



Concevez une application au service de la santé publique

Parcours Data Scientist | projet 3

Rim Bahroun

Novembre 2022

OPENCLASSROOMS



Concevez une application au service de la santé publique

- L'agence "[Santé publique France](#)" a lancé un appel à projets pour trouver des idées innovantes d'applications en lien avec l'alimentation.



- **Mission:** Proposer une idée d'application en liens avec l'alimentation à partir de l'analyse exploratoire des données de **OpenFOODfacts**.



Concevez une application au service de la santé publique



- **Idée d'application**
- **Partie 1: Inspection et nettoyage des données**
 - Analyse de la forme des données
 - Nettoyage sur les variables
 - Nettoyage sur les individus
 - Traitement des valeurs manquantes
- **Partie 2: Exploration des données**
 - Analyse univariée
 - Analyse bivariée
 - Analyse multivariée
 - Application proposée



**Santé
publique**
France

• **Conclusion**



OPENCLASSROOMS

Idée d'application: **RGO-scan**

Nous avons tous ressenti au moins une fois dans notre vie une remontée de liquide acide de l'estomac dans l'œsophage et parfois dans la bouche surtout après un repas copieux . C'est un **reflux gastro-œsophagien (RGO)**.



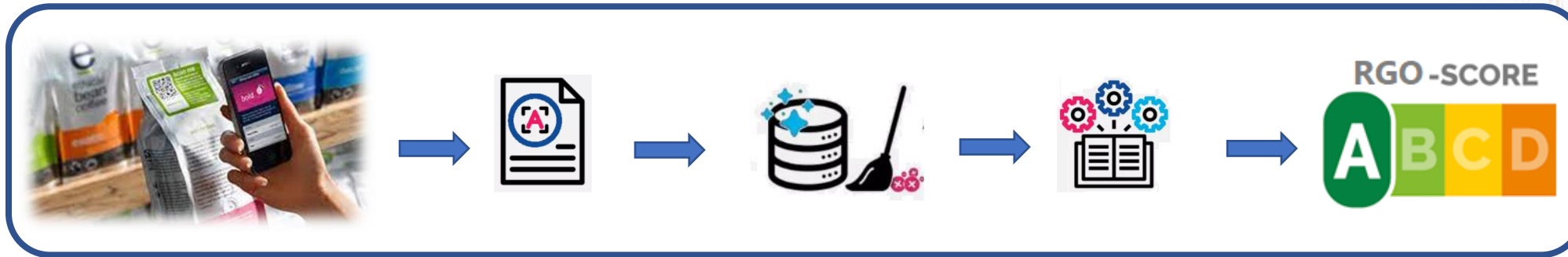
Au moins 20 % des adultes ont des symptômes occasionnels de reflux gastro-œsophagien. 10 % ont des symptômes de RGO chaque jour.

Source: <https://www.ameli.fr>

Idée d'application: RGO-scan

L'idée principale de l'application serait donc d'afficher le **RGO-score** et **RGO-grade** d'un produit en fonction des indications nutritionnelles disponibles sur l'étiquette.

RGO-scan



A: **Nutriment à favoriser** (diminue le reflux gastro-œsophagien)

D: **Nutriment à éviter** (augmente le reflux gastro-œsophagien)



RGO: Reflux gastro-œsophagien.



Concevez une application au service de la santé publique



- **Idée d'application**
- **Partie 1: Inspection et nettoyage des données**
 - Analyse de la forme des données
 - Nettoyage sur les variables
 - Nettoyage sur les individus
 - Traitement des valeurs manquantes
- **Partie 2: Exploration des données**
 - Analyse univariée
 - Analyse bivariée
 - Analyse multivariée
 - Application proposée
- **Conclusion**



1. Inspection et nettoyage des données



1. Analyse de la forme des données



- Données à disposition : 1 fichiers .csv

320772 lignes, 162 colonnes

Il donne les valeurs des caractéristiques de chaque produit alimentaire.

106 variables de type réel et 56 de type chaine de caractères.

```
1 data_Food.sample(4)
```

	code	url	creator	created_t	created_datetime	last_modified_t	last_modified_datetime	product_name
213332	3258561011424	http://world-fr.openfoodfacts.org/produit/3258...	aristoi	1465503014	2016-06-09T20:10:14Z	1486491078	2017-02-07T18:11:18Z	Popcorn micro-ondable sucré
39937	0041220517282	http://world-fr.openfoodfacts.org/produit/0041...	usda-ndb-import	1489071444	2017-03-09T14:57:24Z	1489071444	2017-03-09T14:57:24Z	Original Cola

```
1 data_Food.shape
(320772, 162)
```

```
1 # Types de variables
2 data_Food.dtypes.value_counts()

float64    106
object      56
dtype: int64
```



1. Inspection et nettoyage des données



OPENCLASSROOMS

1. Analyse de la forme des données



Nature - Lay's - 150 g

Certaines informations de ce produit ont été fournies directement par son fabricant PEPSICO FRANCE.

Code-barres: 3168930008958 (EAN / EAN-13)

Nom générique : Chips de pommes de terre

Quantité : 150 g

Conditionnement : Plastique, Sachet

Marques : Lay's

Catégories : Aliments et boissons à base de végétaux, Aliments d'origine végétale, Snacks, Céréales et pommes de terre, Snacks salés, Amuse-gueules, Chips et frites, Chips, Chips de pommes de terre, en:aliments-d-origine-vegetale, en:aliments-et-boissons-a-base-de-vegetaux, en:amuse-gueules, en:cereales-et-pommes-de-terre, en:chips, en:chips-de-pommes-de-terre, en:chips-et-frites, en:snacks-sales

Labels, certifications, récompenses : Sans conservateurs, Nutriscore, Nutriscore C, en:Point Vert, en:Sans colorants artificiels, en:Sans conservateurs, en:Sans huile de palme

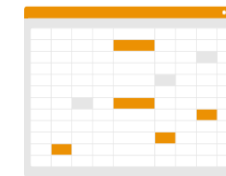
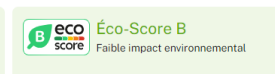
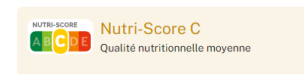
Origine des ingrédients : France, en:Allemagne, en:Belgique

Lieux de fabrication ou de transformation : France

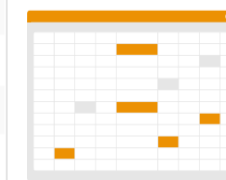
Magasins : Leclerc, Magasins U, Monoprix, carrefour.fr, Vival

Pays de vente : France, en:Allemagne, en:Belgique, en:Espagne, en:Royaume-Uni

Matching with your preferences



1. Inspection et nettoyage des données



OPENCLASSROOMS

1. Analyse de la forme des données



- Données à disposition



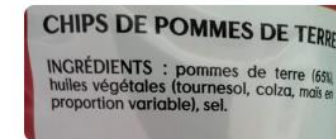
Santé

Ingrédients



5 ingrédients

Pommes de terre, huiles végétales (tournesol, colza, maïs en proportion variable), sel.



Zoom

Transformation des aliments



Aliments transformés

Tableau nutritionnel	Tel que vendu pour 100 g / 100 ml	Tel que vendu par portion (30g)	Comparé à: en:chips-de-pommes-de-terre
Énergie	2 305 kj (551 kcal)	692 kj (165 kcal)	+4 %
Matières grasses	34 g	10,2 g	+4 %
Acides gras saturés	4,2 g	1,26 g	+40 %
Glucides	53 g	15,9 g	+3 %
Sucres	0,5 g	0,15 g	-62 %
Fibres alimentaires	4,2 g	1,26 g	-5 %
Protéines	6,3 g	1,89 g	+6 %
Sel	1,1 g	0,33 g	-13 %
Fruits, légumes, noix et huiles de colza, noix et olive (estimation par analyse de la liste des ingrédients)	0 %	0 %	

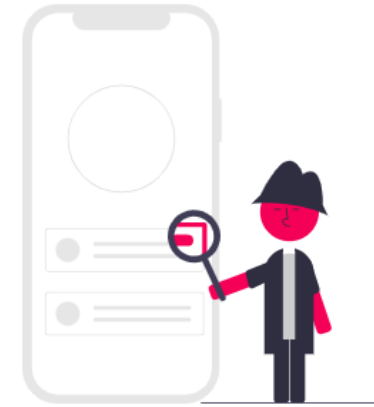
Concevez une application au service de la santé publique



- **Idée d'application**
- **Partie 1: Inspection et nettoyage des données**
 - Analyse de la forme des données
 - Nettoyage sur les variables
 - Nettoyage sur les individus
 - Traitement des valeurs manquantes
- **Partie 2: Exploration des données**
 - Analyse univariée
 - Analyse bivariée
 - Analyse multivariée
 - Application proposée
- **Conclusion**

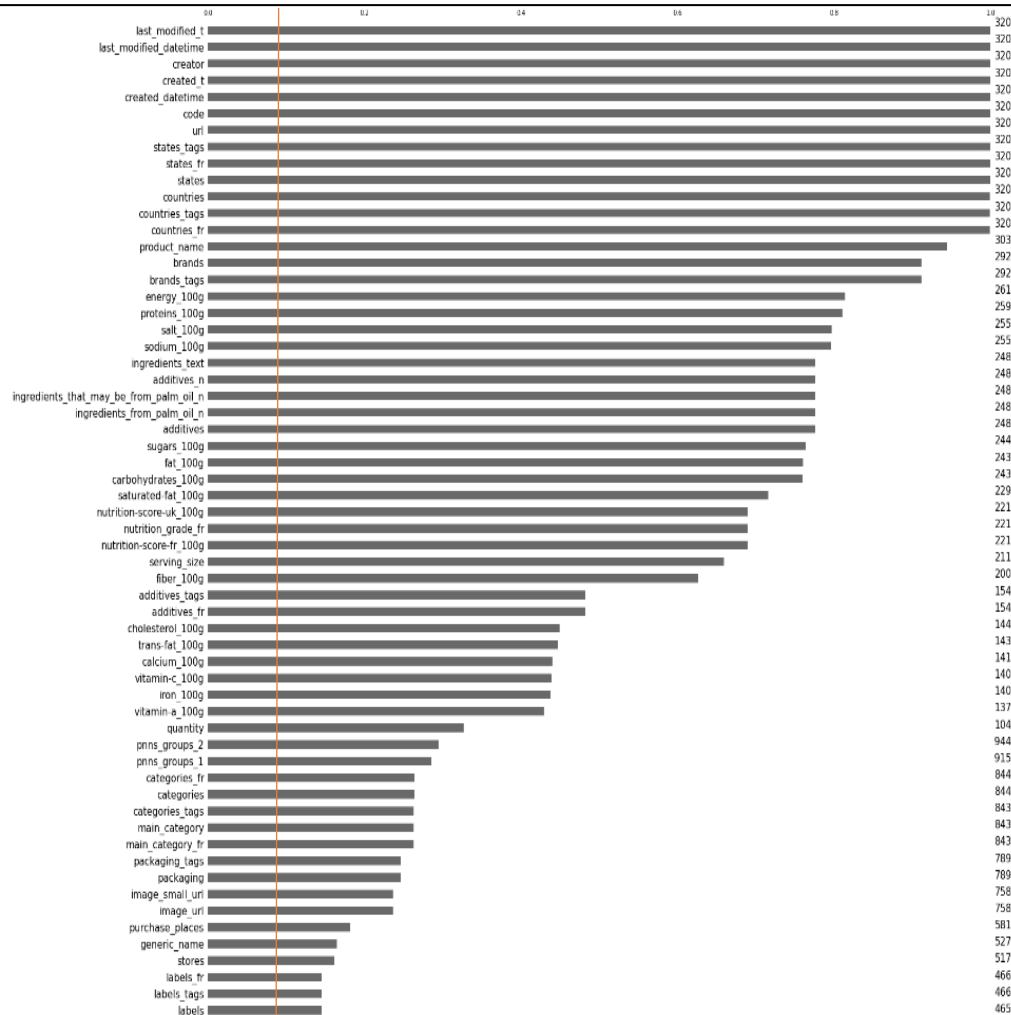


1. Inspection et nettoyage des données



2. Nettoyage sur les variables

- 1 fichiers .csv (320772 lignes, 162 colonnes)



Un grand nombre de variables du jeu de données est très peu renseigné.

1. Inspection et nettoyage des données

2. Nettoyage sur les variables



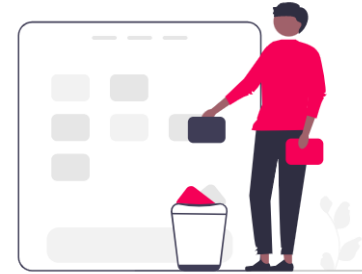
Suppression des variables avec plus de 90% de valeurs manquantes.

On passe de **162** variables à **62** variables.



Suppression des variables redondantes ou non utiles

On passe de **62** variables à **31** variables.



```
1 # supprimer les variables 'information général' non utiles
2 data = data.drop(['created_t', 'last_modified_t', # on garde les variables _datetime
3                  'generic_name', # on garde product_name
4                  'quantity',
5                  'packaging', 'packaging_tags',
6                  'brands', 'brands_tags',
7                  'categories', 'categories_tags', # on garde categories_fr
8                  'manufacturing_places', 'manufacturing_places_tags',
9                  'labels', 'labels_tags', 'labels_fr',
10                 'purchase_places', 'stores',
11                 'countries', 'countries_tags', # on garde countries_fr
12                 'serving_size',
13                 'additives_tags', 'additives', 'additives_fr', # on garde additives_n
14                 'ingredients_from_palm_oil_n', 'ingredients_that_may_be_from_palm_oil_n',
15                 'states', 'states_tags', 'states_fr',
16                 'main_category', # on garde main_category_fr
17                 'image_url', 'image_small_url'
18                 ], axis=1)
```



Concevez une application au service de la santé publique



- **Idée d'application**
- **Partie 1: Inspection et nettoyage des données**
 - Analyse de la forme des données
 - Nettoyage sur les variables
 - Nettoyage sur les individus
 - Traitement des valeurs manquantes
- **Partie 2: Exploration des données**
 - Analyse univariée
 - Analyse bivariée
 - Analyse multivariée
 - Application proposée
- **Conclusion**



1. Inspection et nettoyage des données

3. Nettoyage sur les individus



Suppression des doublons selon le code produit.

22 lignes supprimées.

```
1 data = data_3.copy()
2 # Suppression des doublons en fonction du code
3 data.drop_duplicates(subset = "code", keep = 'last', inplace=True)

(320750, 31)
```

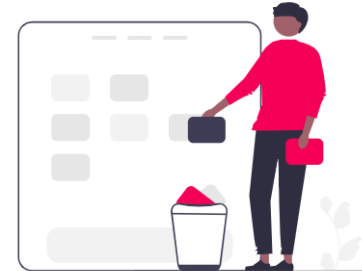


Suppression des produits non vendus en France

222030 lignes supprimées.

```
1 data_fr = data.loc[data["countries_fr"].str.contains('France'),:]
2 data_fr.shape

(98440, 31)
```



Suppression des lignes totalement vides sur les variables quantitatives

24072 lignes supprimées.

```
1 df_num = df.select_dtypes('float')
2 df_num.dropna(axis=0, how = 'all', inplace = True) #eliminer Les lignes avec 100% de valeurs manquantes

(74368, 31)
```



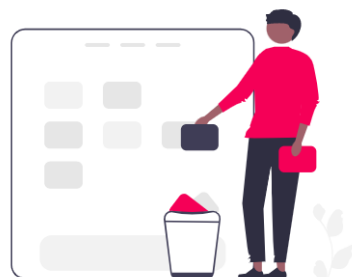
1. Inspection et nettoyage des données

3. Nettoyage sur les individus



Suppression des lignes à valeurs aberrantes.

13 lignes supprimées.



```
1 data_100g.describe()
```

	fat_100g	saturated-fat_100g	trans-fat_100g	cholesterol_100g	carbohydrates_100g	sugars_100g	fiber_100g	proteins_100g	salt_100g	sodium_100g	vitamin-a_100g	vitamin-c_100g	calcium_100g	iron_100g
count	47642.000000	62375.000000	386.000000	415.000000	47211.000000	62515.000000	45723.000000	64318.000000	62574.000000	62571.000000	589.000000	1297.000000	2257.000000	1185.000000
mean	13.332232	5.423696	0.209285	0.041158	27.759277	13.432792	2.559271	7.754531	1.160535	0.456924	0.000627	0.125570	0.325133	0.044379
std	16.926708	8.531083	0.984866	0.535806	27.413340	19.087618	4.634788	7.887373	4.309815	1.696759	0.005574	2.800254	1.739698	0.757976
min	0.000000	0.000000	0.000000	0.000000	0.000000	-0.100000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	1.300000	0.300000	0.000000	0.000000	4.000000	1.000000	0.000000	1.800000	0.080000	0.031496	0.000058	0.012000	0.120000	0.002400
50%	6.800000	2.000000	0.000000	0.000000	14.500000	4.100000	1.380000	6.000000	0.558800	0.220000	0.000120	0.020000	0.130000	0.005000
75%	21.000000	7.400000	0.115000	0.005440	53.000000	17.800000	3.200000	11.000000	1.244600	0.490000	0.000464	0.030000	0.330000	0.008000
max	380.000000	210.000000	17.200000	10.900000	190.000000	105.000000	178.000000	100.000000	211.000000	83.000000	0.120000	100.000000	69.500000	25.000000

```
1 # Suppression des lignes avec des valeurs aberrantes > 100 ou < 0
2 data_fr_100 = data.loc[~ (((data_100g > 100).sum(axis = 1) > 0) | ((data_100g < 0).sum(axis = 1) > 0)), :]
3 data_fr_100.shape
```

(74355, 31)



1. Inspection et nettoyage des données

3. Nettoyage sur les individus

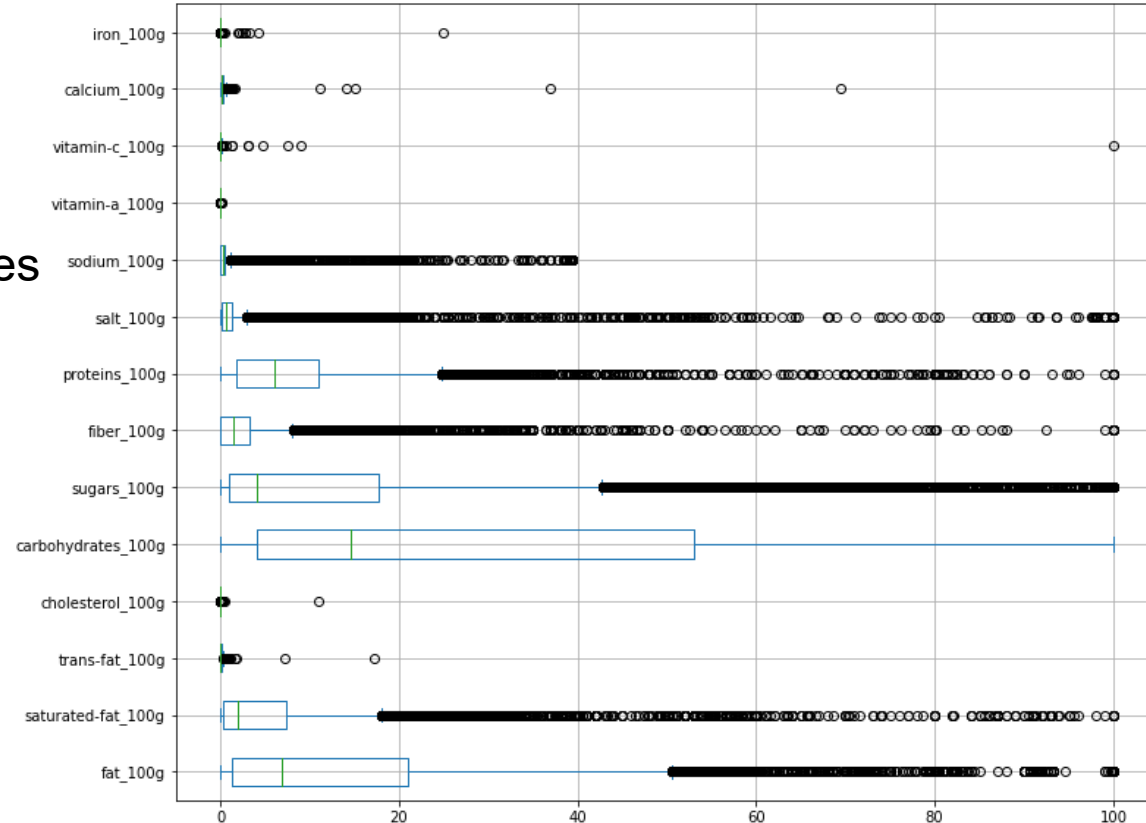
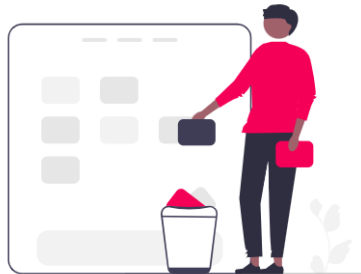
Traitement des lignes à valeurs atypiques



Correction (si possible) ou **suppression** des lignes à valeurs nutritionnelles mal renseignées en se basant sur l'analyse des boîtes à moustache des variables.

133 lignes supprimées.

(74235, 31)



```
1 print(data.shape)
2 # On supprimera les produits avec ['sodium_100g']>30 et qui n'ont pas le mot clés 'sel' ou 'salt' dans 'product_name'.
3 mask = (data['sodium_100g']>30) & (data['product_name'].str.contains("salt|sel", case=False)==False)
4 data = data.loc[~mask,:]
5 print(data.shape)
```

(74266, 31)
(74252, 31)



Concevez une application au service de la santé publique



- **Idée d'application**
- **Partie 1: Inspection et nettoyage des données**
 - Analyse de la forme des données
 - Nettoyage sur les variables
 - Nettoyage sur les individus
 - Traitement des valeurs manquantes
- **Partie 2: Exploration des données**
 - Analyse univariée
 - Analyse bivariée
 - Analyse multivariée
 - Application proposée
- **Conclusion**



1. Inspection et nettoyage des données

4. Traitement des valeurs manquantes et erreurs de type

➡ Les variables qualitatives

➡ Les variables: 'created_datetime' et 'last_modified_datetime'



Correction du type

```
1 # convertir les colonnes 'created_datetime' et 'last_modified_datetime' en type datetime
2 df['created_datetime'] = pd.to_datetime(df['created_datetime'])
3 df['last_modified_datetime'] = pd.to_datetime(df['last_modified_datetime'])
```



1 valeur manquante 'created_datetime' remplacée par 'last_modified_datetime'



1. Inspection et nettoyage des données

4. Traitement des valeurs manquantes

➡ Les variables qualitatives

➡ Les variables: 'pnn_group'



Les valeurs manquantes des variables `pnn_group_1` et `pnn_group_2` remplacées par **'unknown'**.

```
1 # remplacement de pnn_group vide par 'unknown'
2 df['pnn_groups_1'].fillna(value='unknown', inplace=True)
3 df['pnn_groups_2'].fillna(value='unknown', inplace=True)
```



1. Inspection et nettoyage des données

4. Traitement des valeurs manquantes

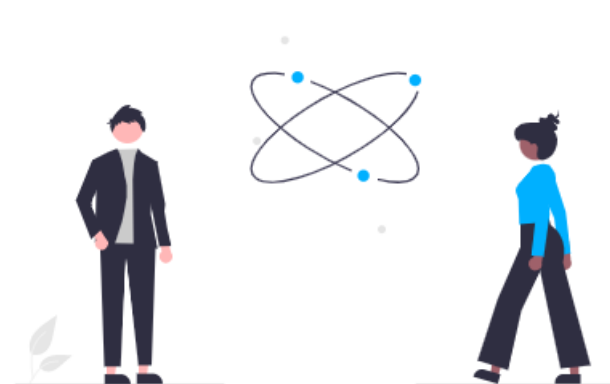
➡ Les variables quantitatives

➡ La variable: 'additives_n'



Les valeurs manquantes de la variable 'additives_n' ont été remplacées par **zéro**. On fera l'hypothèse que si la valeur est manquante cela veut dire qu'il n'y a pas d'additives.

```
1 # Remplacement des valeurs manquantes par zéro
2 df['additives_n'].fillna(value= 0 , inplace=True)
```



1. Inspection et nettoyage des données

4. Traitement des valeurs manquantes

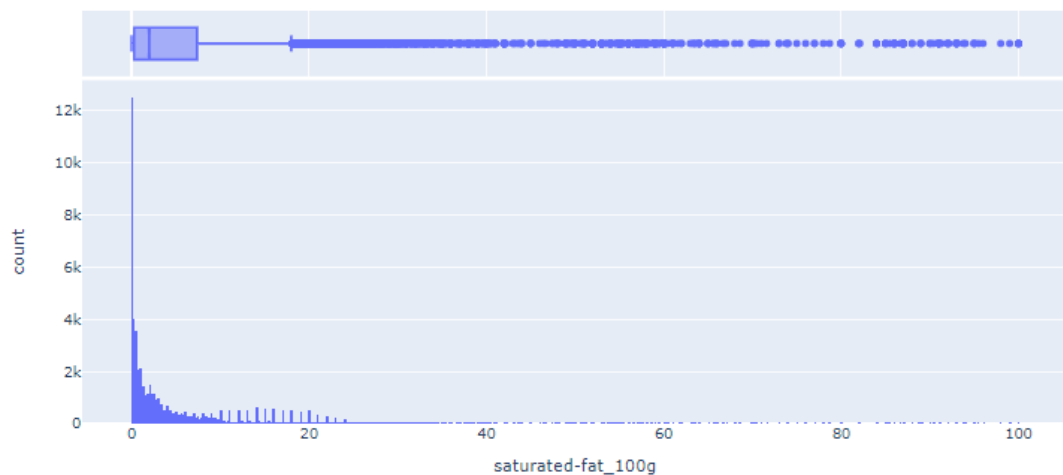
➡ Les variables quantitatives

➡ Les variables nutritionnelles

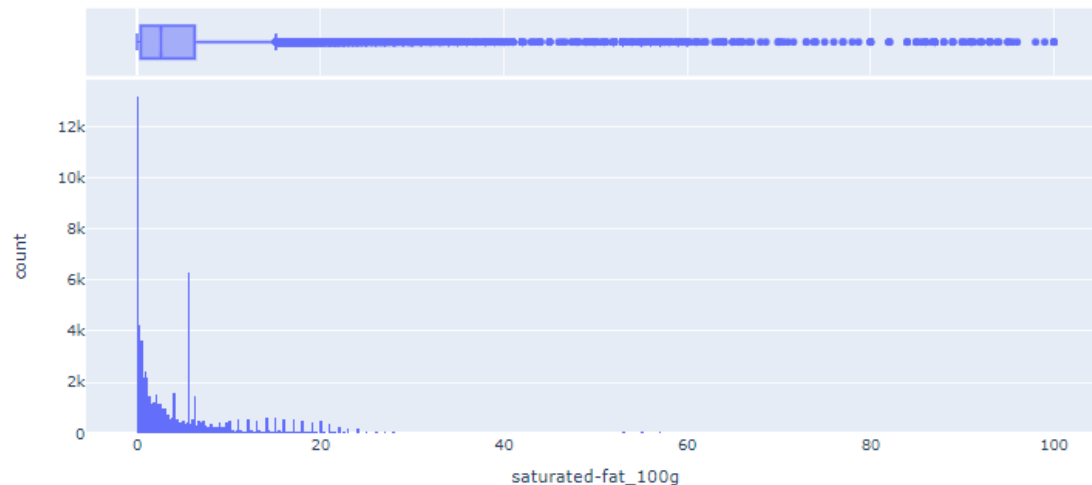


Les valeurs manquantes des variables nutritionnelles '_100g' sont remplacées avec l'algorithme KNN

Avant imputation KNN



Après imputation KNN



```
1 # On entraîne le modèle d'imputation sur un échantillon de données
2 df_sample = df_num.sample(frac=0.5, random_state=1)
3 imputer = KNNImputer(n_neighbors=5)
4 imputer.fit(df_sample)
```

KNNImputer()

```
1 # Puis on applique le modèle sur l'ensemble des données
2 df_num_imputed = imputer.transform(df_num)
```

```
1 ddf_num_imputed = pd.DataFrame(df_num_imputed, columns=df_num.columns)
```



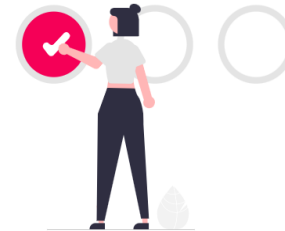
Après imputation de toutes les valeurs manquantes des variables nutritionnelles, les distributions n'ont pas été modifiées comparées aux données d'origine.

1. Inspection et nettoyage des données

Synthèse

Dans cette première partie, nous avons:

- Supprimé les variables à plus de 90% de valeurs manquantes
- Supprimé les variables redondantes ou non utiles
- Supprimé les **doublons** selon le code produit
- Supprimé les produits non vendus en France
- Supprimé les lignes totalement vides sur les variables quantitatives
- Supprimé les lignes à valeurs **aberrantes** pour les nutriments
- Traité les lignes à valeurs **atypiques** : correction ou suppression
- Corrigé le type des variables dates
- **Imputé** les valeurs **manquantes** par une constante ou par l'algorithme KNN
- Vérifié les distributions des variables



Ces opérations ont permis de réduire, nettoyer et imputer notre jeu de données.

(320772 lignes, 162 colonnes) → **(74235 lignes, 31 colonnes)**

Concevez une application au service de la santé publique



- **Idée d'application**
- **Partie 1: Inspection et nettoyage des données**
 - Analyse de la forme des données
 - Nettoyage sur les variables
 - Nettoyage sur les individus
 - Traitement des valeurs manquantes
- **Partie 2: Exploration des données**
 - Analyse univariée
 - Analyse bivariée
 - Analyse multivariée
 - Application proposée
- **Conclusion**

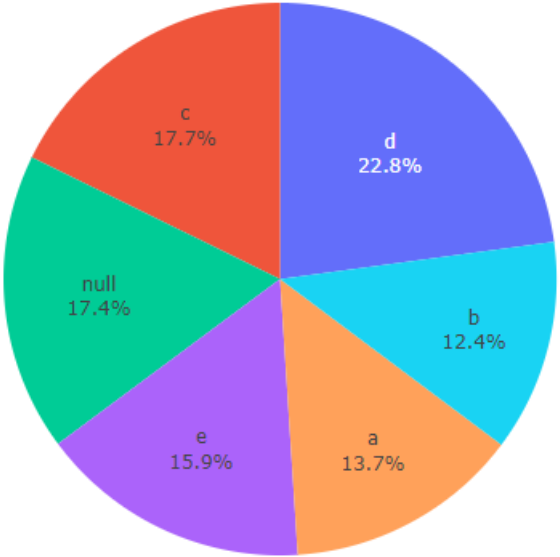


2. Exploration des données

1. Analyse univariée

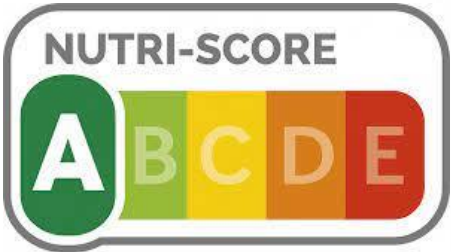
➡ Les variables qualitatives

➡ La variable : qualité nutritionnelle



nutrition_grade_fr

Toutes les classes nutritionnelles sont bien représentées dans ce jeu de données.



2. Exploration des données

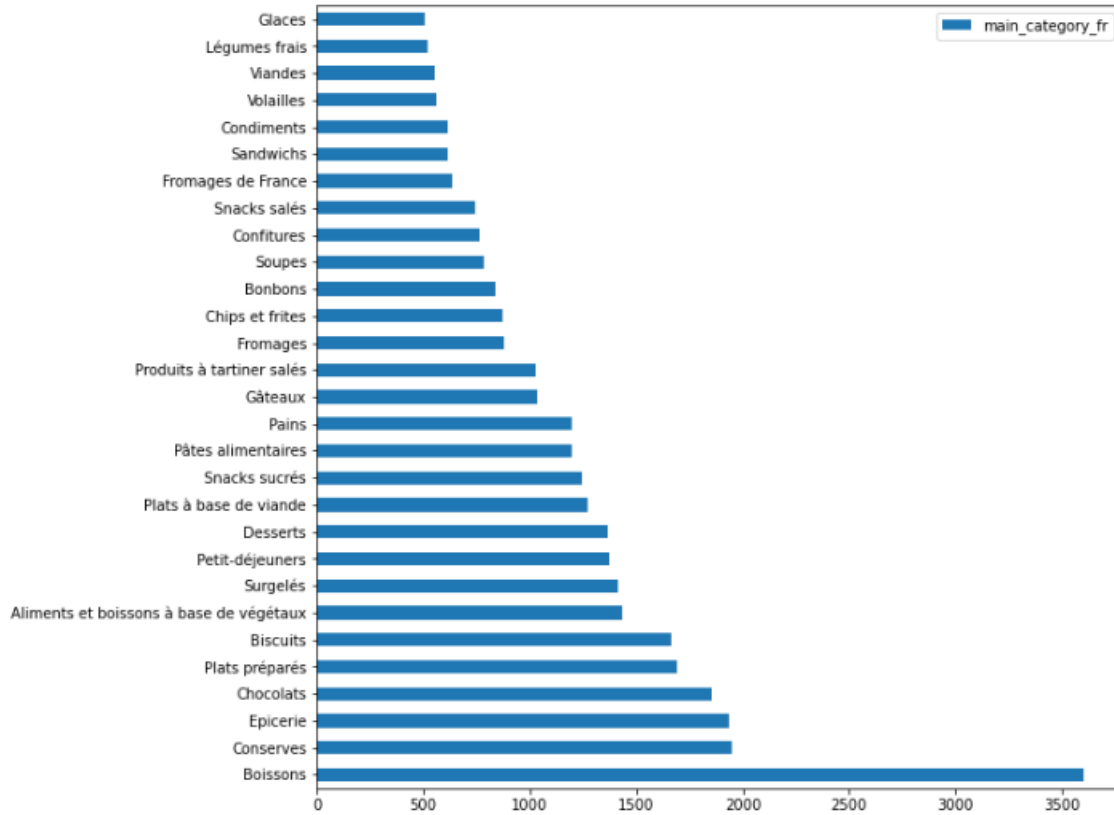
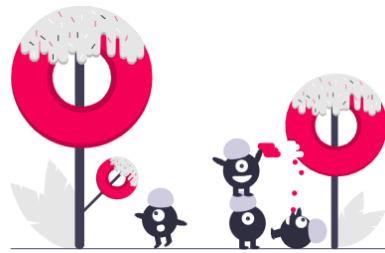
1. Analyse univariée

➡ Les variables qualitatives

➡ La variable : « main_category_fr »

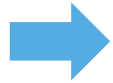
Les groupes les plus représentés dans ce jeu de données sont:

- Boissons,
- Conserves,
- Epicerie,
- Chocolats,
- Plats préparés,
- Biscuits.



2. Exploration des données

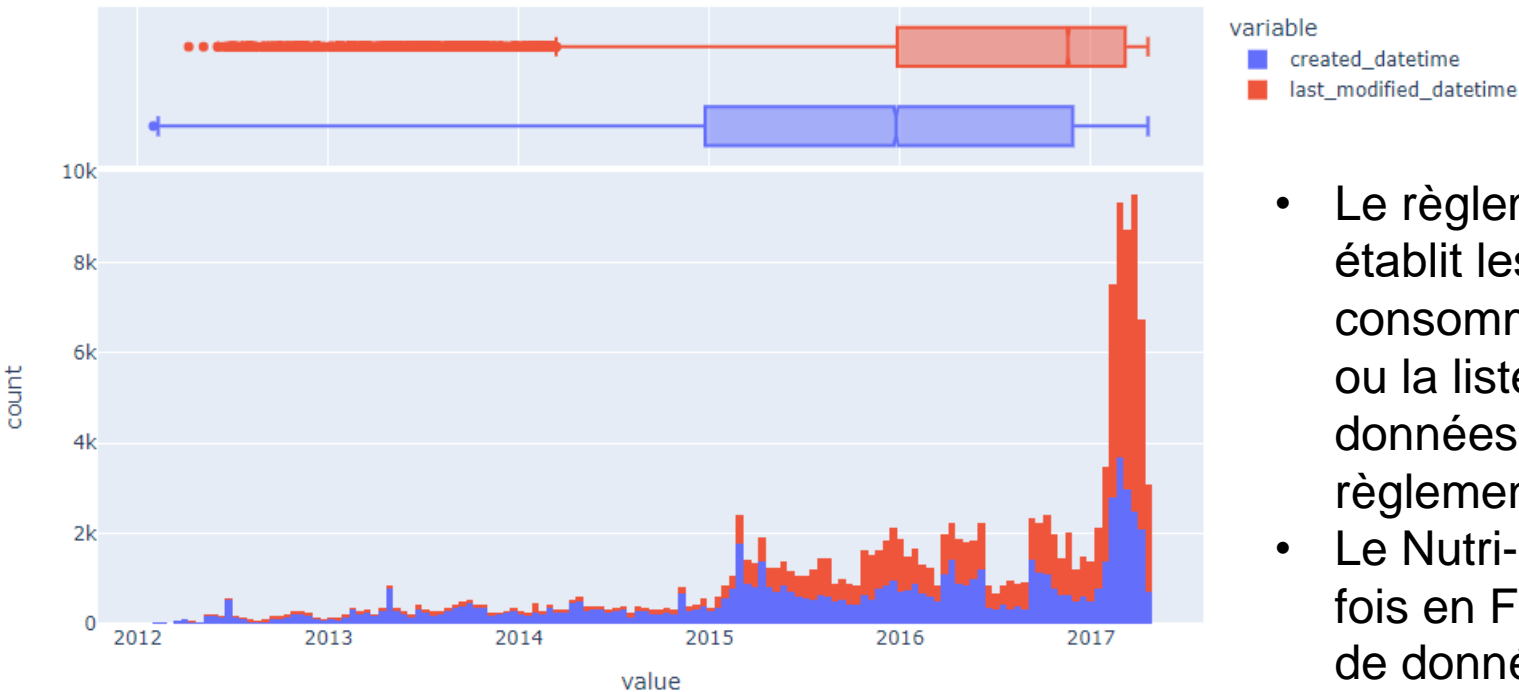
1. Analyse univariée



Les variables quantitatives



Les variables: date de création et date de modification



- Le règlement européen EU n°1169/2011, dit INCO, établit les règles quant à l'information des consommateurs, tel que la déclaration nutritionnelle ou la liste des ingrédients. La création de la base de données a commencé en **2012** juste après ce règlement.
- Le Nutri-Score a été mis en place pour la première fois en France en **2017**. A partir de cette date, la base de données a été le plus modifiée.

2. Exploration des données

1. Analyse univariée

➡ Les variables quantitatives pertinentes pour le **RGO-score**

Quelles variables favorisent le RGO : ...?

Quelles variables atténuent le RGO :...?



RGO: Reflux gastro-œsophagien.



2. Exploration des données

Favorisent le RGO
saturated-fat_100g,
trans-fat_100g,
cholesterol_100g

1. Analyse univariée

➔ Les variables quantitatives pertinentes pour le **RGO-score**

Favorisent le RGO : ...?

Les recommandations pour la gestion du RGO par l'alimentation :

- Limiter les **aliments gras**. Ils retardent la vidange gastrique et réduisent la pression du sphincter œsophagien inférieur SOI, prolongeant ainsi la durée d'exposition de l'œsophage à l'acide gastrique et augmentant le volume d'acide qui peut remonter.



saturated-fat_100g, trans-fat_100g, cholesterol_100g : augment le RGO.



Source: <https://www.passeportsante.net>

OPENCLASSROOMS

RGO: Reflux gastro-œsophagien.

2. Exploration des données

1. Analyse univariée

➔ Les variables quantitatives pertinentes pour le **RGO-score**

Favorisent le RGO : ...?

Les recommandations pour la gestion du RGO par l'alimentation :

- Les ballonnements peuvent être causés par une fermentation excessive dans l'intestin. Ils aggravent le reflux gastro-œsophagien en augmentant la pression dans l'abdomen. La fermentation est générée par certains types de glucides dits fermentescibles. La consommation de **glucides** modifie le pH de l'estomac qui devient trop acide.



carbohydrates_100g : augment le RGO.



Favorisent le RGO
saturated-fat_100g,
trans-fat_100g,
cholesterol_100g,
carbohydrates_100g



Source: <https://www.passeportsante.net>

OPENCLASSROOMS

RGO: Reflux gastro-œsophagien.

2. Exploration des données

1. Analyse univariée

➡ Les variables quantitatives pertinentes pour le **RGO-score**

Favorisent le RGO : ...?

Les recommandations pour la gestion du RGO par l'alimentation :

- L'excès de sel, c'est bien connu, favorise l'hypertension et les maladies cardiovasculaires. Il diminue aussi la pression du sphincter de l'œsophage et augmente donc le risque de reflux.



salt_100g : augment le RGO.

- Autres aliments déconseillés: Produits sucrés



sugars_100g : augment le RGO.



RGO: Reflux gastro-œsophagien.

Favorisent le RGO

saturated-fat_100g,
trans-fat_100g,
cholesterol_100g,
carbohydrates_100g
salt_100g
sugars_100g



Source: <https://www.passeportsante.net>

OPENCLASSROOMS

2. Exploration des données

Atténuent
le RGO
proteins_100g

1. Analyse univariée

➡ Les variables quantitatives pertinentes pour le **RGO-score**

Atténuent le RGO : ...?

Les recommandations pour la gestion du RGO par l'alimentation :

- Assurer une consommation adéquate de protéines. Les protéines augmentent la pression du SOI, ce qui permet ainsi la fermeture du sphincter et réduit le reflux.



proteins_100g : diminue le RGO.



RGO: Reflux gastro-œsophagien.



Source: <https://www.passeportsante.net>

OPENCLASSROOMS

2. Exploration des données

**Atténuent
le RGO**
proteins_100g
fiber_100g
calcium_100g
vitamin-c_100g

1. Analyse univariée

➔ Les variables quantitatives pertinentes pour le **RGO-score**

Atténuent le RGO : ...?

Les recommandations pour la gestion du RGO par l'alimentation :

- Pour prévenir la constipation et les ballonnements liés à la pression abdominale, il faudrait consommer des fibres alimentaires tous les jours.



fiber_100g : diminue le RGO.

- Ce sont des sels (aluminium, calcium, magnésium) qui neutralisent localement l'acidité du contenu de l'estomac.



calcium_100g : diminue le RGO.

- Pensez à consommer suffisamment d'aliments riches en vitamine C



vitamin-c_100g : diminue le RGO.



RGO: Reflux gastro-œsophagien.

Source: <https://www.passeportsante.net>

OPENCLASSROOMS

2. Exploration des données

1. Analyse univariée

➡ Les variables quantitatives pertinentes pour le **RGO-score**

Quels ingrédients ?



Favorisent le RGO
saturated-fat_100g
trans-fat_100g
cholesterol_100g
carbohydrates_100g
salt_100g
sugars_100g

Atténuent le RGO
proteins_100g
fiber_100g
calcium_100g
vitamin-c_100g



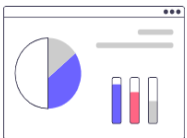
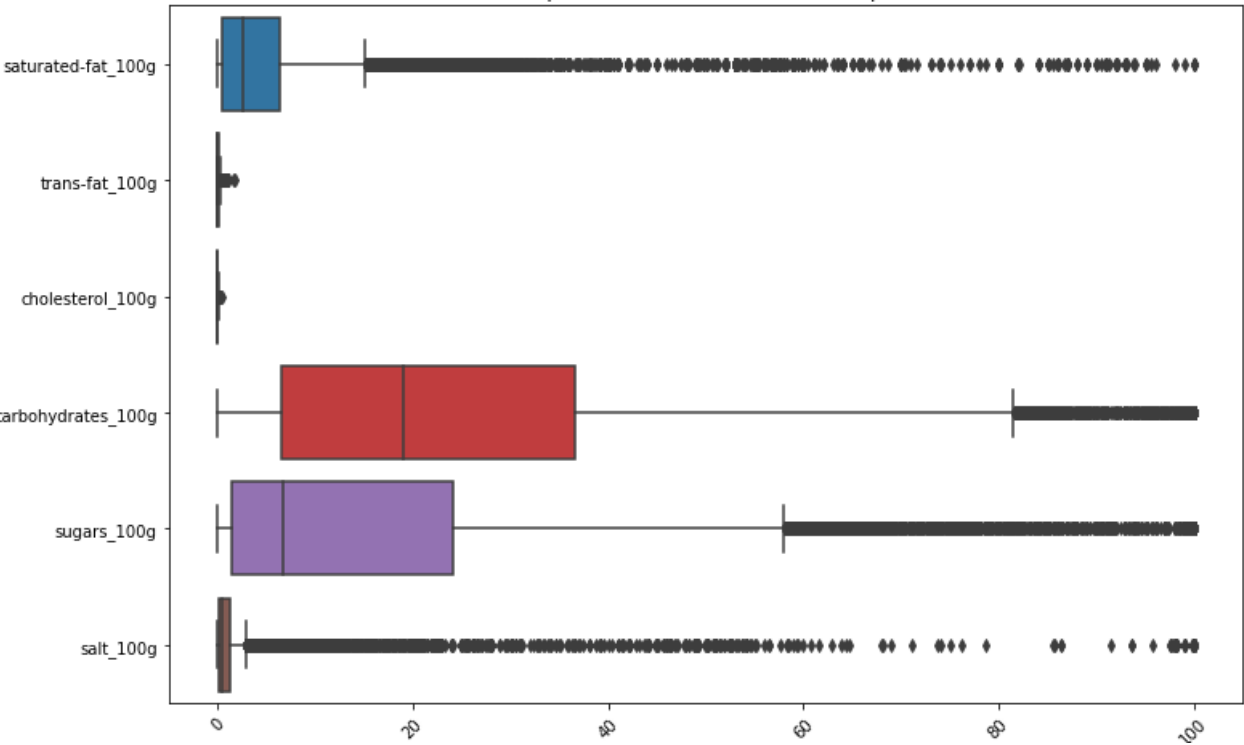
RGO: Reflux gastro-œsophagien.

2. Exploration des données

1. Analyse univariée

➡ Les variables quantitatives pertinentes pour le **RGO-score**

Les variables qui favorisent le RGO (mauvaises pour la santé)



Augmentent le RGO

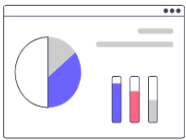
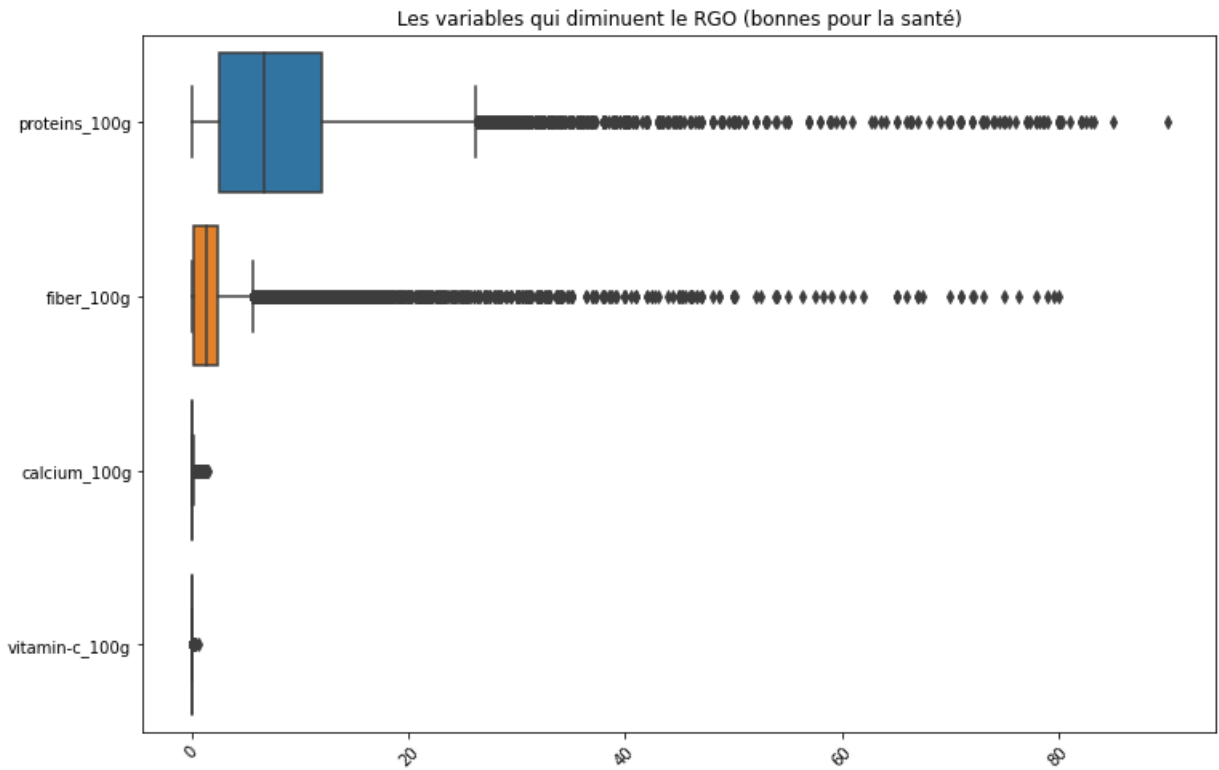
- saturated-fat_100g
- trans-fat_100g
- cholesterol_100g
- carbohydrates_100g
- salt_100g
- sugars_100g

	saturated-fat_100g	trans-fat_100g	cholesterol_100g	carbohydrates_100g	sugars_100g	salt_100g
count	74235.000000	74235.000000	74235.000000	74235.000000	74235.000000	74235.000000
mean	5.381542	0.060425	0.016925	25.824153	15.171771	1.205256
std	7.856967	0.109938	0.019789	23.160358	18.443746	3.935159
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	0.400000	0.000000	0.000200	6.600000	1.500000	0.100000
50%	2.600000	0.014000	0.009400	19.000000	6.700000	0.476000
75%	6.300000	0.100000	0.033822	36.580000	24.100000	1.200000
max	100.000000	1.780000	0.378000	100.000000	100.000000	100.000000

2. Exploration des données

1. Analyse univariée

➡ Les variables quantitatives pertinentes pour le **RGO-score**



	proteins_100g	fiber_100g	calcium_100g	vitamin-c_100g
count	74235.000000	74235.000000	74235.000000	74235.000000
mean	8.120374	2.069124	0.059334	0.022473
std	7.318101	3.473455	0.079199	0.017137
min	0.000000	0.000000	0.000000	0.000000
25%	2.500000	0.100000	0.031720	0.008400
50%	6.670000	1.362000	0.038760	0.020600
75%	12.000000	2.340000	0.057820	0.029320
max	90.000000	80.000000	1.534000	0.582000

**Diminuent
le RGO**

- proteins_100g
- fiber_100g
- calcium_100g
- vitamin-c_100g



Concevez une application au service de la santé publique



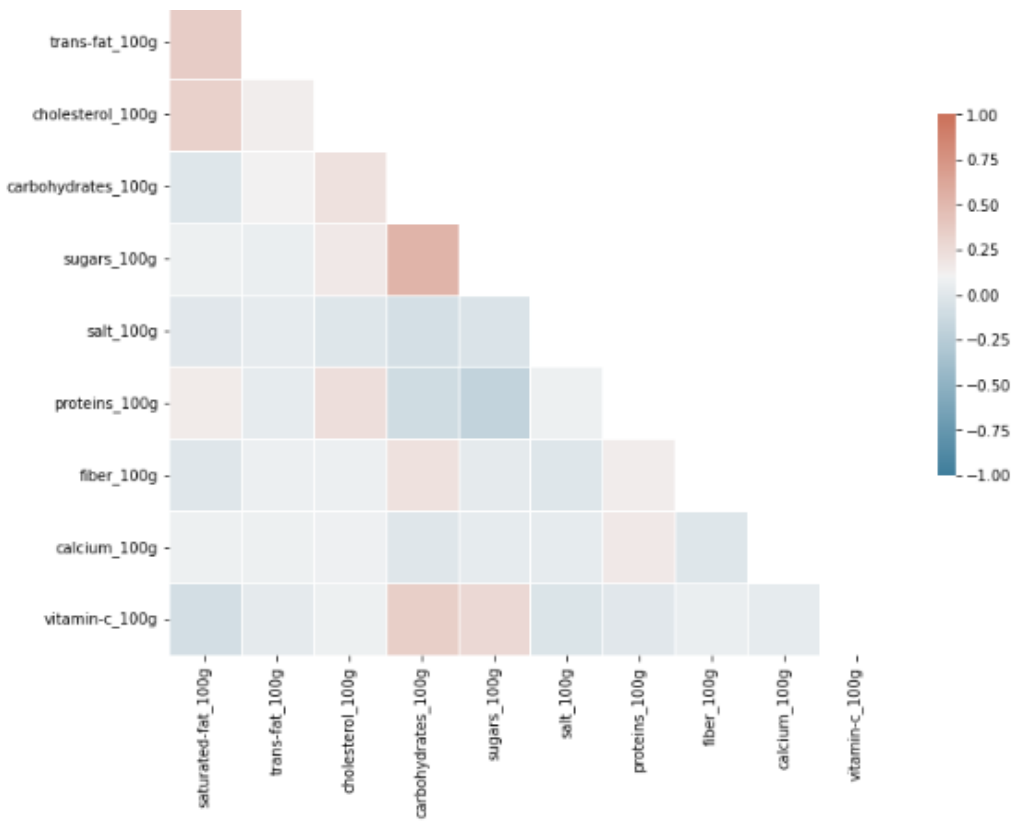
- **Idée d'application**
- **Partie 1: Inspection et nettoyage des données**
 - Analyse de la forme des données
 - Nettoyage sur les variables
 - Nettoyage sur les individus
 - Traitement des valeurs manquantes
- **Partie 2: Exploration des données**
 - Analyse univariée
 - Analyse bivariée
 - Analyse multivariée
 - Application proposée
- **Conclusion**



2. Exploration des données

2. Analyse bivariée

Pour analyser les corrélations linéaires entre nos variables quantitatives, nous allons réaliser un **test de corrélation de Pearson** et afficher ses résultats dans un heatmap :



```
1 data_Food[Features_RGO].corr().round(2)
```

	saturated-fat_100g	trans-fat_100g	cholesterol_100g	carbohydrates_100g	sugars_100g	salt_100g	proteins_100g	fiber_100g	calcium_100g	vitamin-c_100g
saturated-fat_100g	1.00	0.36	0.33	-0.02	0.07	0.00	0.15	-0.00	0.08	-0.08
trans-fat_100g	0.36	1.00	0.13	0.11	0.06	0.04	0.03	0.06	0.08	0.03
cholesterol_100g	0.33	0.13	1.00	0.22	0.17	-0.02	0.23	0.07	0.09	0.07
carbohydrates_100g	-0.02	0.11	0.22	1.00	0.53	-0.07	-0.11	0.22	-0.00	0.34
sugars_100g	0.07	0.06	0.17	0.53	1.00	-0.04	-0.19	0.03	0.03	0.28
salt_100g	0.00	0.04	-0.02	-0.07	-0.04	1.00	0.08	-0.02	0.04	-0.04
proteins_100g	0.15	0.03	0.23	-0.11	-0.19	0.08	1.00	0.14	0.16	0.00
fiber_100g	-0.00	0.06	0.07	0.22	0.03	-0.02	0.14	1.00	-0.01	0.06
calcium_100g	0.08	0.08	0.09	-0.00	0.03	0.04	0.16	-0.01	1.00	0.04
vitamin-c_100g	-0.08	0.03	0.07	0.34	0.28	-0.04	0.00	0.06	0.04	1.00



Matrice de corrélation des variables



- Il y a une corrélation entre les glucides 'carbohydrates_100g' et le sucre 'sugars_100'. Ce résultat est attendu vu que le glucide est un sucre transformé ou simple.
- Le reste des variables semblent ne pas être corrélées entre elles.

Concevez une application au service de la santé publique




- **Idée d'application**
- **Partie 1: Inspection et nettoyage des données**
 - Analyse de la forme des données
 - Nettoyage sur les variables
 - Nettoyage sur les individus
 - Traitement des valeurs manquantes
- **Partie 2: Exploration des données**
 - Analyse univariée
 - Analyse bivariée
 - Analyse multivariée
 - Application proposée
- **Conclusion**

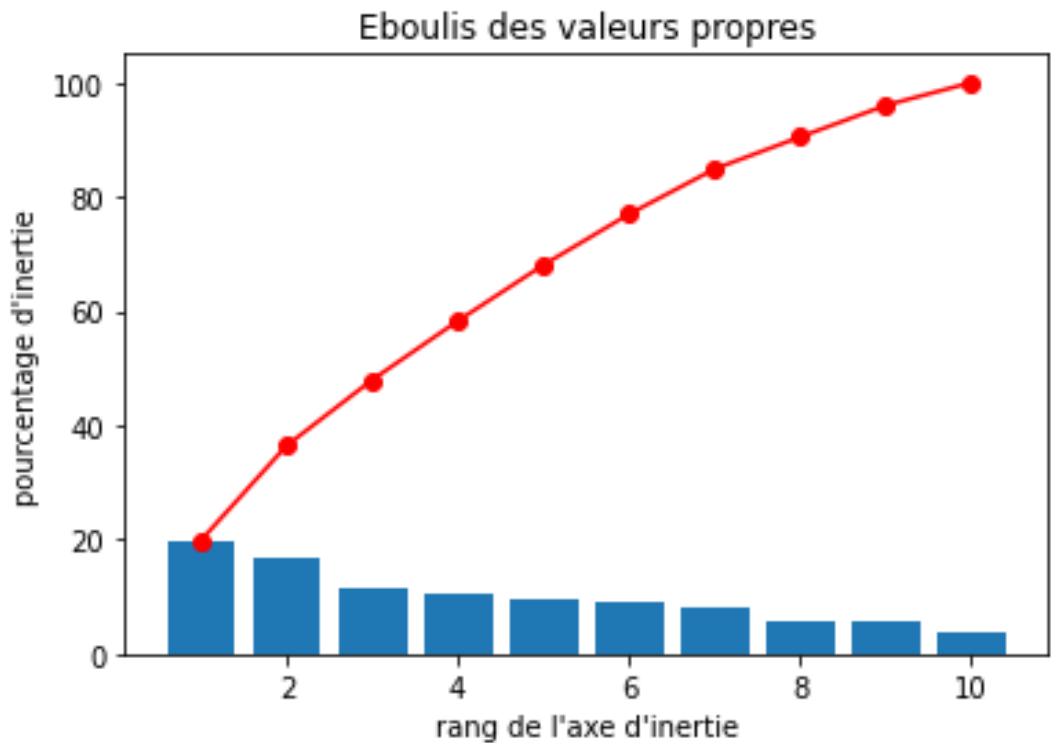



2. Exploration des données


3. Analyse multivariée

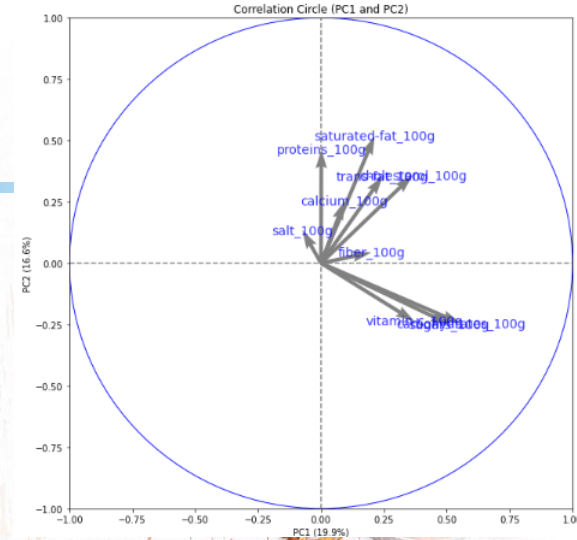
Réduction de dimension: Analyse en composantes Principales ACP

 Peut-on réduire la dimension de l'étude de **10 variables** à moins en utilisant l'ACP sans perdre beaucoup d'information?



 Le premier plan factoriel (axes 1 et 2) couvre une inertie de 36%.

 On obtient une inertie de 80% à partir de 7 composantes sur 10. Une réduction de dimension **n'est pas très intéressantes dans notre cas d'étude.**



Concevez une application au service de la santé publique



- **Idée d'application**
- **Partie 1: Inspection et nettoyage des données**
 - Analyse de la forme des données
 - Nettoyage sur les variables
 - Nettoyage sur les individus
 - Traitement des valeurs manquantes
- **Partie 2: Exploration des données**
 - Analyse univariée
 - Analyse bivariée
 - Analyse multivariée
 - Application proposée
- **Conclusion**



2. Exploration des données

4. Calcul du RGO-score/RGO-grade

- Sélection des variables pertinentes.
- Normalisation des variables: Min-Max Scaler



	saturated-fat_100g	trans-fat_100g	cholesterol_100g	carbohydrates_100g	sugars_100g	salt_100g	proteins_100g	fiber_100g	calcium_100g	vitamin-c_100g
count	74235.000000	74235.000000	74235.000000	74235.000000	74235.000000	74235.000000	74235.000000	74235.000000	74235.000000	74235.000000
mean	0.053815	0.033946	0.044776	0.258242	0.151718	0.012053	0.090226	0.025864	0.038680	0.038613
std	0.078570	0.061763	0.052351	0.231604	0.184437	0.039352	0.081312	0.043418	0.051629	0.029445
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	0.004000	0.000000	0.000529	0.066000	0.015000	0.001000	0.027778	0.001250	0.020678	0.014433
50%	0.026000	0.007865	0.024868	0.190000	0.067000	0.004760	0.074111	0.017025	0.025267	0.035395
75%	0.063000	0.056180	0.089476	0.365800	0.241000	0.012000	0.133333	0.029250	0.037692	0.050378
max	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000

- Calcul du RGO-score et RGO-grade



```
1 data_RGO_scaled['Score_RGO'] = (- data_RGO_scaled[Favorise_RGO].mean(axis=1) + data_RGO_scaled[Diminue_RGO].mean(axis=1))
```

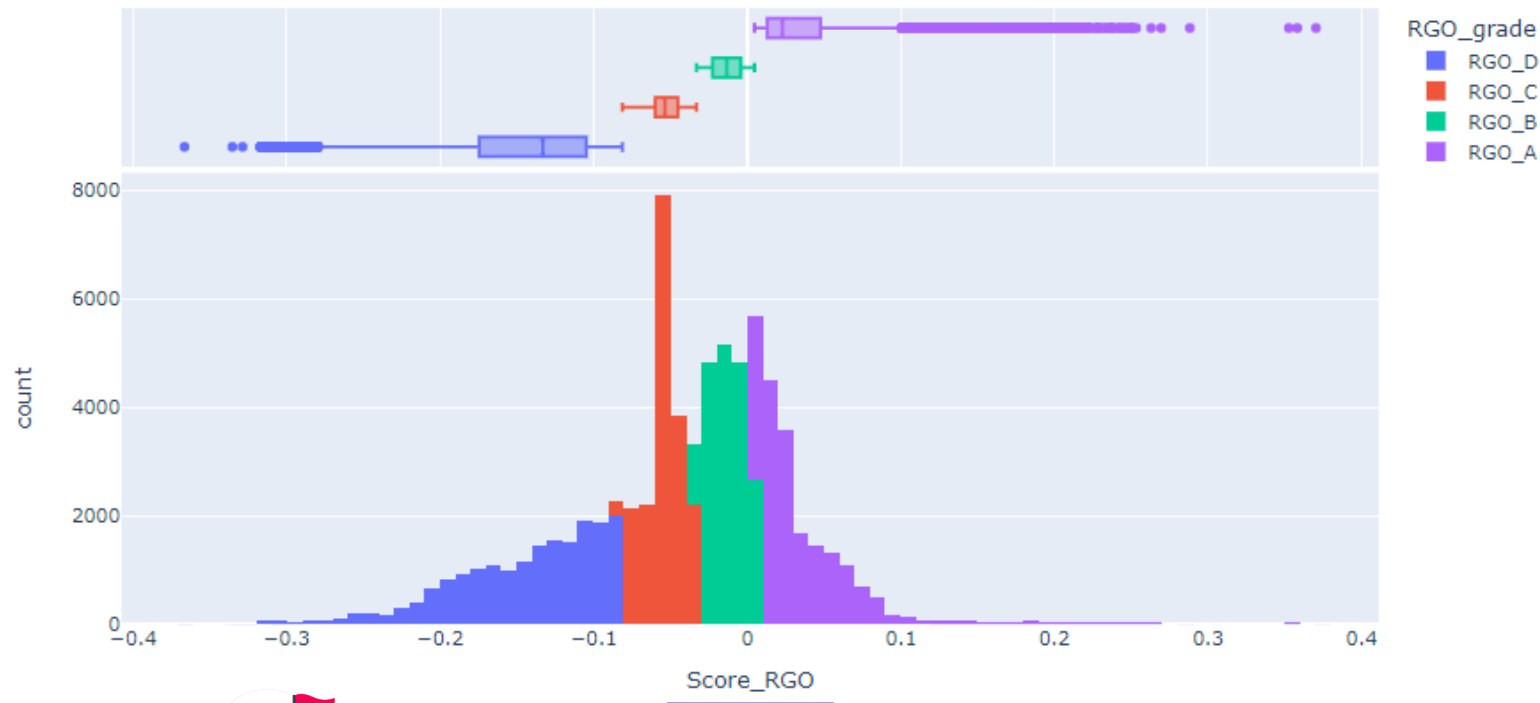
```
1 data_Food["RGO_grade"] = pd.qcut(data_Food['Score_RGO'],4,labels=['RGO_D','RGO_C','RGO_B','RGO_A'])
```


2. Exploration des données

4. Etude RGO-score/RGO-grade

```
1 data_Food["Score_RGO"].describe()
```

count 74235.000000
mean -0.044079
std 0.071648
min -0.366367
25% -0.081280
50% -0.032904
75% 0.004836
max 0.370401
Name: Score_RGO, dtype: float64



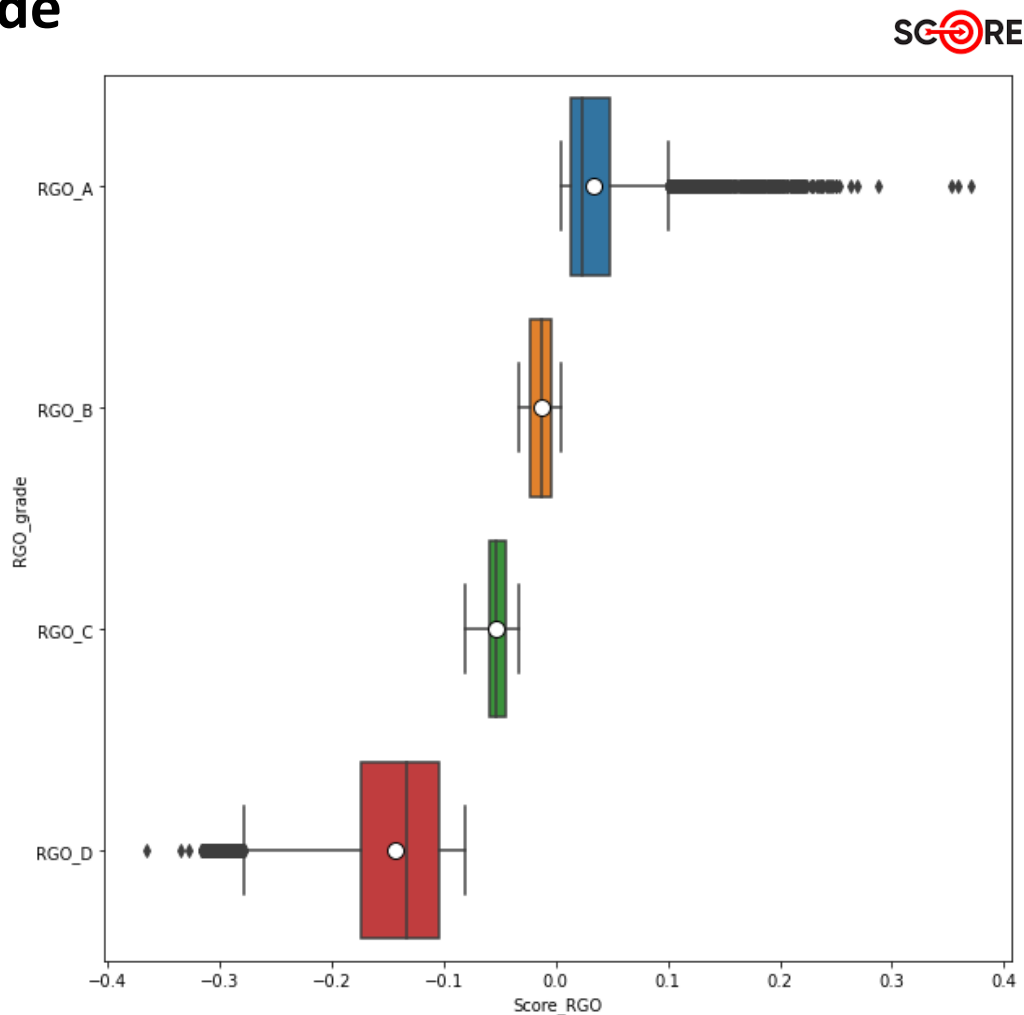
Les différentes classes sont bien représentées



2. Exploration des données

4. Etude du RGO-score/RGO-grade

- A: Nutriment à favoriser (diminue le reflux)
- D: Nutriment à éviter (augmente le reflux)



Plus le RGO-score est grand plus le produit est bon et donc classé en RGO_A.

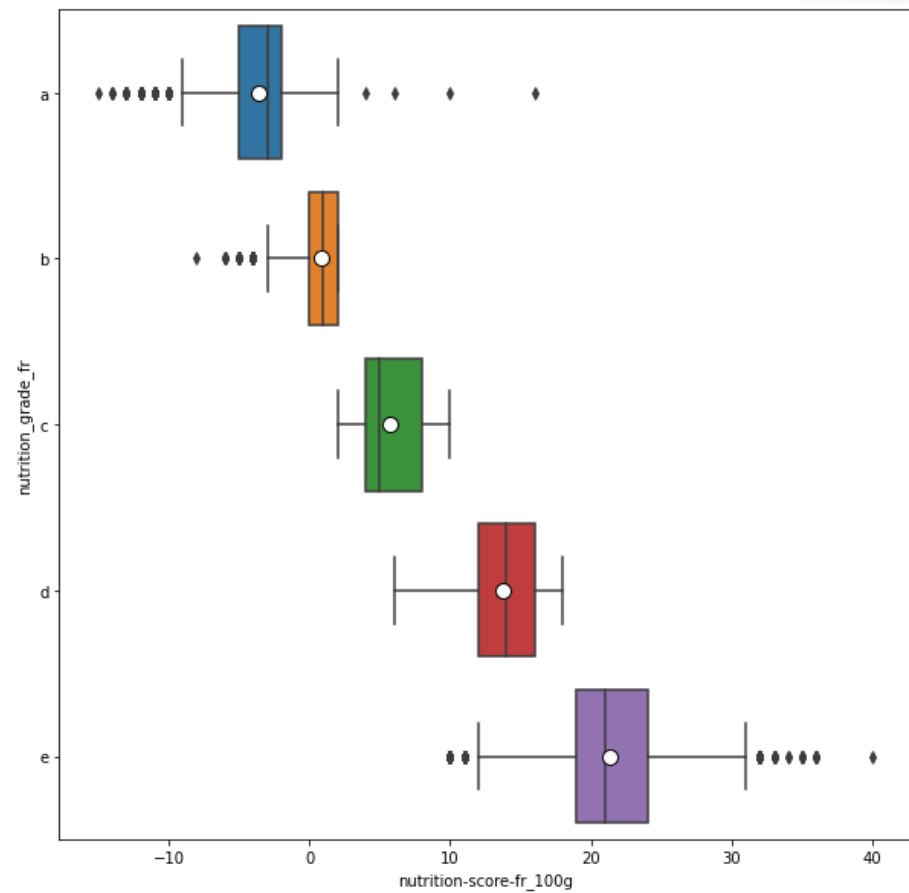
2. Exploration des données

4. Etude du RGO-score/RGO-grade VS Nutri-score/Nutri-grade

Rappel: Nutrition-score/Nutrition-grade ?!

Le calcul du nutri-score se base sur les variables suivantes:

- Energy
- Sucre
- Gras saturé
- Sel
- + Protéines
- + Fibres
- + Fruits et légumes



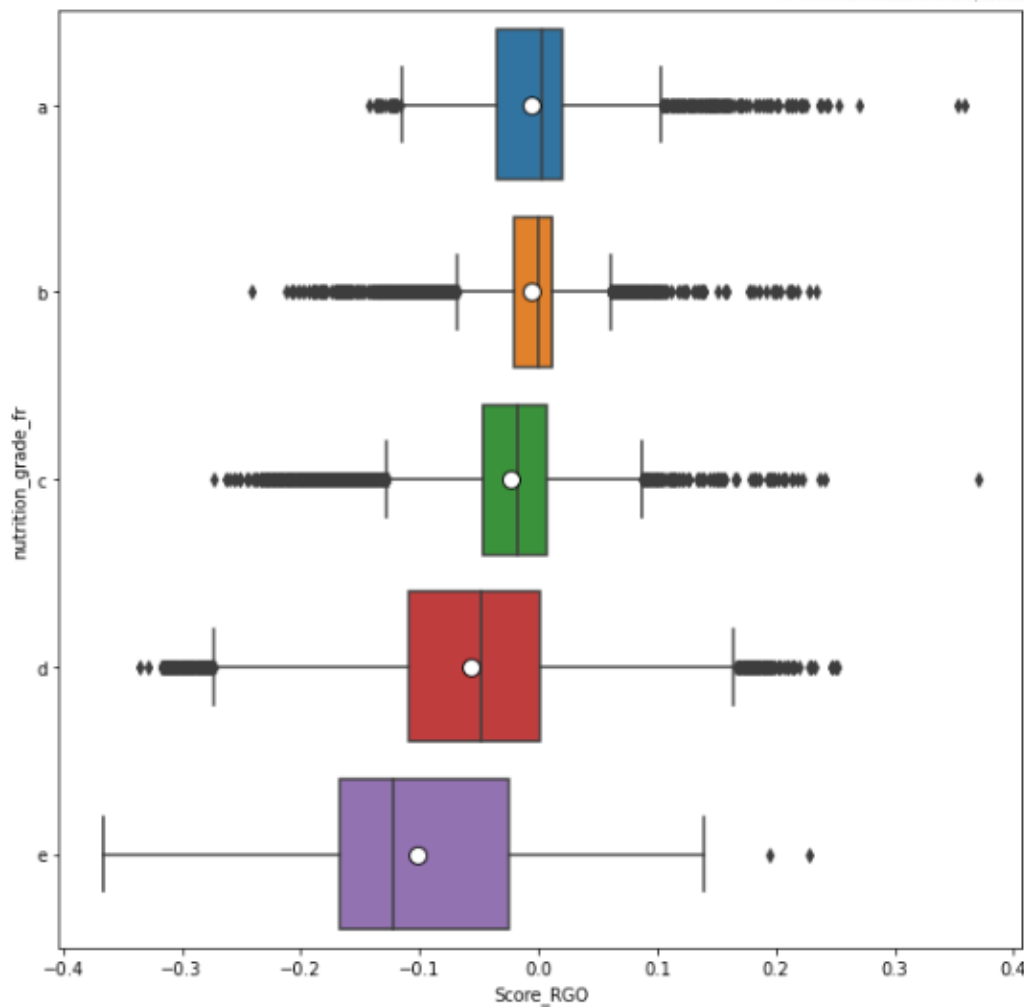
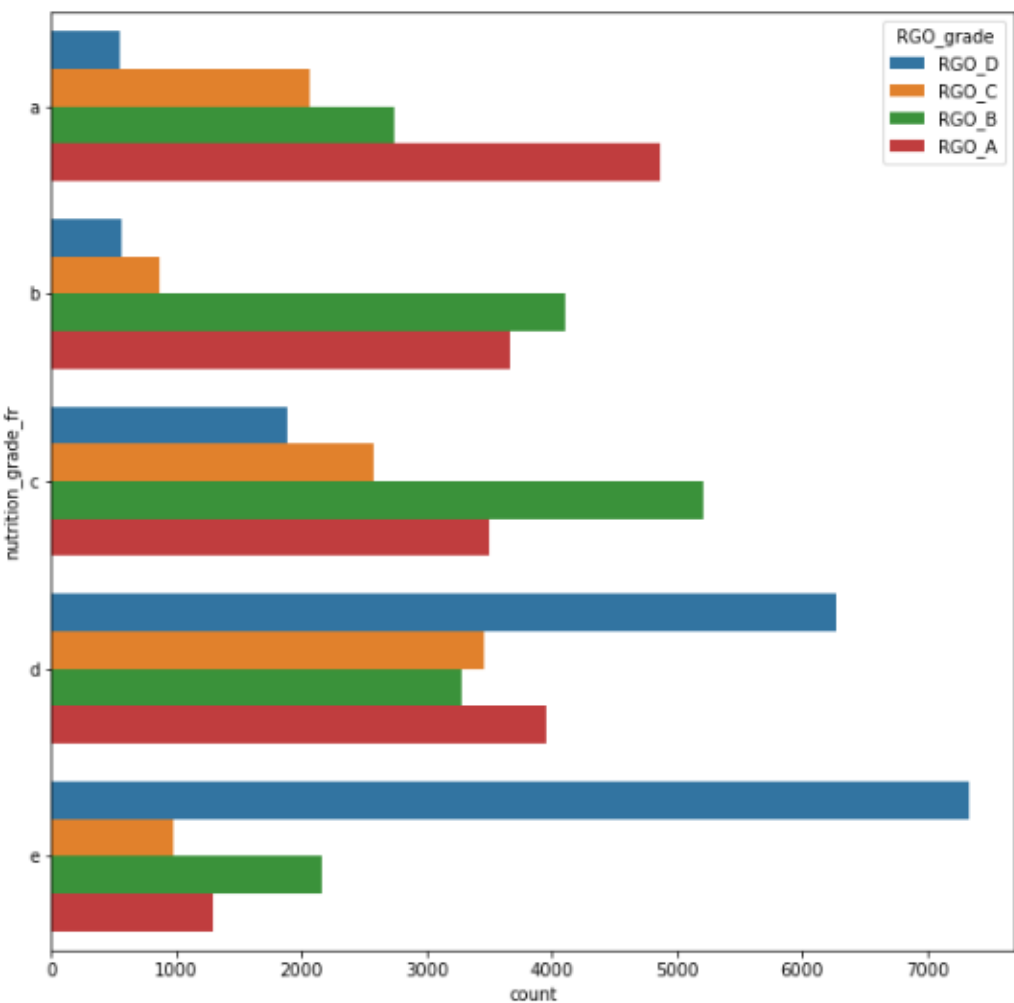
Plus le Nutri-score est faible plus le produit est bon et donc classé en nutri-grade A.



2. Exploration des données



4. Etude du RGO-score/RGO-grade VS Nutrition-score/Nutrition-grade



2. Exploration des données



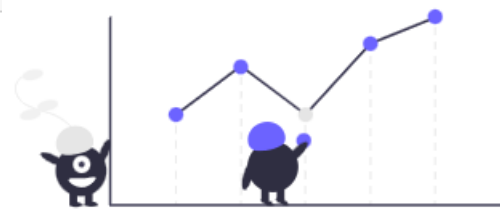
4. Etude du RGO-score/RGO-grade VS Nutrition-score/Nutrition-grade

Le RGO-score et le nutri-score sont corrélés négativement



```
1 data_Food.loc[:,["nutrition-score-fr_100g","Score_RGO"]].corr().round(2)
```

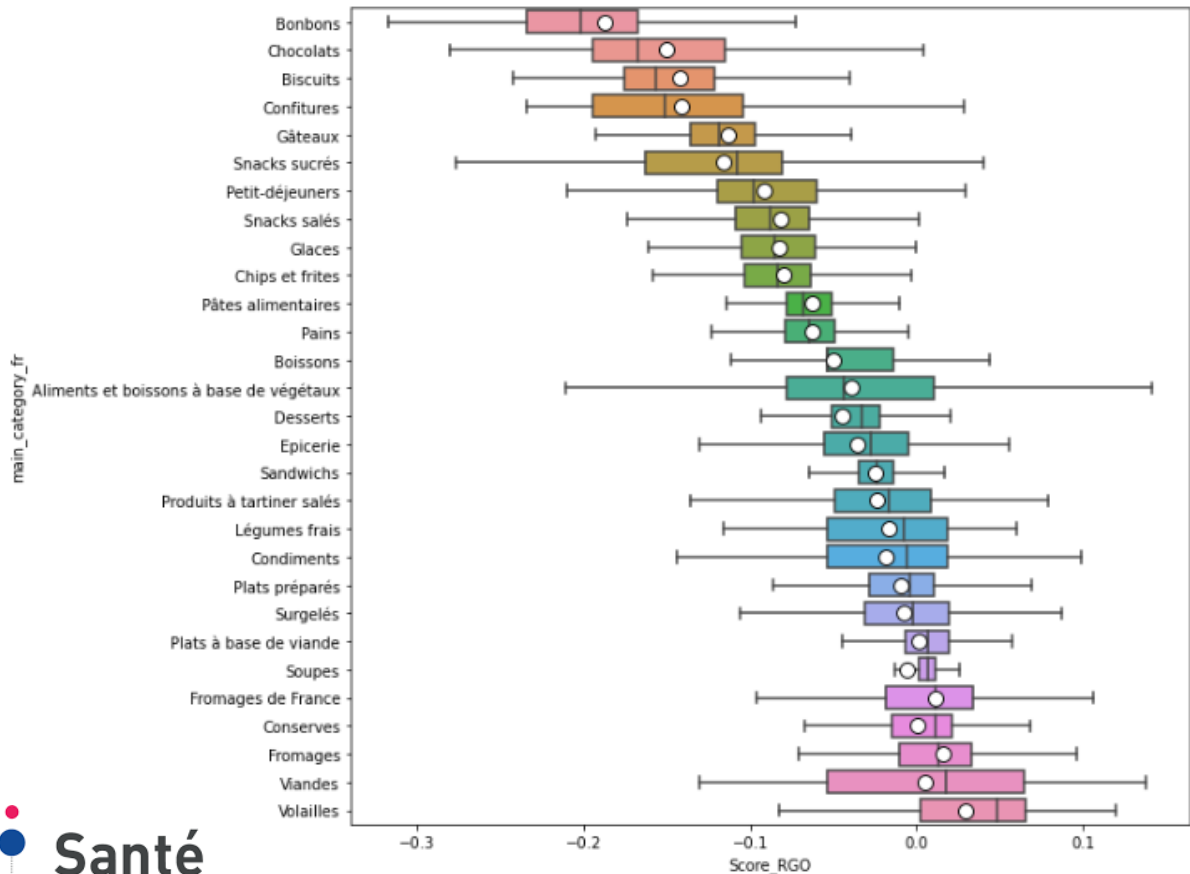
	nutrition-score-fr_100g	Score_RGO
nutrition-score-fr_100g	1.00	-0.46
Score_RGO	-0.46	1.00



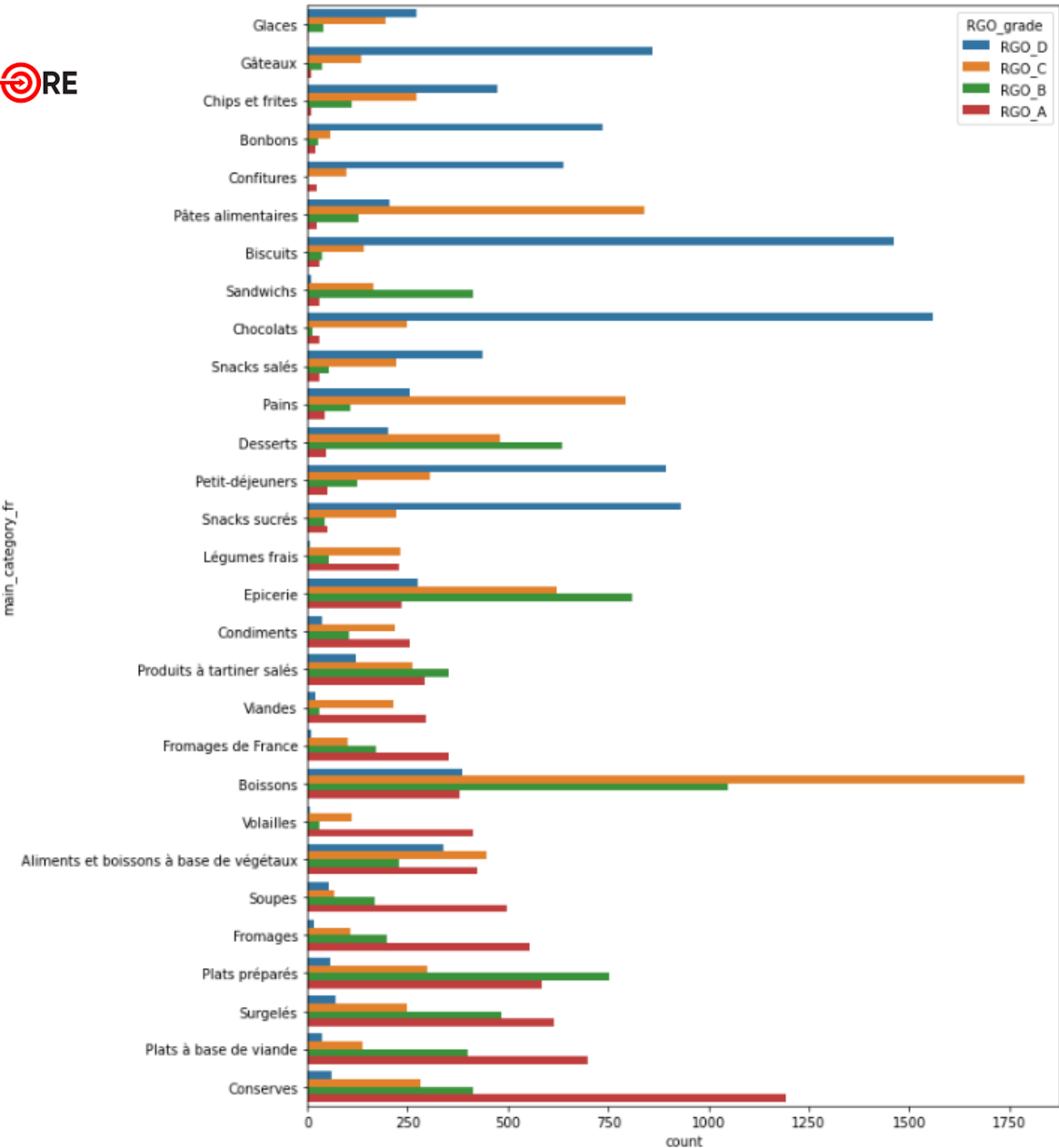
2. Exploration des données



4. Etude du RGO-grade VS les catégories



main_category_fr



La catégorie du produit semble avoir un impact sur la distribution des RGO-scores.

2. Exploration des données

4. Etude du RGO-score VS les catégories : ANOVA (Analyse de la Variance)

Hypothèse :

H0 : La distribution des échantillons est similaire (et donc la catégorie n'a aucune influence sur le RGO-score).

	sum_sq	df	F	PR(>F)
main_category_fr	77.495204	18.0	1730.350206	0.0
Residual	69.241412	27829.0	NaN	NaN

Les résultats du **test de Fisher** nous indiquent ici une p-value de 0 pour les catégories sélectionnées, donc inférieur au niveau de test de 5%. Nous rejetons donc l'hypothèse H0 selon laquelle les distributions sont identiques.



La catégorie de produit a donc bien une influence sur le RGO-score.



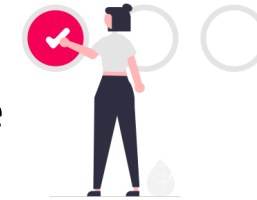
2. Exploration des données

Synthèse

Dans cette deuxième partie, nous avons effectué:

- des **analyses univariées** pour des variables **quantitatives** et **qualitatives**
- l'analyse de la corrélation entre les variables pertinentes (**analyse bivariée**)
- une analyse multivariée pour étudier la possibilité de réduire les dimensions du problème avec l'**ACP**
- le calcul d'un RGO-score/ RGO-grade
- une étude du RGO-score/RGO-grade VS le nutri-score/nutri-grade
- une étude du RGO-score VS la catégorie du produit (**ANOVA**)

SCORE



Ces opérations ont permis de conclure sur la faisabilité de notre projet

Le jeu de données est bien varié.

Les différentes catégories nutritionnels de produit sont présentes.

Peu de variables suffisent pour calculer le RGO-score/RGO-grade.

Selon la catégorie du produit, le RGO-score peut être plus ou moins important.

Concevez une application au service de la santé publique



- **Idée d'application**
- **Partie 1: Inspection et nettoyage des données**
 - Analyse de la forme des données
 - Nettoyage sur les variables
 - Nettoyage sur les individus
 - Traitement des valeurs manquantes
- **Partie 2: Exploration des données**
 - Analyse univariée
 - Analyse bivariée
 - Analyse multivariée
 - Application proposée
- **Conclusion**



Concevez une application au service de la santé publique

Conclusion



 Application: **RGO-scan**: RGO-score/RGO-grade

 Inspection et nettoyage des données de l'OpenFoodFacts

 traitement des valeurs manquantes, aberrantes et atypiques

 Analyses statistiques univariée, bivariée et multivariée

 une base de données riche et variée

 **l'application est faisable**



Amélioration: étude plus poussée des facteurs du reflux

RGO: Reflux gastro-œsophagien.

Concevez une application au service de la santé publique

Merci de votre attention

