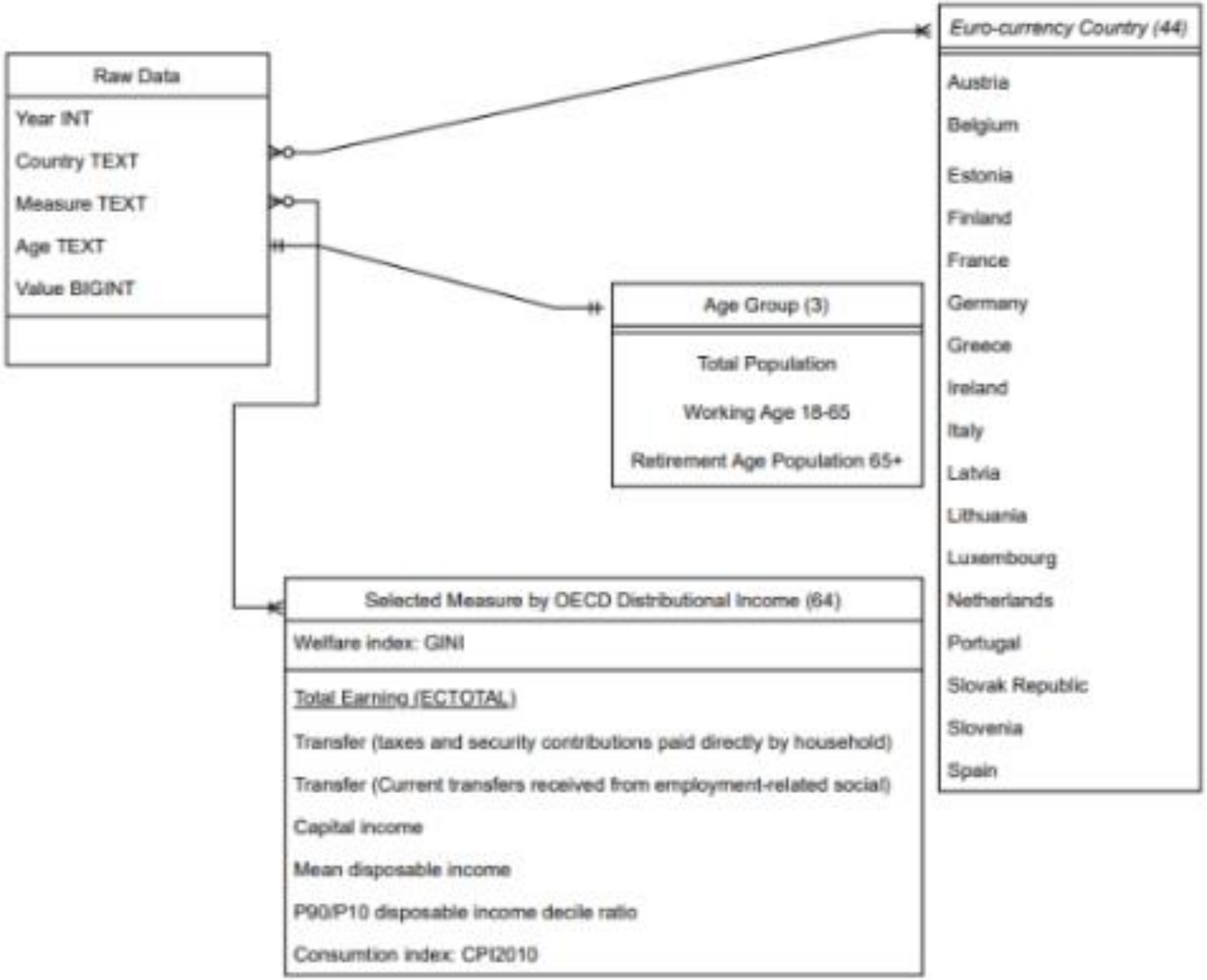
```
yfit = model.predict(Xfit)
```

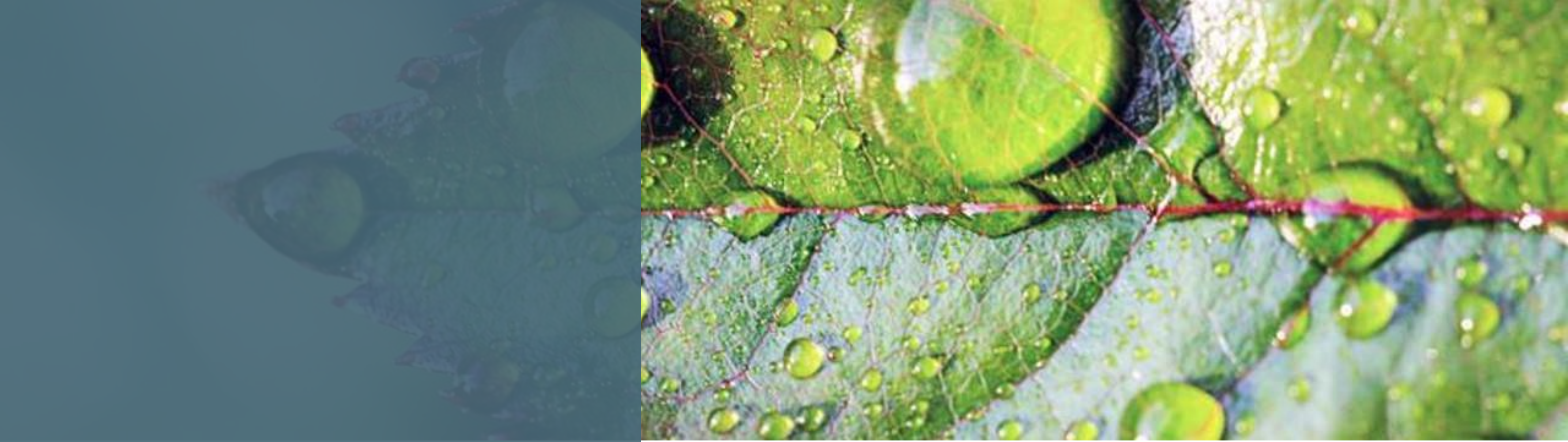


ENTITIES AND ERD

Raw Data. Shape(77182, 21)
(#Number) : Unique at Raw Data)

https://chatlapin.atlassian.net/jira/software/projects/CHAT/boards/1





SQL OR NO SQL

For analyzing entity relation, SQL is useful to check which kind of joints are possibly applied. For cross-country data, it is not easy to make left-join or outer join, the reason is the number of row and variables are reduced due to duplicated elements. When we did pivot in Python, it’s the delicate method from the same reason.

Though, when we use SQL, the entity relation of data is automatically captured. In addition, by using functions such as group by, order by, sum, avg, min, max, it was useful to check the data structure as below.

SQL OR NO SQL

```sql
#function 1: Group by
#show the table with selected Euro-using countries with measures.
SELECT *
FROM OECD.`monde (1)`
GROUP BY Country;

#total generations
SELECT *
FROM OECD.`monde (1)`
WHERE Age='TOT';

#average values for total generations (young and old)
SELECT avg(`values`)
from (select *
FROM OECD.`monde (1)`
group by Age='TOT'
limit 5) summary;

# function 2: Order by
#working-aging generations
SELECT *
FROM OECD.`monde (1)`
WHERE Age='WA'
ORDER BY Country;
```

```sql
# function 3: UNION
SELECT *
FROM OECD.`monde (1)`
WHERE Age='OLD'
UNION
SELECT *
FROM OECD.`monde (1)`
WHERE Age='WA';

#(1) measure: GINI
SELECT *
FROM OECD.`monde (1)`
WHERE Age='OLD'
and Measure='GINI';

#GROUP FUNCTIONS: MAX(4), MIN(5), AVG(6), SUM(7), COUNT(7)
SELECT MAX(`values`) as max,
MIN(`values`) as min,
AVG(`values`) as average,
SUM(`values`) as total,
COUNT(`values`) as NUM_columns
FROM OECD.`monde (1)`
WHERE Age='OLD'
and Measure='GINI';
```

DATA TYPE SELECTION

World data from a toy model of France

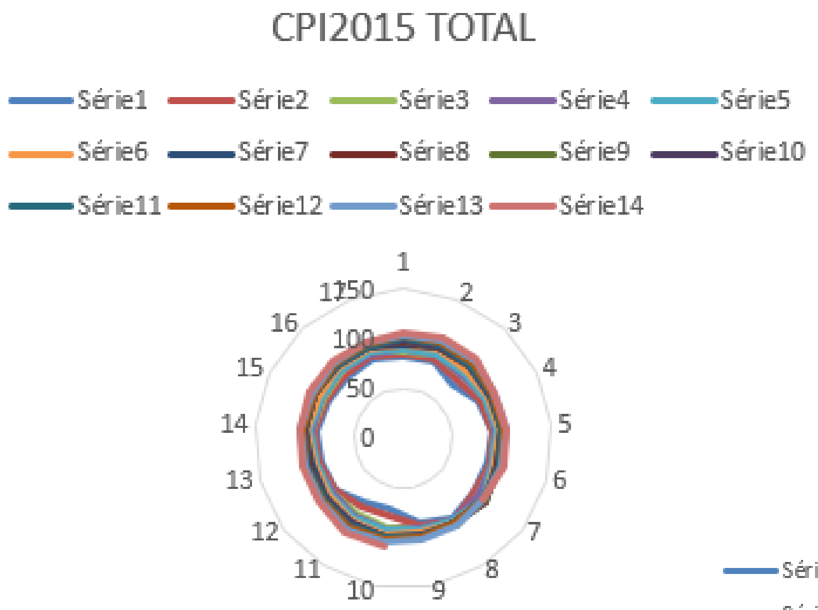
Summary: Stationary Test by Dickey-Fuller test: Except for Capital Income, variables are stationary.

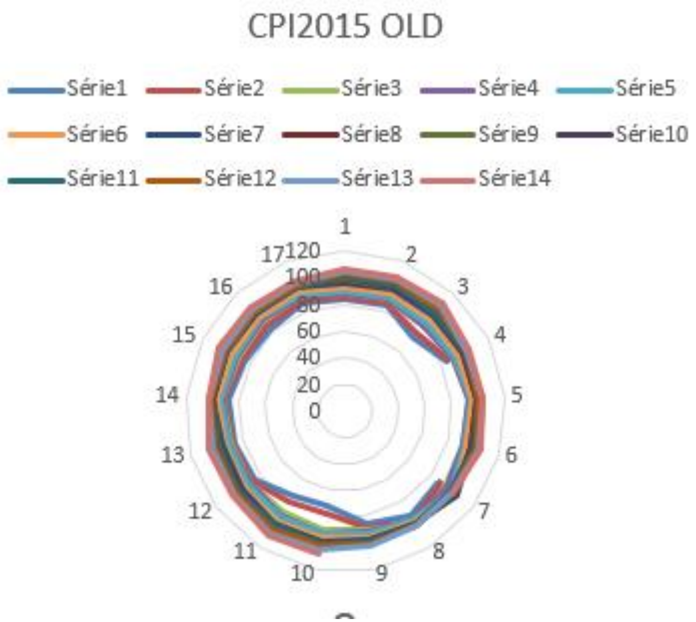
data31['saving']	data31['meandi']	data31['priceindex2015']	data31['transferpaid']
ADF Statistic: -2.09 Critial Values: 1%, -6.05 Critial Values: 5%, -3.93 Critial Values: 10%, -2.99 p-value: 0.25 Stationary	ADF Statistic: 1.32 Critial Values: 1%, -6.05 Critial Values: 5%, -3.93 Critial Values: 10%, -2.99 p-value: 1.00 Stationary	ADF Statistic: -0.44 Critial Values: 1%, -6.05 Critial Values: 5%, -3.93 Critial Values: 10%, -2.99 p-value: 0.90 Stationary	ADF Statistic: -0.72 Critial Values: 1%, -4.94 Critial Values: 5%, -3.48 Critial Values: 10%, -2.84 p-value: 0.84 Stationary

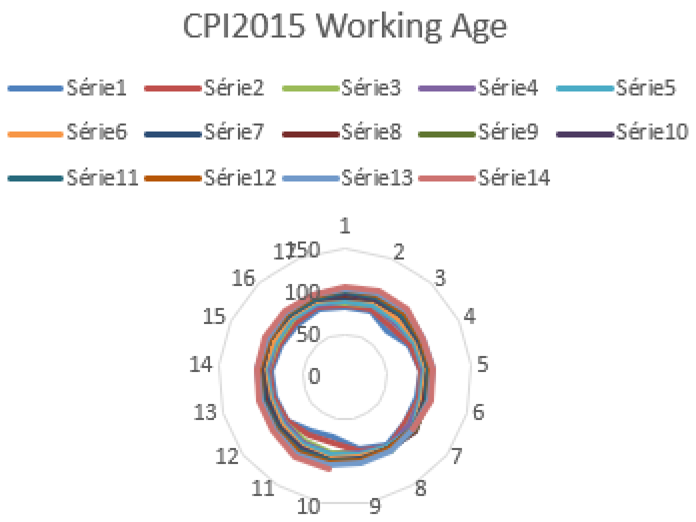
DATA TYPE SELECTION

data31['transferrec']	data31['capitalincome']	data31['totalearning']
ADF Statistic: -0.93 Critial Values: 1%, -4.94 Critial Values: 5%, -3.48 Critial Values: 10%, -2.84 p-value: 0.78 Stationary	ADF Statistic: -8.90 Critial Values: 1%, -6.05 Critial Values: 5%, -3.93 Critial Values: 10%, -2.99 p-value: 0.00 Stationary p-value: 0.00 Non-Stationary	ADF Statistic: 1.85 Critial Values: 1%, -5.35 Critial Values: 5%, -3.65 Critial Values: 10%, -2.90 p-value: 1.00 Stationary

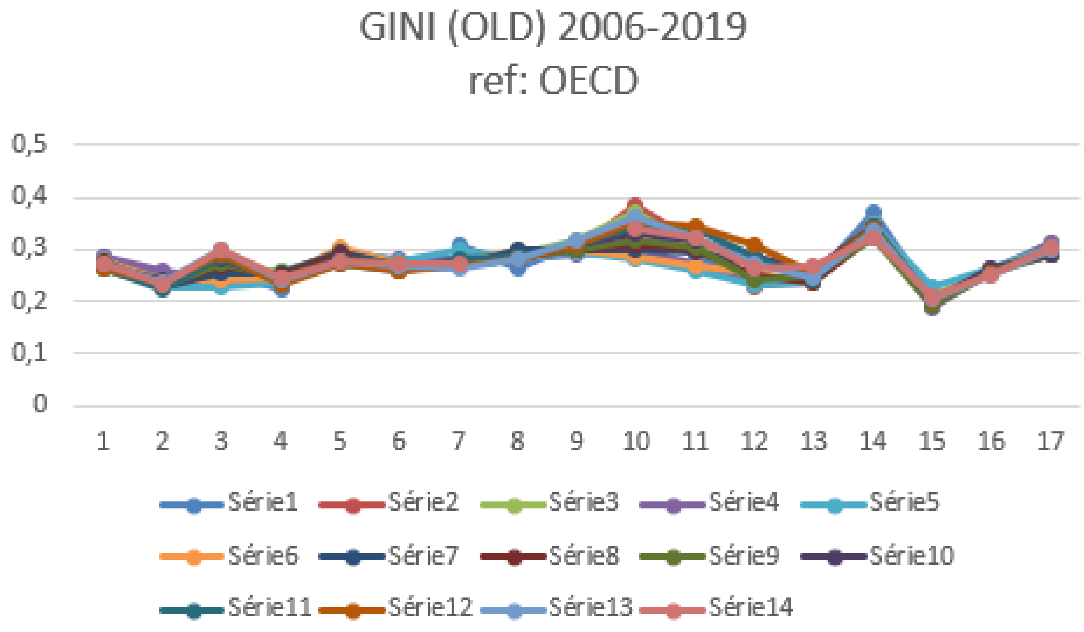
DATA VISUALISATION – INTERGENERATIONAL TRANSFER

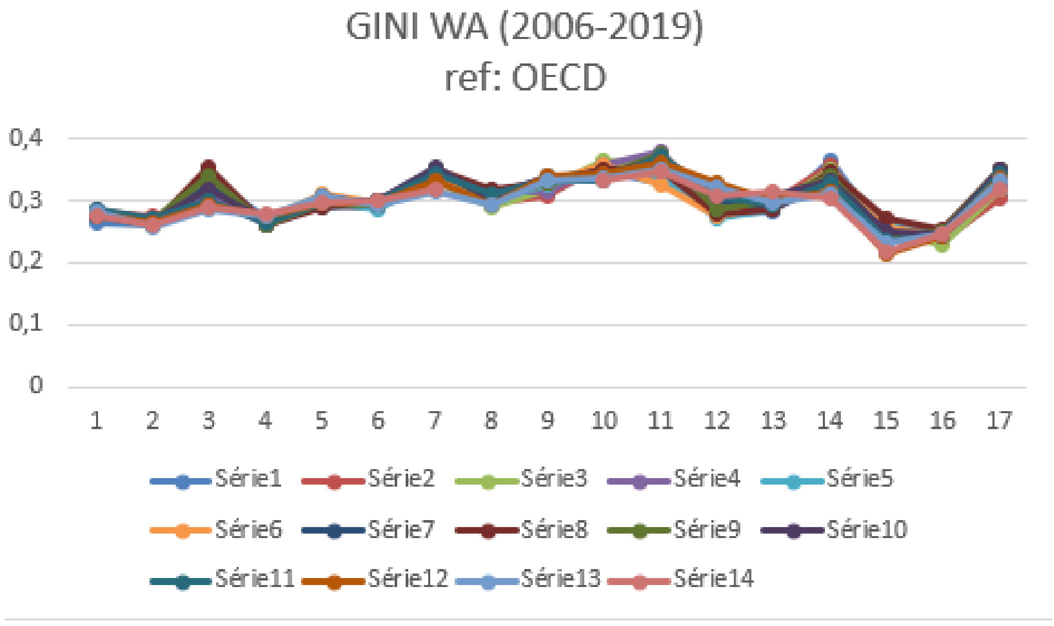




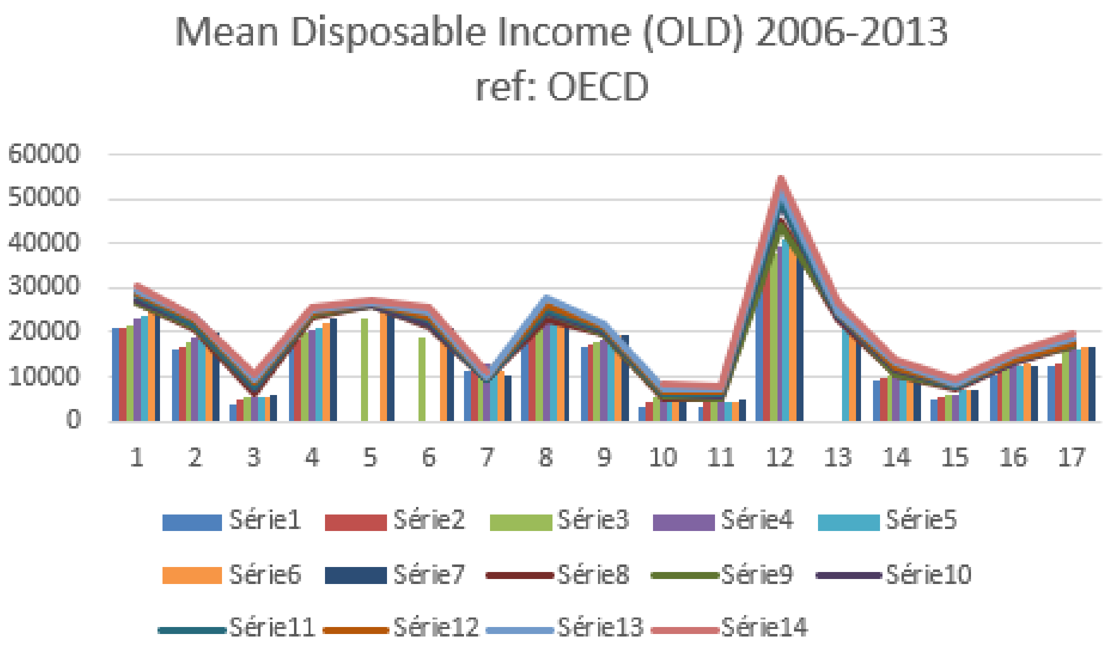


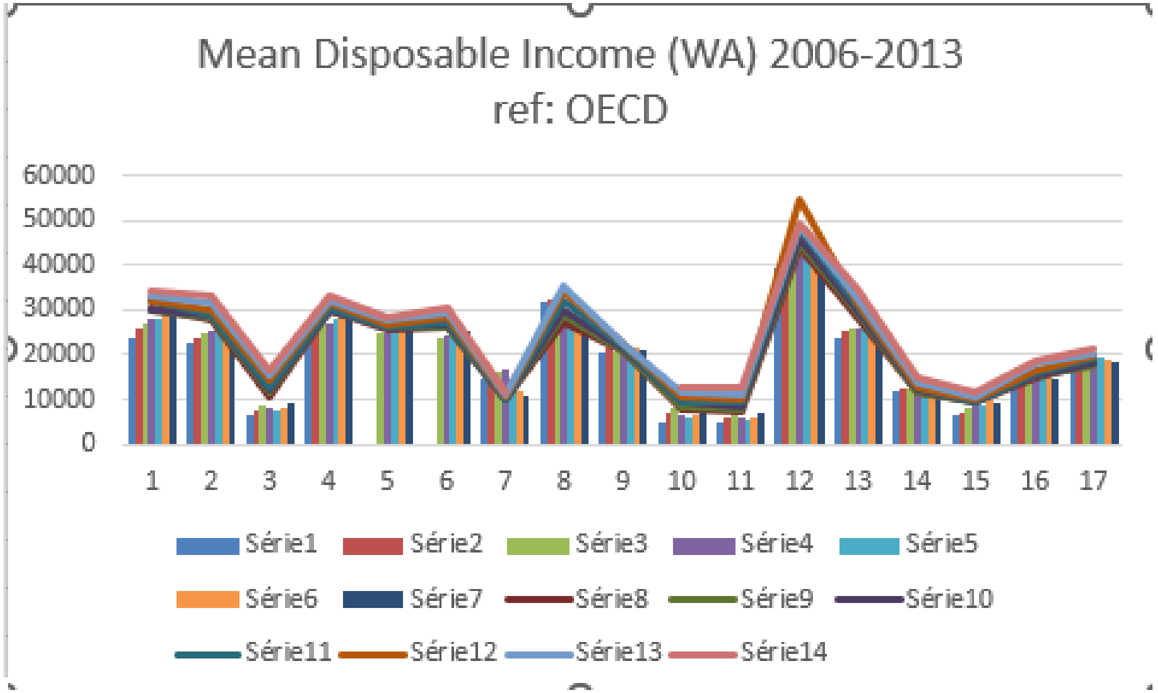
DATA VISUALISATION
INEQUALITY AND GINI INDEX



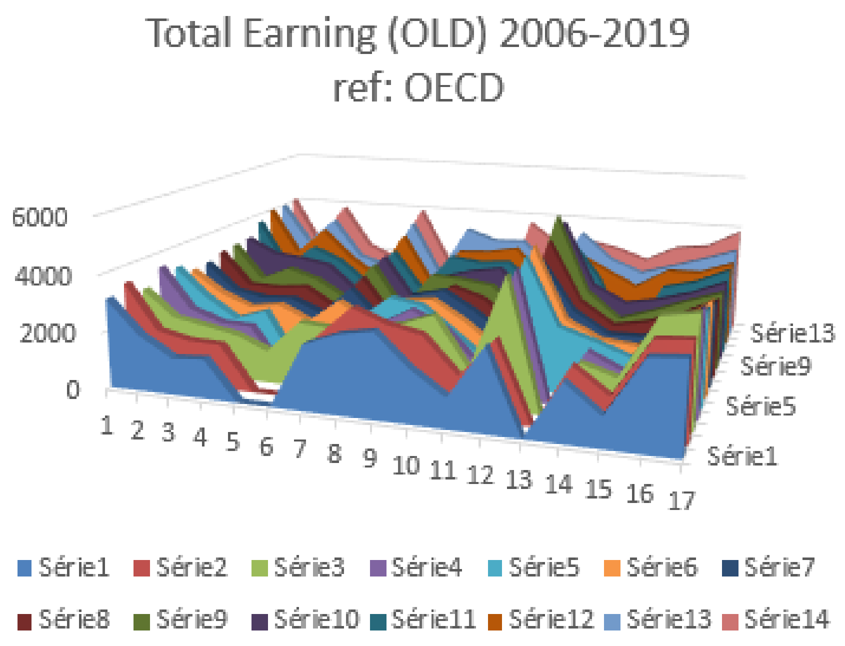


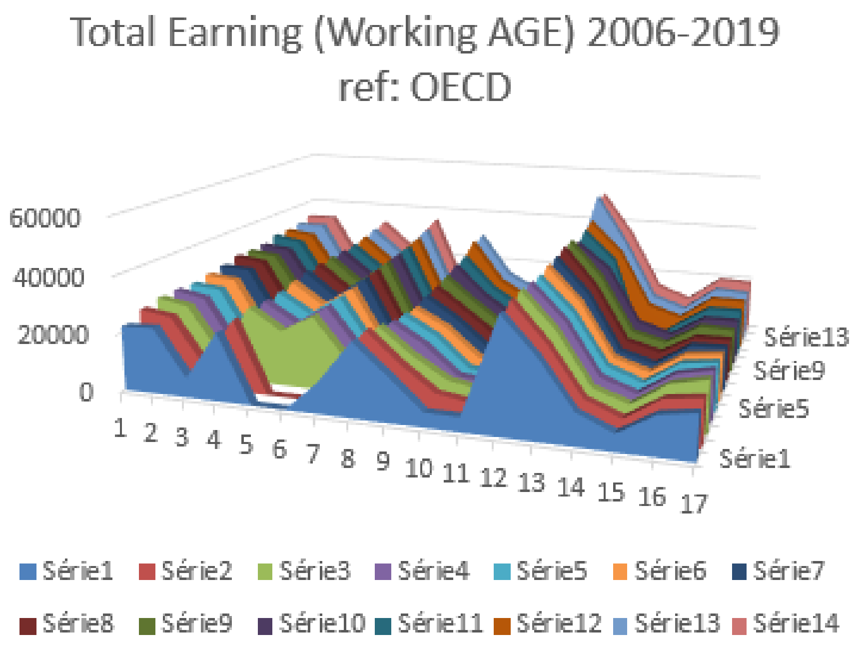
DATA VISUALISATION – MEAN DISPOSABLE INCOME: ESTIMATED AS THE GREATER AMOUNT AMONG VARIABLES. TARGET VARIABLE



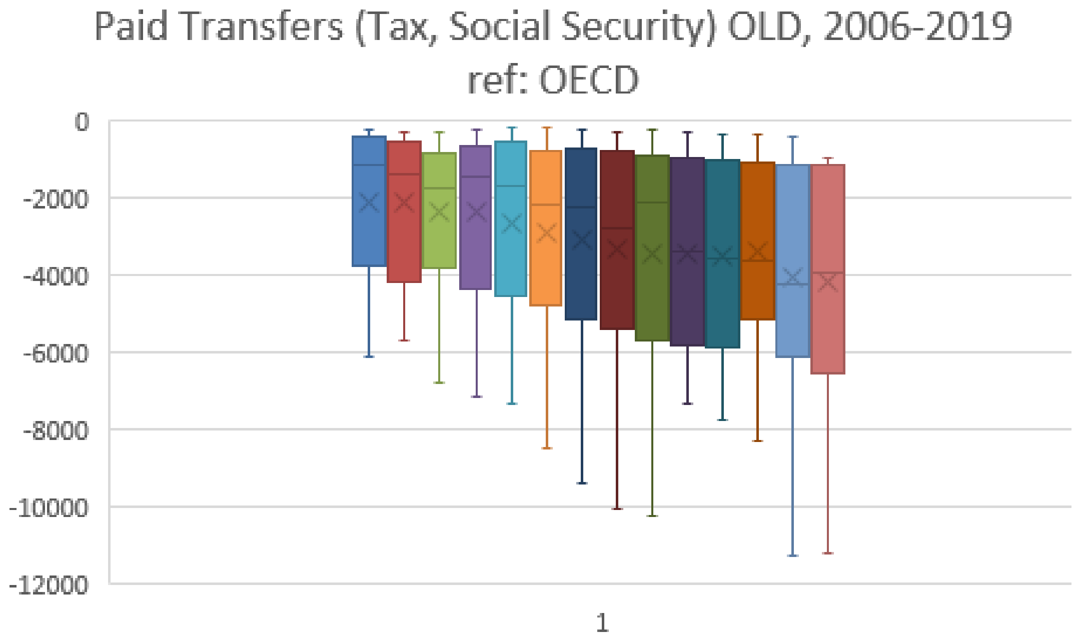


DATA VISUALISATION
TOTAL YEARLY EARNING PER PERSON





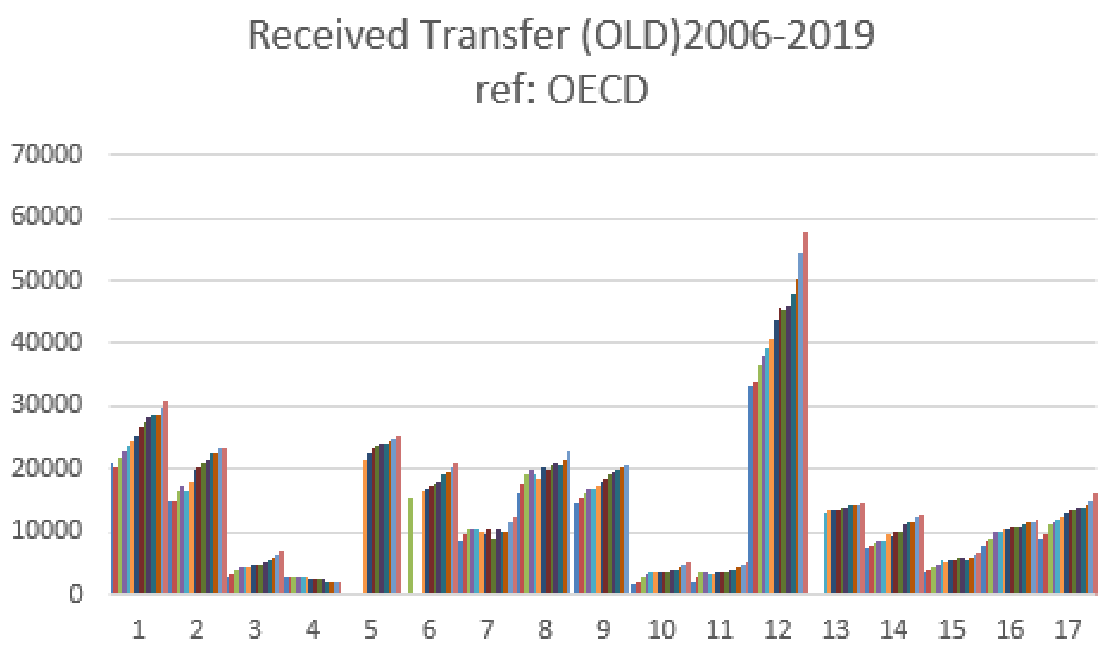
DATA VISUALISATION
PAID TRANSFER (TAX, INSURANCE)

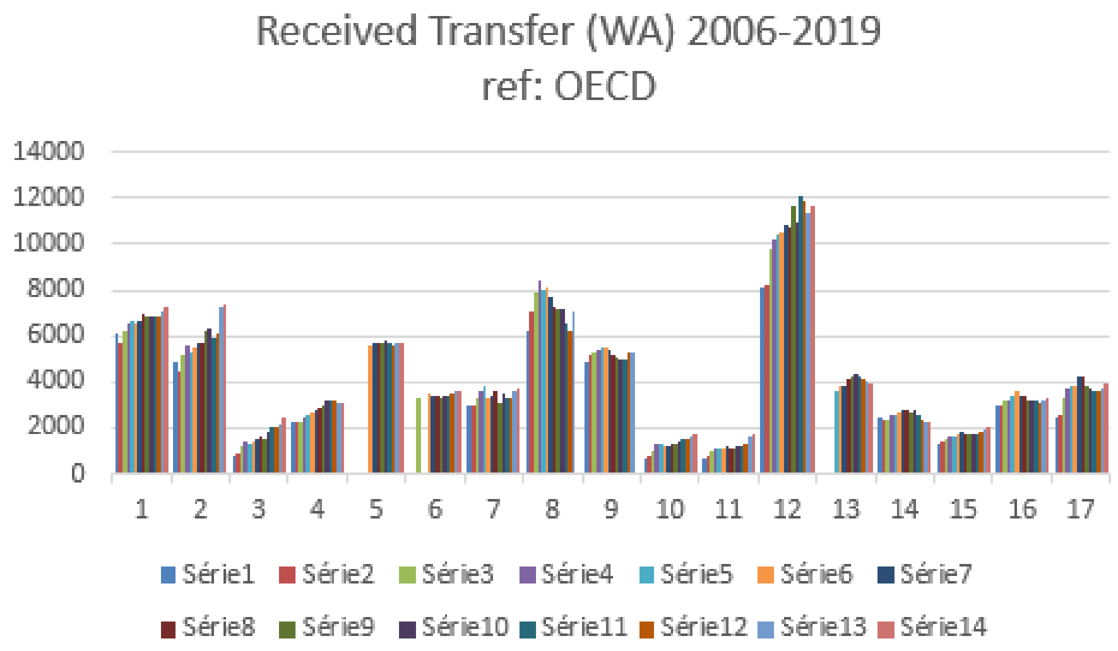


-1864	-1976	-2030	-2143	-2028	-2091

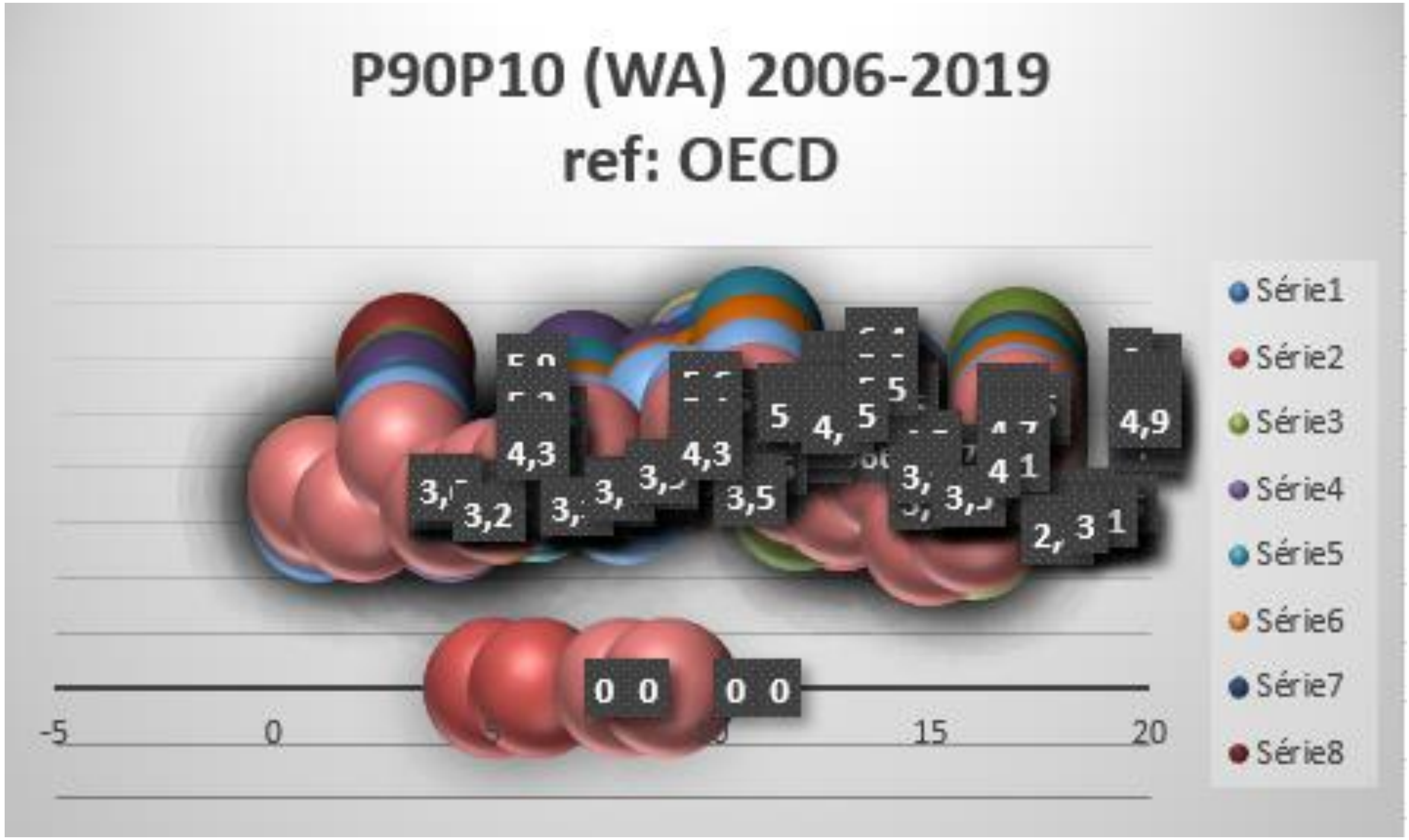
-3751	-3684	-3694	-3786	-3739	-3800

DATA VISUALISATION
RECEIVED TRANSFER (TAX, INSURANCE)





DATA VISUALISATION
INEQUALITY FOR THE WORKING AGE GROUP



Thank you.