



**Concevez une application au  
service de la santé publique**

Alexandre Delaguillaumie

## ***1. Idée d'application***



## *1. Idée d'application*



### ***Le marché***

**16,5 millions d'utilisateurs Yuka**  
Représente 25% des français

**Beaucoup sont prêts à payer plus cher pour manger mieux**  
Représente 75% des français

**Mais il reste un nombre considérable de français**  
Oubliés car ils n'ont pas les moyens

# 1. Idée d'application

**cible**

## Les oubliés !

### **Qui ne veulent pas tout scanner**

Et qui n'ont pas d'application

### **Non sensibles**

Aux enjeux environnementaux et nutritionnels

### **Qui ne cuisinent pas**

Et achète souvent des plats préparés

### **Qui grignotent entre les repas**

Et qui n'ont pas encore de problème de santé

### **Price sensitive**

Qui souhaite consommer mieux

### **Enfants et parents**

Qui ont tendance à remplir les tiroirs de cochonneries



## 1. Idée d'application

### **utilisation**

Il suffit de scanner ou de rentrer le nom du produit dans l'app

**temps**

**argent**

En fonction des 2 seules choses nécessaires pour pouvoir manger

## L'APP VA PROPOSER DIFFÉRENTES SOLUTIONS

Allant de la **recette** diététique alternative...

... Au **produit** plus haut de gamme possédant un meilleur nutriscore que l'on peut se procurer en grande surface.

## *2. Opérations de nettoyage*



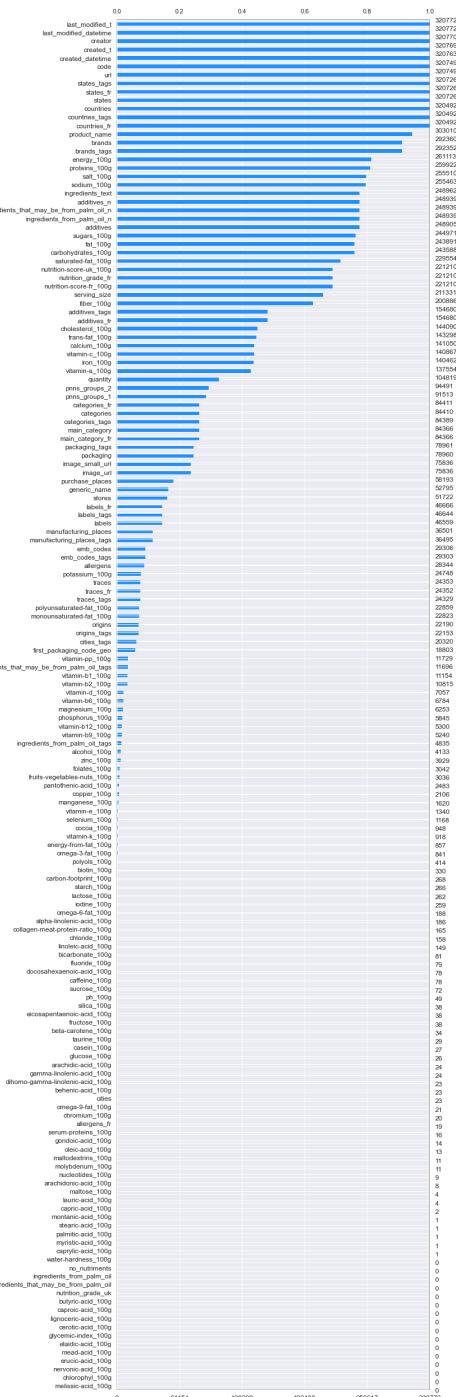
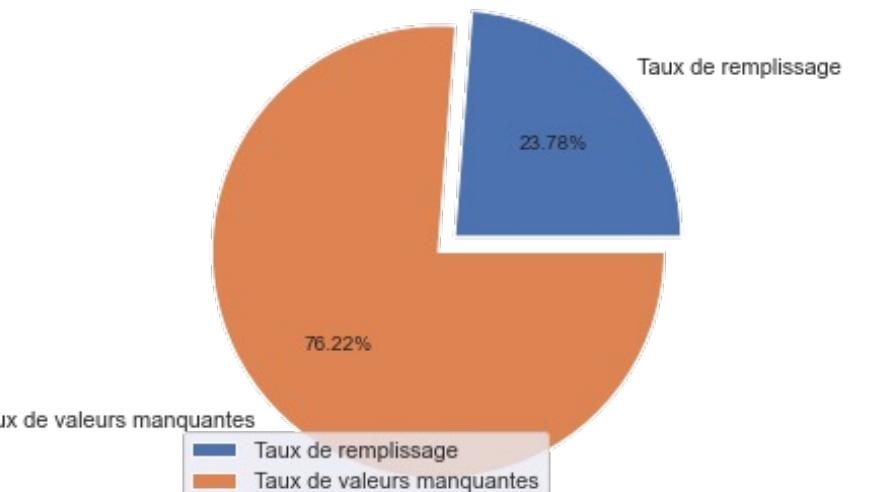
# 2. Opérations de nettoyage

## Overview

### Structure des données

	Caractéristiques	Valeurs
0	Nombre de lignes	320772
1	Nombre de colonnes	162
2	Nombre de variables catégorielles	56
3	Nombre de variables numériques	106
4	Pourcentage de données manquantes	76
5	Nombre de doublons	0

### Taux de remplissage



## *2. Opérations de nettoyage*

### *Les différentes étapes*

1

#### **Retenir les données pertinentes**

##### **Transformer en DataFrame français**

Produits présents uniquement en France

##### **Tri de variables**

Selon la pertinence

##### **Tri de variables**

Selon le taux de remplissage

##### **Catégorisation de données**

Via les PNNS groups

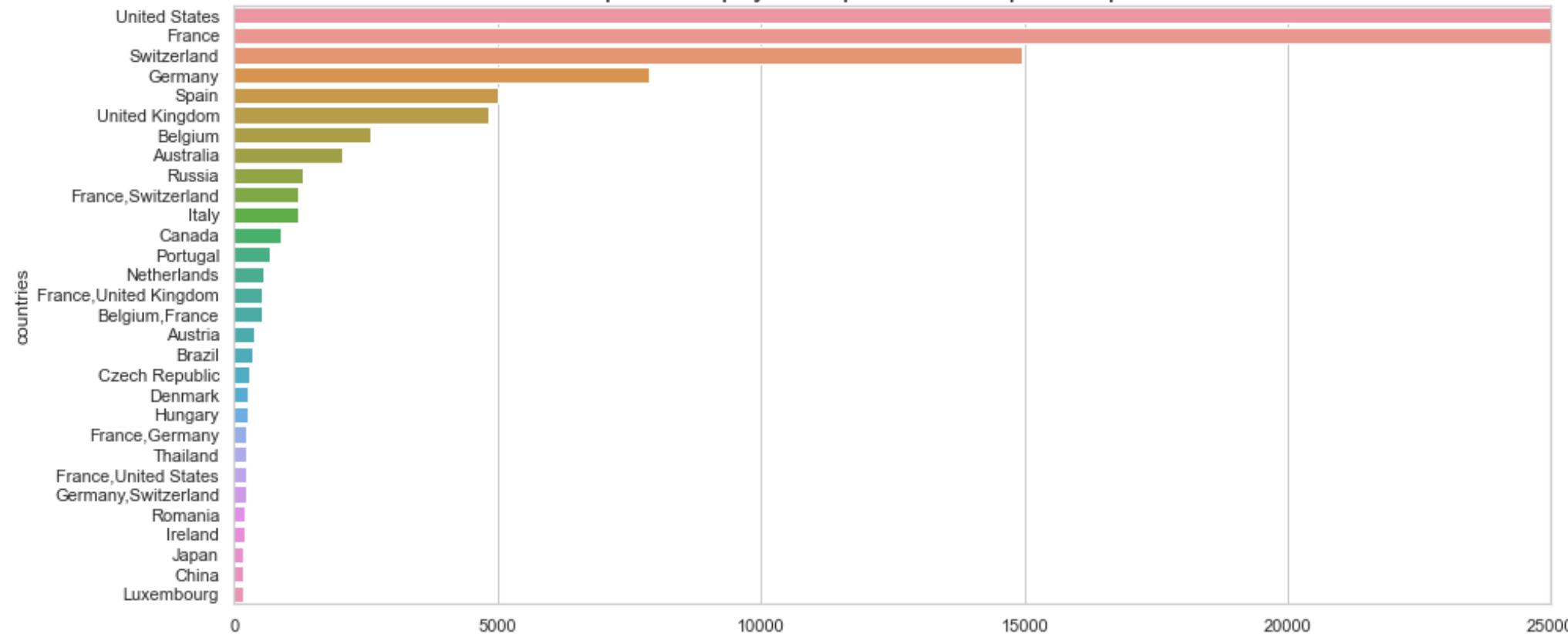
# *Transformer en DataFrame français*

Taille du DataFrame :

**320 772**

**98 447**

Top 20 des pays comptabilisant le plus de produits



## *2. Opérations de nettoyage*

### *Les différentes étapes*

1

#### **Retenir les données pertinentes**

Transformer en DataFrame français

Produits présents uniquement en France

##### **Tri de variables**

Selon la pertinence

##### **Tri de variables**

Selon le taux de remplissage

##### **Catégorisation de données**

Via les PNNS groups

Nombre de colonnes :

162

54

## *2. Opérations de nettoyage*

### *Les différentes étapes*

1

#### **Retenir les données pertinentes**

Transformer en DataFrame français

Produits présents uniquement en France

**Tri de variables**

Selon la pertinence

**Tri de variables**

Selon le taux de remplissage

**Catégorisation de données**

Via les PNNS groups

Nombre de colonnes :

54

22

# Tri des variables

## Pourcentage de remplissage par colonne

	null_percentage
code	0
states_fr	0
states_tags	0
countries_fr	0
countries_tags	0
countries	0
states	0
created_t	0
last_modified_t	0
creator	0
url	0
last_modified_datetime	0
created_datetime	0
product_name	6
brands	9

## Après sélection de produits présents en France

	null_percentage_x	null_percentage_y	évolution
code	0	0	0
pnns_groups_2	71	32	-55
pnns_groups_1	71	34	-52
energy_100g	19	35	84
proteins_100g	19	35	84
categories_fr	74	37	-50
saturated_fat_100g	28	37	32
sodium_100g	20	37	85
sugars_100g	24	37	54
nutrition_grade_fr	31	38	23
nutrition_score_fr_100g	31	38	23
fat_100g	24	52	117
carbohydrates_100g	24	52	117
fiber_100g	37	54	46
alcohol_100g	99	98	-1
polyunsaturated_fat_100g	93	99	6
monounsaturated_fat_100g	93	99	6
omega_3_fat_100g	100	99	-1
cholesterol_100g	55	100	82
omega_9_fat_100g	100	100	0
omega_6_fat_100g	100	100	0
polyols_100g	100	100	0
trans_fat_100g	55	100	82

## *2. Opérations de nettoyage*

### *Les différentes étapes*

#### **Retenir les données pertinentes**

1

**Transformer en DataFrame français**

Produits présents uniquement en France

**Tri de variables**

Selon la pertinence

**Tri de variables**

Selon le taux de remplissage

**Catégorisation de données**

Via les PNNS groups

# Catégorisation de données



## Taux de remplissage

```
Nombre de valeurs dans categories_fr : 61830.  
Nombre de valeurs dans pnns_groups_1 : 64763.  
Nombre de valeurs dans pnns_groups_2 : 66926.  
Nombre de valeurs dans les 3 tags réunis : 66949.
```

## PNNS Group 1

Il y a 11 catégories dans ce tag.

```
[nan,  
 'unknown',  
 'Cereals and potatoes',  
 'Sugary snacks',  
 'Beverages',  
 'Fish Meat Eggs',  
 'Composite foods',  
 'Fruits and vegetables',  
 'Milk and dairy products',  
 'Salty snacks',  
 'Fat and sauces']
```

## PNNS Group 2

Il y a 37 catégories dans ce tag.

```
[nan,  
 'Unknown',  
 'Vegetables',  
 'Biscuits And Cakes',  
 'Sweets',  
 'Non-Sugared Beverages',  
 'Sweetened Beverages',  
 'Meat',  
 'One-Dish Meals',  
 'Soups',  
 'Chocolate Products',  
 'Alcoholic Beverages',  
 'Sandwich',  
 'Cheese',  
 'Appetizers',  
 'Dressings And Sauces',  
 'Dried Fruits',  
 'Nuts',  
 'Breakfast Cereals',  
 'Pizza Pies And Quiche',  
 'Fruits',  
 'Fruit Juices',  
 'Fats',  
 'Cereals',  
 'Bread',  
 'Processed Meat',  
 'Ice Cream',  
 'Fish And Seafood',  
 'Pastries',  
 'Milk And Yogurt',  
 'Dairy Desserts',  
 'Tripe Dishes',  
 'Artificially Sweetened Beverages',  
 'Fruit Nectars',  
 'Potatoes',  
 'Eggs',  
 'Salty And Fatty Products']
```

## *2. Opérations de nettoyage*

### *Les différentes étapes*

1

#### **Retenir les données pertinentes**

Transformer en DataFrame français

Produits présents uniquement en France

**Tri de variables**

Selon la pertinence

**Tri de variables**

Selon le taux de remplissage

**Catégorisation de données**

Via les PNNS groups

## *2. Opérations de nettoyage*

### *Les différentes étapes*

2

#### **Traitement des valeurs aberrantes**

**Valeurs supérieures à 100 pour chaque colonne**

Remplacé par NaN

**Somme supérieure à 110 pour les macronutriments**

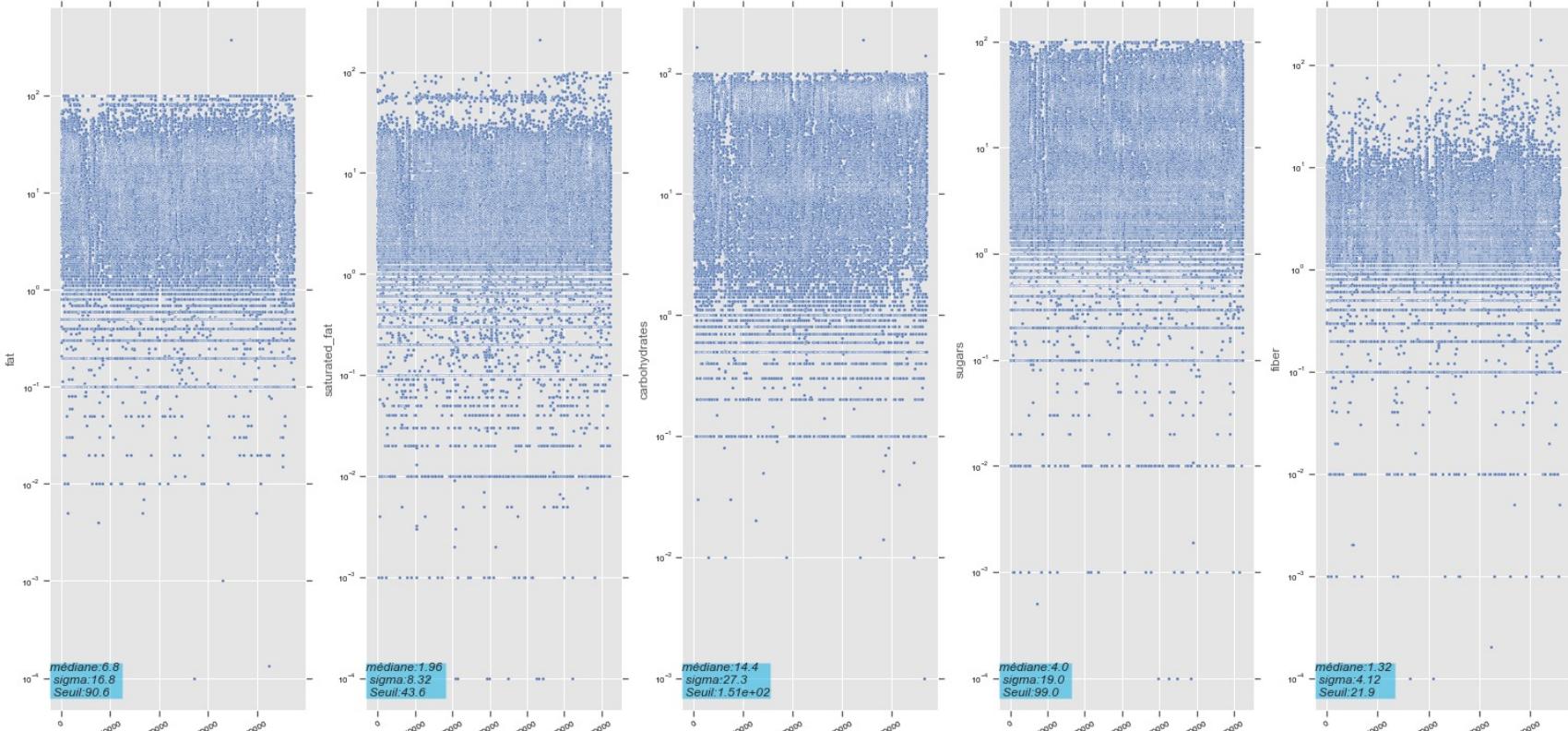
Suppression des produits

**Somme des valeurs nutritionnelle égale à 0**

Suppression des produits

# Traitement des valeurs aberrantes

Dispersion des données nutritionnelles  
Visualisation des outliers



## *2. Opérations de nettoyage*

### *Les différentes étapes*

2

#### **Traitement des valeurs aberrantes**

Valeurs supérieures à 100 pour chaque colonne

Remplacé par NaN

**Somme supérieure à 110 pour les macronutriments**

Suppression des produits

**Somme des valeurs nutritionnelle égale à 0**

Suppression des produits

## *2. Opérations de nettoyage*

### *Les différentes étapes*

2

Taille du DataFrame :

98 438

88 540

#### **Traitement des valeurs aberrantes**

Valeurs supérieures à 100 pour chaque colonne

Remplacé par NaN

Somme supérieure à 110 pour les macronutriments

Suppression des produits

Somme des valeurs nutritionnelle égale à 0

Suppression des produits

## *2. Opérations de nettoyage*

### *Les différentes étapes*

2

#### **Traitement des valeurs aberrantes**

**Valeurs supérieures à 100 pour chaque colonne**

Remplacé par NaN

**Somme supérieure à 110 pour les macronutriments**

Suppression des produits

**Somme des valeurs nutritionnelle égale à 0**

Suppression des produits

## *2. Opérations de nettoyage*

### *Les différentes étapes*

3

#### **Imputation de données**

##### **Remplacer par 0**

Pour les colonnes vides à au moins 97%

##### **Dépassement du seuil de calories possible pour 100g**

Remplacé par NaN puis imputé par le calcul

##### **Calcul du nutriscore**

Pas concluant donc non retenu

##### **KNN imputer sur les variables pertinentes**

K=7, erreur=1.45%

# *Imputation : Remplacer par 0*

## **Variables imputées**

monounsaturated\_fat\_100g  
polyunsaturated\_fat\_100g  
omega\_3\_fat\_100g  
omega\_6\_fat\_100g  
omega\_9\_fat\_100g  
trans\_fat\_100g  
cholesterol\_100g  
polyols\_100g  
alcohol\_100g

## **Pourcentage de données manquantes après imputation :**

---

```
code : 0
nutrition_grade_fr : 31
nutrition_score_fr_100g : 31
energy_100g : 27
pnns_groups_2 : 50
pnns_groups_1 : 52
fat_100g : 46
saturated_fat_100g : 30
monounsaturated_fat_100g : 0
polyunsaturated_fat_100g : 0
omega_3_fat_100g : 0
omega_6_fat_100g : 0
omega_9_fat_100g : 0
trans_fat_100g : 0
cholesterol_100g : 0
carbohydrates_100g : 47
sugars_100g : 30
proteins_100g : 28
sodium_100g : 30
polyols_100g : 0
fiber_100g : 49
alcohol_100g : 0
```

## *2. Opérations de nettoyage*

### *Les différentes étapes*

3

#### **Imputation de données**

##### **Remplacer par 0**

Pour les colonnes vides à au moins 97%

##### **Dépassement du seuil de calories possible pour 100g**

Remplacé par NaN puis imputé par le calcul

##### **Calcul du nutriscore**

Pas concluant donc non retenu

##### **KNN imputer sur les variables pertinentes**

K=7, erreur=1.45%

# *Imputation : Energie via calcul*

```
# Remplir les valeurs nulles par le calcul
def ImputeEnergy(data):

    energy_exception =['polyols_100g','fiber_100g', 'alcohol_100g']

    if data[energy_exception].any in list(data.columns):
        data['energy_100g'].fillna(data['fat_100g']*37.76 + data['carbohydrates_100g']*16.744 + data['proteins_100g']*16.744 + data['alcohol_100g']*29.300\
            + data['fiber_100g']*8.370+ data['polyols_100g']*1.040, inplace=True)
    else :
        data['energy_100g'].fillna(data['fat_100g']*37.674 + data['carbohydrates_100g']*16.744 + data['proteins_100g']*16.744 , inplace=True)

    return data
```

## *2. Opérations de nettoyage*

### *Les différentes étapes*

3

#### **Imputation de données**

##### **Remplacer par 0**

Pour les colonnes vides à au moins 97%

##### **Dépassement du seuil de calories possible pour 100g**

Remplacé par NaN puis imputé par le calcul

##### **Calcul du nutriscore**

Pas concluant donc non retenu

##### **KNN imputer sur les variables pertinentes**

K=7, erreur=1.45%

# Imputation : Nutriscore via calcul

Accuracy Score  
**46.27 %.**

## Attribution d'une note

Pour chaque variable selon son degré de présence dans le produit

## Attribution d'un score

Allant de A à E

## Vérification de fiabilité

En A/B test sur les données déjà remplies

```
#Nutriscore
def CalculNutriGrade(row):
    if row["CalculNutriScore"] < 0 :
        nutriscore = "a"
    elif ((row["CalculNutriScore"] >= 0) & (row["CalculNutriScore"] < 5)) :
        nutriscore = "b"
    elif ((row["CalculNutriScore"] >= 5) & (row["CalculNutriScore"] < 10)) :
        nutriscore = "c"
    elif ((row["CalculNutriScore"] >= 10) & (row["CalculNutriScore"] < 20)) :
        nutriscore = "d"
    else:
        nutriscore = "e"

    return nutriscore
```

```
def CalculNutriScore(row):
    #Energy
    if row["energy_100g"] <= 335:
        a = 0
    elif ((row["energy_100g"] > 335) & (row["energy_100g"] <= 1675)):
        a = 5
    else:
        a = 10
    #Sugar
    if row["sugars_100g"] <= 4.5:
        b = 0
    elif ((row["sugars_100g"] > 4.5) & (row["sugars_100g"] <= 22.5)):
        b = 5
    else:
        b = 10
    #saturated-fat
    if row["saturated_fat_100g"] <= 1:
        c = 0
    elif ((row["saturated_fat_100g"] > 1) & (row["saturated_fat_100g"] <= 5)):
        c = 5
    else:
        c = 10
    #sodium
    if (row["sodium_100g"]/1000) <= 90:
        d = 0
    elif (((row["sodium_100g"]/1000) > 90) & ((row["sodium_100g"]/1000) <= 450)):
        d = 5
    else:
        d = 10
    #fruits-vegetables-rate
    if row["fruits_vegetables_rate_100g"] <= 40:
        e = 0
    elif ((row["fruits_vegetables_rate_100g"] > 40) & (row["fruits_vegetables_rate_100g"] <= 80)):
        e = -2
    else:
        e = -5
    #fiber
    if row["fiber_100g"] <= 0.7:
        f = 0
    elif ((row["fiber_100g"] > 0.7) & (row["fiber_100g"] <= 3.5)):
        f = -2
    else:
        f = -5
    #proteins
    if row["proteins_100g"] <= 1.6:
        g = 0
    elif ((row["proteins_100g"] > 1.6) & (row["proteins_100g"] <= 8)):
        g = -2
    else:
        g = -5

    #Global_score
    global_score = a+b+c+d+e+f+g

    return global_score
```

## *2. Opérations de nettoyage*

### *Les différentes étapes*

3

Taille du DataFrame :

88 540

44 492

#### **Imputation de données**

**Remplacer par 0**

Pour les colonnes vides à au moins 97%

**Dépassement du seuil de calories possible pour 100g**

Remplacé par NaN puis imputé par le calcul

**Calcul du nutriscore**

Pas concluant donc non retenu

**KNN imputer sur les variables pertinentes**

K=7, erreur=1.45%

# Imputation : KNN Imputer

## Utilisation d'un KNeighborsRegressor

Afin de trouver une valeur optimale pour k

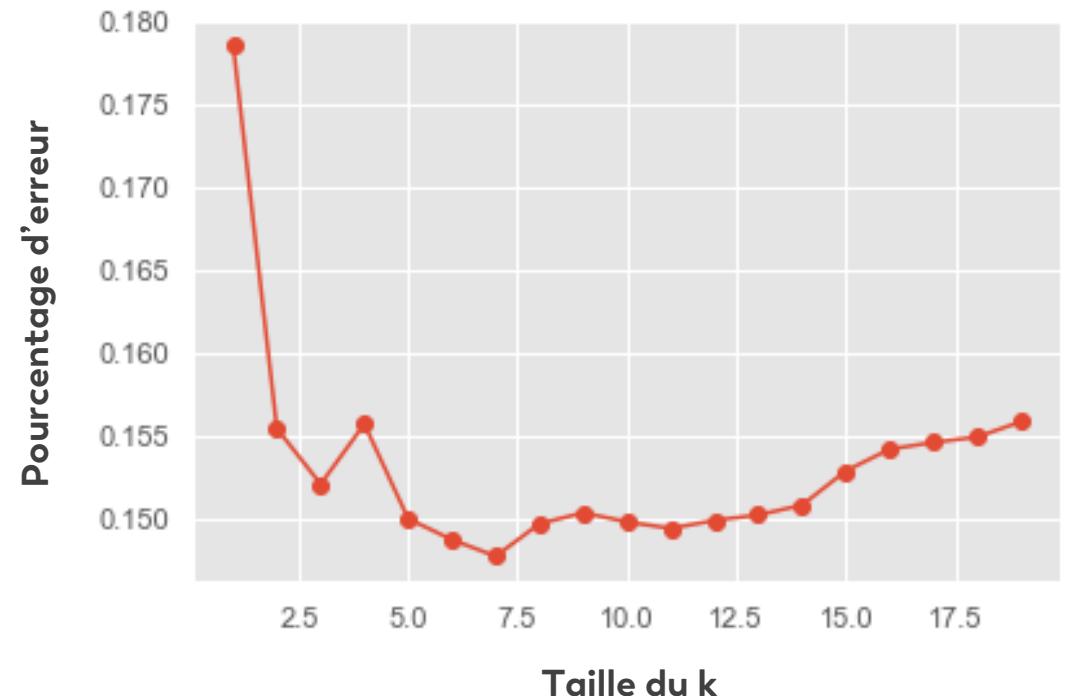
## Puis KNNImputer

Sur toutes les variables numériques

```
imputer = KNNImputer(missing_values=np.nan, n_neighbors=7, weights='distance')
X = df_KNN_2[KNN_features]
df_KNN_3 = pd.DataFrame(imputer.fit_transform(X), index = df_KNN_2.index, columns = X.columns)
```

## Résultats similaires

Pour les variables sugars et nutriscore



## *2. Opérations de nettoyage*

### *Les différentes étapes*

3

#### **Imputation de données**

##### **Remplacer par 0**

Pour les colonnes vides à au moins 97%

##### **Dépassement du seuil de calories possible pour 100g**

Remplacé par NaN puis imputé par le calcul

##### **Calcul du nutriscore**

Pas concluant donc non retenu

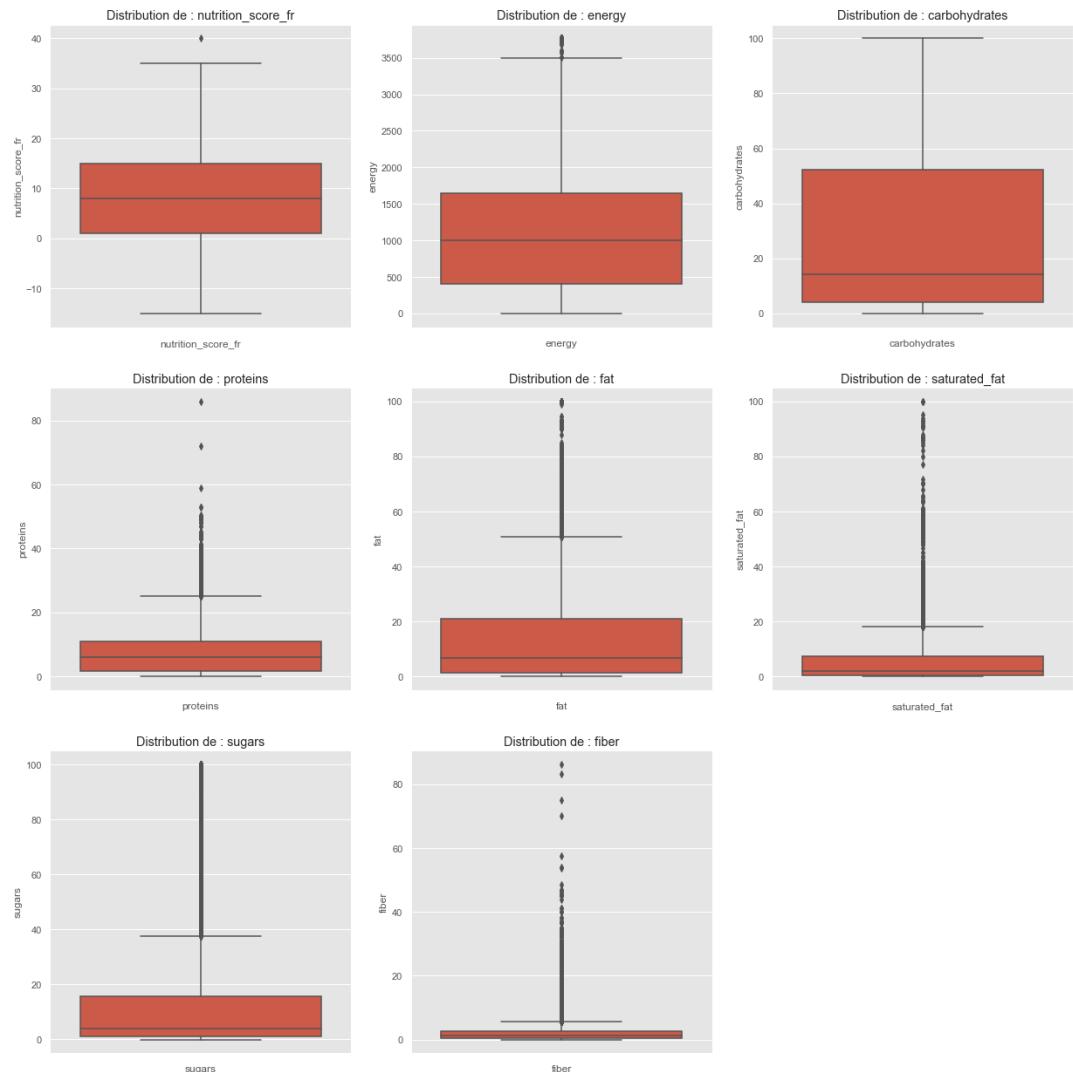
##### **KNN imputer sur les variables pertinentes**

K=3, erreur=13.9%

## 2. Opérations de nettoyage

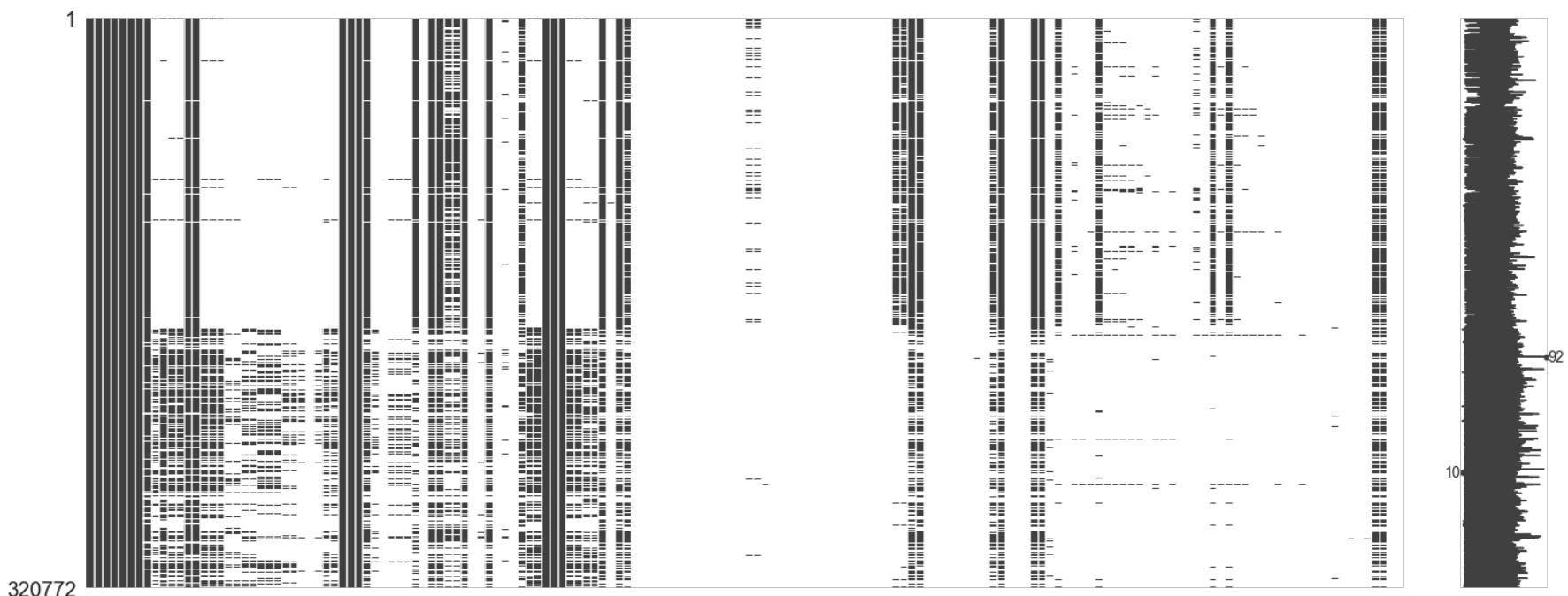
### Synthèse après nettoyage

Caractéristiques	Valeurs
0	Nombre de lignes 39593
1	Nombre de colonnes 14
2	Nombre de variables catégorielles 4
3	Nombre de variables numériques 9
4	Pourcentage de données manquantes 0
5	Nombre de doublons 0



## *2. Opérations de nettoyage*

### *Synthèse après nettoyage*

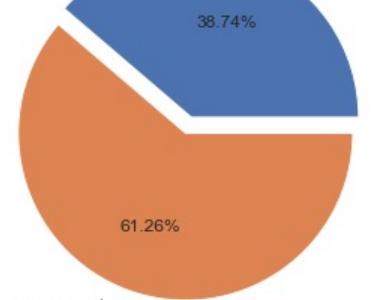


## 2. Opérations de nettoyage

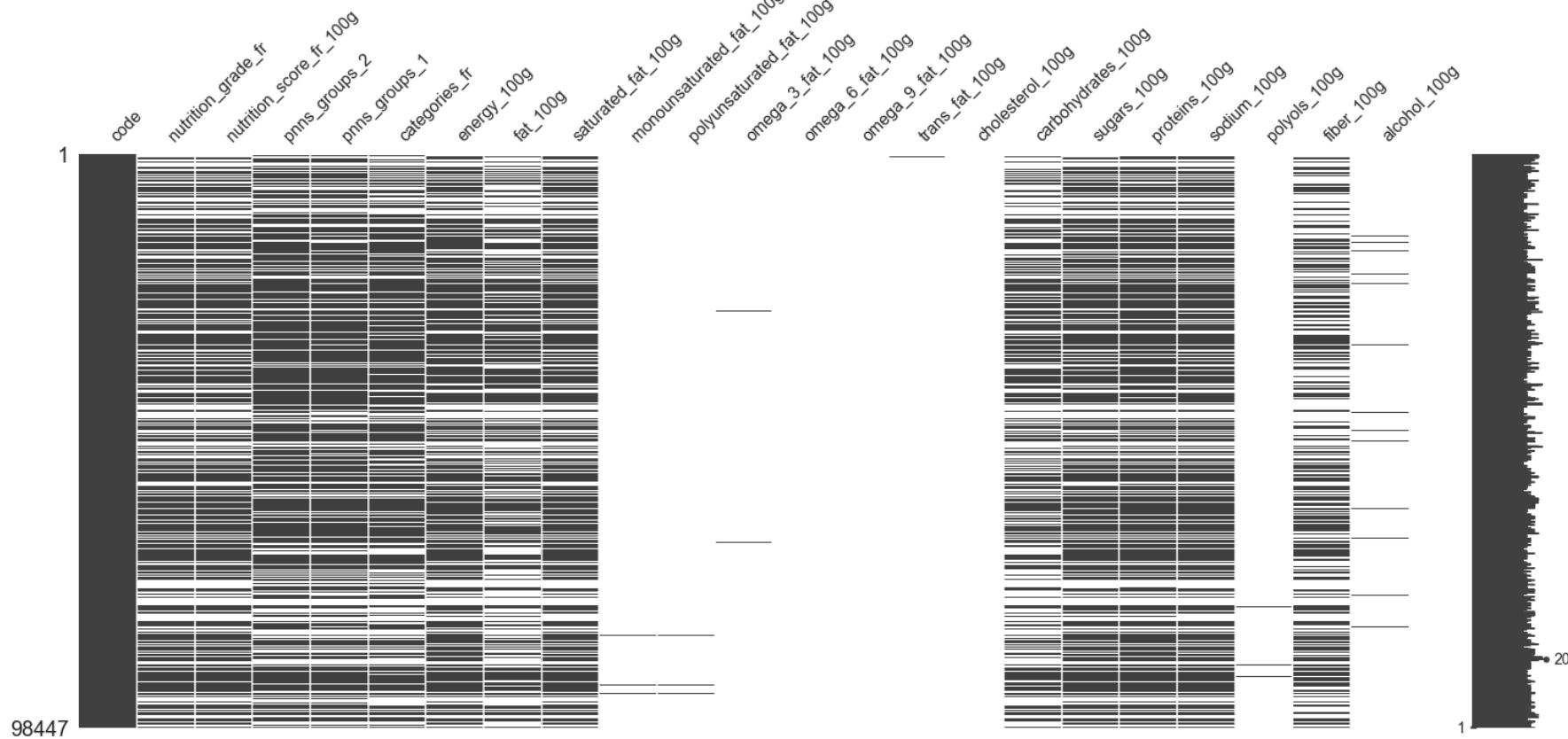
### Synthèse après nettoyage

Taux de remplissage

Taux de remplissage  
Taux de valeurs manquantes

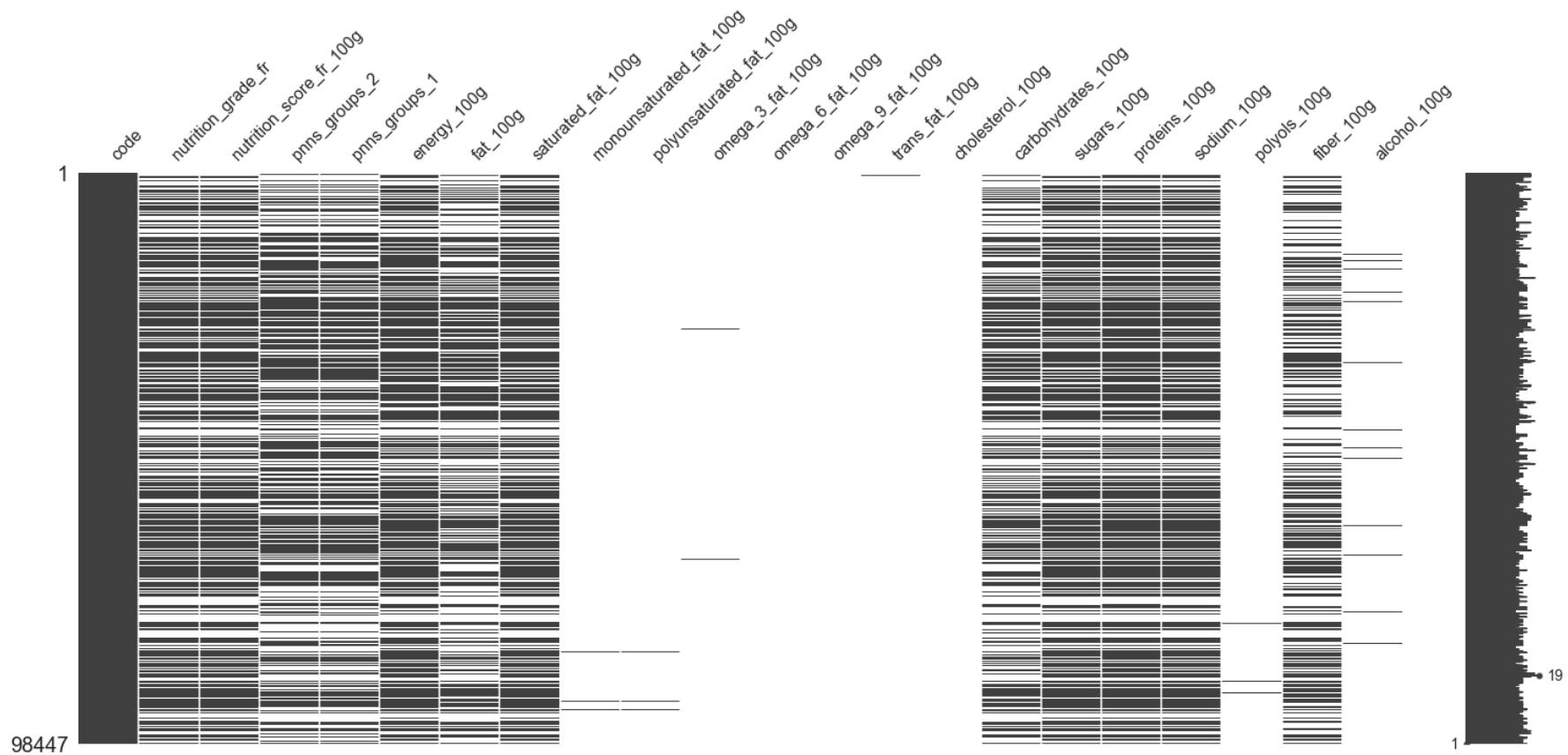


Taux de valeurs manquantes



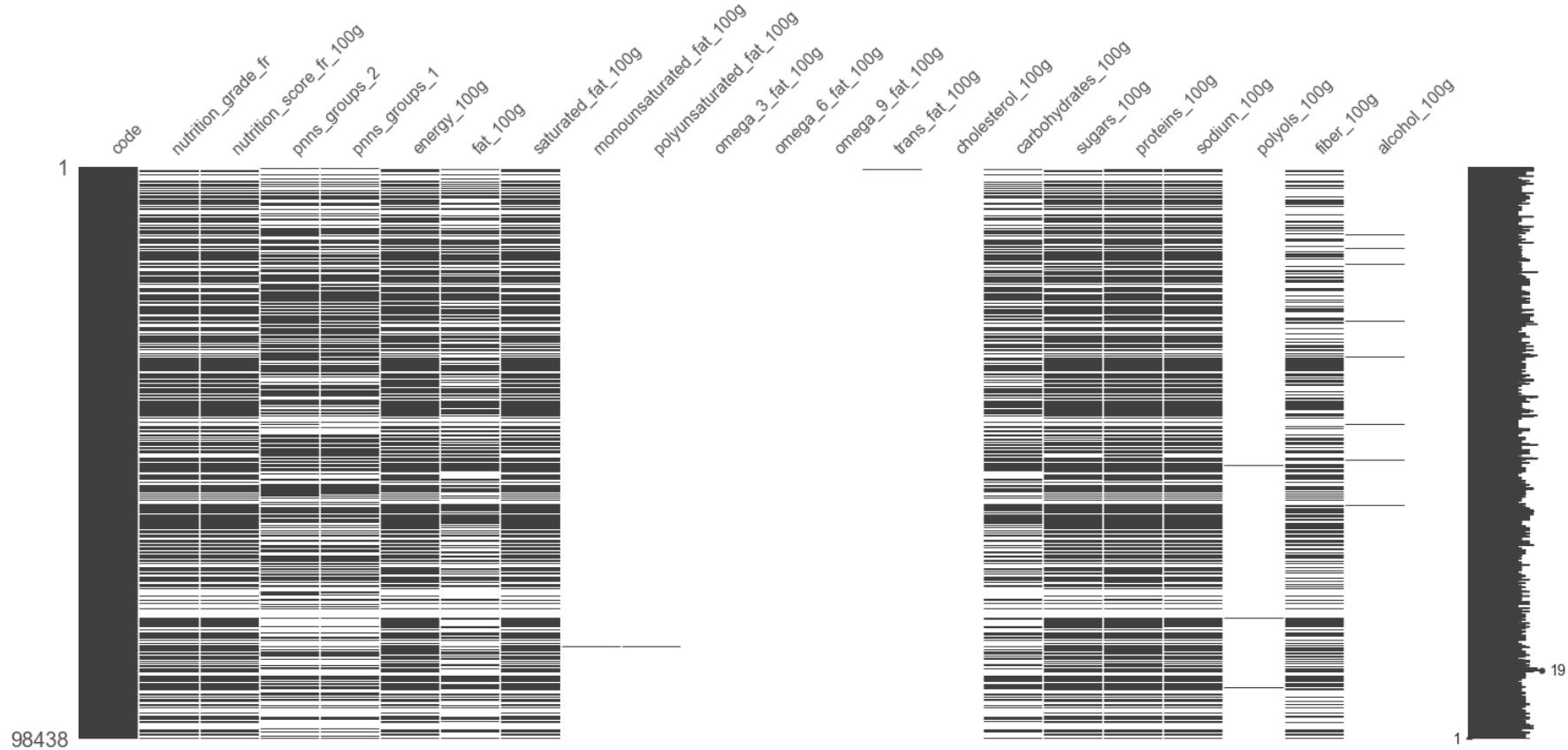
## 2. Opérations de nettoyage

### Synthèse après nettoyage



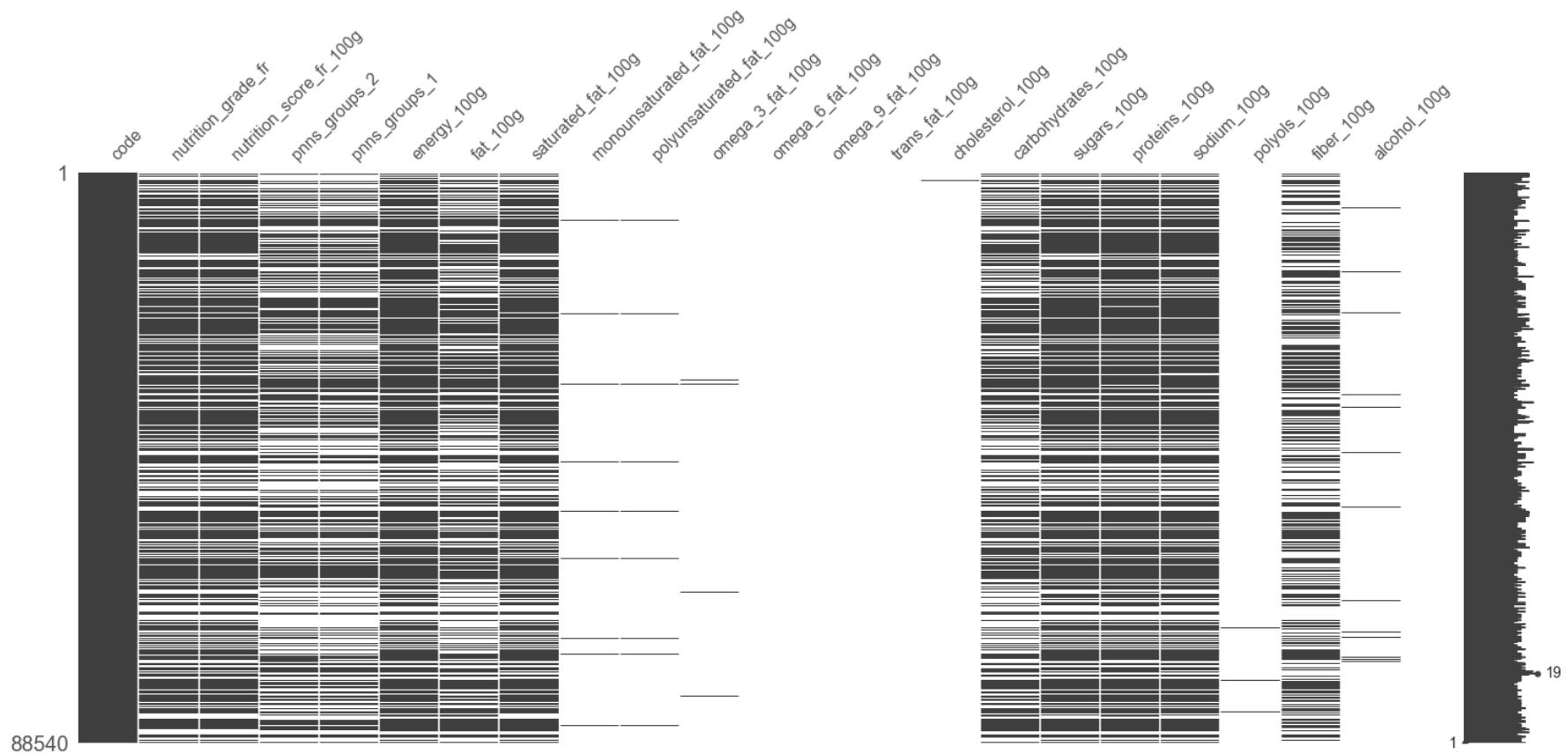
## 2. Opérations de nettoyage

### Synthèse après nettoyage



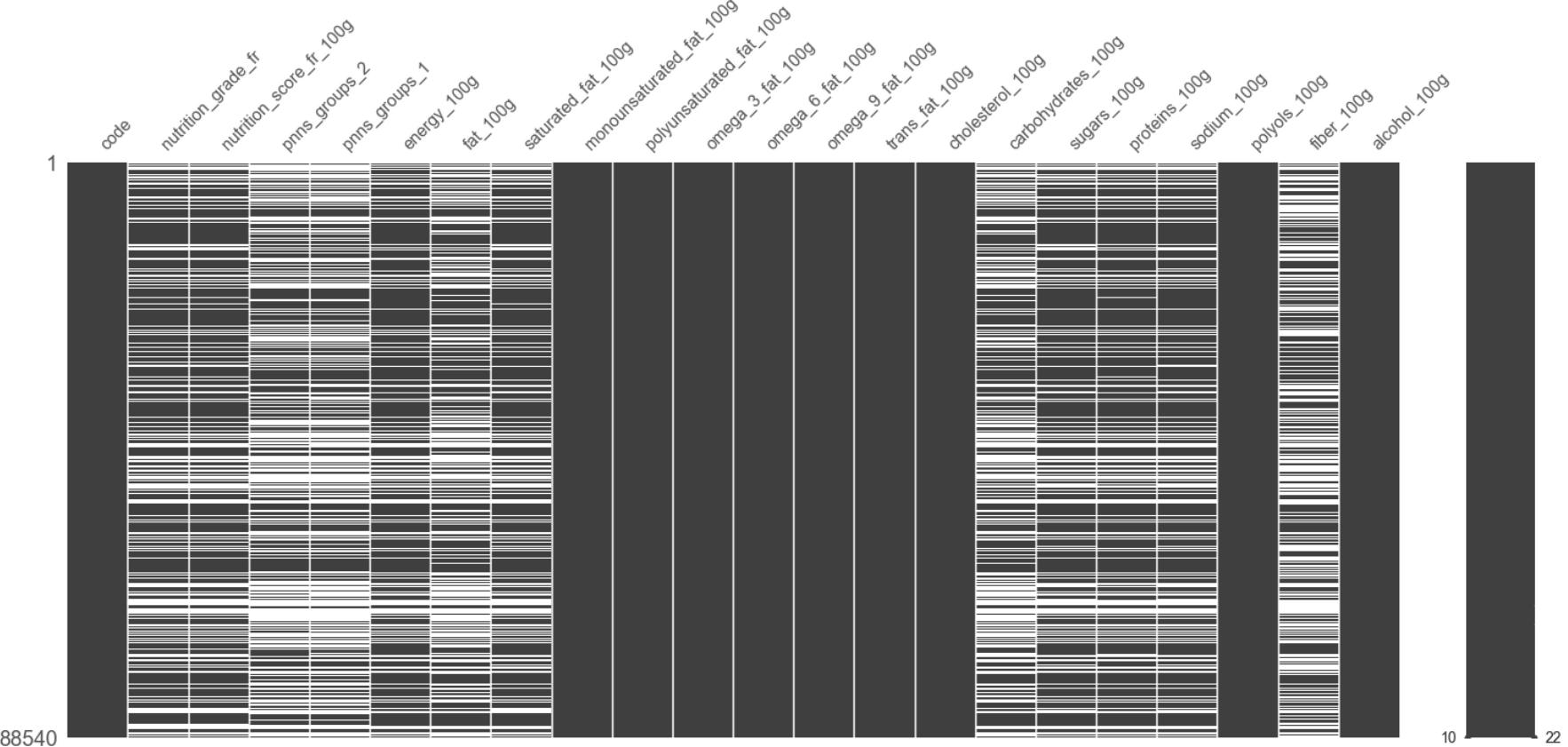
## 2. Opérations de nettoyage

### Synthèse après nettoyage



## 2. Opérations de nettoyage

### Synthèse après nettoyage



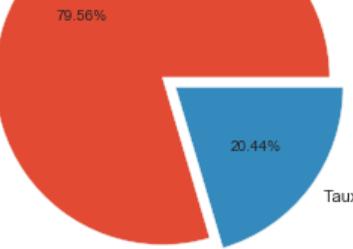
Taux de remplissage

Taux de remplissage

79.56%

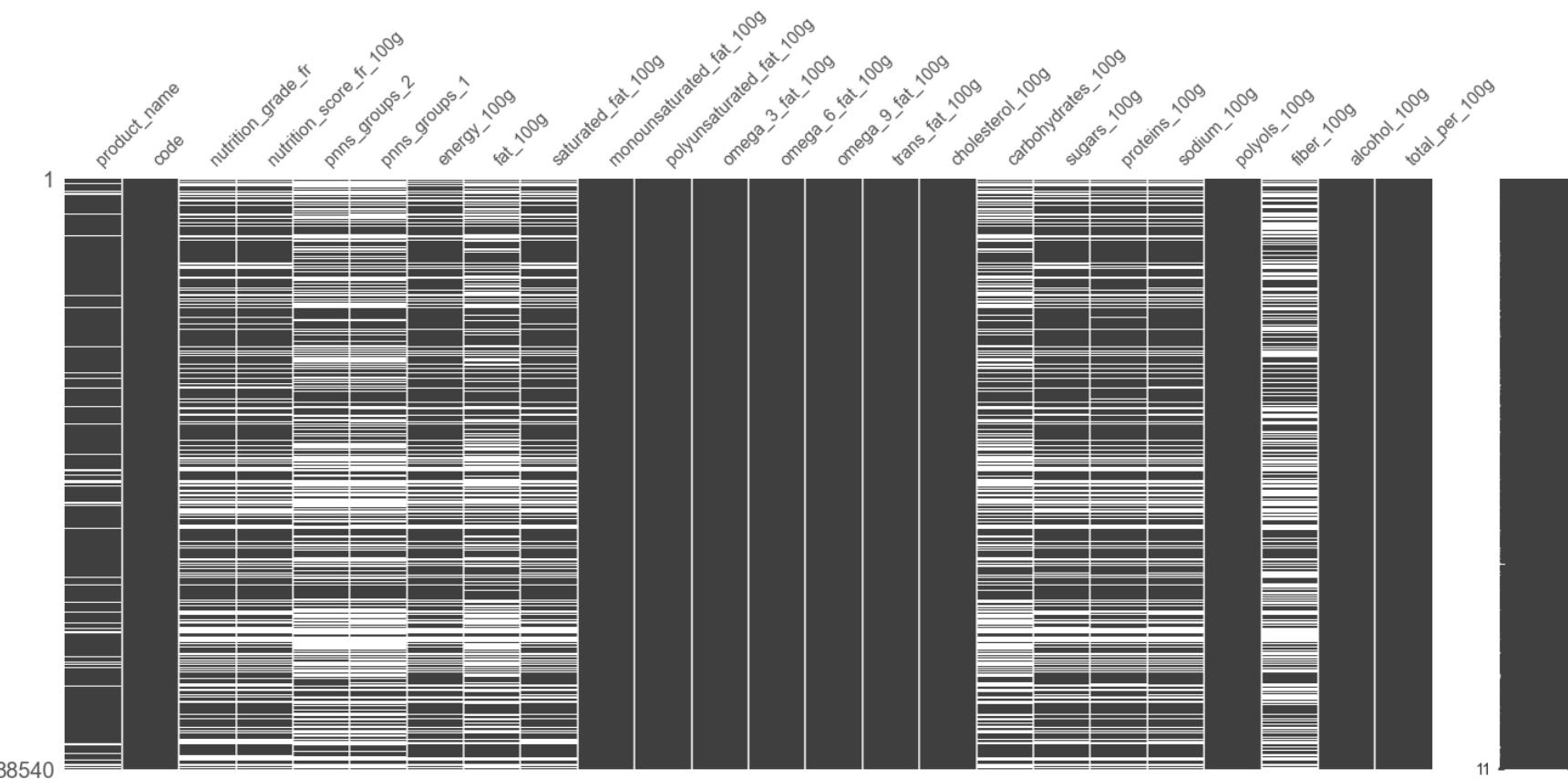
20.44%

Taux de valeurs manquantes



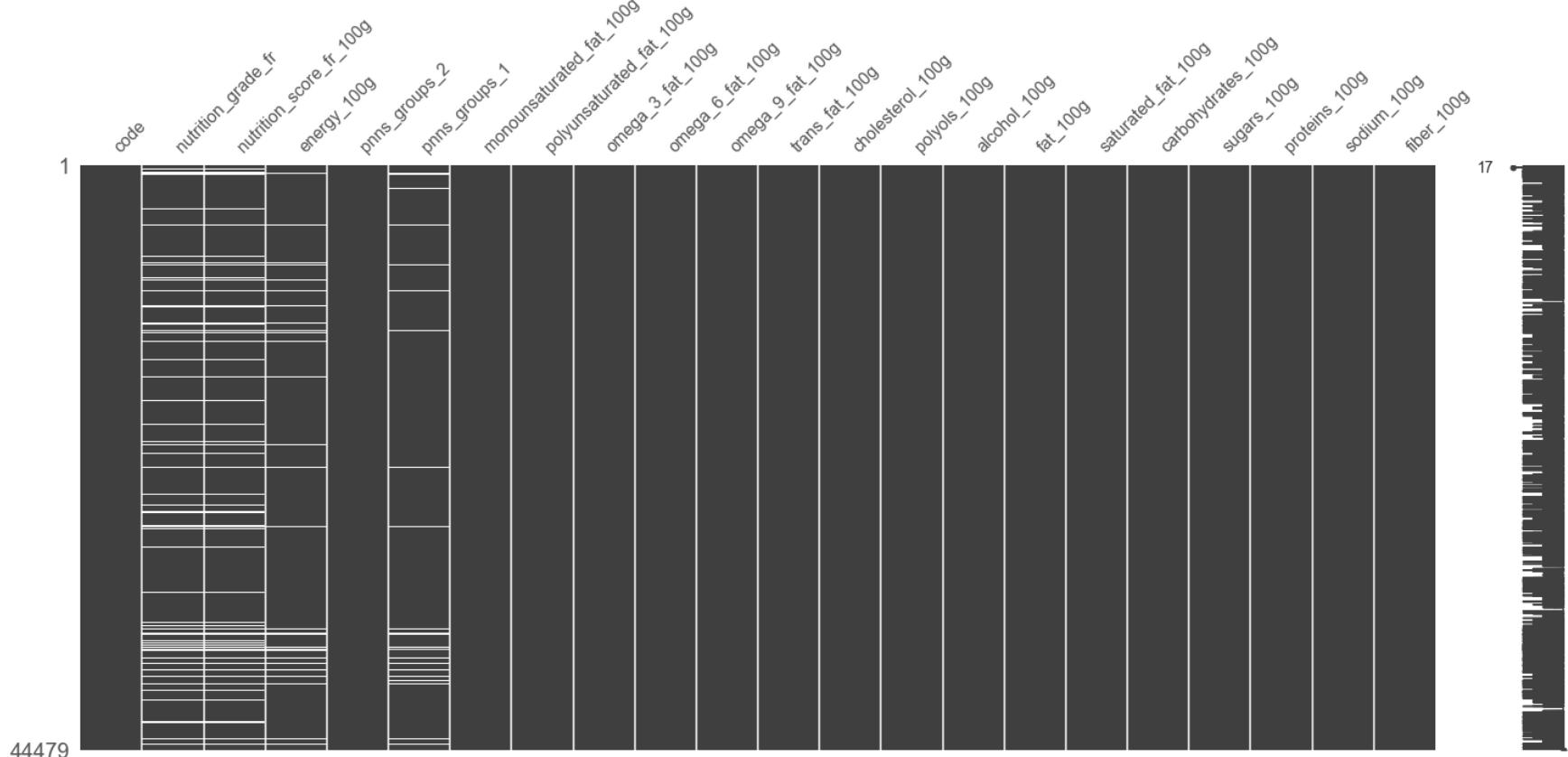
## 2. Opérations de nettoyage

### Synthèse après nettoyage



## 2. Opérations de nettoyage

### Synthèse après nettoyage



Taux de remplissage

Taux de remplissage  
Taux de valeurs manquantes

Taux de remplissage

98.63%

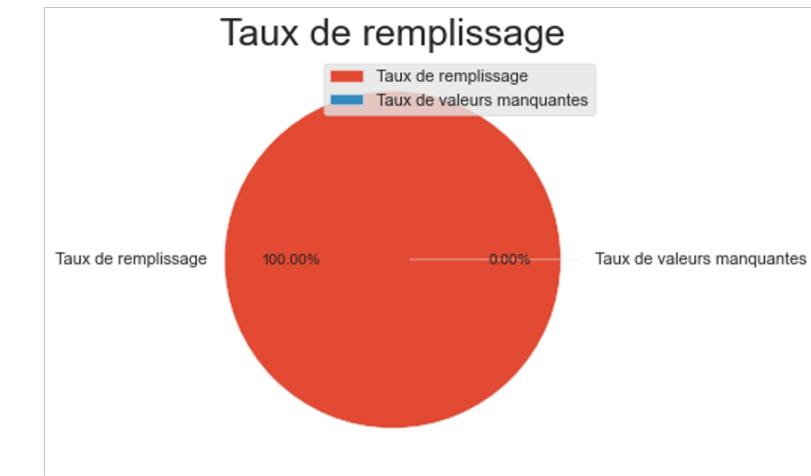
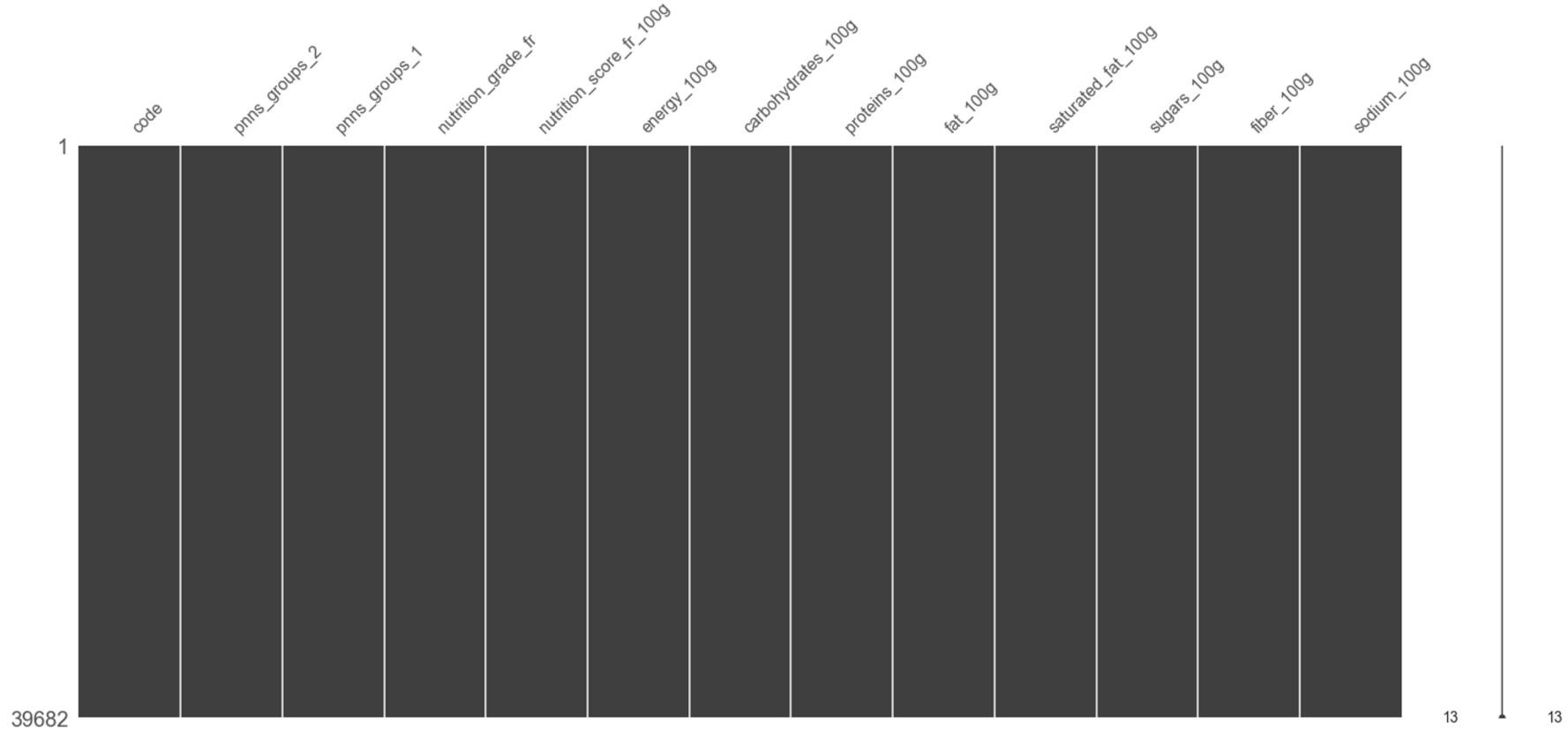
1.37% Taux de valeurs manquantes

1

22

## 2. Opérations de nettoyage

### Synthèse après nettoyage

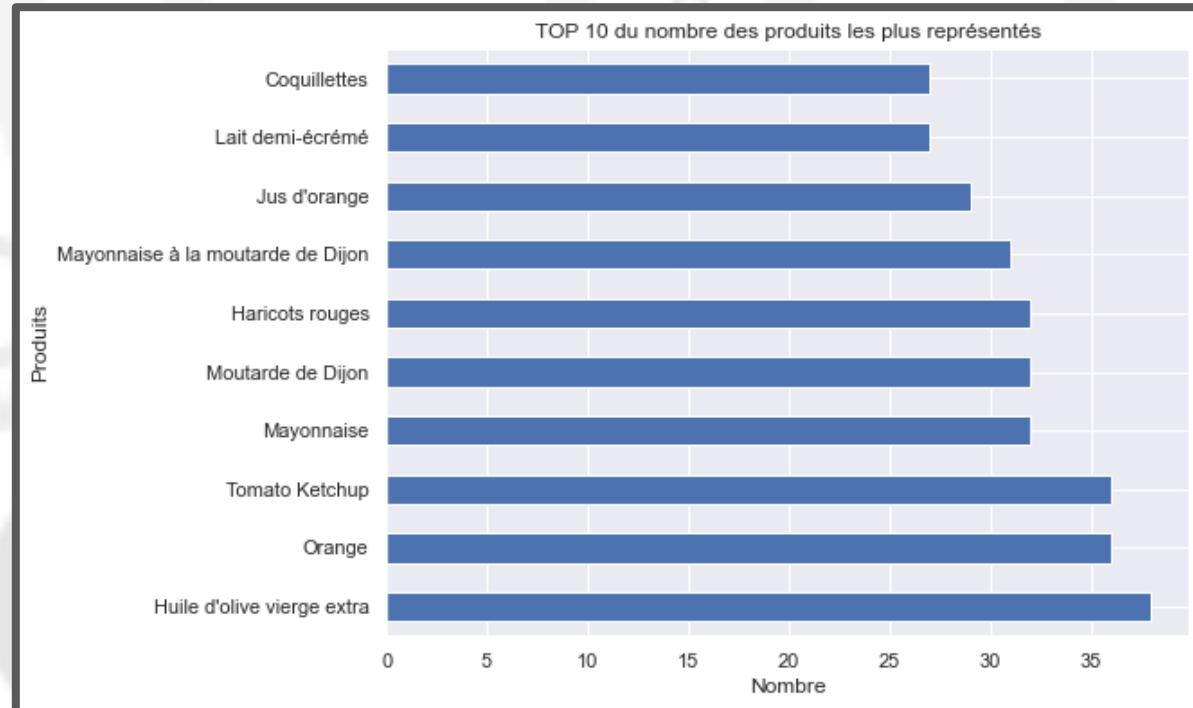


### *3. Exploration de données*



### 3. Exploration de données

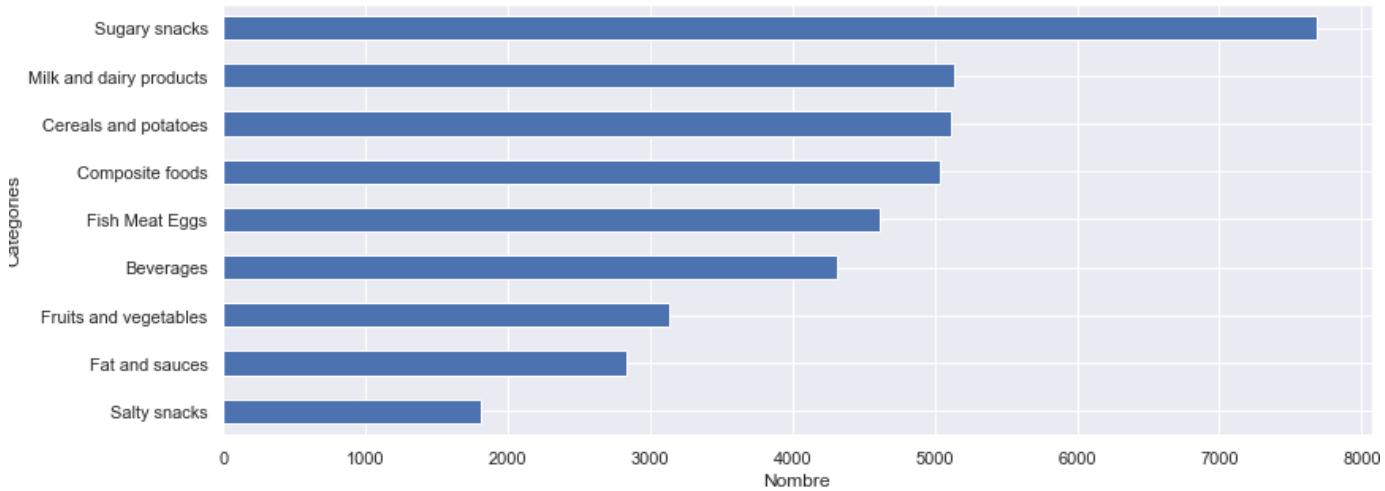
#### Univarié : produits



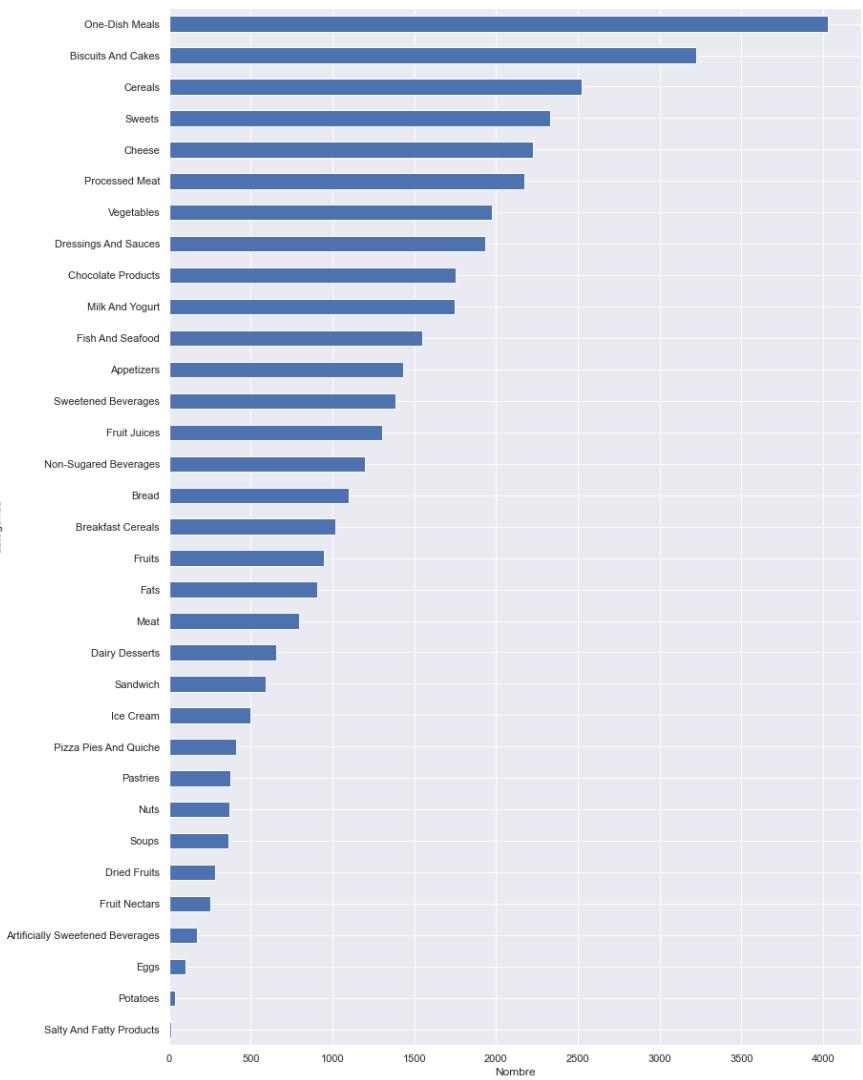
# *3. Exploration de données*

## *Univarié : groupes*

**PNNS Group 1**



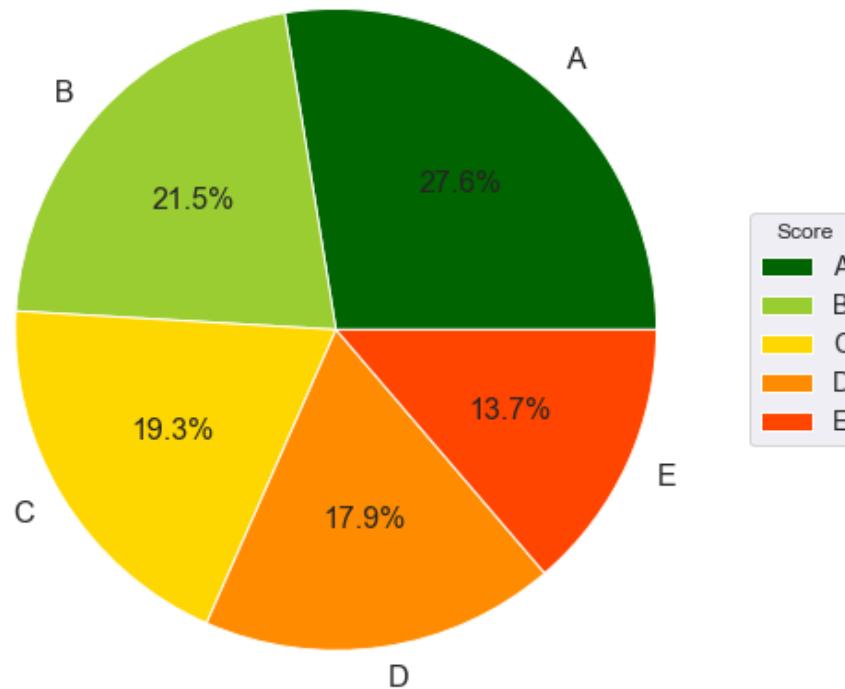
**PNNS Group 2**



### *3. Exploration de données*

#### ***Univarié : nutriscore***

Répartition des Nutriscores



# *3. Exploration de données*

# ***Univarié: variables numériques***

## Interprétation des statistiques :

## Médiane vs moyenne :

## Relativement équilibré

### **Écart interquartile (Q3-Q1) :**

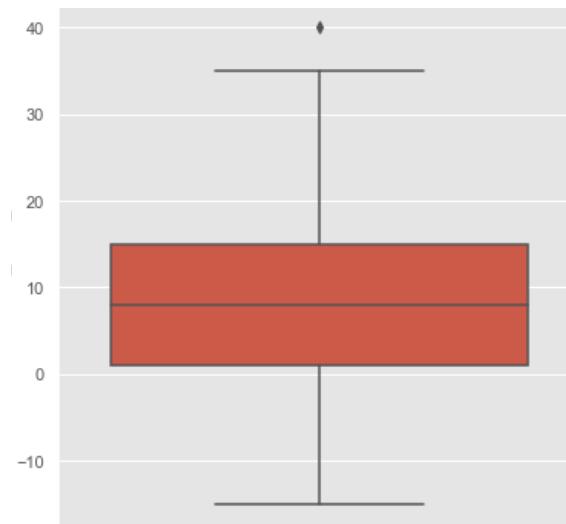
75% des données dans 25% du range

# Nutriscore

Q2	Q3	Q4
1	8	15

**Minimum** | **Maximum**

Ecart-type	Moyenne
9.1	8.4



## *3. Exploration de données*

### **Univarié: variables numériques**

**Interprétation des statistiques :**

**Médiane vs moyenne :**

Déséquilibre léger

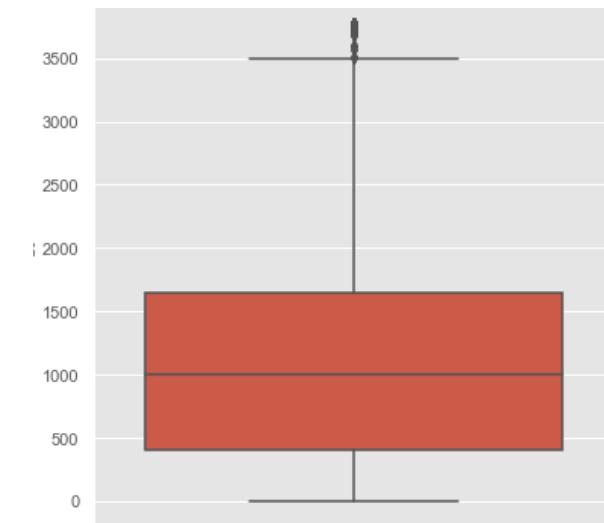
**Écart interquartile (Q3-Q1) :**

75% des données dans 33% du range

**Energie**

Q2      Q3      Q4  
402      1004      1644

Minimum      Maximum  
0      3767.4  
  
Ecart-type      Moyenne  
782      1095



## *3. Exploration de données*

### **Univarié: variables numériques**

**Interprétation des statistiques :**

**Médiane vs moyenne :**

Déséquilibre net sans outliers

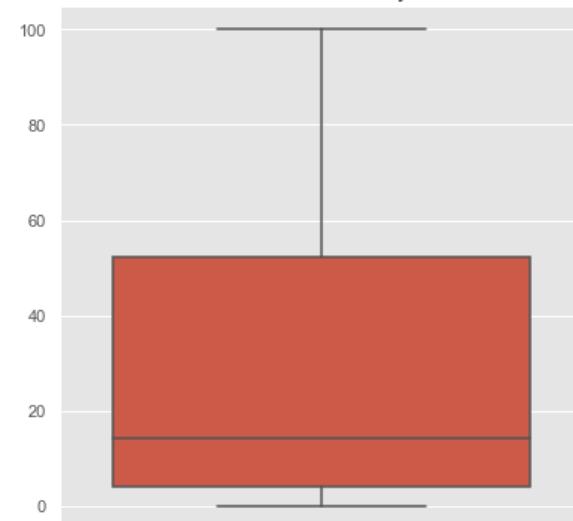
**Écart interquartile (Q3-Q1) :**

75% des données dans 48% du range

#### **Glucides**

Q2      Q3      Q4  
4.2      14.3      52.3

Minimum      Maximum  
0      100  
  
Ecart-type      Moyenne  
27      27.4



## *3. Exploration de données*

### **Univarié: variables numériques**

**Interprétation des statistiques :**

**Médiane vs moyenne :**

Relativement déséquilibré, nombre d'outliers +/- grand

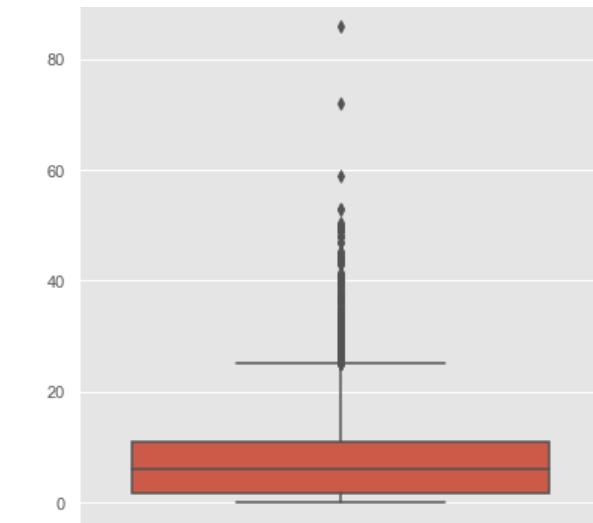
**Écart interquartile (Q3-Q1) :**

75% des données dans 10% du range

#### **Protéines**

Q2      Q3      Q4  
1.6      6      11

Minimum      Maximum  
0      86  
  
Ecart-type      Moyenne  
7.3      7.6



## *3. Exploration de données*

### **Univarié: variables numériques**

**Interprétation des statistiques :**

**Médiane vs moyenne :**

Déséquilibre net, beaucoup d'outliers

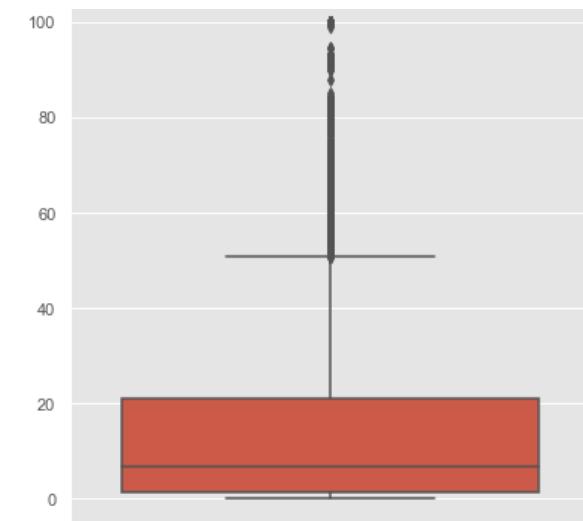
**Écart interquartile (Q3-Q1) :**

75% des données dans 20% du range

#### **Lipides**

Q2      Q3      Q4  
1.2      6.7      21

Minimum      Maximum  
0              100  
  
Ecart-type      Moyenne  
16.9              13.3



## *3. Exploration de données*

### **Univarié: variables numériques**

**Interprétation des statistiques :**

**Médiane vs moyenne :**

Déséquilibre net vers les valeurs faibles,  
beaucoup d'outliers

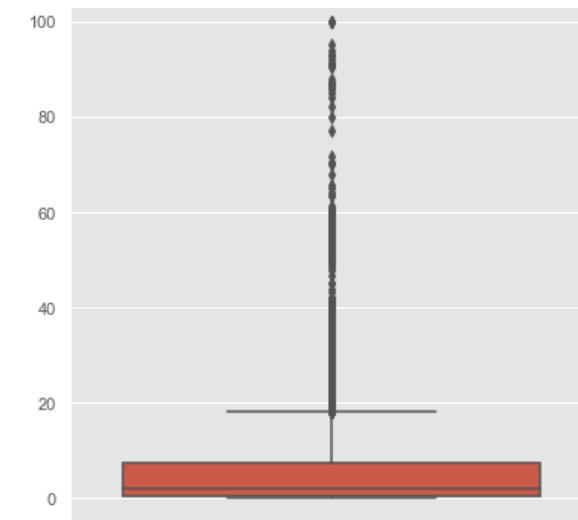
**Écart interquartile (Q3-Q1) :**

75% des données dans 7% du range

#### **Graisse saturée**

Q2      Q3      Q4  
0.3      1.9      7.4

Minimum      Maximum  
0              100  
  
Ecart-type      Moyenne  
8.3              5.3



## *3. Exploration de données*

### **Univarié: variables numériques**

**Interprétation des statistiques :**

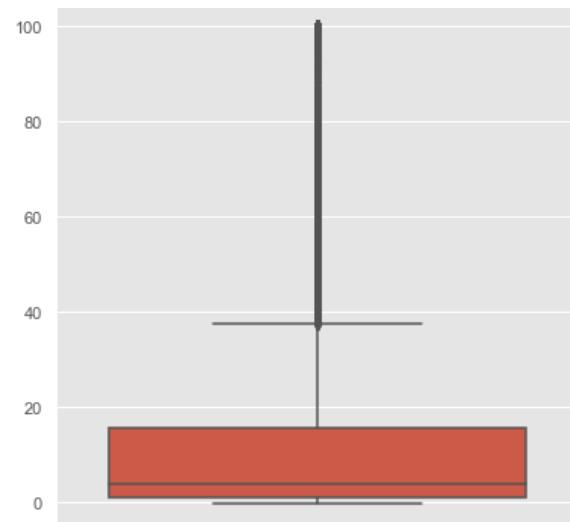
**Médiane vs moyenne :**

Déséquilibre net vers les valeurs faibles,  
beaucoup d'outliers

**Écart interquartile (Q3-Q1) :**

75% des données dans 15% du range

<b>Sucres</b>		
<b>Q2</b> 1	<b>Q3</b> 4	<b>Q4</b> 15.6
<b>Minimum</b> 0		<b>Maximum</b> 100
<b>Ecart-type</b> 18.3		<b>Moyenne</b> 12.7



## *3. Exploration de données*

### **Univarié: variables numériques**

**Interprétation des statistiques :**

**Médiane vs moyenne :**

Relativement équilibré

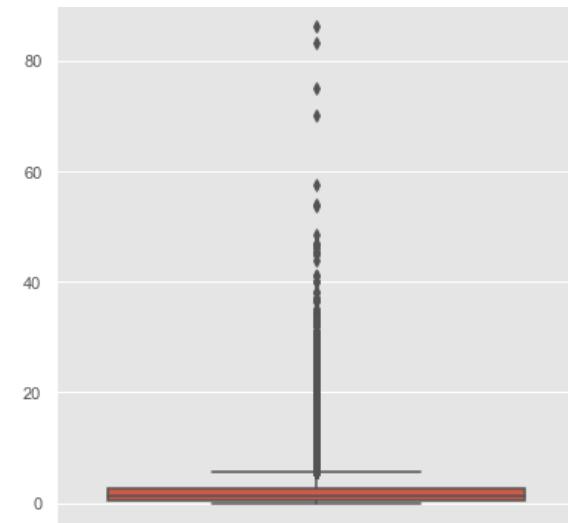
**Écart interquartile (Q3-Q1) :**

75% des données dans 2% du range

**Fibres**

Q2  
0.4 | Q3  
1.2 | Q4  
2.5

Minimum  
0 | Maximum  
86.2  
  
Ecart-type  
3.2 | Moyenne  
2.1



## *3. Exploration de données*

### **Univarié: variables numériques**

**Interprétation des statistiques :**

**Médiane vs moyenne :**

Relativement équilibré

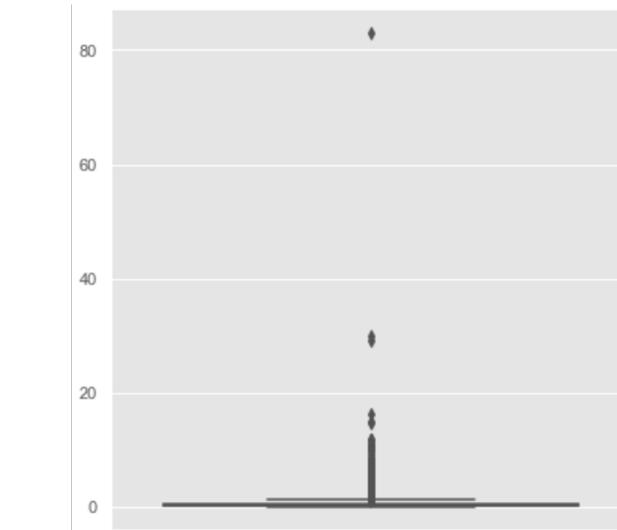
**Écart interquartile (Q3-Q1) :**

75% des données dans 0.5% du range

**Sodium**

Q2      Q3      Q4  
0.03    0.22    0.47

Minimum      Maximum  
0                83  
  
Ecart-type      Moyenne  
0.7              0.34

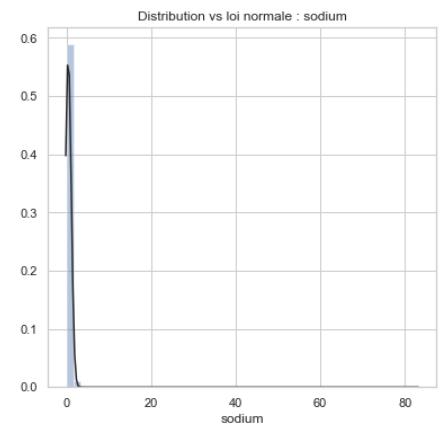
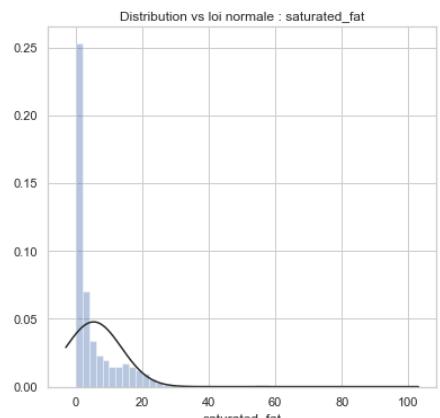
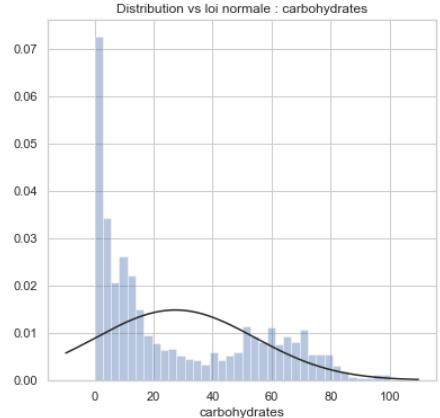
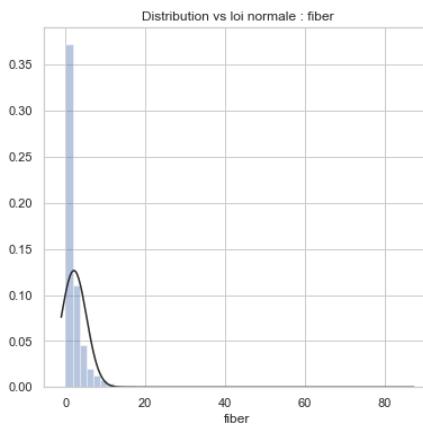
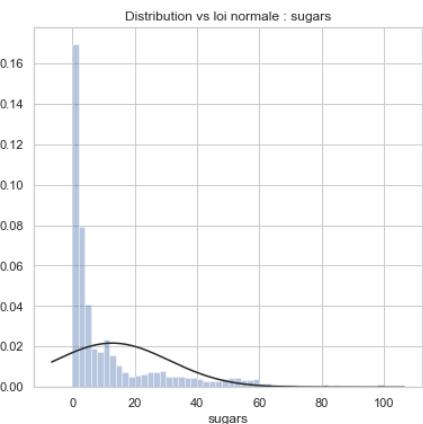
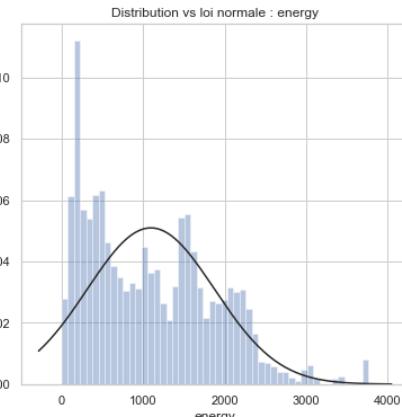
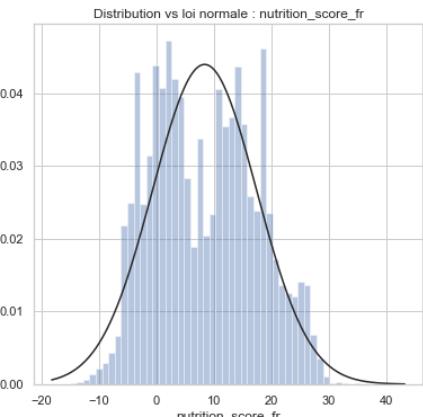


# 3. Exploration de données

## Test de normalité : comparatif visuel

### Affichage pour chaque variable

de sa distribution réelle comparée à une courbe qui délimite théoriquement une répartition gaussienne.



### *3. Exploration de données*

#### ***ANOVA : Kruskal-Wallis***

##### **Nature des données**

- Échantillons indépendants
- Non-normaux
- Variables numérique et catégorielles ordinaires

##### **Interprétation des résultats**

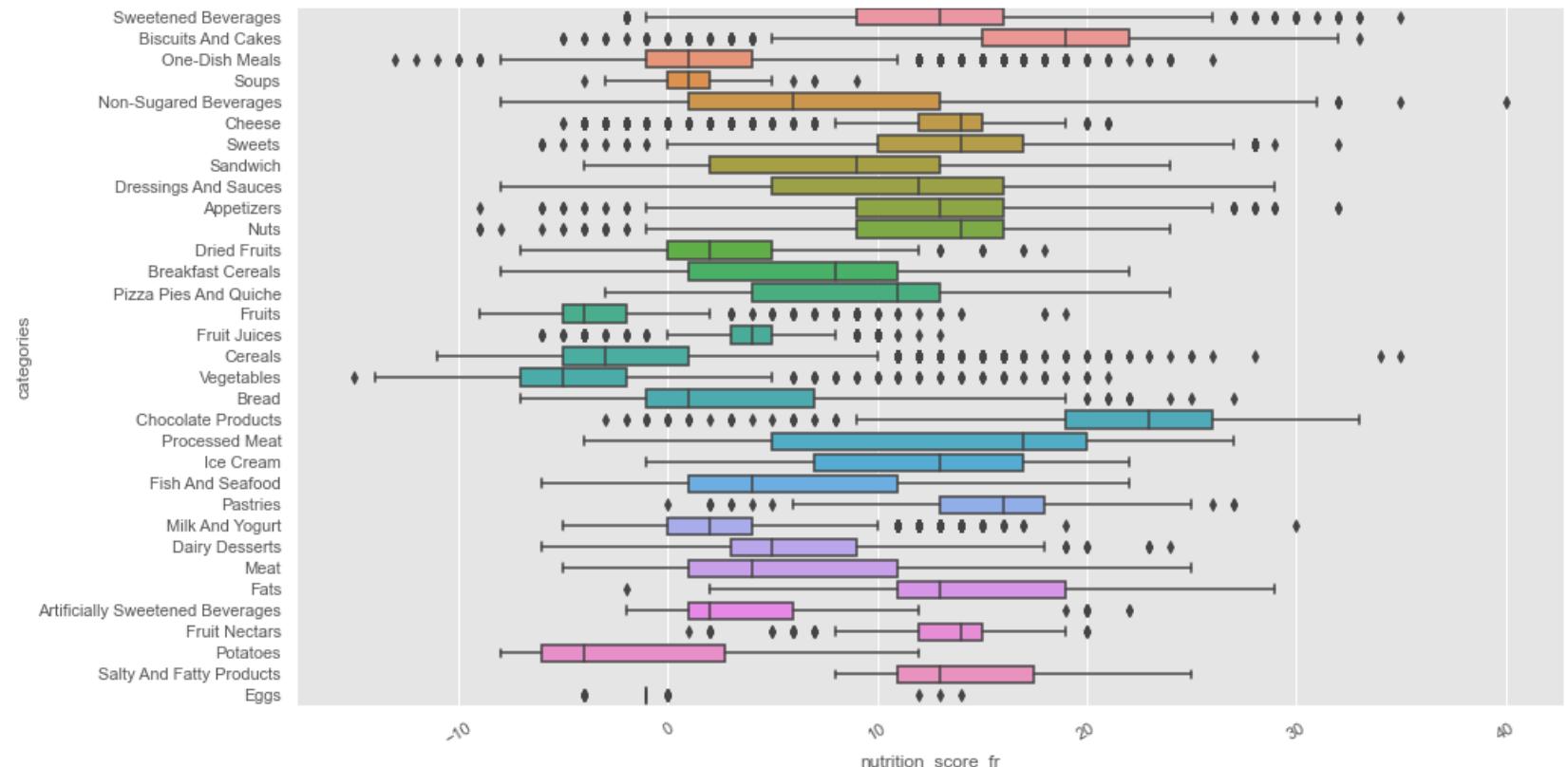
- P-value < 0,05 pour tous les tests
- Moyenne des valeurs centrales est différente
- Nutriscore influe significativement sur chacune des variables numérique.

On procède à ce test car aucune distribution ne suit de loi normale.

```
for all values:  
KruskalResult(statistic=184749.8524840839, pvalue=0.0)  
  
nutrition_score_fr :  
KruskalResult(statistic=5210.527596289104, pvalue=0.0)  
  
energy :  
KruskalResult(statistic=58883.88112667453, pvalue=0.0)  
  
carbohydrates :  
KruskalResult(statistic=19411.3105290711, pvalue=0.0)  
  
proteins :  
KruskalResult(statistic=7072.541516090344, pvalue=0.0)  
  
fat :  
KruskalResult(statistic=4144.76800456862, pvalue=0.0)  
  
saturated_fat :  
KruskalResult(statistic=1697.6646853570312, pvalue=0.0)  
  
sugars :  
KruskalResult(statistic=1187.4798831453822, pvalue=3.2083494072679805e-260)  
  
fiber :  
KruskalResult(statistic=15713.858247874718, pvalue=0.0)  
  
sodium :  
KruskalResult(statistic=56804.12905965278, pvalue=0.0)
```

### *3. Exploration de données*

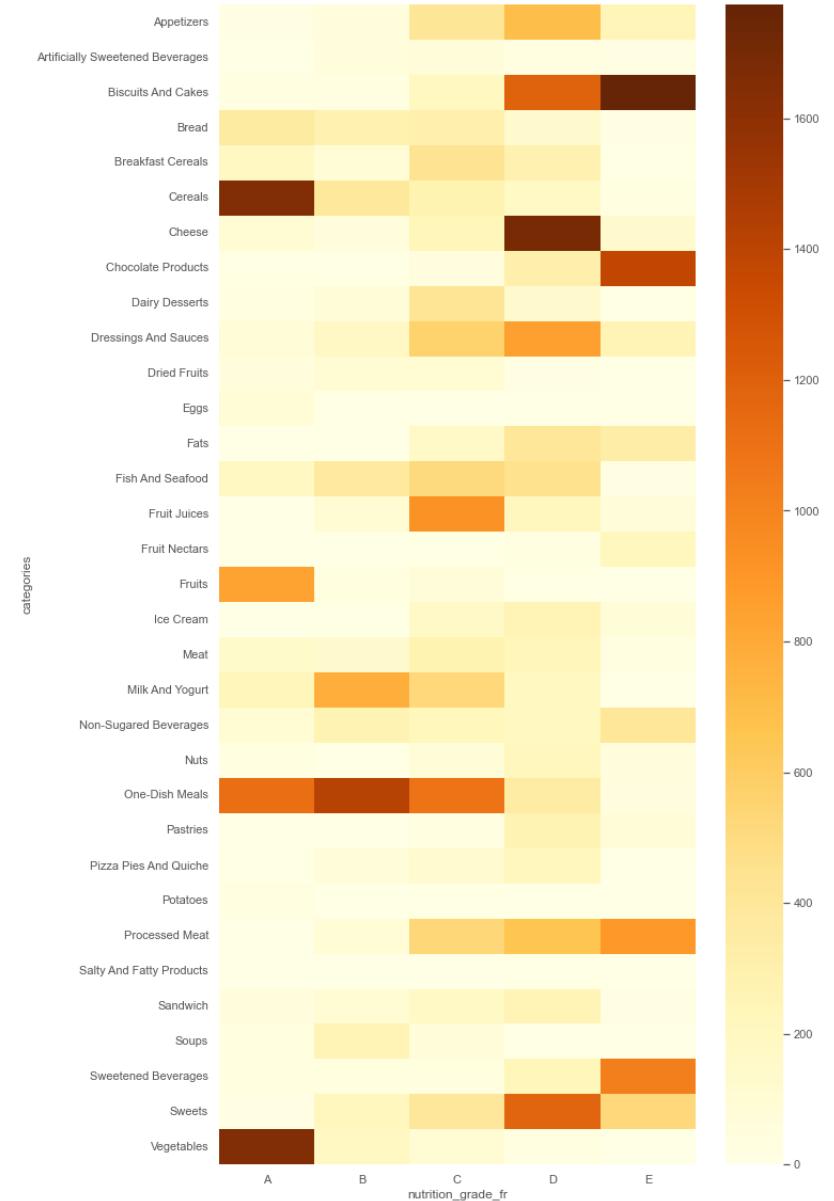
#### *Bivarié : groupes & nutriscore*



### 3. Exploration de données

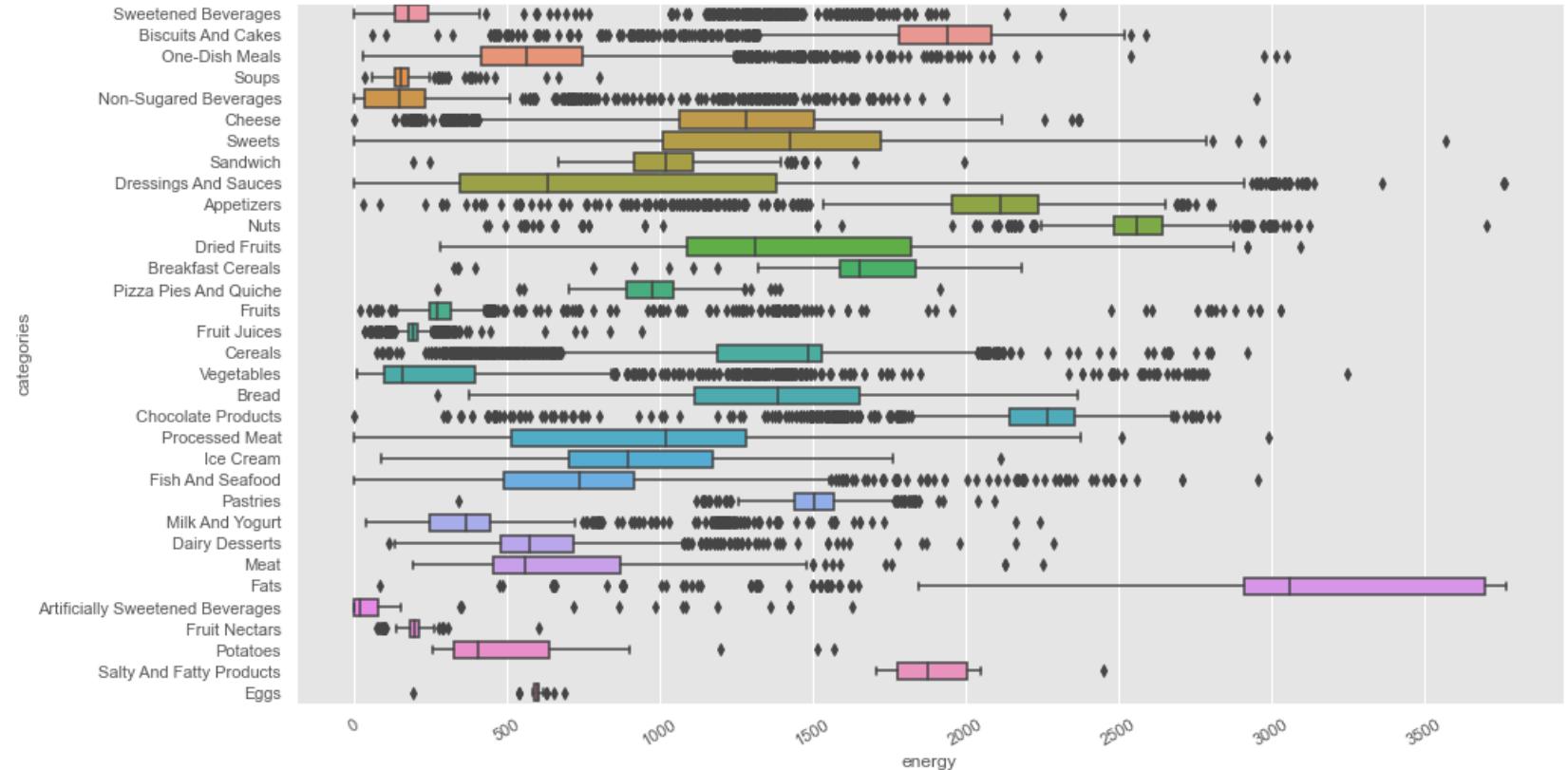
#### Bivarié : groupes & nutriscore

NUTRI-SCORE



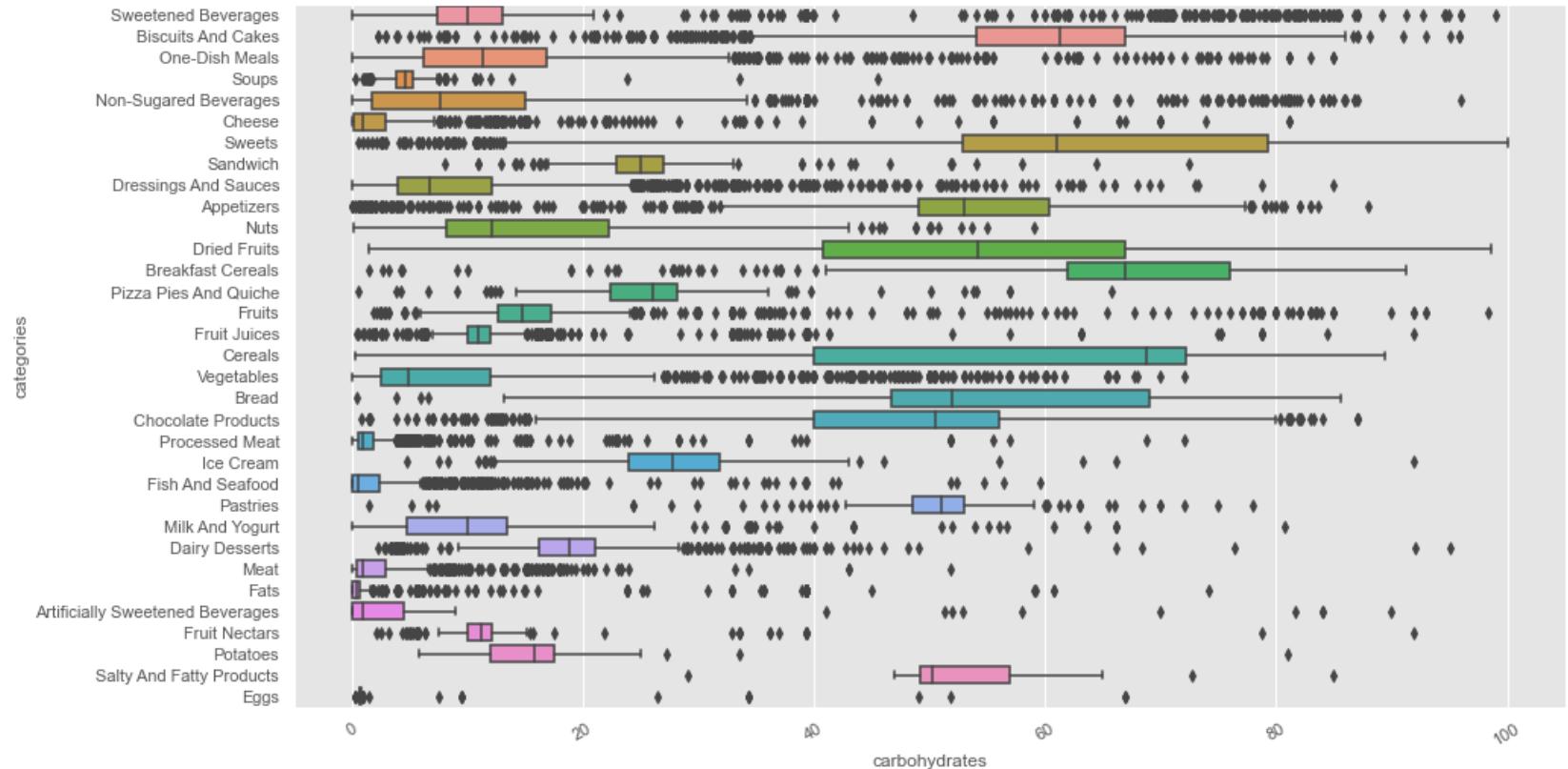
### *3. Exploration de données*

#### *Bivarié : groupes & energie*



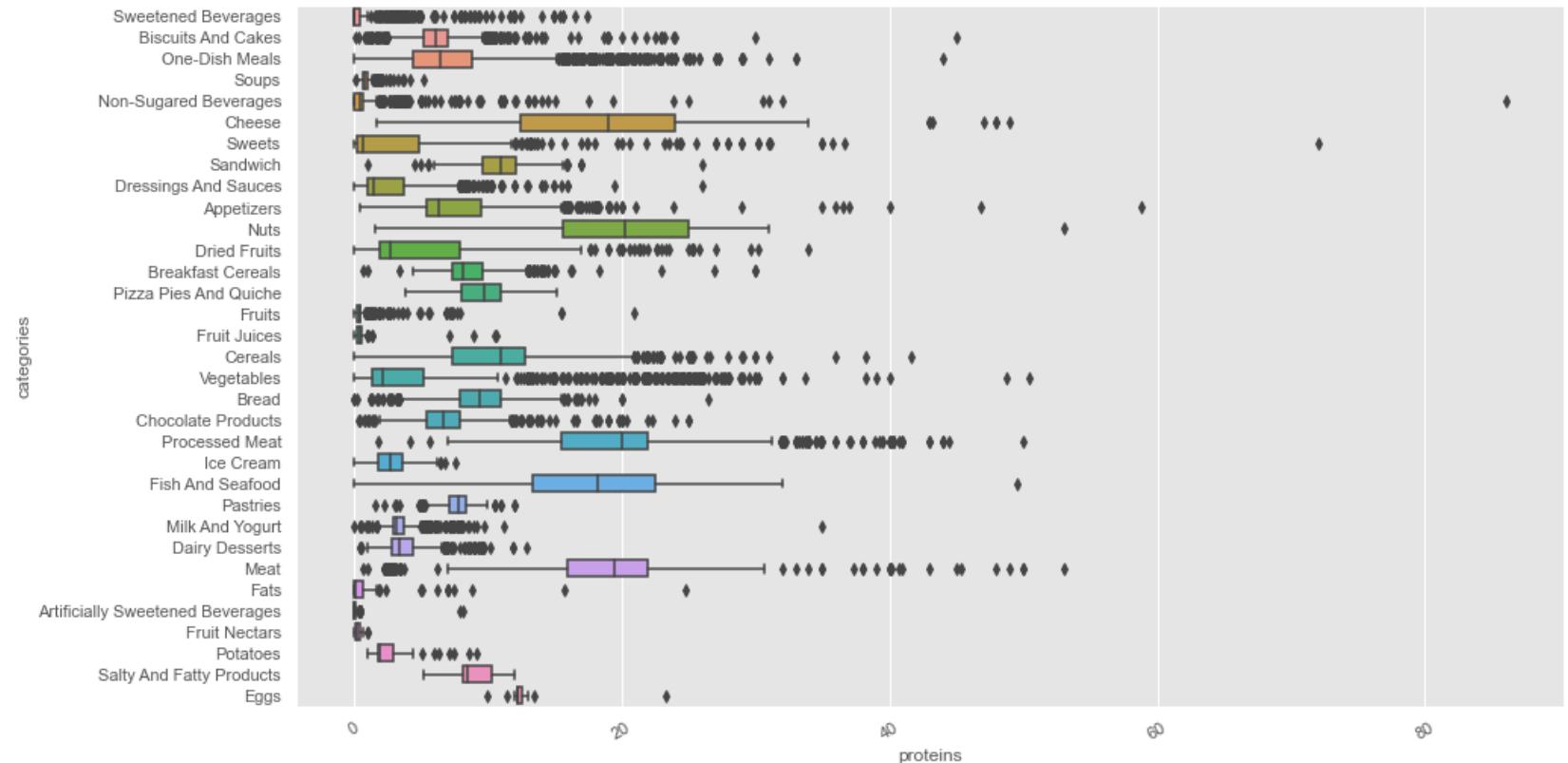
### *3. Exploration de données*

#### *Bivarié : groupes & glucides*



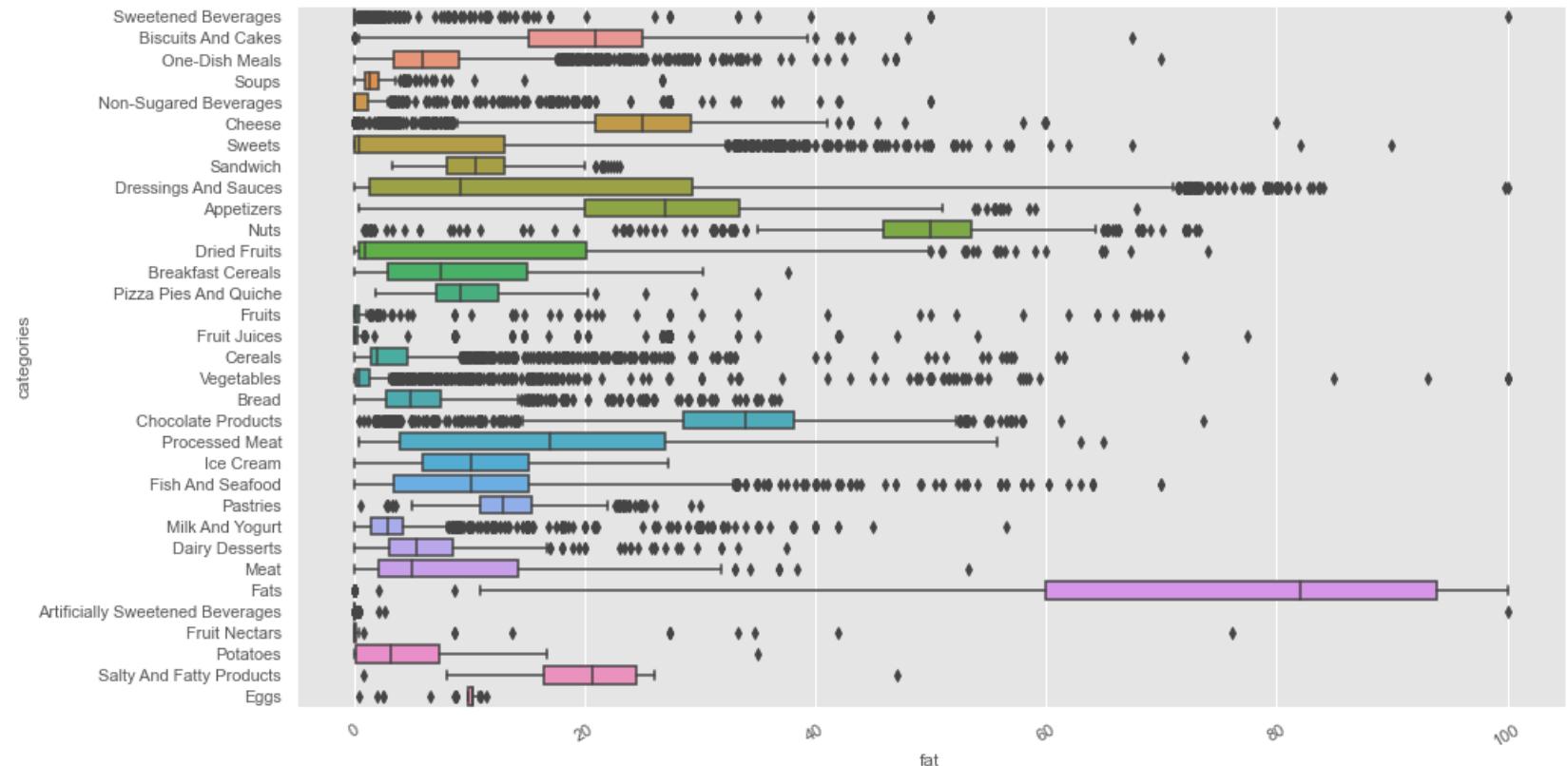
### *3. Exploration de données*

#### *Bivarié : groupes & protéines*



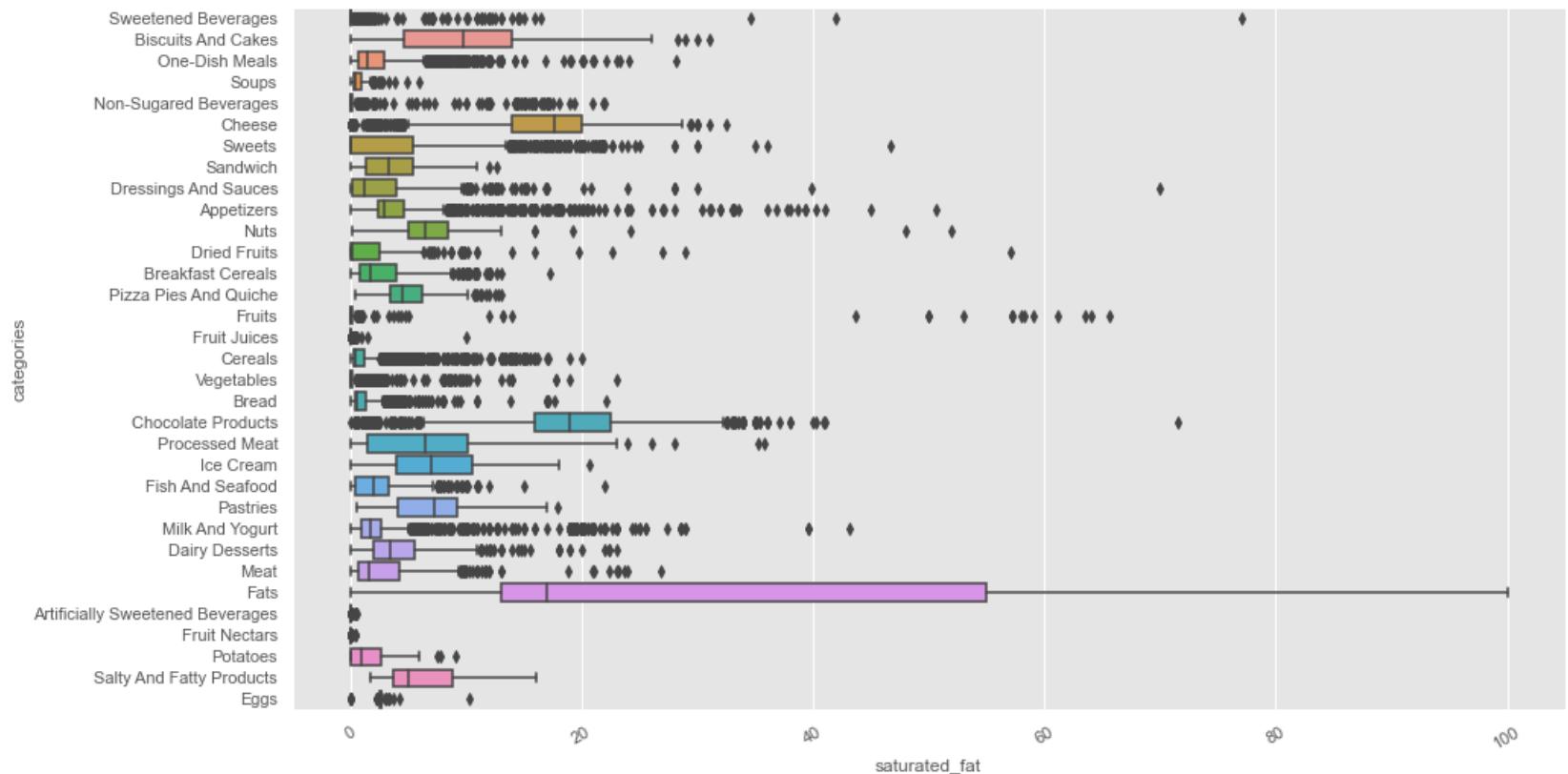
### *3. Exploration de données*

#### *Bivarié : groupes & lipides*



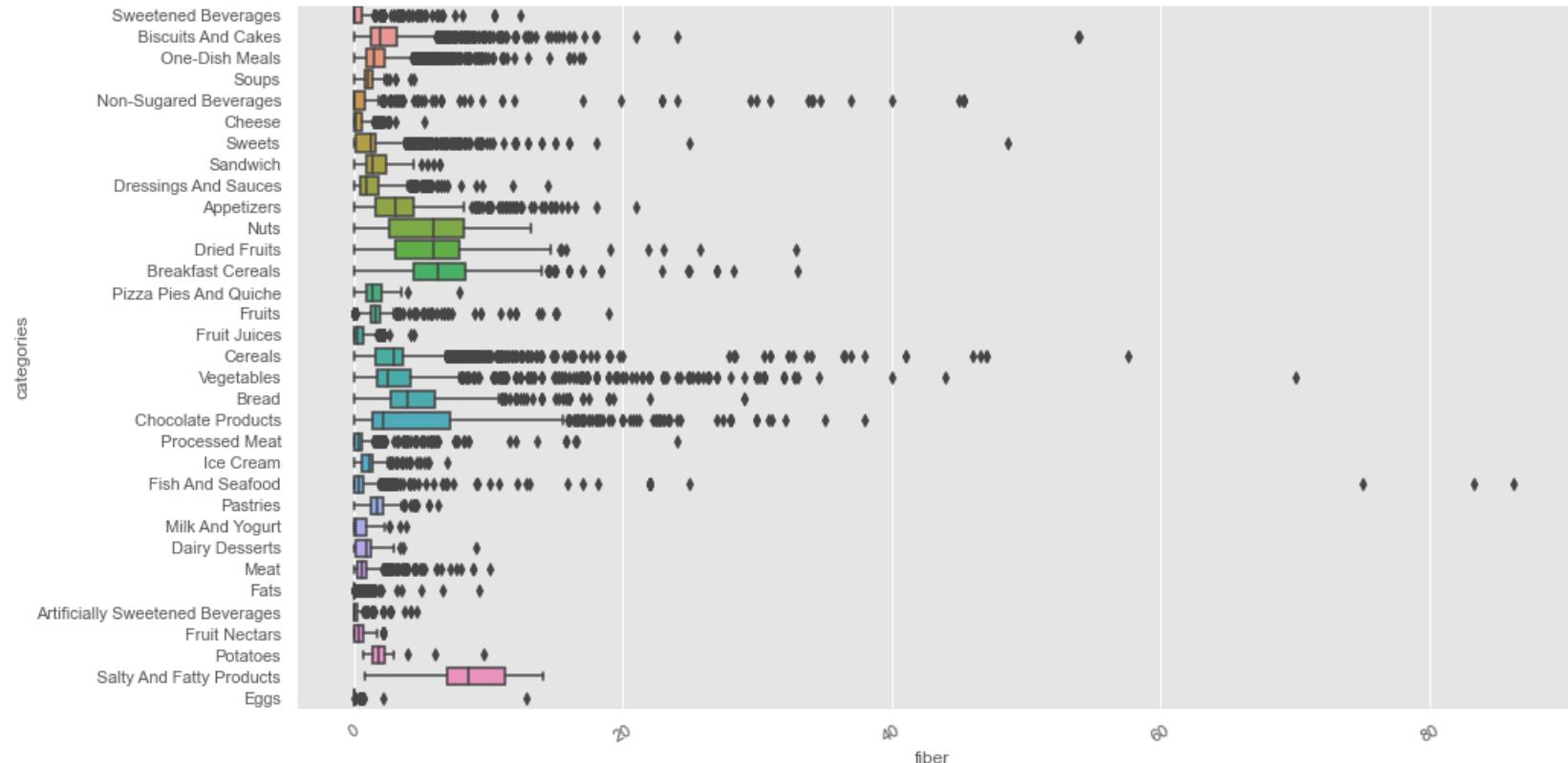
### *3. Exploration de données*

#### *Bivarié : groupes & graisses saturées*



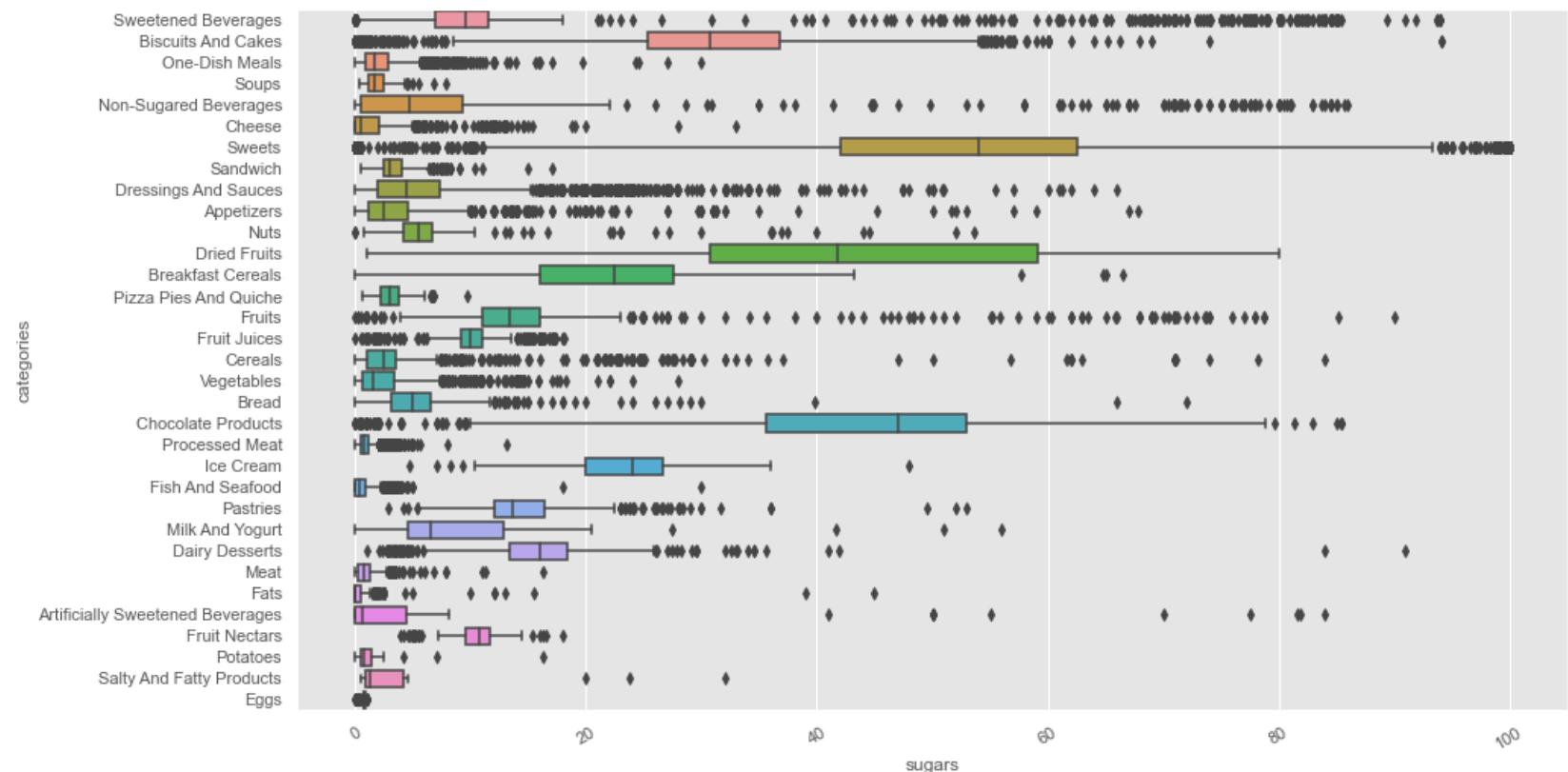
### *3. Exploration de données*

#### *Bivarié : groupes & fibres*



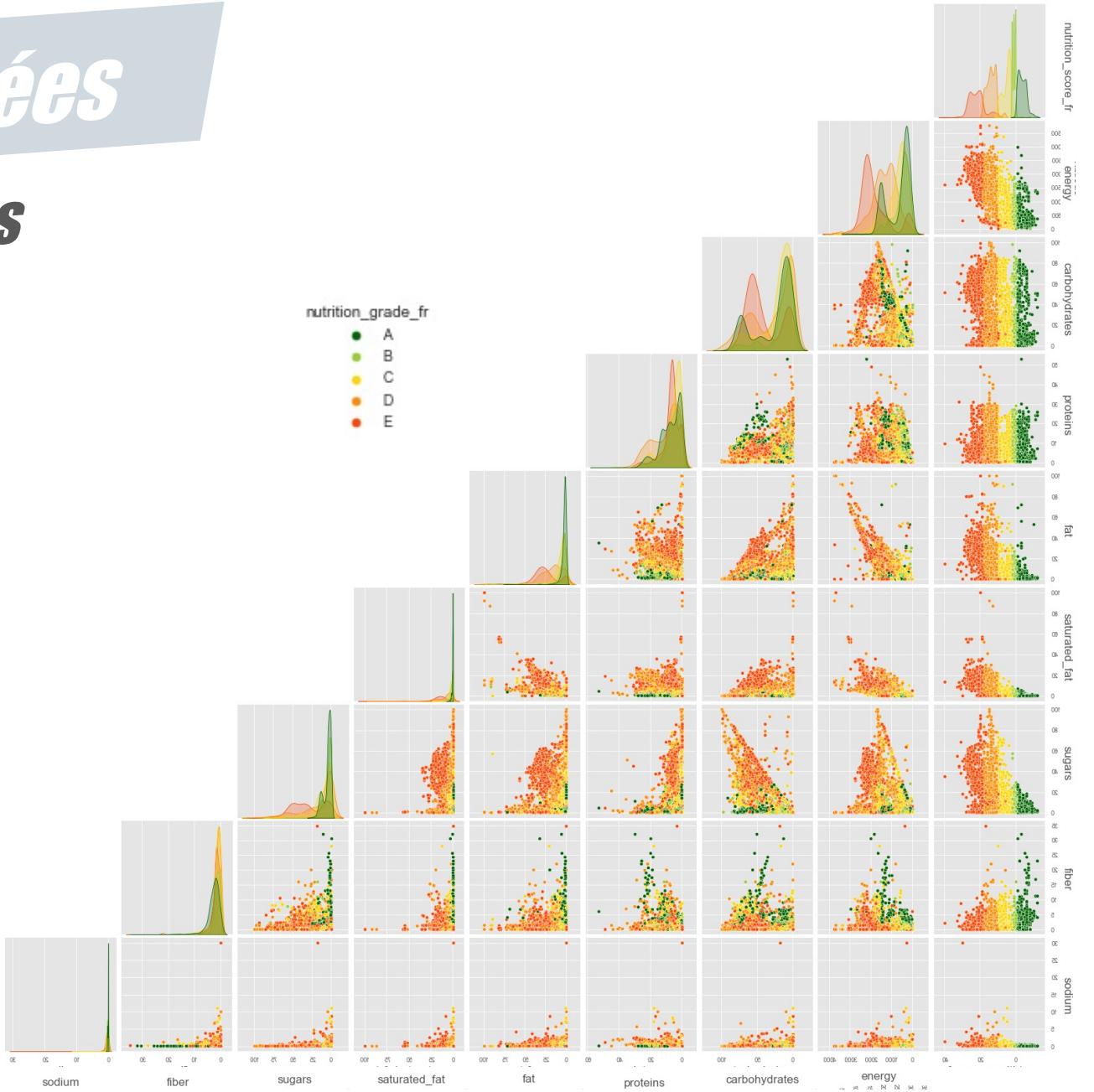
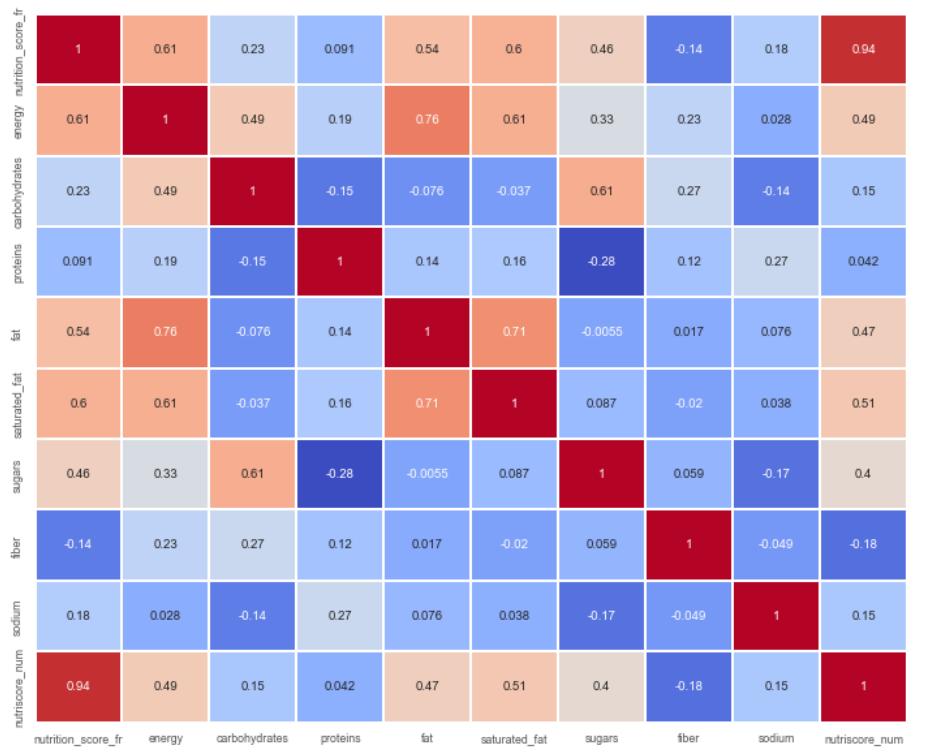
# *3. Exploration de données*

## *Bivarié : groupes & sucre*



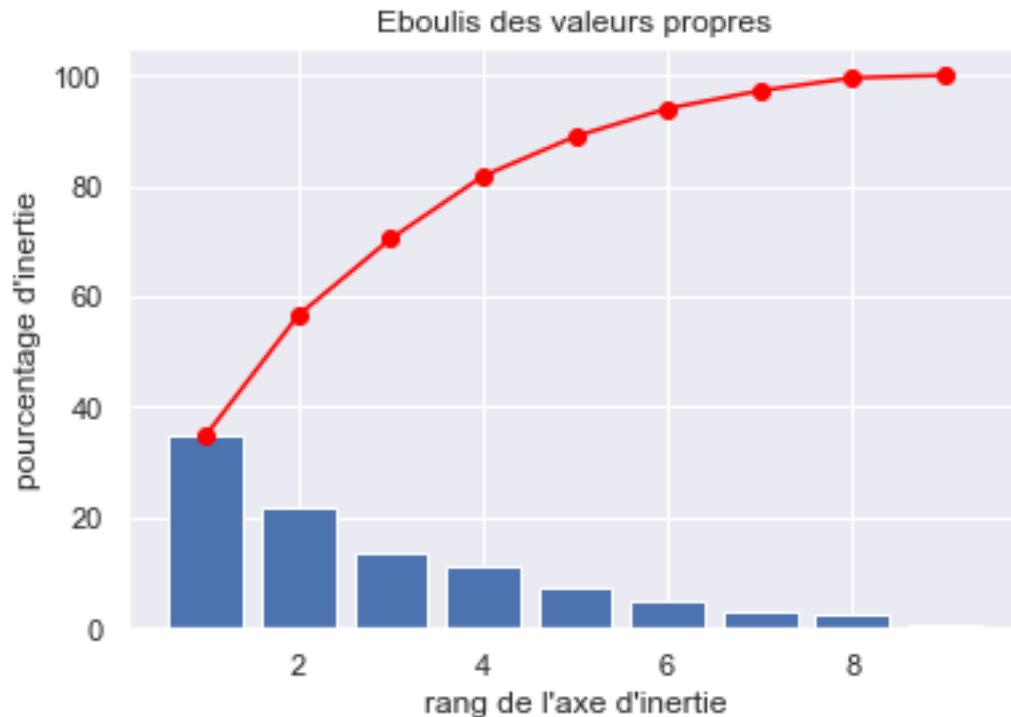
# 3. Exploration de données

## Bivarié : corrélations linéaires



### *3. Exploration de données*

#### **Multivariée : ACP**



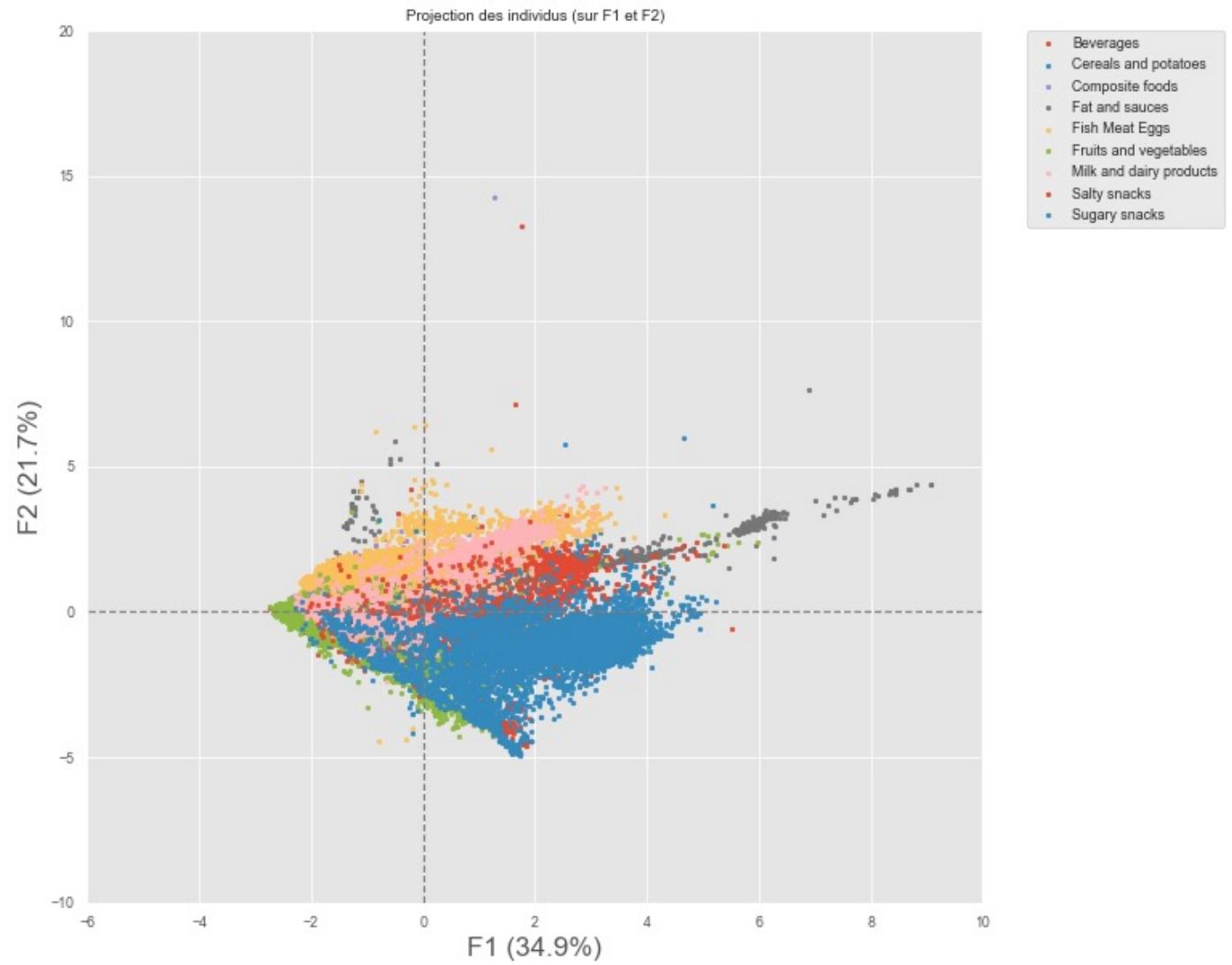
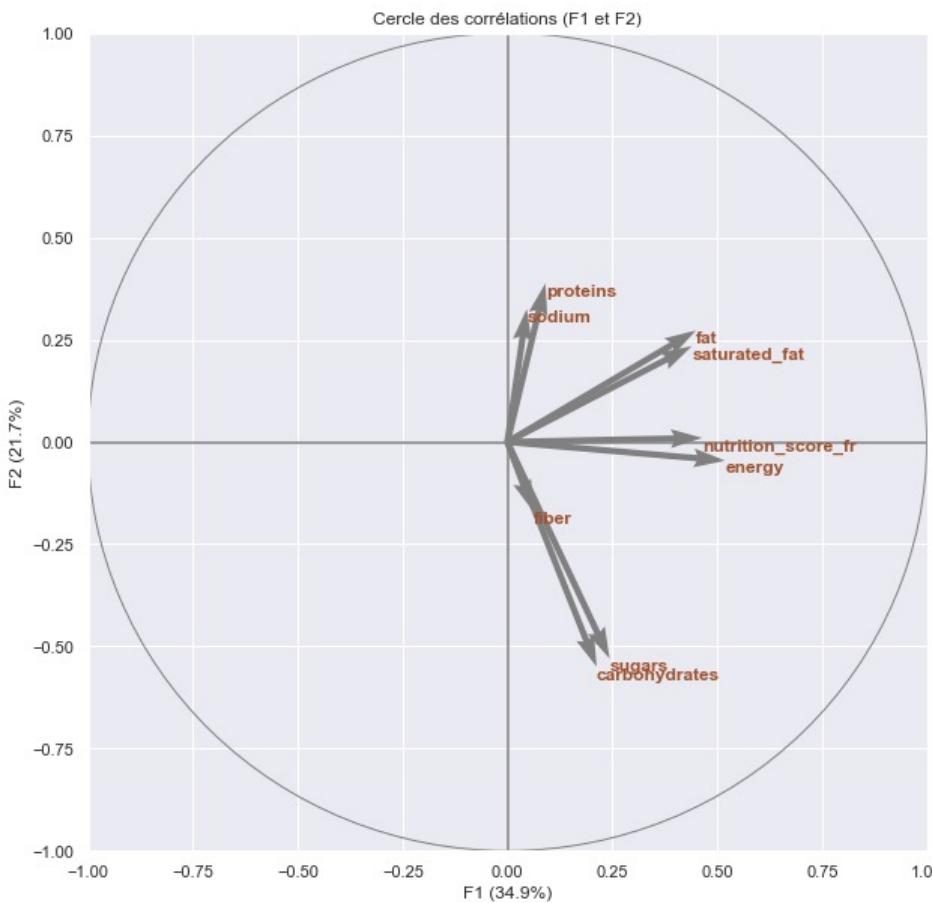
Variables numériques

#### **Dimensions de l'ACP**

	0	1	2	3	4	5	6	7	8
0	0.2	0.3	0.0	0.0	0.2	0.2	0.1	0.0	0.0
1	0.0	0.0	0.3	0.2	0.1	0.1	0.3	0.0	0.1
2	0.1	0.0	0.1	0.2	0.0	0.0	0.0	0.5	0.0
3	0.1	0.0	0.0	0.0	0.1	0.1	0.1	0.1	0.6
4	0.0	0.0	0.0	0.5	0.0	0.0	0.0	0.1	0.3
5	0.1	0.2	0.2	0.0	0.1	0.1	0.2	0.2	0.0
6	0.0	0.0	0.1	0.0	0.2	0.6	0.1	0.0	0.0
7	0.5	0.0	0.0	0.0	0.0	0.0	0.3	0.0	0.0
8	0.0	0.5	0.2	0.0	0.3	0.0	0.0	0.0	0.0

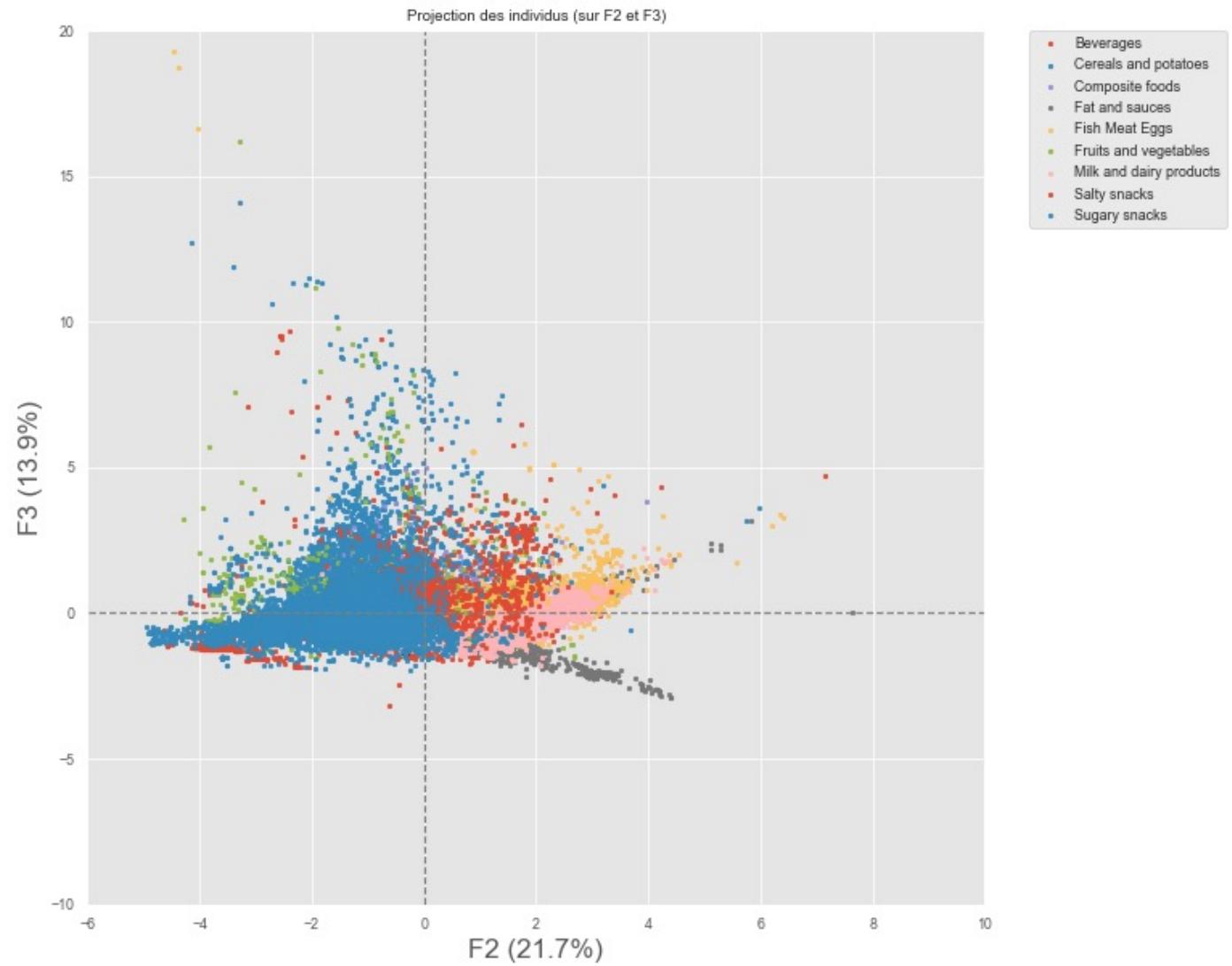
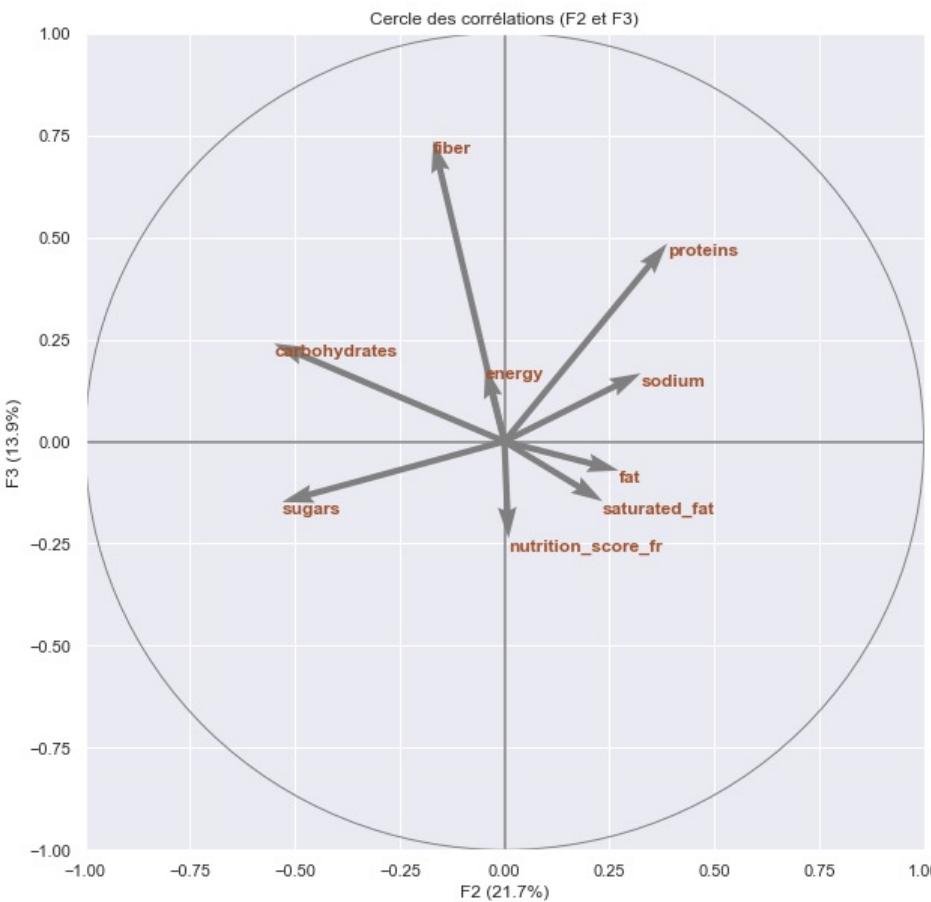
### 3. Exploration de données

#### Multivariée : ACP F1 & F2



### 3. Exploration de données

#### Multivariée : ACP F2 & F3



# *Conclusions pour l'app*



## **Valeurs non normales**

Limite quelque peu l'exploitation de données

## **Le recensement des données**

Manque cruellement de rigueur

## **L'application semble faisable**

Au vu du nombre de produits recensé

## **Les variables nécessaires pour l'application**

Sont bien remplies

**Merci**

**Alexandre Delaguillaumie**