

Définition du besoin client

Réalisation d'un moteur de recommandation par mot-clés

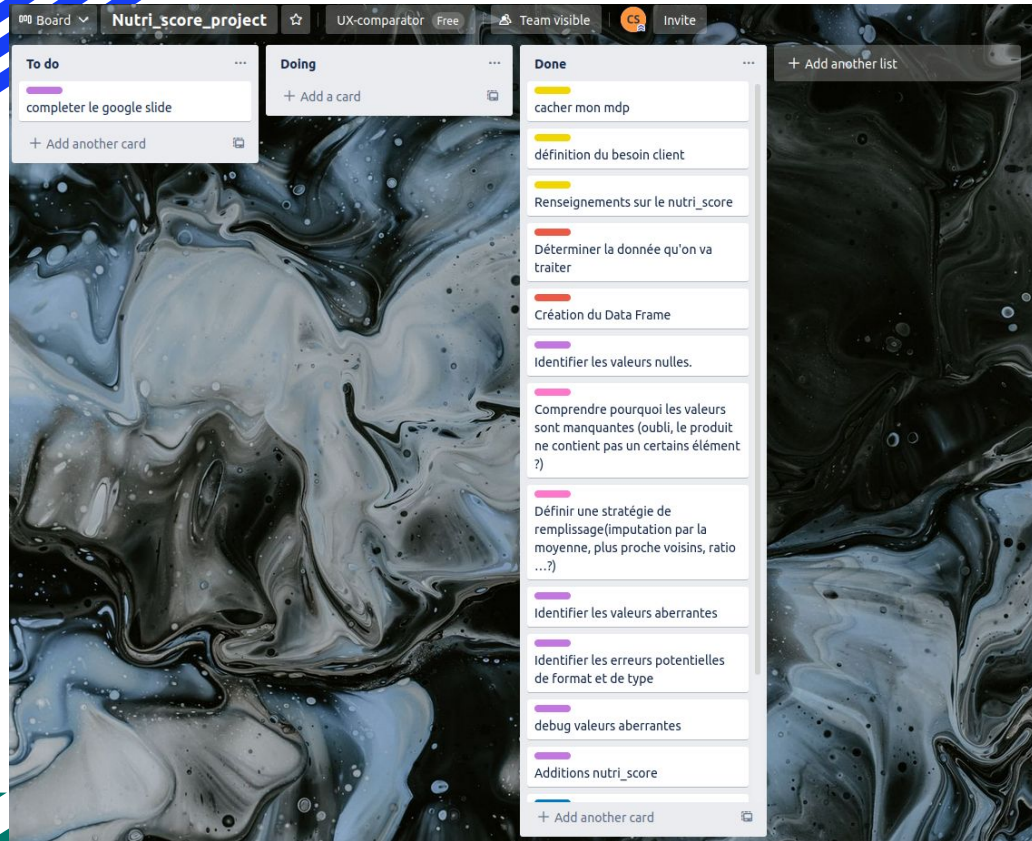
Nutri_score

CALCUL DU NUTRI-SCORE

Attribution des points à différents facteurs nutritionnels

01 Nutriments à limiter

Produit	Énergie (kJ)	Énergie (kcal)	Sodium (mg)	Sucre (g)	Gras saturés (g)	Points
0	≤ 400	≤ 95	≤ 125	≤ 5	≤ 3	≤ 10
1	≤ 800	≤ 190	≤ 250	≤ 10	≤ 6	≤ 20
2	≤ 1200	≤ 285	≤ 375	≤ 15	≤ 9	≤ 30
3	≤ 1600	≤ 380	≤ 500	≤ 20	≤ 12	≤ 40
4	≤ 2000	≤ 475	≤ 625	≤ 25	≤ 15	≤ 50
5	≤ 2400	≤ 570	≤ 750	≤ 30	≤ 18	≤ 60
6	≤ 2800	≤ 665	≤ 875	≤ 35	≤ 21	≤ 70
7	≤ 3200	≤ 760	≤ 1000	≤ 40	≤ 24	≤ 80
8	≤ 3600	≤ 855	≤ 1125	≤ 45	≤ 27	≤ 90
9	≤ 4000	≤ 950	≤ 1250	≤ 50	≤ 30	≤ 100
10	≤ 4400	≤ 1045	≤ 1375	≤ 55	≤ 33	≤ 110
11	≤ 4800	≤ 1140	≤ 1500	≤ 60	≤ 36	≤ 120
12	≤ 5200	≤ 1235	≤ 1625	≤ 65	≤ 39	≤ 130
13	≤ 5600	≤ 1330	≤ 1750	≤ 70	≤ 42	≤ 140
14	≤ 6000	≤ 1425	≤ 1875	≤ 75	≤ 45	≤ 150
15	≤ 6400	≤ 1520	≤ 2000	≤ 80	≤ 48	≤ 160
16	≤ 6800	≤ 1615	≤ 2125	≤ 85	≤ 51	≤ 170
17	≤ 7200	≤ 1710	≤ 2250	≤ 90	≤ 54	≤ 180
18	≤ 7600	≤ 1805	≤ 2375	≤ 95	≤ 57	≤ 190
19	≤ 8000	≤ 1900	≤ 2500	≤ 100	≤ 60	≤ 200
20	≤ 8400	≤ 1995	≤ 2625	≤ 105	≤ 63	≤ 210
21	≤ 8800	≤ 2090	≤ 2750	≤ 110	≤ 66	≤ 220
22	≤ 9200	≤ 2185	≤ 2875	≤ 115	≤ 69	≤ 230
23	≤ 9600	≤ 2280	≤ 3000	≤ 120	≤ 72	≤ 240
24	≤ 10000	≤ 2375	≤ 3125	≤ 125	≤ 75	≤ 250
25	≤ 10400	≤ 2470	≤ 3250	≤ 130	≤ 78	≤ 260
26	≤ 10800	≤ 2565	≤ 3375	≤ 135	≤ 81	≤ 270
27	≤ 11200	≤ 2660	≤ 3500	≤ 140	≤ 84	≤ 280
28	≤ 11600	≤ 2755	≤ 3625	≤ 145	≤ 87	≤ 290
29	≤ 12000	≤ 2850	≤ 3750	≤ 150	≤ 90	≤ 300
30	≤ 12400	≤ 2945	≤ 3875	≤ 155	≤ 93	≤ 310
31	≤ 12800	≤ 3040	≤ 4000	≤ 160	≤ 96	≤ 320
32	≤ 13200	≤ 3135	≤ 4125	≤ 165	≤ 99	≤ 330
33	≤ 13600	≤ 3230	≤ 4250	≤ 170	≤ 102	≤ 340
34	≤ 14000	≤ 3325	≤ 4375	≤ 175	≤ 105	≤ 350
35	≤ 14400	≤ 3420	≤ 4500	≤ 180	≤ 108	≤ 360
36	≤ 14800	≤ 3515	≤ 4625	≤ 185	≤ 111	≤ 370
37	≤ 15200	≤ 3610	≤ 4750	≤ 190	≤ 114	≤ 380
38	≤ 15600	≤ 3705	≤ 4875	≤ 195	≤ 117	≤ 390
39	≤ 16000	≤ 3800	≤ 5000	≤ 200	≤ 120	≤ 400
40	≤ 16400	≤ 3895	≤ 5125	≤ 205	≤ 123	≤ 410
41	≤ 16800	≤ 3990	≤ 5250	≤ 210	≤ 126	≤ 420
42	≤ 17200	≤ 4085	≤ 5375	≤ 215	≤ 129	≤ 430
43	≤ 17600	≤ 4180	≤ 5500	≤ 220	≤ 132	≤ 440
44	≤ 18000	≤ 4275	≤ 5625	≤ 225	≤ 135	≤ 450
45	≤ 18400	≤ 4370	≤ 5750	≤ 230	≤ 138	≤ 460
46	≤ 18800	≤ 4465	≤ 5875	≤ 235	≤ 141	≤ 470
47	≤ 19200	≤ 4560	≤ 6000	≤ 240	≤ 144	≤ 480
48	≤ 19600	≤ 4655	≤ 6125	≤ 245	≤ 147	≤ 490
49	≤ 20000	≤ 4750	≤ 6250	≤ 250	≤ 150	≤ 500
50	≤ 20400	≤ 4845	≤ 6375	≤ 255	≤ 153	≤ 510
51	≤ 20800	≤ 4940	≤ 6500	≤ 260	≤ 156	≤ 520
52	≤ 21200	≤ 5035	≤ 6625	≤ 265	≤ 159	≤ 530
53	≤ 21600	≤ 5130	≤ 6750	≤ 270	≤ 162	≤ 540
54	≤ 22000	≤ 5225	≤ 6875	≤ 275	≤ 165	≤ 550
55	≤ 22400	≤ 5320	≤ 7000	≤ 280	≤ 168	≤ 560
56	≤ 22800	≤ 5415	≤ 7125	≤ 285	≤ 171	≤ 570
57	≤ 23200	≤ 5510	≤ 7250	≤ 290	≤ 174	≤ 580
58	≤ 23600	≤ 5605	≤ 7375	≤ 295	≤ 177	≤ 590
59	≤ 24000	≤ 5700	≤ 7500	≤ 300	≤ 180	≤ 600
60	≤ 24400	≤ 5795	≤ 7625	≤ 305	≤ 183	≤ 610
61	≤ 24800	≤ 5890	≤ 7750	≤ 310	≤ 186	≤ 620
62	≤ 25200	≤ 5985	≤ 7875	≤ 315	≤ 189	≤ 630
63	≤ 25600	≤ 6080	≤ 8000	≤ 320	≤ 192	≤ 640
64	≤ 26000	≤ 6175	≤ 8125	≤ 325	≤ 195	≤ 650
65	≤ 26400	≤ 6270	≤ 8250	≤ 330	≤ 198	≤ 660
66	≤ 26800	≤ 6365	≤ 8375	≤ 335	≤ 201	≤ 670
67	≤ 27200	≤ 6460	≤ 8500	≤ 340	≤ 204	≤ 680
68	≤ 27600	≤ 6555	≤ 8625	≤ 345	≤ 207	≤ 690
69	≤ 28000	≤ 6650	≤ 8750	≤ 350	≤ 210	≤ 700
70	≤ 28400	≤ 6745	≤ 8875	≤ 355	≤ 213	≤ 710
71	≤ 28800	≤ 6840	≤ 9000	≤ 360	≤ 216	≤ 720
72	≤ 29200	≤ 6935	≤ 9125	≤ 365	≤ 219	≤ 730
73	≤ 29600	≤ 7030	≤ 9250	≤ 370	≤ 222	≤ 740
74	≤ 30000	≤ 7125	≤ 9375	≤ 375	≤ 225	≤ 750
75	≤ 30400	≤ 7220	≤ 9500	≤ 380	≤ 228	≤ 760
76	≤ 30800	≤ 7315	≤ 9625	≤ 385	≤ 231	≤ 770
77	≤ 31200	≤ 7410	≤ 9750	≤ 390	≤ 234	≤ 780
78	≤ 31600	≤ 7505	≤ 9875	≤ 395	≤ 237	≤ 790
79	≤ 32000	≤ 7600	≤ 10000	≤ 400	≤ 240	≤ 800
80	≤ 32400	≤ 7695	≤ 10125	≤ 405	≤ 243	≤ 810
81	≤ 32800	≤ 7790	≤ 10250	≤ 410	≤ 246	≤ 820
82	≤ 33200	≤ 7885	≤ 10375	≤ 415	≤ 249	≤ 830
83	≤ 33600	≤ 7980	≤ 10500	≤ 420	≤ 252	≤ 840
84	≤ 34000	≤ 8075	≤ 10625	≤ 425	≤ 255	≤ 850
85	≤ 34400	≤ 8170	≤ 10750	≤ 430	≤ 258	≤ 860
86	≤ 34800	≤ 8265	≤ 10875	≤ 435	≤ 261	≤ 870
87	≤ 35200	≤ 8360	≤ 11000	≤ 440	≤ 264	≤ 880
88	≤ 35600	≤ 8455	≤ 11125	≤ 445	≤ 267	≤ 890
89	≤ 36000	≤ 8550	≤ 11250	≤ 450	≤ 270	≤ 900
90	≤ 36400	≤ 8645	≤ 11375	≤ 455	≤ 273	≤ 910
91	≤ 36800	≤ 8740	≤ 11500	≤ 460	≤ 276	≤ 920
92	≤ 37200	≤ 8835	≤ 11625	≤ 465	≤ 279	≤ 930
93	≤ 37600	≤ 8930	≤ 11750	≤ 470	≤ 282	≤ 940
94	≤ 38000	≤ 9025	≤ 11875	≤ 475	≤ 285	≤ 950
95	≤ 38400	≤ 9120	≤ 12000	≤ 480	≤ 288	≤ 960
96	≤ 38800	≤ 9215	≤ 12125	≤ 485	≤ 291	≤ 970
97	≤ 39200	≤ 9310	≤ 12250	≤ 490	≤ 294	≤ 980
98	≤ 39600	≤ 9405	≤ 12375	≤ 495	≤ 297	≤ 990
99	≤ 40000	≤ 9500	≤ 12500	≤ 500	≤ 300	≤ 1000
100	≤ 40400	≤ 9595	≤ 12625	≤ 505	≤ 303	≤ 1010
101	≤ 40800	≤ 9690	≤ 12750	≤ 510	≤ 306	≤ 1020
102	≤ 41200	≤ 9785	≤ 12875	≤ 515	≤ 309	≤ 1030
103	≤ 41600	≤ 9880	≤ 13000	≤ 520	≤ 312	≤ 1040
104	≤ 42000	≤ 9975	≤ 13125	≤ 525	≤ 315	≤ 1050
105	≤ 42400	≤ 10070	≤ 13250	≤ 530	≤ 318	≤ 1060
106	≤ 42800	≤ 10165	≤ 13375	≤ 535	≤ 321	≤ 1070
107	≤ 43200	≤ 10260	≤ 13500	≤ 540	≤ 324	≤ 1080
108	≤ 43600	≤ 10355	≤ 13625	≤ 545	≤ 327	≤ 1090
109	≤ 44000	≤ 10450	≤ 13750	≤ 550	≤ 330	≤ 1100
110	≤ 44400	≤ 10545	≤ 13875	≤ 555	≤ 333	≤ 1110
111	≤ 44800	≤ 10640	≤ 14000	≤ 560	≤ 336	≤ 1120
112	≤ 45200	≤ 10735	≤ 14125	≤ 565	≤ 339	≤ 1130
113	≤ 45600	≤ 10830	≤ 14250	≤ 570	≤ 342	≤ 1140
114	≤ 46000	≤ 10925	≤ 14375	≤ 575	≤ 345	≤ 1150
115	≤ 46400	≤ 11020	≤ 14500	≤ 580	≤ 348	≤ 1160
116	≤ 46800	≤ 11115	≤ 14625	≤ 585	≤ 351	≤ 1170
117	≤ 47200	≤ 11210	≤ 14750	≤ 590	≤ 354	≤ 1180
118	≤ 47600	≤ 11305	≤ 14875	≤ 595	≤ 357	≤ 1190
119	≤ 48000	≤ 11400	≤ 15000	≤ 600	≤ 360	≤ 1200
120	≤ 48400	≤ 11495	≤ 15125	≤ 605	≤ 363	≤ 1210
121	≤ 48800	≤ 11590	≤ 15250	≤ 610	≤ 366	≤ 1220
122	≤ 49200	≤ 11685	≤ 15375	≤ 615	≤ 369	≤ 1230
123	≤ 49600	≤ 11780	≤ 15500	≤ 620	≤ 372	≤ 1240
124	≤ 50000	≤ 11875	≤ 15625	≤ 625	≤ 375	≤ 1250
125	≤ 50400	≤ 11970	≤ 15750	≤ 630	≤ 378	≤ 1260
126	≤ 50800	≤ 12065	≤ 15875	≤ 635	≤ 381	≤ 1270
127	≤ 51200	≤ 12160	≤ 16000	≤ 640	≤ 384	≤ 1280
128	≤ 51600	≤ 12255	≤ 16125	≤ 645	≤ 387	≤ 1290
129	≤ 52000	≤ 12350	≤ 16250	≤ 650	≤ 390	≤ 1300
130	≤ 52400	≤ 12445	≤ 16375	≤ 655	≤ 393	≤ 1310
131	≤ 52800	≤ 12540	≤ 16500	≤ 660	≤ 396	≤ 1320
132	≤ 53200	≤ 12635	≤ 16625	≤ 665	≤ 399	≤ 1330
133	≤ 53600	≤ 12730	≤ 16750	≤ 670	≤ 402	≤ 1340
134	≤ 54000	≤ 12825	≤ 16875	≤ 675	≤ 405	≤ 1350
135	≤ 54400	≤ 12920	≤ 17000	≤ 680	≤ 408	≤ 1360
136	≤ 54800	≤ 13015	≤ 17125	≤ 685	≤ 411	≤ 1370
137	≤ 55200	≤ 13110	≤ 17250	≤ 690	≤ 414	≤ 1380
138	≤ 55600	≤ 13205	≤ 17375	≤ 695	≤ 417	≤ 1390
139	≤ 56000	≤ 13300	≤ 17500	≤ 700	≤ 420	≤ 1400
140	≤ 56400	≤ 13395	≤ 17625	≤ 705	≤ 423	≤ 1410
141	≤ 56800	≤ 13490	≤ 17750	≤ 710	≤ 426	≤ 1420
142	≤ 57200					



J'organise mon travail sur trello

```
main.py x __init__.py x conf.py x
1
2 import pandas as pd
3 import seaborn as sns
4 import matplotlib.pyplot as plt
5 import mysql.connector
6 import sqlalchemy
7 import sys
8 sys.path.insert(0, "/home/apprenant/PycharmProjects/db_recommandation/conf/conf.py")
9
10 def mysql_connect():
11     from conf.conf import mysql_pseudo, mysql_mdp
12     mysql_username = mysql_pseudo
13     mysql_password = mysql_mdp
14     database_name = 'nutri_score'
15     database_connection = sqlalchemy.create_engine(
16         'mysql+mysqlconnector://{0}:{1}@localhost/{2}'.format(mysql_username, mysql_password, database_name),
17         pool_recycle=1, pool_timeout=57600).connect()
18     return database_connection
19 print("done")
20 database_connection = mysql_connect()
21 print("done")
22 df = pd.read_csv("/home/apprenant/Downloads/recommandation.tsv", sep='\t', low_memory=False)
23 print("done")
24 #df.to_sql('nutri_all', database_connection, if_exists='replace', index=False)
25 print("done")
```

db_recommandation > main.py

- Project
- db_recommandation ~/PycharmProjects/db_recommandation
 - .ipynb_checkpoints
 - conf
 - __init__.py
 - conf.py
 - Analyse Nutriscore.ipynb
 - main.py
 - Traitement Nutriscore.ipynb
- External Libraries
- Scratches and Consoles

Je connecte ma data à sql et je lis mon fichier tsv



Analyse

Structure de table Vue relationnelle

#	Nom	Type	Interclassement	Attributs	Null	Valeur par défaut	Commentaires	Extra	Action
<input type="checkbox"/>	1 code	text	utf8mb4_0900_ai_ci		Oui				Modifier Supprimer Plus
<input type="checkbox"/>	2 url	text	utf8mb4_0900_ai_ci		Oui				Modifier Supprimer Plus
<input type="checkbox"/>	3 creator	text	utf8mb4_0900_ai_ci		Oui				Modifier Supprimer Plus
<input type="checkbox"/>	4 created_t	text	utf8mb4_0900_ai_ci		Oui				Modifier Supprimer Plus
<input type="checkbox"/>	5 created_datetime	text	utf8mb4_0900_ai_ci		Oui				Modifier Supprimer Plus
<input type="checkbox"/>	6 last_modified_t	text	utf8mb4_0900_ai_ci		Oui				Modifier Supprimer Plus
<input type="checkbox"/>	7 last_modified_datetime	text	utf8mb4_0900_ai_ci		Oui				Modifier Supprimer Plus
<input type="checkbox"/>	8 product_name	text	utf8mb4_0900_ai_ci		Oui				Modifier Supprimer Plus
<input type="checkbox"/>	9 generic_name	text	utf8mb4_0900_ai_ci		Oui				Modifier Supprimer Plus
<input type="checkbox"/>	10 quantity	text	utf8mb4_0900_ai_ci		Oui				Modifier Supprimer Plus
<input type="checkbox"/>	11 packaging	text	utf8mb4_0900_ai_ci		Oui				Modifier Supprimer Plus
<input type="checkbox"/>	12 packaging_tags	text	utf8mb4_0900_ai_ci		Oui				Modifier Supprimer Plus
<input type="checkbox"/>	13 brands	text	utf8mb4_0900_ai_ci		Oui				Modifier Supprimer Plus
<input type="checkbox"/>	14 brands_tags	text	utf8mb4_0900_ai_ci		Oui				Modifier Supprimer Plus
<input type="checkbox"/>	15 categories	text	utf8mb4_0900_ai_ci		Oui				Modifier Supprimer Plus
<input type="checkbox"/>	16 categories_tags	text	utf8mb4_0900_ai_ci		Oui				Modifier Supprimer Plus
<input type="checkbox"/>	17 categories_en	text	utf8mb4_0900_ai_ci		Oui				Modifier Supprimer Plus
<input type="checkbox"/>	18 origins	text	utf8mb4_0900_ai_ci		Oui				Modifier Supprimer Plus
<input type="checkbox"/>	19 origins_tags	text	utf8mb4_0900_ai_ci		Oui				Modifier Supprimer Plus
<input type="checkbox"/>	20 manufacturing_places	text	utf8mb4_0900_ai_ci		Oui				Modifier Supprimer Plus
<input type="checkbox"/>	21 manufacturing_places_tags	text	utf8mb4_0900_ai_ci		Oui				Modifier Supprimer Plus
<input type="checkbox"/>	22 labels	text	utf8mb4_0900_ai_ci		Oui				Modifier Supprimer Plus
<input type="checkbox"/>	23 labels_tags	text	utf8mb4_0900_ai_ci		Oui				Modifier Supprimer Plus
<input type="checkbox"/>	24 labels_en	text	utf8mb4_0900_ai_ci		Oui				Modifier Supprimer Plus
<input type="checkbox"/>	25 emb_codes	text	utf8mb4_0900_ai_ci		Oui				Modifier Supprimer Plus
<input type="checkbox"/>	26 emb_codes_tags	text	utf8mb4_0900_ai_ci		Oui				Modifier Supprimer Plus
<input type="checkbox"/>	27 first_packaging_code_geo	text	utf8mb4_0900_ai_ci		Oui				Modifier Supprimer Plus

Je visualise ma data sur PHPmyAdmin

Importation du fichier CSV et des packages + Mise en lien avec Pycharm

```
In [1]: #!/pip3 install seaborn
```

```
In [2]: import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
print("Setup Complete")
```

Setup Complete

```
In [3]: df = pd.read_csv("/home/apprenant/Downloads/recommandation.tsv", sep='\t', low_memory=False)
```


ANALYSE DE MA DONNÉE

```
In [4]: def info_df(df):  
        print("Dataset : ",  
              dict([(v, df[v].apply(lambda r: len(str(r)) if r != None else 0).max()) for v in df.columns.values]))  
  
        print(info_df(df))
```

```
Dataset : {'code': 40, 'url': 652, 'creator': 68, 'created_t': 37, 'created_datetime': 46, 'last_modified_t': 31,  
'last_modified_datetime': 35, 'product_name': 234, 'generic_name': 870, 'quantity': 194, 'packaging': 248, 'packag  
ing_tags': 240, 'brands': 201, 'brands_tags': 198, 'categories': 651, 'categories_tags': 771, 'categories_en': 65  
1, 'origins': 4240, 'origins_tags': 462, 'manufacturing_places': 558, 'manufacturing_places_tags': 539, 'labels':  
666, 'labels_tags': 679, 'labels_en': 676, 'emb_codes': 331, 'emb_codes_tags': 308, 'first_packaging_code_geo': 1  
9, 'cities': 3, 'cities_tags': 247, 'purchase_places': 89, 'stores': 341, 'countries': 328, 'countries_tags': 286,  
'countries_en': 204, 'ingredients_text': 4468, 'allergens': 672, 'allergens_en': 77, 'traces': 229, 'traces_tags':  
238, 'traces_en': 238, 'serving_size': 171, 'no_nutriments': 3, 'additives_n': 4, 'additives': 141411, 'additives  
tags': 281, 'additives_en': 869, 'ingredients_from_palm_oil_n': 3, 'ingredients_from_palm_oil': 3, 'ingredients_fr  
om_palm_oil_tags': 58, 'ingredients_that_may_be_from_palm_oil_n': 3, 'ingredients_that_may_be_from_palm_oil': 3, '  
ingredients_that_may_be_from_palm_oil_tags': 315, 'nutrition_grade_uk': 3, 'nutrition_grade_fr': 3, 'pnns_groups_1  
' : 23, 'pnns_groups_2': 32, 'states': 390, 'states_tags': 378, 'states_en': 339, 'main_category': 191, 'main_categ  
ory_en': 191, 'image_url': 87, 'image_small_url': 87, 'energy_100g': 8, 'energy-from-fat_100g': 6, 'fat_100g': 13,  
'saturated-fat_100g': 7, '-butyric-acid_100g': 3, '-caproic-acid_100g': 3, '-caprylic-acid_100g': 3, '-capric-acid  
_100g': 4, '-lauric-acid_100g': 7, '-myristic-acid_100g': 4, '-palmitic-acid_100g': 3, '-stearic-acid_100g': 3, '-  
arachidic-acid_100g': 5, '-behenic-acid_100g': 4, '-lignoceric-acid_100g': 3, '-cerotic-acid_100g': 3, '-montanic  
-acid_100g': 4, '-melissic-acid_100g': 3, 'monounsaturated-fat_100g': 6, 'polyunsaturated-fat_100g': 7, 'omega-3-fa  
t_100g': 7, '-alpha-linolenic-acid_100g': 7, '-eicosapentaenoic-acid_100g': 5, '-docosahexaenoic-acid_100g': 6, 'o  
mega-6-fat_100g': 5, '-linoleic-acid_100g': 6, '-arachidonic-acid_100g': 6, '-gamma-linolenic-acid_100g': 6, '-dih  
omo-gamma-linolenic-acid_100g': 18, 'omega-9-fat_100g': 4, '-oleic-acid_100g': 4, '-elaidic-acid_100g': 3, '-gondo  
ic-acid_100g': 8, '-mead-acid_100g': 3, '-erucic-acid_100g': 3, '-nervonic-acid_100g': 3, 'trans-fat_100g': 7, 'ch  
olesterol_100g': 14, 'carbohydrates_100g': 13, 'sugars_100g': 12, '-sucrose_100g': 4, '-glucose_100g': 4, '-fructo  
se_100g': 5, '-lactose_100g': 5, '-maltose_100g': 4, '-maltodextrins_100g': 4, 'starch_100g': 5, 'polyols_100g':  
5, 'fiber_100g': 13, 'proteins_100g': 13, 'casein_100g': 4, 'serum-proteins_100g': 3, 'nucleotides_100g': 6, 'salt  
_100g': 18, 'sodium_100g': 20, 'alcohol_100g': 5, 'vitamin-a_100g': 10, 'beta-carotene_100g': 8, 'vitamin-d_100g':  
10, 'vitamin-e_100g': 8, 'vitamin-k_100g': 9, 'vitamin-c_100g': 8, 'vitamin-b1_100g': 10, 'vitamin-b2_100g': 9, 'v  
itamin-pp_100g': 9, 'vitamin-b6_100g': 8, 'vitamin-b9_100g': 9, 'folates_100g': 10, 'vitamin-b12_100g': 10, 'bioti  
n_100g': 8, 'pantothenic-acid_100g': 9, 'silica_100g': 8, 'bicarbonate_100g': 8, 'potassium_100g': 9, 'chloride_10  
0g': 8, 'calcium_100g': 8, 'phosphorus_100g': 7, 'iron_100g': 8, 'magnesium_100g': 9, 'zinc_100g': 8, 'copper_100g  
' : 9, 'manganese_100g': 8, 'fluoride_100g': 8, 'selenium_100g': 8, 'chromium_100g': 8, 'molybdenum_100g': 8, 'iodi  
ne_100g': 8, 'caffeine_100g': 7, 'taurine_100g': 7, 'ph_100g': 6, 'fruits-vegetables-nuts_100g': 5, 'fruits-vegeta  
bles-nuts-estimate_100g': 5, 'collagen-meat-protein-ratio_100g': 4, 'cocoa_100g': 5, 'chlorophyll_100g': 3, 'carbon  
-footprint_100g': 9, 'nutrition-score-fr_100g': 5, 'nutrition-score-uk_100g': 5, 'glycemic-index_100g': 3, 'water-  
hardness_100g': 3}  
None
```



```
In [7]: df.iloc[:, [0,66]]
```

```
Out[7]:
```

	code	saturated-fat_100g
0	0000000003087	NaN
1	0000000004530	28.57
2	0000000004559	0.00
3	0000000016087	5.36
4	0000000016094	NaN
...
356022	99567453	0.00
356023	9970229501521	NaN
356024	9977471758307	NaN
356025	9980282863788	NaN
356026	999990026839	NaN

Je regarde en détail la colonne saturated_fat

#J'hésitais à inclure cette colonne dans mon dataframe. J'ai donc voulu en savoir plus. Savoir si elle était vraiment utile.

```
In [8]: print(df['fruits-vegetables-nuts_100g'].shape)
print(df['fruits-vegetables-nuts_100g'].isnull().sum())

(356027,)
352799
```

#Je sélectionne les colonnes qui me semblent essentielles pour le nutri_score et je crée un dataframe indépendant pour passer au nettoyage et au traitement de ma data.

```
In [9]: df_nutri_score = df[['product_name', 'energy_100g', 'sugars_100g', 'fiber_100g', 'proteins_100g', 'sodium_100g', 'saturated-fat_100g', 'nutrition-score-fr_100g', 'nutrition-grade_fr']]
```

#Visualiser mes colonnes

```
In [10]: print(df_nutri_score.columns)

Index(['product_name', 'energy_100g', 'sugars_100g', 'fiber_100g',
       'proteins_100g', 'sodium_100g', 'saturated-fat_100g',
       'nutrition-score-fr_100g', 'nutrition-grade_fr'],
      dtype='object')
```

Je regarde le nombre de valeurs nulles dans mes colonnes

#Je renomme mes colonnes

```
In [12]: df_nutri_score = df_nutri_score.rename(columns={'energy_100g': 'energy', 'sugars_100g': 'sugar', 'fiber_100g': 'fiber'})
```

#Nombre de valeurs nulles dans chacune de mes colonnes

```
In [13]: print(df_nutri_score.isnull().sum())
```

```
product_name      17512
energy            60660
sugar             76841
fiber            135344
protein           61866
sodium            66333
saturated_fat     92204
nutrition_score   101171
nutrition_echelon 101171
dtype: int64
```

Je renomme mes colonnes pour faciliter leur manipulation

#Visualiser le type de données de mes colonnes

```
In [14]: print(df_nutri_score.dtypes)
```

```
product_name      object
energy            float64
sugar             float64
fiber             float64
protein           float64
sodium            float64
saturated_fat      float64
nutrition_score    float64
nutrition_echelon  object
dtype: object
```

#Nombre de valeurs uniques dans chacune de mes colonnes

```
In [15]: print(df_nutri_score.nunique())
```

```
product_name      249245
energy            4093
sugar             4194
fiber             1150
protein           2633
sodium            5636
saturated_fat      2307
nutrition_score     55
nutrition_echelon    5
dtype: int64
```

Je regarde les valeurs uniques de mes colonnes



Data cleaning

Importation du fichier CSV et des packages + Mise en lien avec Pycharm

```
In [1]: import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
import missingno as msno
import numpy as np; np.random.seed(42)
print("Setup Complete")
```

Setup Complete

```
In [2]: df = pd.read_csv("/home/apprenant/Downloads/recommandation.tsv", sep='\t', low_memory=False)
```

Création de mon DataFrame

```
In [3]: #Je sélectionne les colonnes qui me semblent essentielles pour le nutri_score et je crée un dataframe indépendant p
df_nutri_score = df[['product_name', 'energy_100g', 'sugars_100g', 'fiber_100g', 'proteins_100g', 'sodium_100g', 's'
```

```
In [4]: #Je renomme mes colonnes
df_nutri_score = df_nutri_score.rename(columns={'energy_100g': 'energy', 'sugars_100g': 'sugar', 'fiber_100g': 'fib'
```

Sur Jupyter notebook, je connecte mon pycharm et mes packages.
Je crée ensuite mon dataframe.

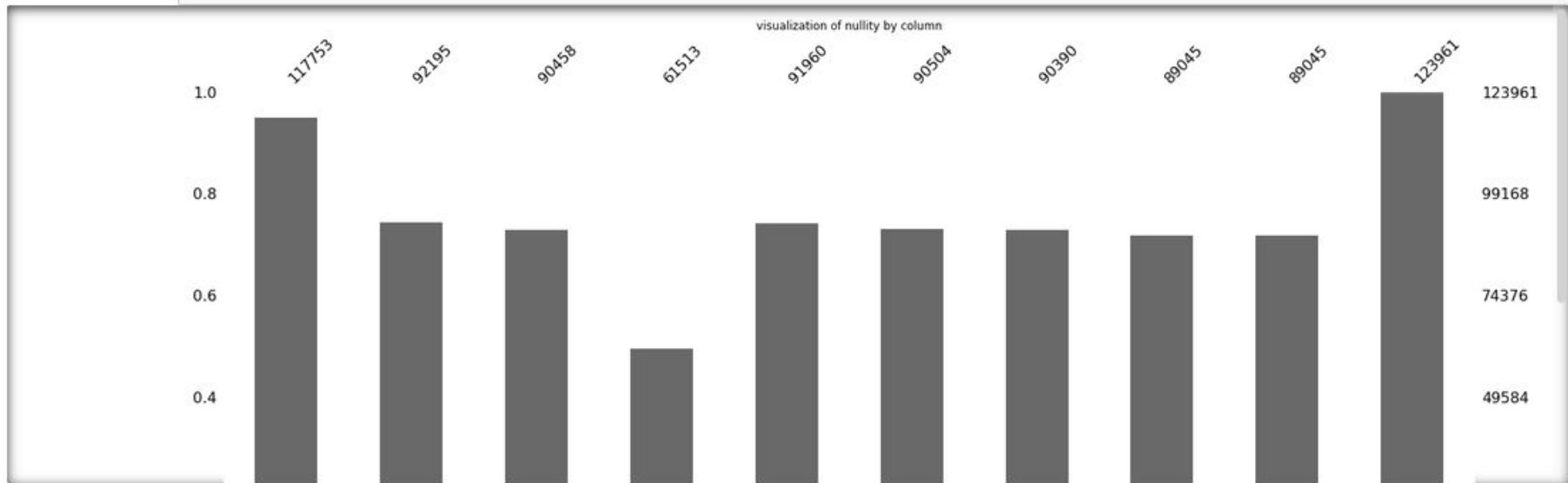
Sélection de la France comme pays principal

```
In [5]: #test = df_nutri_score[df_nutri_score.distribution ]  
#df_nutri_score[df_nutri_score.distribution] = df_nutri_score[df_nutri_score.distribution == 'France']  
#print(df_nutri_score.head())  
  
df_nutri_score_fr = df_nutri_score[df_nutri_score.distribution == 'France']
```

Je sélectionne la France comme pays principal

MISSING DATA VISUALIZATION

```
In [6]: msno.bar(df_nutri_score_fr);  
plt.title("visualization of nullity by column");
```



Je visualise ma donnée manquante avec missingno

Pourcentage de données présentes par colonne

In [7]:

```
percentage_per_column_notmissing_value = ((df_nutri_score_fr.isnull().sum() * 100 / len(df_nutri_score_fr)) - 100)..  
print(permission_per_column_notmissing_value)
```

```
product_name      94.991973  
energy            74.374198  
sugar             72.972951  
fiber            49.622865  
protein           74.184623  
sodium           73.010060  
saturated fat     72.918095  
nutrition_score   71.833077  
nutrition echelon 71.833077  
distribution      100.000000  
dtype: float64
```

Pourcentage de données manquantes par colonne

In [8]:

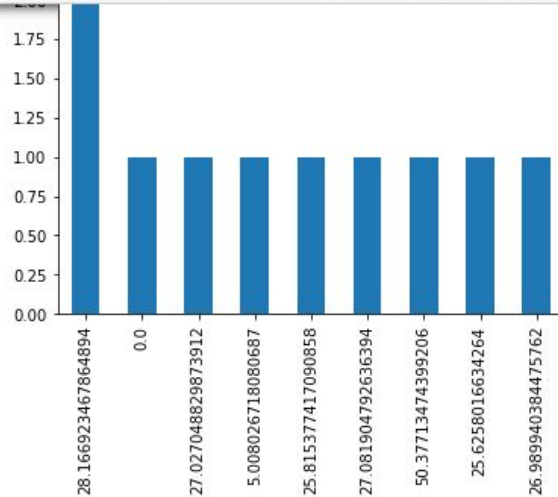
```
percentage_per_column_missing_value = df_nutri_score_fr.isnull().sum() * 100 / len(df_nutri_score_fr)  
print(permission_per_column_missing_value)
```

```
product_name      5.008027  
energy            25.625802  
sugar             27.027049  
fiber            50.377135  
protein           25.815377  
sodium           26.989940  
saturated fat     27.081905  
nutrition_score   28.166923  
nutrition echelon 28.166923  
distribution       0.000000  
dtype: float64
```

Je compare le pourcentage de données manquantes au pourcentage de données présentes

Visualiser le pourcentage de valeurs manquantes face à l'ensemble des valeurs:

```
In [9]: percentage_per_column_missing_value.value_counts().plot(kind='bar');  
plt.title("Percentage per column of missig values");
```



Je visualise le pourcentage de données manquantes

Cleaning valeurs manquantes

Je supprime les produits manquants:

```
In [11]: df_nutri_score_fr=df_nutri_score_fr.dropna(subset = ['product_name'])
print(df_nutri_score_fr.isnull().sum())
```

```
product_name      0
energy            25919
sugar             27647
fiber             56435
protein           26153
sodium            27604
saturated_fat      27711
nutrition_score    29051
nutrition_echelon  29051
distribution       0
dtype: int64
```

Visualiser le tableau dans son ensemble

```
In [12]: df_nutri_score_fr
```

Out[12]:

	product_name	energy	sugar	fiber	protein	sodium	saturated_fat	nutrition_score	nutrition_echelon	distribution
0	Farine de blé noir	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	France
46	Naturablue original	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	France
47	Filet de bœuf	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	France
51	Naturakrill original	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	France
138	Twix x2	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	France
...
356017	Thé vert Earl grey	21.0	0.5	0.2	0.5	0.01	0.2	NaN	NaN	France
356018	Cheese cake thé vert, yuzu	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	France
356019	Rillettes d'ole	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	France
356024	Biscottes bio	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	France
356025	Tomates aux Vermicelles	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	France

117753 rows × 10 columns

Je supprime les lignes où le nom du produit est manquant

Je supprime les lignes où l'ensemble des colonnes hors celle de product_name sont vides

```
In [13]: df_nutri_score_fr = df_nutri_score_fr.dropna(axis=0, subset=['energy', 'sugar', 'fiber', 'sodium', 'protein', 'satu
```

Il existe dans la db beaucoup de valeurs manquantes. J'ai supprimé les noms des produits manquants pour ne pas avoir de données orphelines. Je vais traiter les valeurs nulles de l'énergie, fibres, protéines etc. Pour ces colonnes, je ne peux pas me permettre de dupliquer la valeur d'à côté ou de supprimer la ligne en entier. La meilleure solution me semble être celle de remplacer la valeur manquante par 0. Après avoir regardé plusieurs produits, il me semble plus probable que NaN équivaut à une valeur nulle soit 0 et n'est pas un manque d'informations. exemple: Real Salt Granular NaN NaN NaN NaN 37.857 NaN NaN NaN

Je supprime les lignes où seul le nom du produit est présent et les restes des données des autres colonnes manquantes


```
In [51]: #Je remplace les données nulles de la colonne sucre par 0
df_nutri_score_fr['sugar'] = df_nutri_score_fr['sugar'].fillna(0)
```

```
In [52]: #Je remplace les données nulles de la colonne energy par 0
df_nutri_score_fr['energy'] = df_nutri_score_fr['energy'].fillna(0)
```

```
In [53]: #Je remplace les données nulles de la colonne fiber par 0
df_nutri_score_fr['fiber'] = df_nutri_score_fr['fiber'].fillna(0)
```

```
In [54]: #Je remplace les données nulles de la colonne protein par 0
df_nutri_score_fr['protein'] = df_nutri_score_fr['protein'].fillna(0)
```

```
In [55]: #Je remplace les données nulles de la colonne sodium par 0
df_nutri_score_fr['sodium'] = df_nutri_score_fr['sodium'].fillna(0)
```

```
In [56]: #Je remplace les données nulles de la colonne saturated fat par 0
df_nutri_score_fr['saturated_fat'] = df_nutri_score_fr['saturated_fat'].fillna(0)
```

```
In [20]: #Je regarde ma quantité de data
df_nutri_score_fr.shape
```

```
Out[20]: (60781, 10)
```

```
In [21]: #Je regarde mes 10 premières lignes
df_nutri_score_fr
```

```
Out[21]:
```

	product_name	energy	sugar	fiber	protein	sodium	saturated_fat	nutrition_score	nutrition_echelon	distribution
185	Root Beer	215.0	13.60	0.0	0.00	0.024200	0.00	18.0	e	France
193	Preparation mug cake chocolat-caramel au beurr...	1632.0	42.00	0.0	7.00	0.383858	4.50	21.0	e	France
194	Mini Confettis	1753.0	87.70	0.9	0.60	0.003937	0.80	14.0	d	France
195	Praliné Amande Et Noisette	2406.0	50.30	3.9	9.50	0.001181	2.90	14.0	d	France
231	Pepsi, Nouveau goût !	177.0	10.40	0.0	0.00	0.010000	0.00	13.0	e	France

Je remplace les données manquantes de mes éléments par 0

```
In [22]: #le lien entre nutri_core et nutrition echelon(grade) est évident du fait du même nombre de valeurs manquantes
print(df_nutri_score_fr['nutrition_score'].isnull().sum())
print(df_nutri_score_fr['nutrition echelon'].isnull().sum())

592
592
```

Je traite les valeurs manquantes de nutriscore et Echelon grade / Nutri score vide / Nutri score Grade vide

```
In [58]: #Je remplace les données nulles de la colonne nutrition echelon par 'undefined' pour ne pas induire dans l'erreur l
df_nutri_score_fr['nutrition echelon'] = df_nutri_score_fr['nutrition echelon'].fillna('undefined')
print(df_nutri_score_fr['nutrition echelon'].isnull().sum())

0
```

```
In [24]: #éléments le plus fréquent par colonne
print(df_nutri_score_fr.mode())
```

product name	energy	sugar	fiber	protein	sodium	saturated fat \
0 Spaghetti	0.0	0.0	0.0	0.0	0.0	0.0
nutrition_score	nutrition echelon	distribution				
0	0.0	d France				

```
In [25]: df_nutri_score_limit = df_nutri_score_fr[(df_nutri_score_fr['nutrition_score'] >= -15) & (df_nutri_score_fr['nutriti
```

```
In [26]: df_nutri_score_limit.shape
```

```
Out[26]: (60189, 10)
```

```
In [27]: df_nutri_score_fr = df_nutri_score_limit
```

```
In [28]: df_nutri_score_fr.shape
```

```
Out[28]: (60189, 10)
```

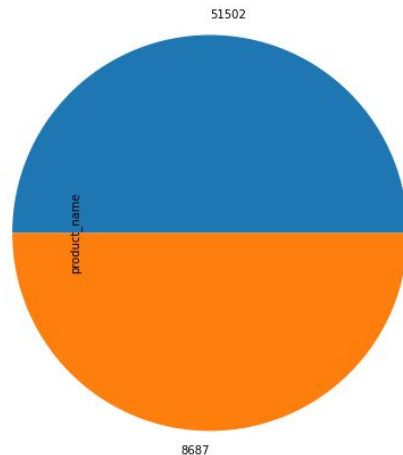
```
In [57]: #Je remplace les données nulles de la colonne nutrition echelon par 'undefined' pour ne pas induire dans l'erreur l
df_nutri_score_fr['nutrition_score'] = df_nutri_score_fr['nutrition_score'].fillna('undefined')
print(df_nutri_score_fr['nutrition_score'].isnull().sum())

0
```

Le lien entre nutrition grade et échelon est évident du fait du nombre identique de données manquantes. Je remplace leur données manquantes par undefined. Je supprime simplement le nutrition score plus petit que -15 et plus grand que 40 car ce n'est pas possible.

JE TRAITE LES VALEURS DOUBLES

```
In [30]: #j'affiche les valeurs doubles dans la colonne product_name  
valeurs_doubles_product_name = df_nutri_score_fr['product_name'].duplicated().value_counts()  
valeurs_doubles_product_name.value_counts().plot(kind='pie', radius=2);
```



```
In [31]: #Je mets à jour mon dataframe en supprimant les valeurs doubles  
df_nutri_score_fr = df_nutri_score_fr.drop_duplicates(subset="product_name")
```

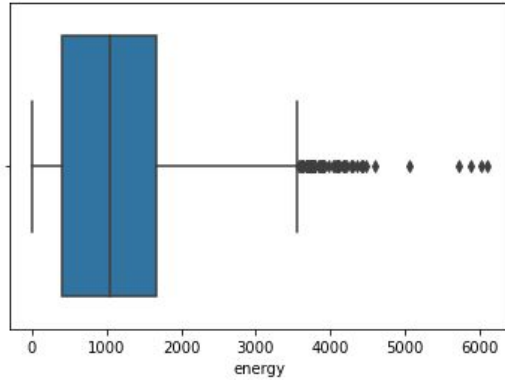
```
In [32]: #les valeurs doubles dans la colonne 'produits' ont bien été supprimés  
df_nutri_score_fr['product_name'].duplicated().value_counts()
```

```
Out[32]: False      51502  
Name: product_name, dtype: int64
```

Je traite les valeurs doubles de product_name

JE TRAITE LES VALEURS ABERRANTES (<0, >100, Somme > 100)

```
In [34]: ax = sns.boxplot(x=df_nutri_score_fr['energy'])
```



Je traite les valeur aberrantes et visualise le boxplot de énergie

```

In [37]: df_nutri_score_energy = df_nutri_score_sugar[(df_nutri_score_sugar['energy'] >= 0) & (df_nutri_score_sugar['energy']
Out[37]:
In [38]: #je visualise à chaque étape mon nombre de valeurs pour ne pas faire de fausses manipulations
df_nutri_score_energy.shape
Out[38]: (51473, 10)

In [39]: df_nutri_score_fiber = df_nutri_score_energy[(df_nutri_score_energy['fiber'] >= 0) & (df_nutri_score_energy['fiber']
Out[39]:
In [40]: #je visualise à chaque étape mon nombre de valeurs pour ne pas faire de fausses manipulations
df_nutri_score_fiber.shape
Out[40]: (51465, 10)

In [41]: df_nutri_score_protein = df_nutri_score_fiber[(df_nutri_score_fiber['protein'] >= 0) & (df_nutri_score_fiber['prote
Out[41]:
In [42]: #je visualise à chaque étape mon nombre de valeurs pour ne pas faire de fausses manipulations
df_nutri_score_protein.shape
Out[42]: (51464, 10)

In [43]: df_nutri_score_sodium = df_nutri_score_protein[(df_nutri_score_protein['sodium'] >= 0) & (df_nutri_score_protein['s
Out[43]:
In [44]: #je visualise à chaque étape mon nombre de valeurs pour ne pas faire de fausses manipulations
df_nutri_score_sodium.shape
Out[44]: (51464, 10)

In [45]: df_nutri_score_saturated_fat = df_nutri_score_sodium[(df_nutri_score_sodium['saturated_fat'] >= 0) & (df_nutri_score
Out[45]:
In [46]: #je visualise à chaque étape mon nombre de valeurs pour ne pas faire de fausses manipulations
df_nutri_score_saturated_fat.shape
Out[46]: (51464, 10)

In [47]: #j'écrase mon tableau sur la dernière version mise à jour
df_nutri_score_fr = df_nutri_score_saturated_fat
Out[47]:
In [48]: #Je compare le nombre de valeurs avec df_nutri_score_saturated_fat pour voir si mon nombre de valeurs est identique
df_nutri_score_fr.shape

```

Je ne prends que les éléments compris entre 0 et 100g

```
In [49]: df_nutri_score_fr.head(10)
```

```
Out[49]:
```

	product_name	energy	sugar	fiber	protein	sodium	saturated_fat	nutrition_score	nutrition_echelon	distribution
185	Root Beer	215.0	13.6	0.0	0.0	0.024200	0.0	18.0	e	France
193	Preparation mug cake chocolat-caramel au beurr...	1632.0	42.0	0.0	7.0	0.383858	4.5	21.0	e	France
194	Mini Confettis	1753.0	87.7	0.9	0.6	0.003937	0.8	14.0	d	France
195	Praliné Amande Et Noisette	2406.0	50.3	3.9	9.5	0.001181	2.9	14.0	d	France
231	Pepsi, Nouveau goût !	177.0	10.4	0.0	0.0	0.010000	0.0	13.0	e	France
240	Crêpes jambon fromage	678.0	3.7	0.9	8.2	0.287402	1.7	0.0	b	France
242	Tarte Poireaux Et Lardons	1079.0	1.0	1.4	7.5	0.314961	11.0	15.0	d	France
248	Marmite Original Pate A Tartiner 125G	1046.0	1.0	3.5	39.0	3.858268	0.1	9.0	c	France
251	Madeleines nature	1900.0	26.0	1.5	6.0	0.259843	2.5	12.0	d	France
288	Cakes Raisins	1768.0	28.0	1.8	5.8	0.255906	2.5	13.0	d	France

```
In [50]: df_nutri_score_fr.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 51464 entries, 185 to 356005
Data columns (total 10 columns):
#   Column          Non-Null Count  Dtype
---  -
0   product_name    51464 non-null  object
1   energy          51464 non-null  float64
2   sugar          51464 non-null  float64
3   fiber          51464 non-null  float64
4   protein         51464 non-null  float64
5   sodium          51464 non-null  float64
6   saturated_fat   51464 non-null  float64
7   nutrition_score  51464 non-null  float64
8   nutrition_echelon 51464 non-null  object
9   distribution     51464 non-null  object
dtypes: float64(7), object(3)
memory usage: 4.3+ MB
```

Je visualise mon tableau final après son nettoyage