Social Inequalities in Chronic Diseases

How the social class impacts the inequalities in Chronic Diseases in France

Huyen Tran

Description

Chronic Diseases

A chronic condition is a health condition or disease that is persistent or otherwise long-lasting in its effects or a disease that comes with time. The term "chronic" is often applied when the course of the disease lasts for more than three months. Wikipedia

Context

In Vietnam, most of the patients for chronic diseases are low-income workers. This issue could be explained by the pollution, poor quality of foods, unhealthy way of living as well as high medical costs. In France, despite of a clean environment, good quality of food and healthcare assistance from the government, social inequalities in chronic diseases still exist.

Objective

- ❏ To explore further the social inequalities in chronic diseases in France to get the insights.
- ❏ To build a model to predict which psychotropic treatment will get highest weighted number of cases and which social class will get highest weighted number of patients for psychotropic treatment in France.

Planning

Data Collection

- Healthcare
- Data.drees.solidarites-sante.gouv.fr

Data Cleaning

- Rename columns
- Drop columns
- Handle missing values
- Handle duplicated
- Columns definition

EDA & Visualization

- Data distribution
- Prevalence vs incidence
- Weighted number of patients
- Violin charts
- Standardised rate with the direct & indirect method

Planning

MySQL ERD / Database Schema

- ❏ Create dataframes with Python
- ❏ Export dataframes as tables in MySQL
- ❏ Identify primary keys and foreign keys
- ❏ Entity Relationship Diagram

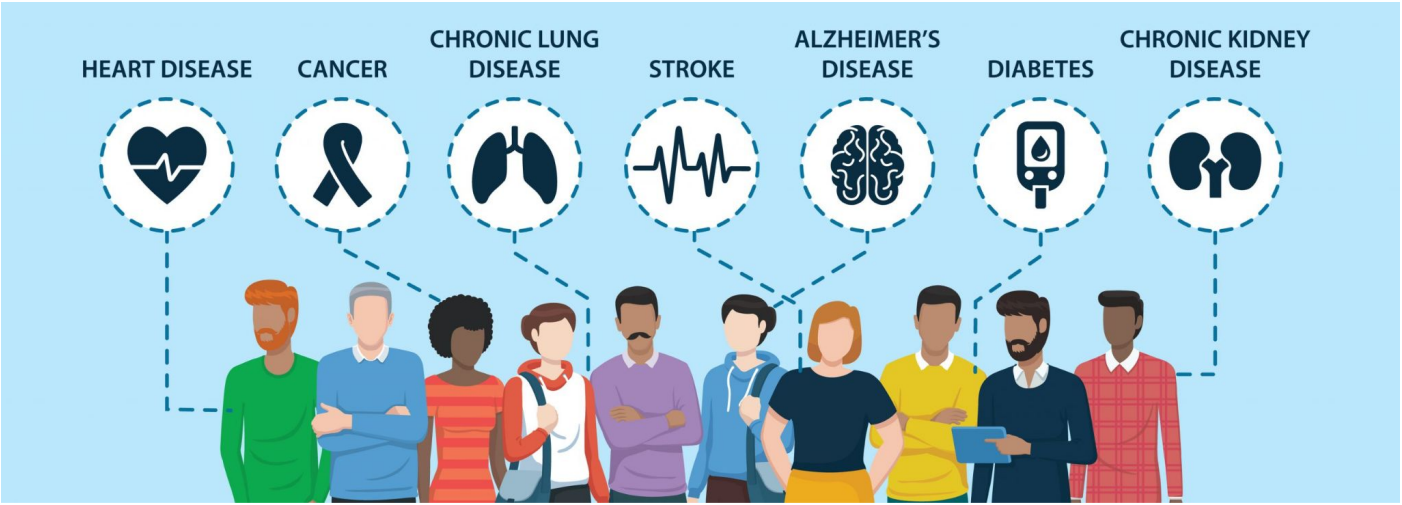
MySQL Queries

- ❏ Top 10 new cases for the poorest population
- ❏ Top 10 new cases for the wealthiest population
- ❏ Cases by disease group at each profession
- ❏ Average weighted number of cases by disease group at each level of education
- ❏ Highest cases by disease group & gender

Machine Learning

- ❏ LinearRegression
- ❏ DecisionTreeRegressor
- ❏ RandomForestRegressor
- ❏ XGBRegressor
- ❏ SVM

Introduction



State of Health in the EU - France - Country Health Profile 2021

“Most French people reported good health, but nearly two in five adults (38%) have a chronic condition in 2019, a slightly higher proportion than the EU average.”

“However, as in other countries, people on higher incomes are more likely to report being a good health: 72% in the highest income quintile reported being in good health compared with 58% in the lowest.”

Data Collection

La Direction de la recherche, de l'évaluation, des études et des statistiques (DREES)

The incidence and prevalence rates of chronic diseases based on various socio-demographic variables such as Gender, Age Group, Region, Tenth of Standard of Living, Socio-professional group, and Education.

The period covered is from 2016 to 2017 and the population studied includes those living in metropolitan France or in the overseas territories except for Mayotte, for which there is not sufficient data.

EDA - Exploratory Data Analysis

1. Data Cleaning

MORE DATA INFO :

Data shape : (46176, 15)

Total rows in the dataset : 46,176
Total columns in the dataset : 15
Total duplicated values : 0
Total null values : 7,722

RATIO OF MISSING AND DUPLICATED VALUES IN OUR DATA :

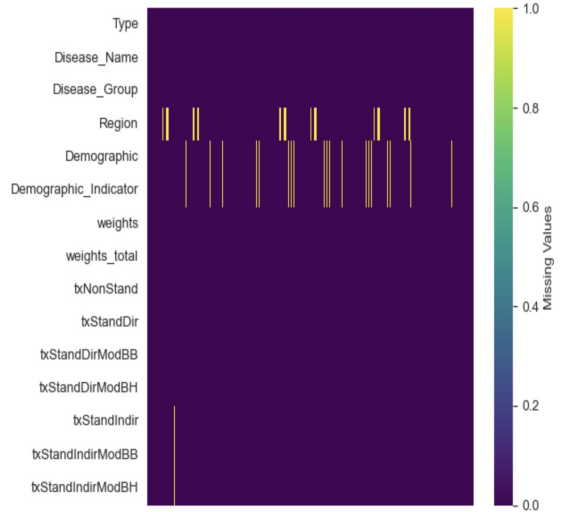
Percentage of null values in the data : 16.72%
Percentage of duplicates in the data : 0.0%

TOTAL MISSING VALUES FOR EACH COLUMN :

	MISSING VALUES
Type	0
Disease_Name	0
Disease_Group	0
Region	3016
Demographic	1768
Demographic_Indicator	1768
weights	4
weights_total	4
txNonStand	4
txStandDir	4
txStandDirModBB	4
txStandDirModBH	4
txStandIndir	382
txStandIndirModBB	382
txStandIndirModBH	382

VISAULIZE MISSNG VALUES :

METHOD 1



Columns definition:

- “Type”: Type of value ("incidence" or "prevalence")
- “Disease_Name”: Name of the Disease
- “Disease_Group”: Group of the Disease
- “Demographic”: Demographic Information such as Social Class, Genders, Age Group, Jobs and Degrees.
- “Demographic_Indicator”: Indicators that explain each Demographic Information
- “weights”: Weighted numbers of people who are ill (prevalence) or fall ill (incidence).
- “weights_total”: Total Weighted numbers of people who are ill (prevalence) or fall ill (incidence).
- “txNonStand”: Non-Standardised Rate: weights / weights_total
- “txStandDir”: Standardised Rate with the Direct Method
- “txStandDirModBB”: Standardised Rate with the Direct Method (lower limit of the 95% confidence interval)
- “txStandDirModBH”: Standardised Rate with the Direct Method (upper limit of the 95% confidence interval)
- “txStandIndir”: Standardised Rate with the Indirect Method
- “txStandIndirModBB”: Standardised Rate with the Indirect Method (lower limit of the 95% confidence interval)
- “txStandIndirModBH”: Standardised Rate with the Indirect Method (upper limit of the 95% confidence interval)

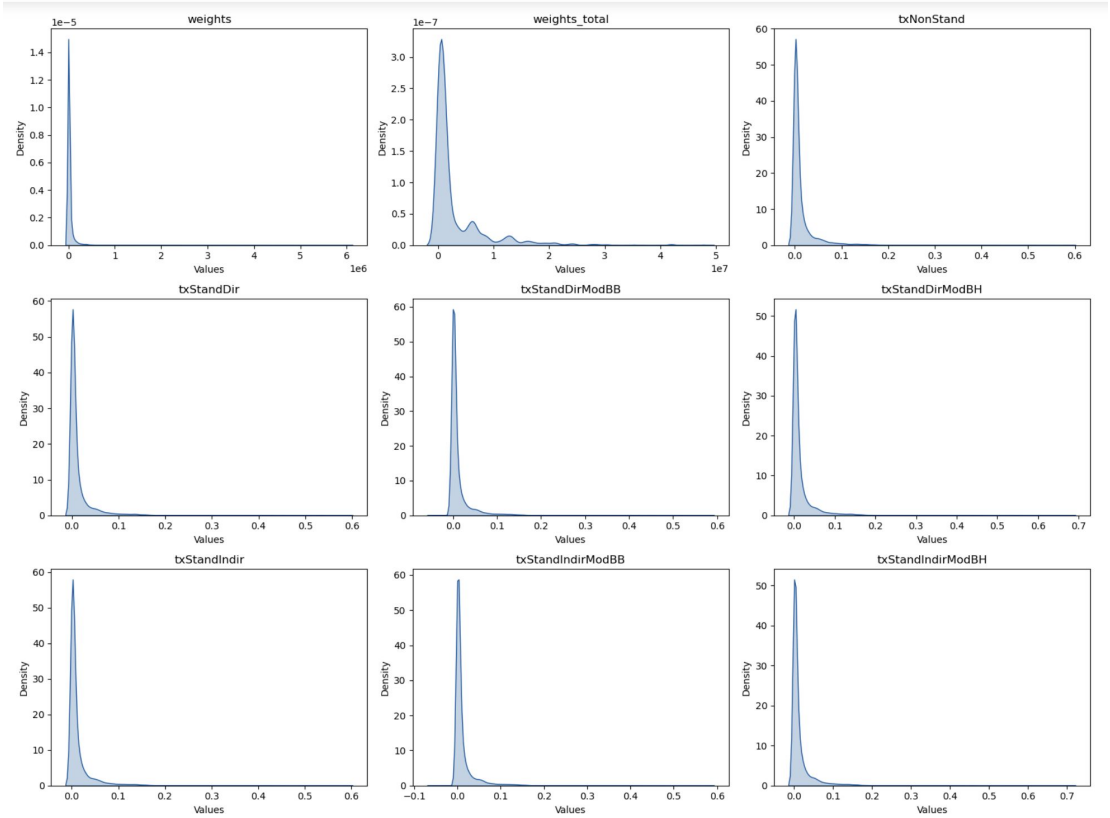
“Region” column has high number of missing values (3106).

Option: filling them with KNN Imputer → producing bias into data.

Painfully removing “Region” column.

2. Data Distribution

Distribution of numeric columns



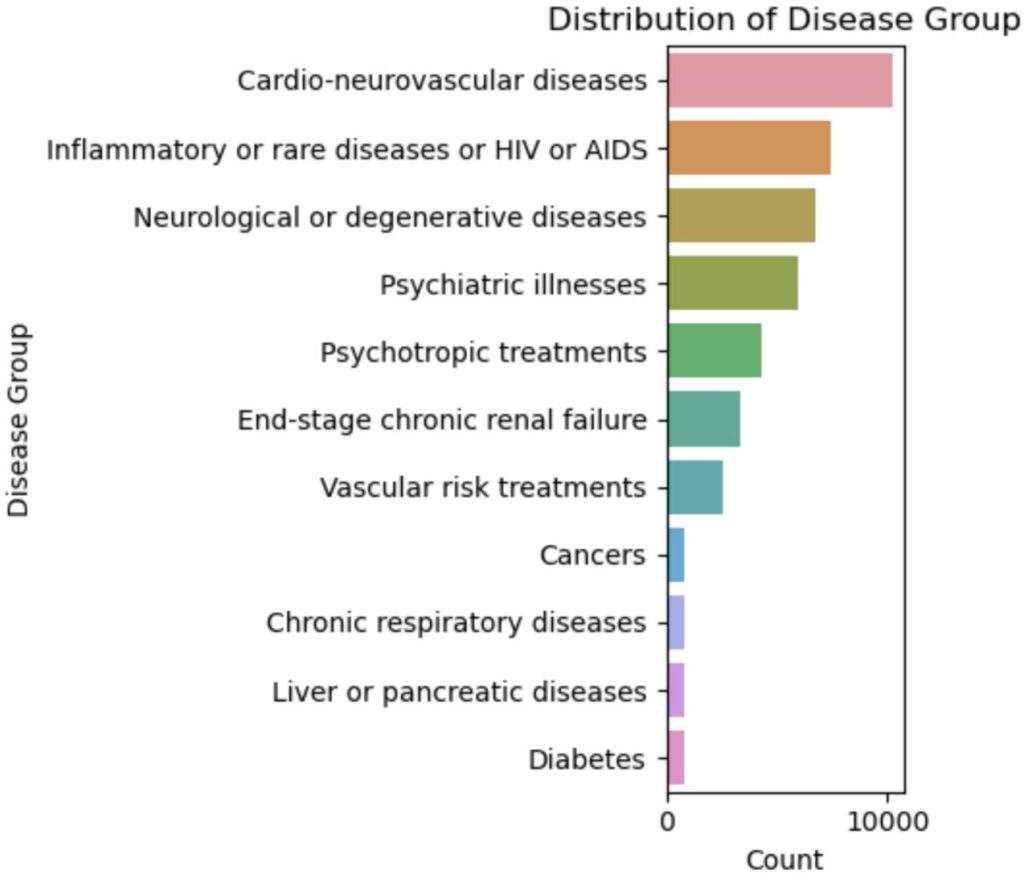
8 columns display a right skew.

The occurrences of higher values are less frequent but possessing greater magnitude.

The existence of outliers or extreme values on the higher end of the distribution.

The outliers may hold valuable information or represent significant events or conditions.

Retain these outliers for subsequent analysis and interpretation.

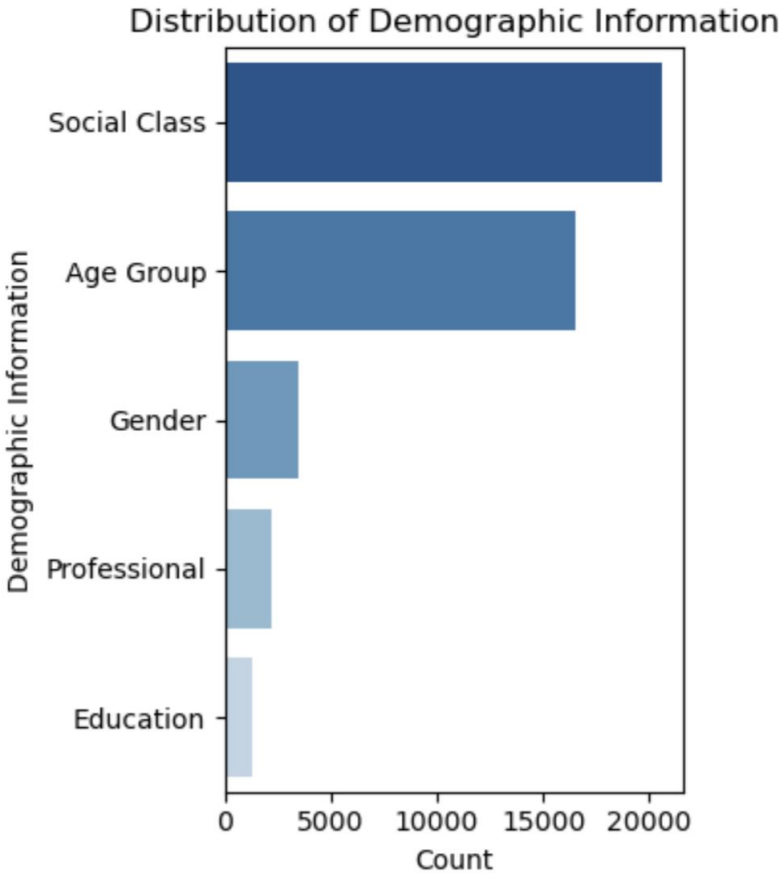


Overview of how the data categorises present each disease group.

Cardio-neurovascular: highest prevalence with 12 different diseases.

Inflammatory or rare diseases or HIV or AIDS: 9 diseases.

Cancers : lower representation because there is no breakdown of specific cancer types.



Highest number of rows corresponds to the social class variable, indicating a primary focus on capturing social inequalities across different classes in France.

3. Prevalence vs Incidence

Prevalence includes all cases, both new and pre-existing, in the population at the specified time (in this case, the weighted number of patients in France from 2016 to 2017).

Incidence is limited to new case only.

```
prevalence    22124
incidence     21920
Name: Type, dtype: int64
```

4. Weighted Number of Patients

What is Weighted Number of Patients?

By assigning weights based on the severity or frequency of the condition, the data provides a more accurate representation of the burden on our healthcare system.

$$500 \rightarrow 100 + \text{severe symptoms / longer duration of illness} = 120$$

$$500 \rightarrow 100 \rightarrow 120$$

This adjustment accounts for the fact that some patients may require more intensive treatment or have a greater impact on healthcare resources.

❖ *Social class has an significant impact on the prevalence cases*

Total weighted number of patients by Social Class for each Disease Group

Type	Disease_Group \ Demographic_Indicator	1	2	3	4	5	6	7	8	9	10
incidence	Cancers	212,172	254,828	296,205	302,589	308,269	295,988	299,974	316,566	348,876	384,580
	Cardio-neurovascular diseases	1,272,886	1,753,830	1,998,851	1,886,429	1,725,999	1,605,273	1,536,623	1,484,984	1,597,562	1,649,315
	Chronic respiratory diseases	640,698	653,756	633,626	606,400	592,856	575,793	554,705	531,173	530,143	491,877
	Diabetes	187,211	177,702	163,761	153,988	145,311	140,101	134,426	126,906	124,341	114,260
	End-stage chronic renal failure	20,535	20,694	19,592	17,657	19,284	15,012	20,212	15,625	13,379	13,050
	Inflammatory or rare diseases or HIV or AIDS	155,470	168,539	175,561	181,315	162,798	180,618	173,959	166,797	174,305	174,405
	Liver or pancreatic diseases	120,976	120,442	124,261	108,193	106,713	91,653	88,637	85,130	81,652	83,376
	Neurological or degenerative diseases	317,868	427,126	468,664	441,495	395,598	359,330	337,192	317,701	324,503	346,207
	Psychiatric illnesses	823,147	842,624	820,829	773,816	690,145	623,875	595,430	554,316	535,100	479,440
	Psychotropic treatments	1,715,879	1,933,990	2,054,173	2,098,965	2,038,025	2,051,986	2,010,759	2,040,744	2,110,078	2,065,059
	Vascular risk treatments	1,067,946	1,115,657	1,159,775	1,161,111	1,153,475	1,198,774	1,190,134	1,245,659	1,329,575	1,450,640
prevalence	Cancers	1,192,879	1,560,473	1,859,944	1,911,986	1,933,671	1,928,459	1,941,225	2,070,328	2,285,752	2,542,571
	Cardio-neurovascular diseases	5,373,920	7,353,257	8,400,943	8,052,555	7,502,942	6,980,628	6,639,148	6,499,843	6,872,951	6,971,750
	Chronic respiratory diseases	2,370,765	2,513,779	2,446,051	2,306,487	2,189,939	2,103,372	1,992,911	1,892,337	1,822,468	1,641,065
	Diabetes	2,406,421	2,591,320	2,670,841	2,484,489	2,346,200	2,146,647	2,009,323	1,957,522	1,878,108	1,675,478
	End-stage chronic renal failure	136,108	132,360	127,205	127,638	105,525	92,675	89,915	79,722	83,453	78,545
	Inflammatory or rare diseases or HIV or AIDS	1,302,858	1,486,927	1,504,716	1,555,082	1,479,614	1,436,906	1,429,144	1,457,773	1,473,509	1,347,288
	Liver or pancreatic diseases	461,146	428,069	392,064	350,616	335,090	295,137	284,619	265,656	259,814	240,067
	Neurological or degenerative diseases	1,942,533	2,409,381	2,556,265	2,387,721	2,097,733	1,957,818	1,775,650	1,728,172	1,744,962	1,787,834
	Psychiatric illnesses	5,368,761	5,474,137	4,737,868	4,078,321	3,530,604	3,113,311	2,801,501	2,571,054	2,399,883	2,077,016
	Psychotropic treatments	6,316,054	7,608,863	8,416,848	8,318,375	7,942,055	7,657,693	7,334,214	7,384,218	7,557,545	7,321,155
	Vascular risk treatments	7,613,148	9,829,293	11,534,845	11,857,824	12,122,995	12,346,350	12,309,596	13,061,666	13,986,157	14,313,414

- ➔ Cardio-neurovascular (5.1 million people treated in 2019)
- ➔ Diabetes (4.0 million)
- ➔ Chronic respiratory diseases (3.7 million)
- ➔ Cancer (3.3 million)
- ➔ Psychiatric diseases (2.5 million) Neurological or Degenerative diseases (1.7 million)
- ➔ Inflammatory or rare diseases or HIV/AIDS (1.3 million)
- ➔ Liver or pancreatic diseases (0.6 million) [CNAM, 2021]

❖ *Ranking of Weighted Number of Cases by Education Level*

Type	Disease_Group	Demographic_Indicator 1	2	3	4
incidence	Cancers	7	6	6	6
	Cardio-neurovascular diseases	1	2	3	3
	Chronic respiratory diseases	6	5	5	5
	Diabetes	8	9	9	9
	End-stage chronic renal failure	11	11	11	11
	Inflammatory or rare diseases or HIV or AIDS	9	8	8	8
	Liver or pancreatic diseases	10	10	10	10
	Neurological or degenerative diseases	5	7	7	7
	Psychiatric illnesses	4	4	4	4
	Psychotropic treatments	2	1	1	1
	Vascular risk treatments	3	3	2	2

Type	Disease_Group	Demographic_Indicator 1	2	3	4
prevalence	Cancers	7	6	5	5
	Cardio-neurovascular diseases	2	3	3	3
	Chronic respiratory diseases	8	7	6	6
	Diabetes	6	5	9	8
	End-stage chronic renal failure	11	11	11	11
	Inflammatory or rare diseases or HIV or AIDS	9	9	8	7
	Liver or pancreatic diseases	10	10	10	10
	Neurological or degenerative diseases	5	8	7	9
	Psychiatric illnesses	4	4	4	4
	Psychotropic treatments	3	2	2	2
	Vascular risk treatments	1	1	1	1

1	Pas de diplôme	No degree
2	BEP/CAP	Professional degree
3	Baccalauréat	High school dgree
4	Enseignement supérieur	Higher education

❖ *Ranking of Weighted Number of Cases by Profession*

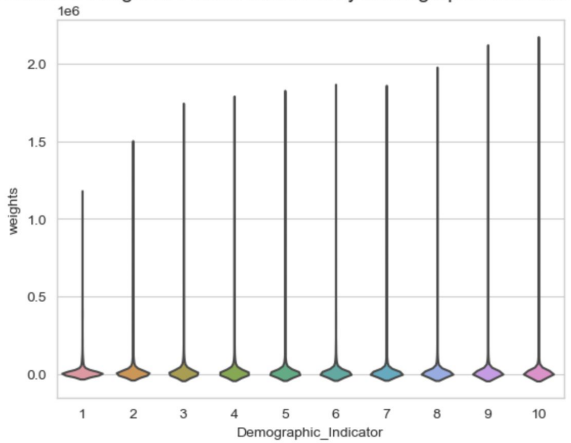
Type	Disease_Group	1	2	3	4	5	6	8
incidence	Cancers	7	7	6	6	7	7	8
	Cardio-neurovascular diseases	1	1	2	2	2	1	5
	Chronic respiratory diseases	6	5	4	5	5	5	2
	Diabetes	9	8	9	9	9	8	9
	End-stage chronic renal failure	11	11	11	11	11	11	11
	Inflammatory or rare diseases or HIV or AIDS	8	9	8	8	8	9	7
	Liver or pancreatic diseases	10	10	10	10	10	10	10
	Neurological or degenerative diseases	4	6	7	7	6	6	6
	Psychiatric illnesses	5	4	5	4	4	4	3
	Psychotropic treatments	2	2	1	1	1	2	1
	Vascular risk treatments	3	3	3	3	3	3	4

Type	Disease_Group	1	2	3	4	5	6	8
prevalence	Cancers	7	6	4	5	7	8	9
	Cardio-neurovascular diseases	2	2	3	3	3	2	5
	Chronic respiratory diseases	8	8	7	8	8	7	4
	Diabetes	6	4	6	6	5	5	8
	End-stage chronic renal failure	11	11	11	11	11	11	11
	Inflammatory or rare diseases or HIV or AIDS	9	9	8	9	9	9	7
	Liver or pancreatic diseases	10	10	10	10	10	10	10
	Neurological or degenerative diseases	4	7	9	7	6	6	6
	Psychiatric illnesses	5	5	5	4	4	4	1
	Psychotropic treatments	3	3	2	2	2	3	3
	Vascular risk treatments	1	1	1	1	1	1	2

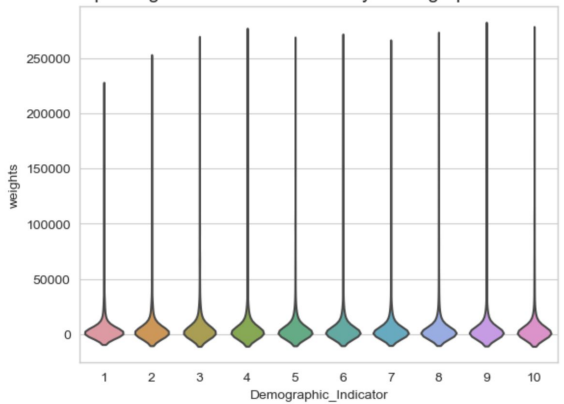
1	Agriculteurs exploitants	Farmers
2	Artisans, commerçants, chefs d'entreprise	Craftsmen, merchants, business owners
3	Cadres et professions intellectuelles supérieures	Executives and higher intellectual professions
4	Professions intermédiaires	Intermediate professions
5	Employés	Employees
6	Ouvriers	Workers
7	Retraités	Retired
8	Autres	Others

Violin Charts

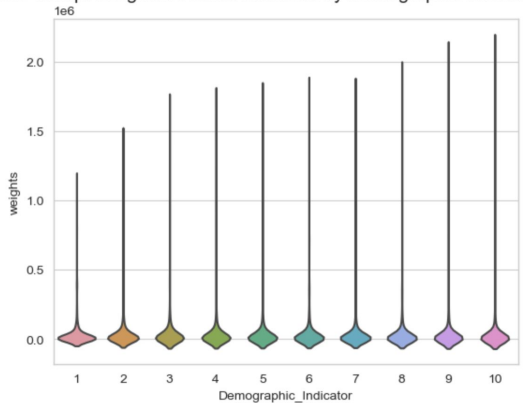
Whole Dataset: Weighted Patient Numbers by Demographic Indicator (Violin Plot)



Incidence Group: Weighted Patient Numbers by Demographic Indicator (Violin Plot)



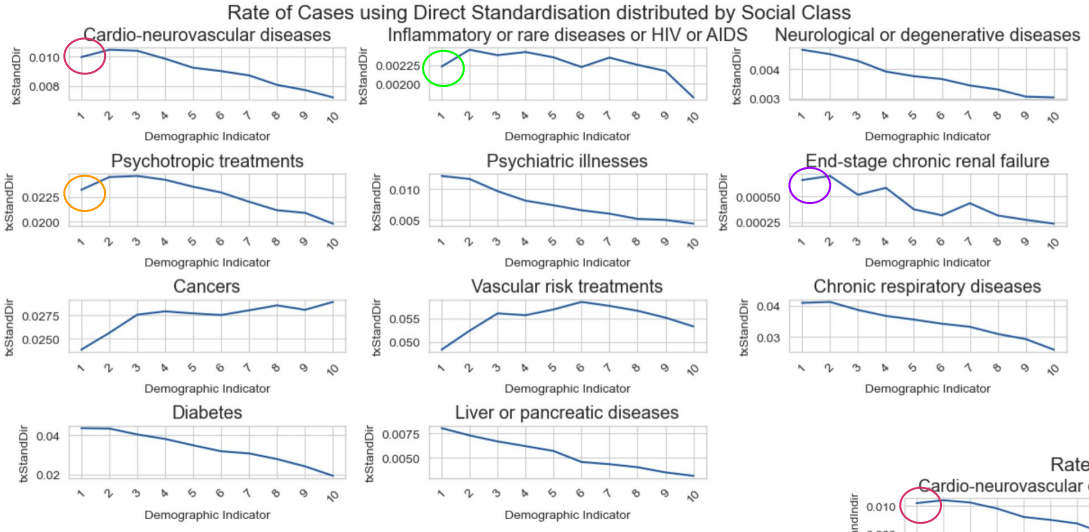
Prevalence Group: Weighted Patient Numbers by Demographic Indicator (Violin Plot)



5. Standardised Rate with the Direct & Indirect Method

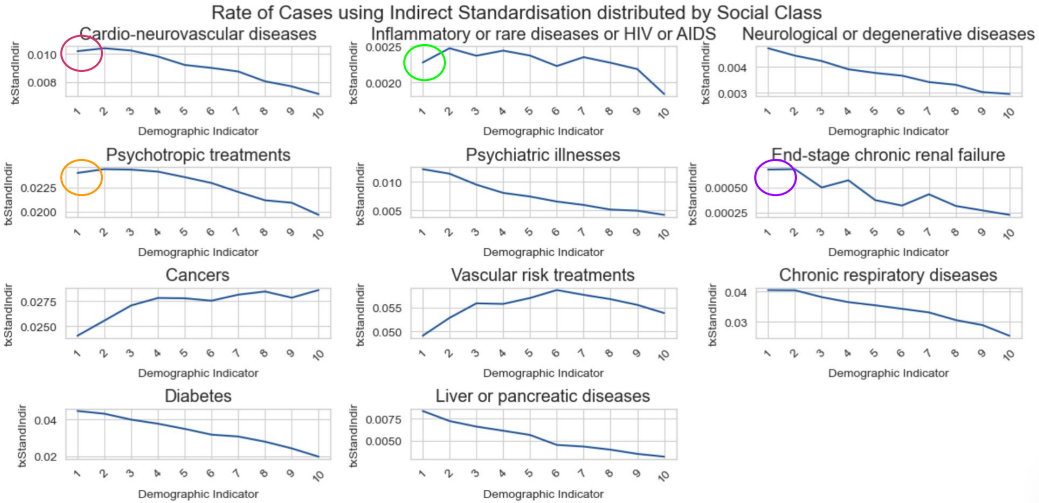
Epidemiological studies or healthcare research

Standardised Rate with the Direct Method and Standardised Rate with the Indirect Method to compare the **observed** and **expected** outcomes of a population or a group by accounting for differences in demographic or clinical characteristics between them.



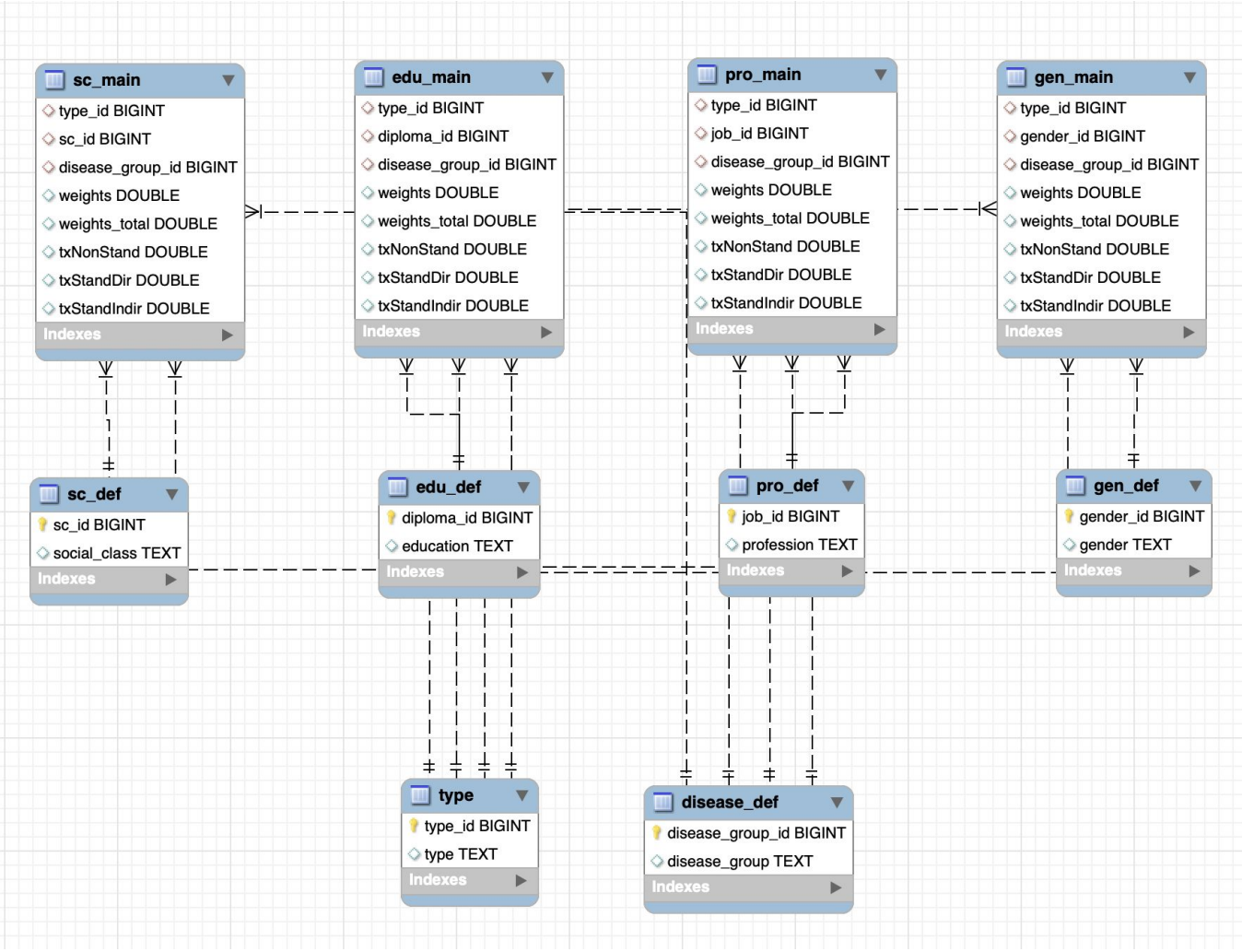
→ Adjusts for differences in the age structure.

Adjust for differences in a specific variable, such as education, profession & socioeconomic status. ←



SQL

1. Entity Relationship Diagram


sc_main
type_id BIGINT
sc_id BIGINT
disease_group_id BIGINT
weights DOUBLE
weights_total DOUBLE
txNonStand DOUBLE
txStandDir DOUBLE
txStandIndir DOUBLE
Indexes
edu_main
type_id BIGINT
diploma_id BIGINT
disease_group_id BIGINT
weights DOUBLE
weights_total DOUBLE
txNonStand DOUBLE
txStandDir DOUBLE
txStandIndir DOUBLE
Indexes
pro_main
type_id BIGINT
job_id BIGINT
disease_group_id BIGINT
weights DOUBLE
weights_total DOUBLE
txNonStand DOUBLE
txStandDir DOUBLE
txStandIndir DOUBLE
Indexes
gen_main
type_id BIGINT
gender_id BIGINT
disease_group_id BIGINT
weights DOUBLE
weights_total DOUBLE
txNonStand DOUBLE
txStandDir DOUBLE
txStandIndir DOUBLE
Indexes
sc_def
sc_id BIGINT
social_class TEXT
Indexes
edu_def
diploma_id BIGINT
education TEXT
Indexes
pro_def
job_id BIGINT
profession TEXT
Indexes
gen_def
gender_id BIGINT
gender TEXT
Indexes
type
type_id BIGINT
type TEXT
Indexes
disease_def
disease_group_id BIGINT
disease_group TEXT
Indexes

2. Queries

```sql
# Top 10 highest weighted number of new cases (incidence) for the poorest population
select sc_def.social_class, sc_main.type_id as disease_type, disease_def.disease_group as diease_name, sc_main.weights as highest_weights
from chronic_diseases.sc_main
left join chronic_diseases.disease_def on sc_main.disease_group_id=disease_def.disease_group_id
left join chronic_diseases.sc_def on sc_main.sc_id = sc_def.sc_id
where sc_main.type_id = 2 and sc_main.sc_id = 1
order by sc_main.weights desc limit 10;
```

social_class	disease_type	diease_name	highest_weights
10th decile (least affluent segment)	2	Psychotropic treatments	218766.553084868
10th decile (least affluent segment)	2	Chronic respiratory diseases	213565.92720769
10th decile (least affluent segment)	2	Psychotropic treatments	163763.34822732
10th decile (least affluent segment)	2	Vascular risk treatments	159227.810961478
10th decile (least affluent segment)	2	Vascular risk treatments	139291.372545082
10th decile (least affluent segment)	2	Psychotropic treatments	135866.886315331
10th decile (least affluent segment)	2	Cardio-neurovascular diseases	122229.751244613
10th decile (least affluent segment)	2	Chronic respiratory diseases	115861.708246178
10th decile (least affluent segment)	2	Psychiatric illnesses	109733.440878773
10th decile (least affluent segment)	2	Psychotropic treatments	105827.363807957

```sql
# Top 10 highest weighted number of incidence cases for the wealthiest population
select sc_def.social_class, sc_main.type_id as disease_type, disease_def.disease_group as diease_name, sc_main.weights as highest_weights
from chronic_diseases.sc_main
left join chronic_diseases.disease_def on sc_main.disease_group_id=disease_def.disease_group_id
left join chronic_diseases.sc_def on sc_main.sc_id = sc_def.sc_id
where sc_main.type_id = 2 and sc_main.sc_id = 10
order by sc_main.weights desc limit 10;
```

social_class	disease_type	diease_name	highest_weights
10th decile (most affluent segment)	2	Psychotropic treatments	267585.066322509
10th decile (most affluent segment)	2	Vascular risk treatments	207681.435790414
10th decile (most affluent segment)	2	Vascular risk treatments	179934.80629602
10th decile (most affluent segment)	2	Psychotropic treatments	174320.920325922
10th decile (most affluent segment)	2	Cardio-neurovascular diseases	165503.285047724
10th decile (most affluent segment)	2	Chronic respiratory diseases	163958.877236791
10th decile (most affluent segment)	2	Psychotropic treatments	158532.086183803
10th decile (most affluent segment)	2	Psychotropic treatments	154027.913491984
10th decile (most affluent segment)	2	Cancers	128193.395339375
10th decile (most affluent segment)	2	Psychotropic treatments	109052.980138706

```sql
# How many weighted number of patients each disease group has at each profession.
select dd.disease_group_id as "disease_id", dd.disease_group as "disease_group", pd.profession as "profession",
sum(pm.weights) as "Number of weighted number"
from chronic_diseases.disease_def dd
inner join chronic_diseases.pro_main pm
    on dd.disease_group_id = pm.disease_group_id
inner join chronic_diseases.pro_def pd
    on pm.job_id = pd.job_id
group by dd.disease_group_id, pd.profession
order by dd.disease_group_id asc;
```

38:16

Result Grid | Filter Rows: Search | Export:

disease_id	disease_group	profession	Number of weighted num...
1	Cardio-neurovascular diseases	Craftsmen, merchants, business owners	5143335.269752712
1	Cardio-neurovascular diseases	Employees	13918568.967766207
1	Cardio-neurovascular diseases	Executives and higher intellectual professions	5096274.879106675
1	Cardio-neurovascular diseases	Farmers	3399447.060048797
1	Cardio-neurovascular diseases	Intermediate professions	9340338.109076591
1	Cardio-neurovascular diseases	Others	4185668.6685799346
1	Cardio-neurovascular diseases	Workers	15960150.029456185
2	Inflammatory or rare diseases or HIV or AIDS	Craftsmen, merchants, business owners	615095.824984554
2	Inflammatory or rare diseases or HIV or AIDS	Employees	2970139.000938609
2	Inflammatory or rare diseases or HIV or AIDS	Executives and higher intellectual professions	980610.0714420708
2	Inflammatory or rare diseases or HIV or AIDS	Farmers	282767.3331271705
2	Inflammatory or rare diseases or HIV or AIDS	Intermediate professions	1963431.7828026263
2	Inflammatory or rare diseases or HIV or AIDS	Others	1722062.6232495978
2	Inflammatory or rare diseases or HIV or AIDS	Workers	1949805.103500285
3	Neurological or degenerative diseases	Craftsmen, merchants, business owners	1050995.8857681814
3	Neurological or degenerative diseases	Employees	4325366.733803928
3	Neurological or degenerative diseases	Executives and higher intellectual professions	1033596.6659461475
3	Neurological or degenerative diseases	Farmers	900935.541349854
3	Neurological or degenerative diseases	Intermediate professions	2311327.475184918
3	Neurological or degenerative diseases	Others	2904271.314835079
3	Neurological or degenerative diseases	Workers	3491694.606244939
4	Psychotropic treatments	Craftsmen, merchants, business owners	3914701.828916698
4	Psychotropic treatments	Employees	20411848.95241939

```sql
# The average weighted number of patients of each disease at each education group.
select dd.disease_group_id as "disease_id", dd.disease_group as "disease_group", ed.education as "education",
avg(em.weights) as "average_weights"
from chronic_diseases.disease_def dd
inner join chronic_diseases.edu_main em
    on dd.disease_group_id = em.disease_group_id
inner join chronic_diseases.edu_def ed
    on em.diploma_id = ed.diploma_id
group by dd.disease_group_id, dd.disease_group, ed.education
order by dd.disease_group asc;
```

76:42

Result Grid Filter Rows: Search Export:

disease_id	disease_group	education	average_weights
7	Cancers	Higher education	506755.8562264973
7	Cancers	High school dgree	309421.17751023255
7	Cancers	Professional degree	785693.8244984302
7	Cancers	No degree	808122.2953587114
1	Cardio-neurovascular diseases	No degree	359188.06427971163
1	Cardio-neurovascular diseases	Higher education	105869.62296305157
1	Cardio-neurovascular diseases	High school dgree	79681.08673818524
1	Cardio-neurovascular diseases	Professional degree	243080.516863615
9	Chronic respiratory diseases	High school dgree	335942.55222853326
9	Chronic respiratory diseases	Higher education	455634.5856764645
9	Chronic respiratory diseases	Professional degree	803351.2575363609
9	Chronic respiratory diseases	No degree	827820.2048055949
10	Diabetes	Higher education	340355.51058952766
10	Diabetes	No degree	1096592.0862263225
10	Diabetes	Professional degree	830403.382762995
10	Diabetes	High school dgree	253429.70705653736
6	End-stage chronic renal failure	High school dgree	3771.672723486887
6	End-stage chronic renal failure	Higher education	4991.950767167749
6	End-stage chronic renal failure	Professional degree	10472.557686542472
6	End-stage chronic renal failure	No degree	13978.666882120284
2	Inflammatory or rare diseases...	No degree	52554.5010033112
2	Inflammatory or rare diseases...	Professional degree	63468.05702918372
2	Inflammatory or rare diseases...	High school dgree	29176.642187995123

```sql
# The highest total weighted number of patients by disease group and gender
select a.disease_id, a.disease_group, max(a.total_weights) as max_total_weights, a.gender as "gender"
from
(
select dd.disease_group_id as "disease_id", dd.disease_group as "disease_group", sum(gm.weights) as "total_weights", gd.gender as "gender"
from chronic_diseases.disease_def dd
inner join chronic_diseases.gen_main gm
    on dd.disease_group_id = gm.disease_group_id
inner join chronic_diseases.gen_def gd
    on gm.gender_id = gd.gender_id
group by dd.disease_group_id, dd.disease_group, gd.gender
) as a
group by a.disease_id, a.disease_group, a.gender
order by max_total_weights desc;
```

34:35

Result Grid | Filter Rows: Search | Export:

disease_id	disease_group	max_total_weights	gender
8	Vascular risk treatments	25345657.373912442	F
4	Psychotropic treatments	20983903.922390006	F
8	Vascular risk treatments	18347333.252581067	M
1	Cardio-neurovascular diseases	16814660.6492235	M
1	Cardio-neurovascular diseases	12249989.322056545	F
4	Psychotropic treatments	11020402.260922272	M
5	Psychiatric illnesses	8251748.368059484	F
5	Psychiatric illnesses	6057794.726825952	M
3	Neurological or degenerative diseases	4709314.910996418	F
9	Chronic respiratory diseases	4599245.2364405105	F
9	Chronic respiratory diseases	4435060.949771437	M
10	Diabetes	4297189.045655614	M
7	Cancers	3942680.9838899574	F
10	Diabetes	3583967.112749868	F
7	Cancers	3475251.373446934	M
3	Neurological or degenerative diseases	3335631.093025194	M
2	Inflammatory or rare diseases or HIV...	3090135.78936938	F

Result Grid
Form Editor
Field Types
Query Stats

Machine Learning

1. Which psychotropic treatment will get highest weighted number of cases in France?

In [1928]:
```
psy_ml = pd.get_dummies(sc_psy, columns=['Disease_Name'])

psy_ml
```

tor	weights	txStandDir	txStandIndir	Disease_Name_Traitements antidépresseurs ou régulateurs de l'humeur	Disease_Name_Traitements anxiolytiques	Disease_Name_Traitements hypnotiques	Disease_Name_Traitements neuroleptiques	Disease_Name_Traitements psychotropes
1	218766.55	0.02	0.02	0	0	0	0	1
1	61497.39	0.01	0.01	0	0	1	0	0
2	242739.06	0.02	0.02	0	0	0	0	1
2	126362.29	0.01	0.01	1	0	0	0	0
2	67534.68	0.01	0.01	0	0	1	0	0
...	...	...	...	...	...	...	...	...
8	52063.81	0.01	0.01	1	0	0	0	0
9	57684.08	0.01	0.01	1	0	0	0	0
10	109052.98	0.02	0.02	0	0	0	0	1
10	57884.32	0.01	0.01	1	0	0	0	0

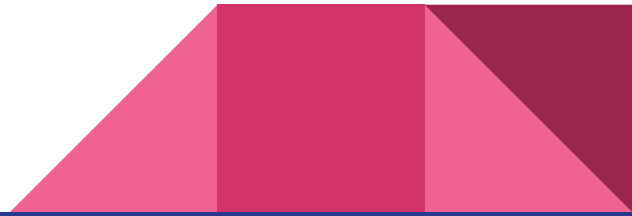
In [1935]:
```
from sklearn.model_selection import train_test_split

features =['Demographic_Indicator',"Disease_Name_Traitements antidépresseurs ou régulateurs de l'humeur",'Disease_Na
target =['weights']

X = psy_ml[features]
y = psy_ml[target]
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

print("Training set - X shape:", X_train.shape)
print("Training set - y shape:", y_train.shape)
print("Testing set - X shape:", X_test.shape)
print("Testing set - y shape:", y_test.shape)
```

```
Training set - X shape: (800, 6)
Training set - y shape: (800, 1)
Testing set - X shape: (200, 6)
Testing set - y shape: (200, 1)
```



```
regression_model = LinearRegression()
regression_model.fit(X_train, y_train)

# Predict the target variable for the testing set
y_pred = regression_model.predict(X_test)

# Evaluate the performance of the regression model
mse = mean_squared_error(y_test, y_pred)
rmse = np.sqrt(mse)
r_squared = regression_model.score(X_test, y_test)

print("Mean Squared Error (MSE):", mse)
print("Root Mean Squared Error (RMSE):", rmse)
print("R-squared:", r_squared)
```

```
Mean Squared Error (MSE): 1185094010.4238503
Root Mean Squared Error (RMSE): 34425.19441374079
R-squared: 0.09330110516750412
```

```
rf_model = RandomForestRegressor(n_estimators=100, random_state=42)

# Fit the model on the training data
rf_model.fit(X_train, y_train)

# Make predictions on the test data
y_pred = rf_model.predict(X_test)

# Evaluate the model
mse = mean_squared_error(y_test, y_pred)
rmse = np.sqrt(mse)
r2 = r2_score(y_test, y_pred)

print("Mean Squared Error (MSE):", mse)
print("Root Mean Squared Error (RMSE):", rmse)
print("R-squared:", r2)
```

```
Mean Squared Error (MSE): 1313788803.5539427
Root Mean Squared Error (RMSE): 36246.224680012434
R-squared: -0.005161485711693725
```

```
dt_regressor = DecisionTreeRegressor(random_state=42)

# Fit the model to the training data
dt_regressor.fit(X_train, y_train)

# Make predictions on the test data
y_pred = dt_regressor.predict(X_test)

# Evaluate the model
mse = mean_squared_error(y_test, y_pred)
rmse = np.sqrt(mse)
r2 = r2_score(y_test, y_pred)

# Print the evaluation metrics
print("Mean Squared Error (MSE):", mse)
print("Root Mean Squared Error (RMSE):", rmse)
print("R-squared:", r2)
```

```
Mean Squared Error (MSE): 1316895047.1464257
Root Mean Squared Error (RMSE): 36289.04858419997
R-squared: -0.00753802935094261
```

```
xgb_reg = xgb.XGBRegressor()

# Fit the model to the training data
xgb_reg.fit(X_train, y_train)

# Make predictions on the test data
y_pred = xgb_reg.predict(X_test)

# Evaluate the model
mse = mean_squared_error(y_test, y_pred)
rmse = np.sqrt(mse)
r2 = r2_score(y_test, y_pred)

# Print the evaluation metrics
print("Mean Squared Error (MSE):", mse)
print("Root Mean Squared Error (RMSE):", rmse)
print("R-squared:", r2)
```

```
Mean Squared Error (MSE): 1316895056.0203526
Root Mean Squared Error (RMSE): 36289.048706467256
R-squared: -0.007538036140260385
```

```
svr = SVR()

# Fit the model to the training data
svr.fit(X_train, y_train)

# Make predictions on the test data
y_pred = svr.predict(X_test)

# Evaluate the model
mse = mean_squared_error(y_test, y_pred)
rmse = np.sqrt(mse)
r2 = r2_score(y_test, y_pred)

# Print the evaluation metrics
print("Mean Squared Error (MSE):", mse)
print("Root Mean Squared Error (RMSE):", rmse)
print("R-squared:", r2)
```

```
Mean Squared Error (MSE): 1427536954.5857
Root Mean Squared Error (RMSE): 37782.76001810482
R-squared: -0.09218860923318362
```

2. Which social class will get highest weighted number of patients for psychotropic treatment in France?

```
features =['weights','txStandDir','txStandIndir','Demographic_Indicator',"Disease_Name_Traitements antidépresseurs o
target =['Demographic_Indicator']

# Split the data into training and testing sets
X = psy_ml[features]
y = psy_ml[target]
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

print("Training set - X shape:", X_train.shape)
print("Training set - y shape:", y_train.shape)
print("Testing set - X shape:", X_test.shape)
print("Testing set - y shape:", y_test.shape)
```

```
Training set - X shape: (800, 9)
Training set - y shape: (800, 1)
Testing set - X shape: (200, 9)
Testing set - y shape: (200, 1)
```

```
regression_model = LinearRegression()
regression_model.fit(X_train, y_train)

# Predict the target variable for the testing set
y_pred = regression_model.predict(X_test)

# Evaluate the performance of the regression model
mse = mean_squared_error(y_test, y_pred)
rmse = np.sqrt(mse)
r_squared = regression_model.score(X_test, y_test)

print("Mean Squared Error (MSE):", mse)
print("Root Mean Squared Error (RMSE):", rmse)
print("R-squared:", r_squared)
```

```
Mean Squared Error (MSE): 3.318040052222573e-22
Root Mean Squared Error (RMSE): 1.8215488058854126e-11
R-squared: 1.0
```

```
bine the predictions with the actual social classes
ctions = pd.DataFrame({'Predicted_weighted_number': y_pred, 'Social_Class': X_test['Demographic_Indicator

d the social class with the highest predicted weighted number
st_weighted_social_class = predictions.groupby('Social_Class')['Predicted_weighted_number'].sum().idxmax(

("Social class with the highest predicted weighted number for psychotropic treatment:", highest_weighted_
```

```
Social class with the highest predicted weighted number for psychotropic treatment: 9
```

```
xgb_reg = xgb.XGBRegressor()

# Fit the model to the training data
xgb_reg.fit(X_train, y_train)

# Make predictions on the test data
y_pred = xgb_reg.predict(X_test)

# Evaluate the model
mse = mean_squared_error(y_test, y_pred)
rmse = np.sqrt(mse)
r2 = r2_score(y_test, y_pred)

# Print the evaluation metrics
print("Mean Squared Error (MSE):", mse)
print("Root Mean Squared Error (RMSE):", rmse)
print("R-squared:", r2)
```

```
Mean Squared Error (MSE): 1.2759896605984976e-09
Root Mean Squared Error (RMSE): 3.5720997474853607e-05
R-squared: 0.9999999998367418
```

```
Combine the predictions with the actual social classes
redictions = pd.DataFrame({'Predicted_weighted_number': y_pred, 'Social_Class': X_test['Demographic_Indicato

Find the social class with the highest predicted weighted number
ighest_weighted_social_class = predictions.groupby('Social_Class')['Predicted_weighted_number'].sum().idxmax

rint("Social class with the highest predicted weighted number for psychotropic treatment:", highest_weighted
```

```
Social class with the highest predicted weighted number for psychotropic treatment: 9
```

```
rf_model = RandomForestRegressor(n_estimators=100, random_state=42)

# Fit the model on the training data
rf_model.fit(X_train, y_train)

# Make predictions on the test data
y_pred = rf_model.predict(X_test)

# Evaluate the model
mse = mean_squared_error(y_test, y_pred)
rmse = np.sqrt(mse)
r2 = r2_score(y_test, y_pred)

print("Mean Squared Error (MSE):", mse)
print("Root Mean Squared Error (RMSE):", rmse)
print("R-squared:", r2)
```

```
Mean Squared Error (MSE): 0.0
Root Mean Squared Error (RMSE): 0.0
R-squared: 1.0
```

```
ine the predictions with the actual social classes
tions = pd.DataFrame({'Predicted_weighted_number': y_pred, 'Social_Class': X_test['Demographic_Indicator']})

 the social class with the highest predicted weighted number
t_weighted_social_class = predictions.groupby('Social_Class')['Predicted_weighted_number'].sum().idxmax()

"Social class with the highest predicted weighted number for psychotropic treatment:", highest_weighted_social_class
```

```
Social class with the highest predicted weighted number for psychotropic treatment: 9
```

```
dt_regressor = DecisionTreeRegressor(random_state=42)

# Fit the model to the training data
dt_regressor.fit(X_train, y_train)

# Make predictions on the test data
y_pred = dt_regressor.predict(X_test)

# Evaluate the model
mse = mean_squared_error(y_test, y_pred)
rmse = np.sqrt(mse)
r2 = r2_score(y_test, y_pred)

# Print the evaluation metrics
print("Mean Squared Error (MSE):", mse)
print("Root Mean Squared Error (RMSE):", rmse)
print("R-squared:", r2)
```

```
Mean Squared Error (MSE): 0.0
Root Mean Squared Error (RMSE): 0.0
R-squared: 1.0
```

```
ombine the predictions with the actual social classes
dictions = pd.DataFrame({'Predicted_weighted_number': y_pred, 'Social_Class': X_test['Demographic_Indicator']

ind the social class with the highest predicted weighted number
hest_weighted_social_class = predictions.groupby('Social_Class')['Predicted_weighted_number'].sum().idxmax()

nt("Social class with the highest predicted weighted number for psychotropic treatment:", highest_weighted_so
```

```
Social class with the highest predicted weighted number for psychotropic treatment: 9
```

Conclusion