Projet 2:
Analyse exploratoire de données nutritionnelles

Mission: Aider le site Lamarmite à construire un générateur de recettes saines avec les conclusions d'une AED* à réaliser à partir d'une base de donnée de produits de consommation.

Sommaire

Champ d'analyse

- Plus une mission qu'une problématique
  - Fournir de l'information pertinente qui pourra être prise en compte dans la création de recettes "saines" => recherche sur alimentation saine
- Information sur la base de données:
  - Nutrition_grade (qualitative: a -> e) : définie par http://fr.openfoodfacts.org/score-nutritionnel-experimental-france
  - Nutrition-score (quantitative) : définie par le UK Food Standards Administration (FSA)
  - Plus un produit est de bonne qualité, plus leur Nutrition-score est bas ((glucides saturés+lipides+sodium) -(fibres+proteins+FruitsLégumesNuts))
- Plusieurs questions possibles:
  - Existe-il une corrélation entre les deux façons de noter les produits (FR, UK)?
  - Est-ce que le pays d'origine (countries) a un impact sur les deux scores?
  - Quels sont les ingrédients (nutrition facts) qu'il faut prendre en compte en priorité car forte lien avec les scores?
- L'angle d'attaque choisi:
  - Le champ d'analyse se portera principalement à donner des éléments de réponses à la question3.
  - Apparition de patterns pas encore identifiés à cette étape de l'analyse.

1. Data Profiling: les questions préalables

Objectif: Après avoir importé le fichier csv (read_csv()), utiliser les statistiques descriptives pour analyser la qualité des données et mieux les appréhender.

Questions préalables

- Savoir d'où viennent les données et comment ont-elles été collectées pour éviter une mauvaise interprétation dans les conclusions de l'analyse.
- La base de données doit être structurée sous le format de la troisième forme normale (3FN) ou "tidy data".

1. Data Profiling: découvrir le contenu du fichier

- Mesures pour comprendre la structure de la base de données et identifier le ratio de valeurs manquantes par variable: len(), count().

- Definition et type de chaque variable du fichier (colonnes) ? Dtypes

- Echantillons sous forme de tableaux: tail(), head(), sample(5)

```
print(df.count()/len(df))
```

```
Nombre de colonnes:162
Nombre de lignes:320772
Nombre d'observations non nulles pour chaque colonne:
code                                0.999928
url                                 0.999928
creator                             0.999994
created_t                           0.999991
created_datetime                    0.999972
last_modified_t                     1.000000
last_modified_datetime              1.000000
product_name                        0.944627
generic_name                        0.164587
quantity                            0.326771
packaging                           0.246159
```

```
In [6]: df.dtypes
```

```
cities                        object
cities_tags                   object
purchase_places               object
                              ...
biotin_100g                  float64
pantothenic-acid_100g        float64
silica_100g                  float64
bicarbonate_100g             float64
potassium_100g               float64
```

```
In [9]: df.sample(5)
```

Out[9]:

	code	url	creator	created_t	created_datetime	last_modified_t	last_modified_datetime	product_name
117360	0607012351993	http://world-fr.openfoodfacts.org/produit/0607...	usda-ndb-import	1489075006	2017-03-09T15:56:46Z	1489075006	2017-03-09T15:56:46Z	Mr. Martin, Salted Sunflower Seeds
141656	0753695280621	http://world-fr.openfoodfacts.org/produit/0753...	usda-ndb-import	1489076580	2017-03-09T16:23:00Z	1489076580	2017-03-09T16:23:00Z	The Carolina Nut Co., Dill Pickle Peanuts

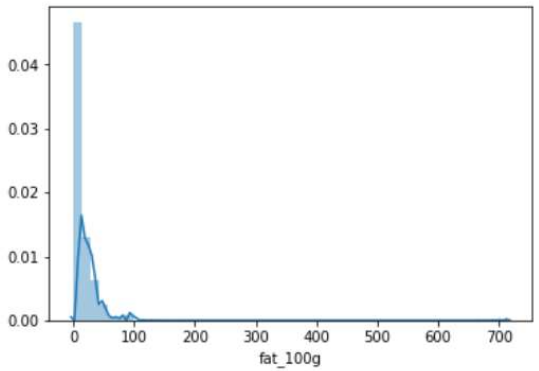
1. Data Profiling: statistiques et visualization graphique

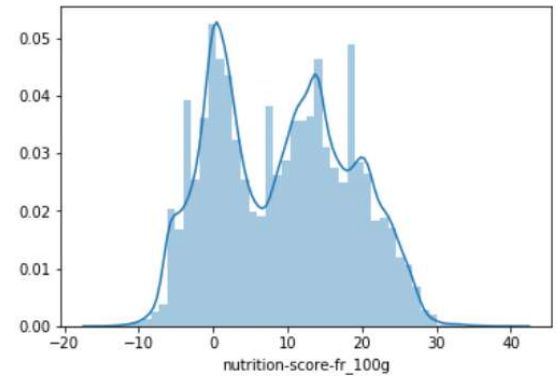
- Statistiques pour des variables ciblées: describe(), min(), max(), mean(), mode(), median(), std(), quantile()
- Graphes sur les observations de certaines variables avec la librairie seaborn et la function distplot().

```
In [10]: df.describe()
```

Out[10]:

	no_nutriments	additives_n	ingredients_from_palm_oil_n	ingredients_from_palm_oil	ingredients_that_may_be_from_palm_oil_n	ingredients_that_may_be
count	0.0	248939.000000	248939.000000	0.0	248939.000000	
mean	NaN	1.936024	0.019659	NaN	0.055246	
std	NaN	2.502019	0.140524	NaN	0.269207	
min	NaN	0.000000	0.000000	NaN	0.000000	
25%	NaN	0.000000	0.000000	NaN	0.000000	
50%	NaN	1.000000	0.000000	NaN	0.000000	
75%	NaN	3.000000	0.000000	NaN	0.000000	
max	NaN	31.000000	2.000000	NaN	6.000000	





1. Data Profiling – Pandas_profiling

- Executer une fonction ProfileReport(df) de la librairie pandas_profiling pour obtenir un résumé global des caractéristiques des données du fichier

Overview

Dataset info

Number of variables	162
Number of observations	320772
Total Missing (%)	44.0%
Total size in memory	396.5 MiB
Average record size in memory	1.3 KiB

Variables types

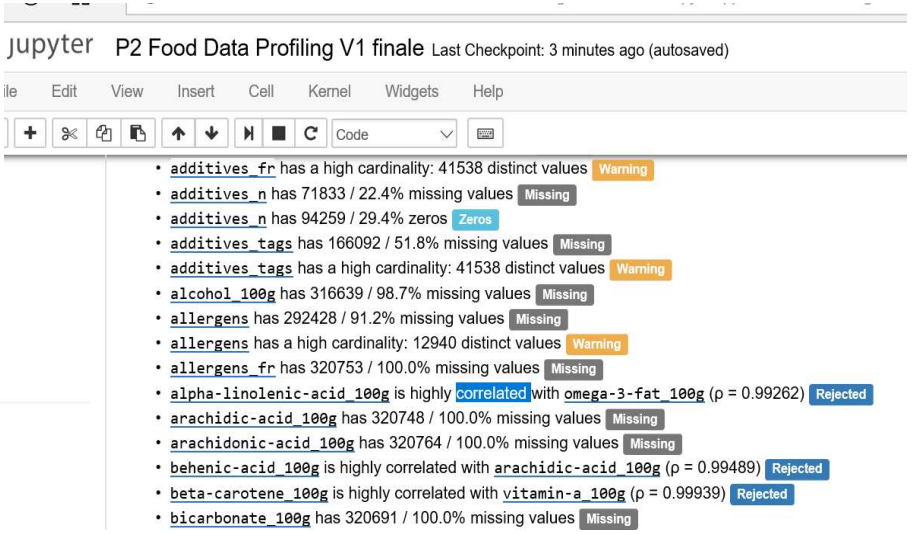
Numeric	28
Categorical	56
Boolean	5
Date	0
Text (Unique)	0
Rejected	73
Unsupported	0

sugars_100g
Numeric

Distinct count	4069
Unique (%)	1.3%
Missing (%)	23.6%
Missing (n)	75801
Infinite (%)	0.0%
Infinite (n)	0

Mean	16.003
Minimum	-17.86
Maximum	3520
Zeros (%)	11.6%

Toggle details



2. Nettoyage des données: traitement des colonnes

Objectif: Détecter puis corriger ou supprimer les données numériques incomplètes afin de réaliser une analyse plus pertinente.

- Option 1: utiliser le résultat du data profiling avec liste des variables identifiées.
- Option 2: Calcul du taux de remplissage pour chaque variable numérique.

```
nutrition-score-uk_100g
0.689617547666
Traitement de REMPLISSAGE des valeurs manquantes par la médiane car remplie à hauteur de 68.96175476662552%.
mediane de la colonne : 9.0.
Taux de remplissage de nutrition-score-uk_100g est désormais de : 1.0
glycemic-index_100g
0.0
Traitement de SUPPRESSION de la colonne car n'est remplie qu'à hauteur de 0.0%.
water-hardness_100g
0.0
Traitement de SUPPRESSION de la colonne car n'est remplie qu'à hauteur de 0.0%.
42
320772
```

Pour les deux options, supprimer la colonne pour TxRempl <= 35% et remplacer les valeurs manquantes par la mediane si TxRempl entre 35% et 80%. La variable intéressante pour notre étude Fruits-vegetables-nuts_100g a trop de valeurs manquantes, donc ne sera pas prise en compte!

2. Nettoyage des données: traitement des lignes et comparaison avec état initial

- Identification des lignes dont le taux de non remplissage est > 20%
- Créer une liste contenant les indices de ces lignes détectées et créér un nouveau dataframe excluant ces indices.
- Data profiling pour comparer le nouveau dataset avec le fichier initial

Avant traitement

```
import pandas_profiling

pandas_profiling.ProfileReport(df)
```

Overview

Dataset info

Number of variables	162
Number of observations	320772
Total Missing (%)	44.0%
Total size in memory	396.5 MiB
Average record size in memory	1.3 KiB

Après traitement

```
matplotlib.use(BACKEND)
```

Overview

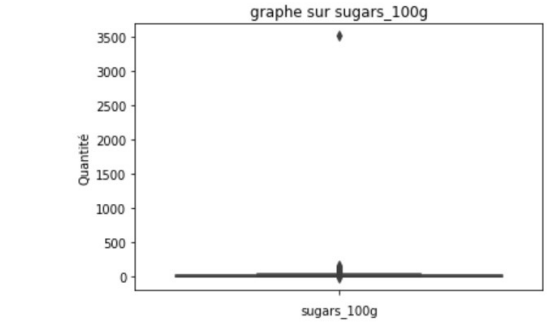
Dataset info

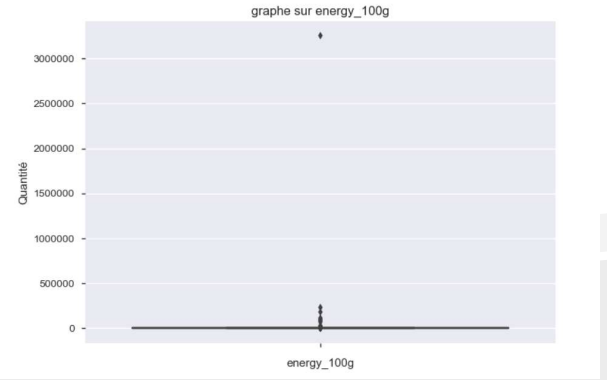
Number of variables	42
Number of observations	296822
Total Missing (%)	5.1%
Total size in memory	95.1 MiB
Average record size in memory	336.0 B

2. Nettoyage des données – données extrêmes avec graphes

Objectif: Détecter puis supprimer certaines données extrêmes voir aberrantes (ou outliers) afin de réaliser une analyse plus pertinente.

- Techniques graphiques pour identifier les données qui sont éloignées de la mass of data
  - Box plot: graphe utilisant la médiane et les deux quartiles 25^{ième} (Q1) et 75^{ième} (Q3), (Q3 - Q1) étant le IQ. Outliers visibles car bien au-dessus du plus grand quartile.
  - Scatter plot: Les outliers sont séparés de la mass of data





2. Nettoyage des données – données extrêmes avec IQ et comparaison avec état initial

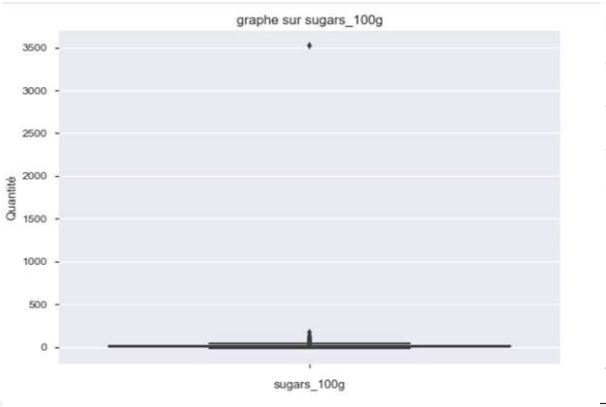
- Identification des valeurs extrêmes avec comparaison avec l'IQ (du Box plot)
  - Un point en-dessous ou au-dessus "inner fence" (Q1 - 1.5*IQ et Q3 + 1.5*IQ): **mild outlier**
  - Un point en-dessous ou au-dessus "outer fence" (Q1 - 3*IQ et Q3 + 3*IQ): **extreme outlier**
- Data profiling pour comparer le nouveau dataset

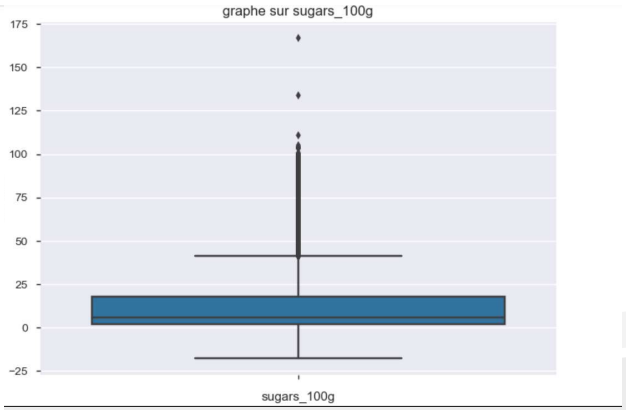
Avec outliers

```
In [54]: df1['sugars_100g'].describe()

Out[54]: count    296822.000000
         mean         14.202963
         std          20.653593
         min         -17.860000
         25%           2.100000
         50%           5.710000
         75%          17.780000
         max        3520.000000
         Name: sugars_100g, dtype: float64
```

Sans outliers

```
In [79]: df1['sugars_100g'].describe()

Out[79]: count    296754.000000
         mean         14.189582
         std          19.623256
         min         -17.860000
         25%           2.100000
         50%           5.710000
         75%          17.780000
         max         166.670000
         Name: sugars_100g, dtype: float64
```





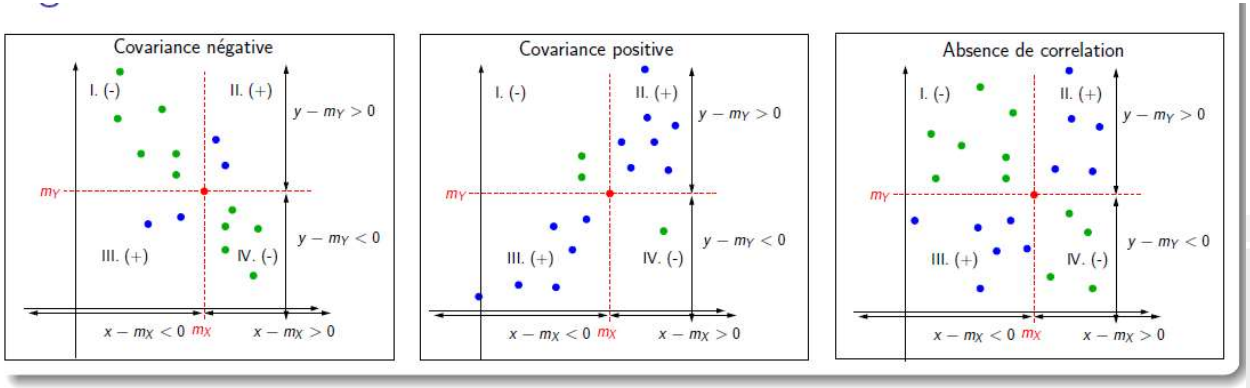
3. Exploration des données – Corrélation

Objectif: Identifier les variables qui sont liées entre elles avec plus ou moins d'intensité.

Le coefficient de corrélation de Pearson mesure l'intensité de de la relation linéaire entre deux variables.

- Plus la valeur du coefficient est proche de + 1 ou de - 1, plus les deux variables sont associées fortement.
- Plus le coefficient est près de 0, moins les variables partagent de covariance et donc, moins l'association est forte.

$$\rho_{X,Y} = \frac{\mathrm{cov}(X,Y)}{\sigma_X \times \sigma_Y} \qquad \mathrm{cov}(X,Y) = \frac{1}{n}\left(\sum_{i=1}^{n}(x_i - m_x)(y_i - m_y)\right)$$



Plusieurs points:

- Corrélation n'implique pas causalité.
- Coeff r nul n'implique pas l'indépendance des variables sauf si les variables suivent une loi normale.
- Si deux variables sont indépendantes, leur coefficient de corrélation est nul, la réciproque est fausse.
- Les points extrêmes et un mélange de population sont deux cas pouvant créer une fausse corrélation.

3. Exploration des données – Corrélation résultats

Objectif: Identifier les variables qui sont liées entre elles avec plus ou moins d'intensité.

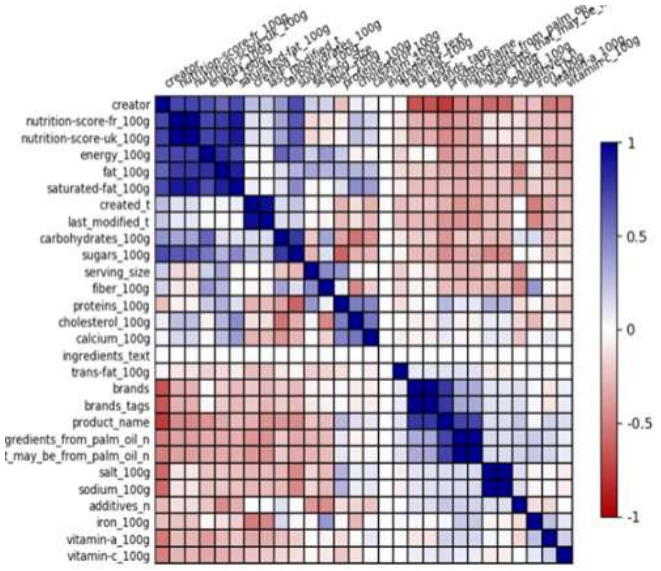
- Matrice de correlation sous forme graphique entre toutes les variables (new library biokit.viz, corrplot())
- Coefficient de Corrélation de toutes les variables par rapport à la variable d'intérêt: nutrition-score-fr_100g (corr()).
- Création d'un nouveau DataFrame

```
In [9]: df=pd.DataFrame(data=products, columns=new_liste)
        df=df.dropna()
        df
```

Out[9]:

	nutrition-score-fr_100g	saturated-fat_100g	energy_100g	proteins_100g	sugars_100g	salt_100g	fiber_100g
1	14.0	28.57	2243.00	3.570	14.29	0.00000	3.600
2	0.0	0.00	1941.00	17.860	17.86	0.63500	7.100
3	12.0	5.36	2540.00	17.860	3.57	1.22428	7.100
4	10.0	1.79	1552.00	8.570	5.71	0.58166	5.700
5	10.0	1.92	1933.00	13.460	11.54	0.58166	7.700
6	10.0	1.79	1490.00	8.890	5.71	0.58166	1.500

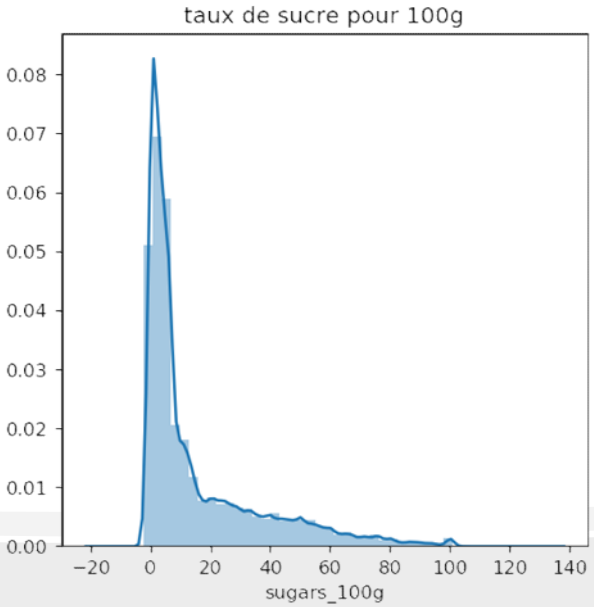
```
fiber_100g                              -0.110688
product_name                            -0.084110
vitamin-a_100g                          -0.022836
brands                                  -0.020872
vitamin-c_100g                          -0.020676
brands_tags                             -0.019631
serving_size                            -0.006258
iron_100g                               -0.003002
ingredients_that_may_be_from_palm_oil_n  0.016530
ingredients_from_palm_oil_n              0.016530
last_modified_t                          0.044859
trans-fat_100g                           0.046563
created_t                                0.066126
proteins_100g                            0.135305
additives_n                              0.146554
calcium_100g                             0.161805
cholesterol_100g                         0.198328
salt_100g                                0.198802
sodium_100g                              0.198811
carbohydrates_100g                       0.237748
sugars_100g                              0.444579
fat_100g                                 0.536985
energy_100g                              0.571530
saturated-fat_100g                       0.697870
creator                                  0.846295
nutrition-score-uk_100g                  0.986459
nutrition-score-fr_100g                  1.000000
ingredients_text                              NaN
Name: nutrition-score-fr_100g, dtype: float64
```

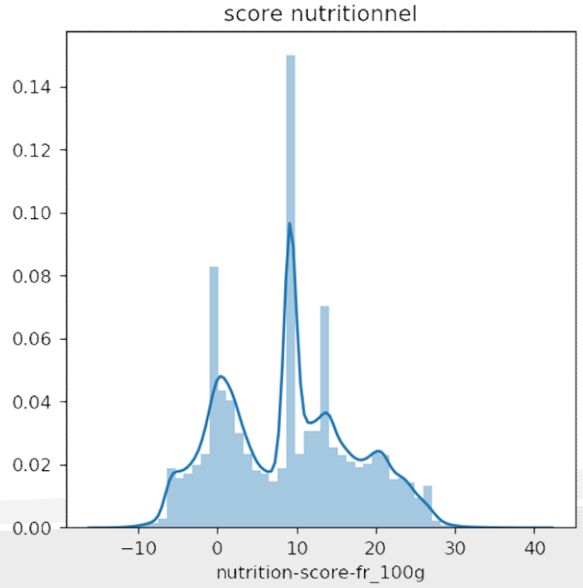


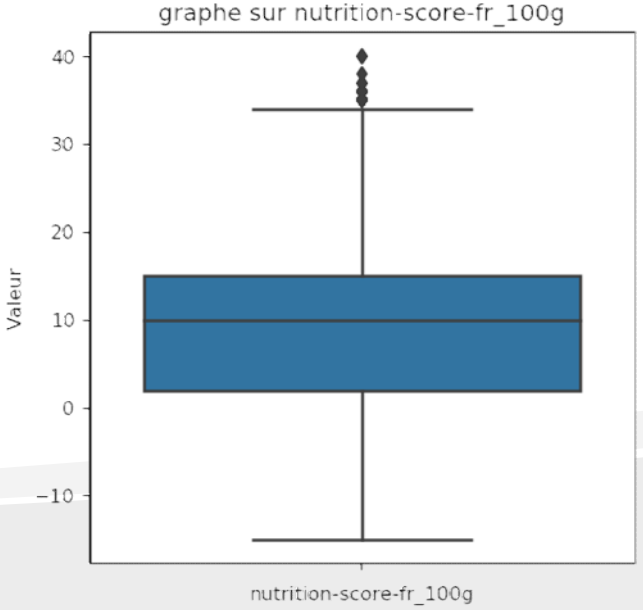
3. Exploration des données – Analyse univariée

Objectif: Dans l'analyse univariée, un critère est analysé sans tenir compte des autres.

- Aucune des deux variables suit une loi normale.
- Les deux histogrammes montrent les fréquences associées aux observations des deux variables.
- Le boxplot montre les trois valeurs quartiles, la médiane, les valeurs min et max ainsi que des valeurs extrêmes de la distribution de la variable nutrition-score-fr_100g.





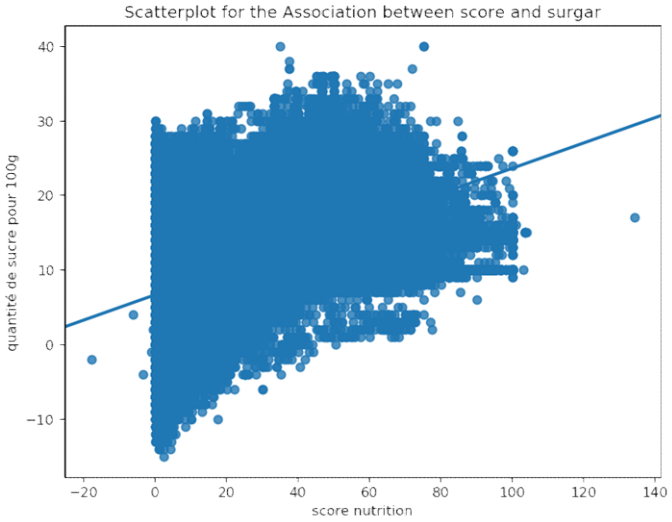


3. Exploration des données – Analyse multivariée

Objectif: Dans l'analyse multivariée, l'analyse est faite en tenant compte de l'interaction des critères les uns avec les autres.

- **Analyse bi-variée** entre deux variables nutrition-score-fr_100g et sugars_100g avec seaborn.regplot()

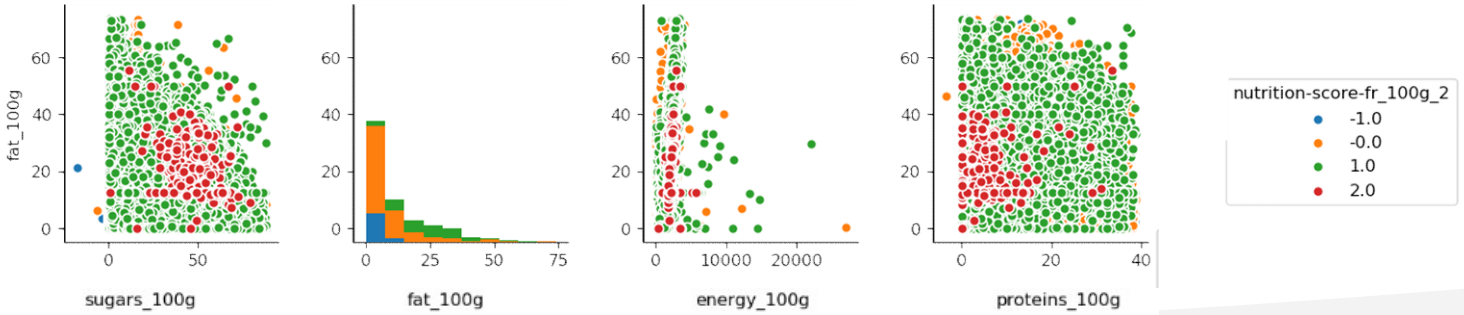
- => Covariance plutôt positive donc une liaison croissante entre les deux variables. Il semblerait avoir un lien entre le sucre et le score.



3. Exploration des données – Analyse multivariée

- **Analyse multivariée** avec seaborn.pairplot(), décrivant les relations entre toutes les variables et mettant en avant des clusters (couleurs différentes) pour chaque valeur de la variable d'intérêt nutrition-score-fr_100g

- => 2 clusters avec un score = 2 (mauvais score)
  - sugars et fat ont des valeurs hautes
  - proteins ont des valeurs basses

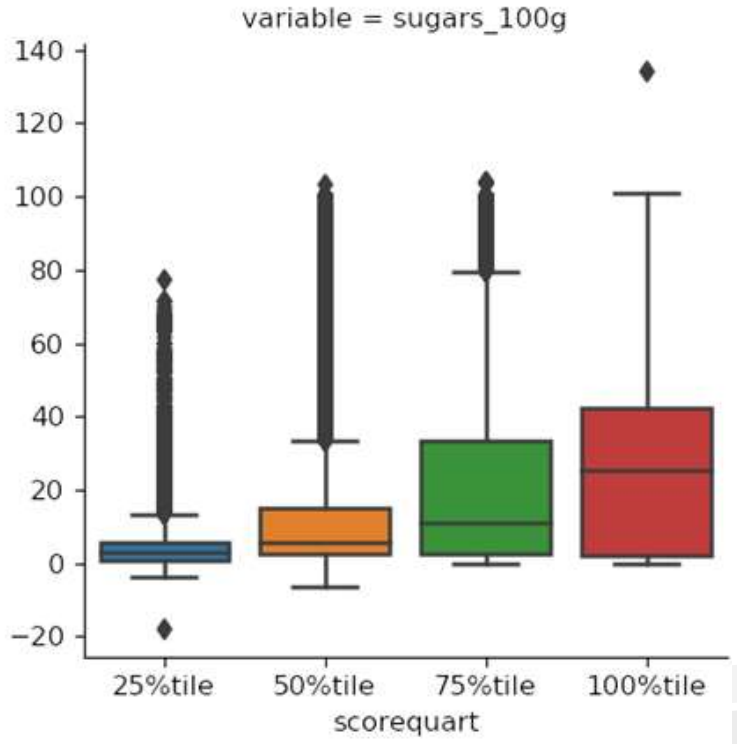


3. Exploration des données – Analyse multivariée

- **Analyse multivariée** avec seaborn.factorplot(), pour montrer la distribution d'une variable par rapport à des catégories.

- Boxplot de la variable sugars_100g, décomposé en quatre boxplot, chaque boite représentant un quartile de la variable d'intérêt nutrition-score-fr_100g (scorequart). L'axe des ordonnées illustre les valeurs des observations de la variable sugars_100g.

- => Pour un score bas (quartile 25%tile) donc un bon score, la variable sugars_100g a des valeurs plutôt basses.
- => Plus le score est haut donc score mauvais, plus le sugars_100g présentent des valeurs hautes.



3. Exploration des données – Test de significativité

Objectif: Test correspondant au test de nullité du coefficient (= test de "non corrélation").
Le coefficient de corrélation est-il significativement différent de 0 ?

- **La question statistique:** Existe t'il une liaison linéaire entre les 2 variables sugars_100g et nutrition-score-fr_100g

- **Les hypothèses statistiques:**
  - H0 : $\rho = 0$ → Absence de liaison linéaire entre les deux variables (ne peut se prononcer sur l'indépendantes car elles ne suivent pas des loi normales)
  - H1 : ρ diff de zéro → Existence d'une liaison linéaire

- **Probabilité critique (p-value):** On calcule la probabilité critique (p-value) que l'on compare au risque α que l'on s'est fixé (5%). On peut interpréter la p-value comme le plus petit seuil de significativité pour lequel l'hypothèse nulle est acceptée. Si la p-value est plus petite que α , alors nous rejetons l'hypothèse nulle et ainsi nous pouvons déclarer qu'il y a liaison linéaire entre les deux variables. La p-value n'est pas entièrement fiable mais acceptable pour de échantillons > 500.

3. Exploration des données – Test de significativité résultats

- Coefficient de corrélation = 0.4064 donc une faible liaison linéaire
- p_value < 0.05 donc nous rejetons l'hypothèse nulle et pouvons déclarer qu'il y a liaison linéaire entre les deux variables nutrition-score-fr_100g et sugars_100g

In [17]:
```
import numpy as np
from scipy import stats

p_value = stats.pearsonr(products['nutrition-score-fr_100g'],products['sugars_100g'])
print(p_value)
```

```
(0.40641635268944887, 0.0)
```

4. Exploration des données – Feature Engineering

- **Objectif:** FE aide à extraire plus d'informations de nos données. Une technique est de générer une nouvelle variable à partir des variables existantes (feature creation)

Formule du calcul d'un nouveau score nutritionnel:

- https://fr.openfoodfacts.org/score-nutritionnel-experimental-france
- Points donnés aux produits en fonction de la quantité de nutriments qu'ils contiennent pour 100 g.
  - - Points A pour les nutriments jugés "mauvais"
  - Points C sont donnés pour les "bons" nutriments
- Calcul du Nouveau Score = points A - points C

=> Création d'un dict avec le nouveau score associé à chaque aliment (observation).

```
1653
<class 'numpy.float64'>
calcul pointsA
12.5
calcul pointsC
10
dictionnaire des nouveaux scores
{0: 6.0, 1: 8.5, 2: -2.5, 3: 2.5, 4: -2.5, 5: -2.5, 6: 1.0, 7: -2.5, 8: 12.5, 9: 11.0, 10: -2.5, 11: 0.0, 12: -2.5, 13: -2.5,
14: 2.5, 15: -2.5, 16: -2.5, 17: 11.0, 18: 0.0, 19: -2.5, 20: 10.0, 21: 7.5, 22: 2.5, 23: 0.0, 24: 2.5, 25: 13.5, 26: -2.5, 2
7: 5.0, 28: 5.0, 29: -2.5, 30: 7.5, 31: -2.5, 32: 12.5, 33: 10.0, 34: -2.5, 35: 2.5, 36: 13.5, 37: 7.5, 38: 7.5, 39: -2.5, 4
0: -2.5, 41: -2.5, 42: -2.5, 43: 12.5, 44: -2.5, 45: -2.5, 46: 6.0, 47: -2.5, 48: -2.5, 49: -2.5, 50: -2.5, 51: -2.5, 52: -2.
5, 53: -2.5, 54: -2.5, 55: -2.5, 56: -2.5, 57: 2.5, 58: -2.5, 59: -2.5, 60: 2.5, 61: -2.5, 62: 0.0, 63: 0.0, 64: -2.5, 65:
```

Points A

Les points A sont la somme des points pour l'énergie, les graisses saturées, les sucres et le sodium.

Points	Energie (kJ)	Graisses saturées (g)	Sucres (g)	Sodium (mg)
0	≤335	≤1	≤4.5	≤90
1	>335	>1	>4.5	>90
2	>670	>2	>9	>180
3	>1005	>3	>13.5	>270
4	>1340	>4	>18	>360
5	>1675	>5	>22.5	>450
6	>2010	>6	>27	>540
7	>2345	>7	>31	>630
8	>2680	>8	>36	>720
9	>3015	>9	>40	>810
10	>3350	>10	>45	>900

Points C

Les points C sont la somme des points pour les fruits, légumes et noix, pour les fibres et pour les protéines.

Points	Fruits, légumes et noix (%)	Fibres (g)	Protéines (g)
0	≤40	≤0.7	≤1.6
1	>40	>0.7	>1.6
2	>60	>1.4	>3.2
3	-	>2.1	>4.8
4	-	>2.8	>6.4
5	>80	>3.5	>8.0

Conclusions

Traitement de la base de donnée: Cleaning

Traitement	Nbr colonnes (critères)	Nbr lignes (observations)
Etat initial	162	320 772
Valeurs manquantes		
Au niveau des colonnes	42	320 772
Au niveau des lignes	42	296 822
Valeurs aberrantes (outliers)	42	295 009
Valeurs doublons ou inutiles	39	295 009
Etat après traitement Cleaning	39	295 009

Dataset info

Number of variables	162
Number of observations	320772
Total Missing (%)	44.0%
Total size in memory	396.5 MiB
Average record size in memory	1.3 KiB

Dataset info

Number of variables	39
Number of observations	295009
Total Missing (%)	5.5%
Total size in memory	90.0 MiB
Average record size in memory	320.0 B

Exploration des données:

- Quels sont les ingrédients (nutrition facts) qu'il faut prendre en compte en priorité car fort lien avec les scores?

⇒ identification de plusieurs critères à prendre en compte car corrélés: le sucre, l'énergie, les proteins.
⇒ Le sucre à une liaison croissante avec le score (covariance positive)
⇒ Liaison linéaire démontrée entre le score et le sucre, test à exécuter pour tous les autres critères afin de les comparer
⇒ Feature creation nous permet d'obtenir un autre score pour chaque nutriment (certainement assez proche du nutrition-score-fr déjà fourni)

Conclusions

A retenir:
- Passer plus de temps sur la définition des critères
- Réfléchir plus à la problématique et faire un premier tri dans les critères à étudier
- Garder en tête la problématique tout le long de l'étude pour mieux l'exécuter

A approfondir:
- Les librairies Python
- Revoir l'interprétation de tous les graphes disponibles
- Etudier les différents tests
- Savoir refaire le même projet en R et comparer