Projet 3 - Concevez une application au service de la santé publique

Appel à projets pour trouver des idées innovantes d'applications en lien avec l'alimentation.

Sommaire

Mon idée d’application

Nettoyage des données

Analyse des données (univariée et multivariée)

Le projet et sa faisabilité

puis remarques.

L'application

- Scanner un produit
- Obtenir son nutri score

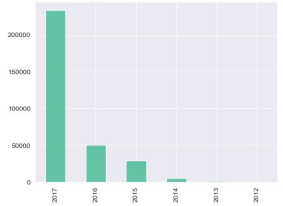
Si sa catégorie est détectée

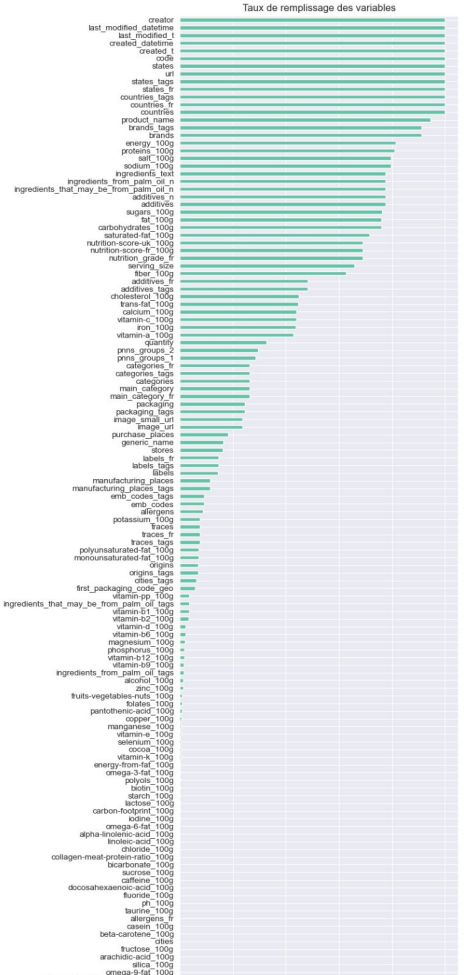
➔ Trouver 3 alternatives avec un meilleur nutri score

NETTOYAGE

Nettoyage des données

Les données sont à jour





- 320 772 individus pour 162 variables
- Conservation des variables remplies à plus de 60% avant sélection pour l'application

```
list_var = ['code','url' ,'product_name' ,'brands',
                'countries', 'countries_tags', 'countries_fr',
                'ingredients_text','additives_n', 'additives',
                 'energy_100g', 'fat_100g','saturated-fat_100g', 'carbohydrates_100g',
                'sugars_100g', 'fiber_100g','proteins_100g', 'salt_100g', 'sodium_100g',
                'nutrition-score-fr_100g','nutrition_grade_fr',
                'pnns_groups_1', 'pnns_groups_2',  'main_category_fr']
```

- Suppression des individus sans code ou sans product_name, et ceux qui n'ont aucune donnée à partir d'ingredients_text
- Suppression des duplicates sur la variable code en conservant l'individu le plus rempli

320 772 → 282988

-12%

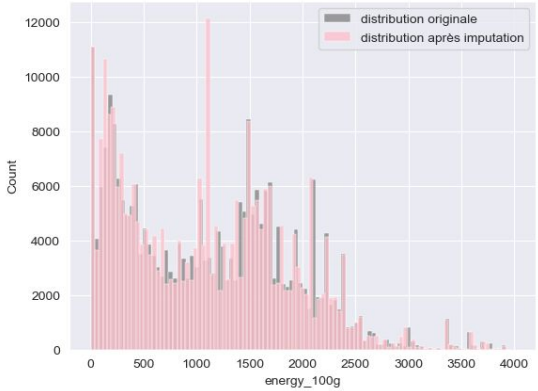
Nettoyage des données

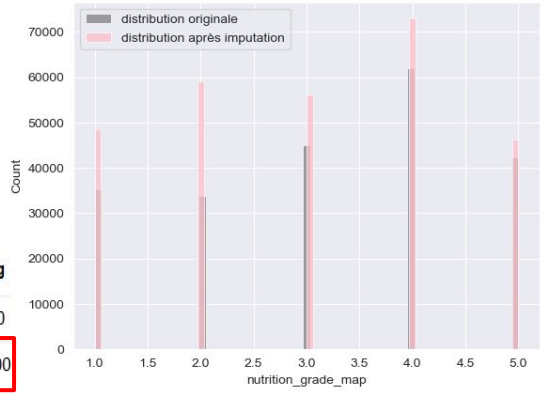
Valeurs aberrantes

- energy_100g ⇒ NaN pour > 4000kcal (aliment le plus calorique possible)
- autres valeurs en 100g ⇒ val abs>100 : NaN , val abs<100 : val abs

Valeurs manquantes

- valeurs en 100g ⇒ NaN : median du pnns_groups_2 } univariée
- energy_100g ⇒ iterative imputer (25,5 k données) } multivariée
- nutri score ⇒ knn imputer (64,5 k données)



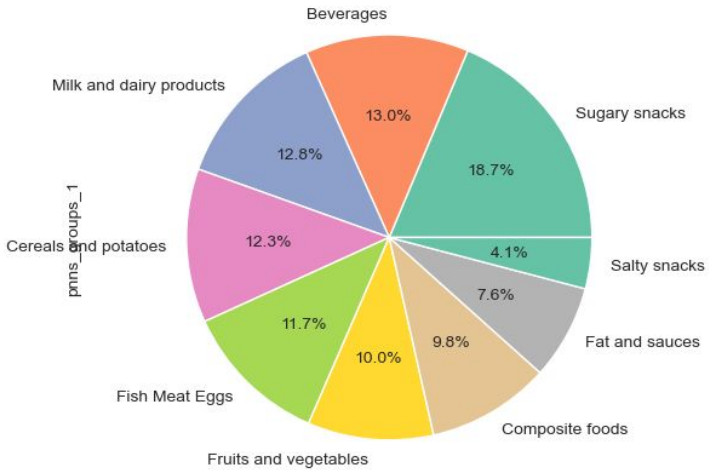


	additives_n	energy_100g	fat_100g	saturated-fat_100g	carbohydrates_100g	sugars_100g	fiber_100g	proteins_100g	salt_100g	sodium_100g
min	0.000000	0.000000e+00	0.000000	0.000000	0.000000	-17.860000	-6.700000	-800.000000	0.000000	0.000000
max	31.000000	3.251373e+06	714.290000	550.000000	2916.670000	3520.000000	5380.000000	430.000000	64312.800000	25320.000000

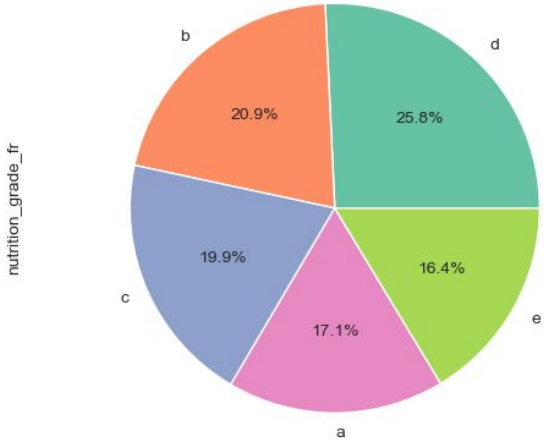

ANALYSE

Analyse des données - Univariée

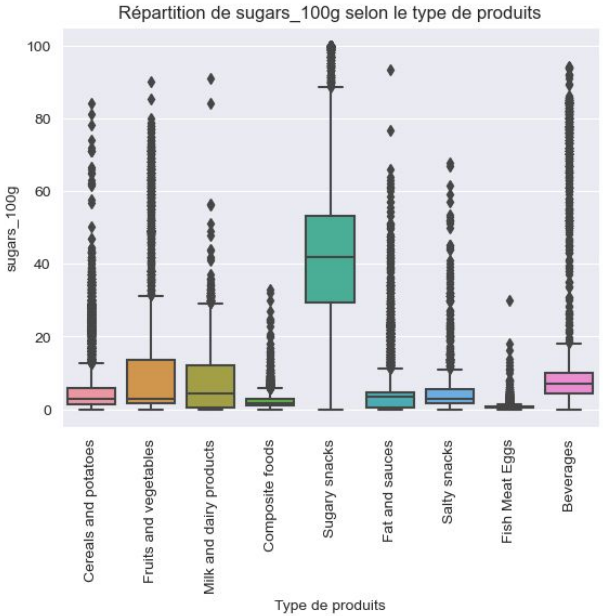
Catégorie des produits (68k renseignées)

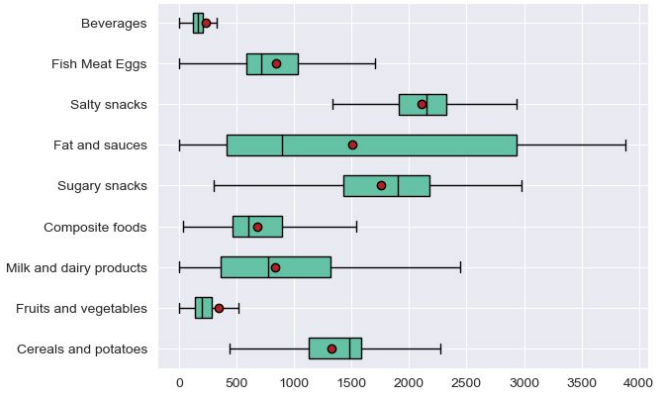


Nutri Score

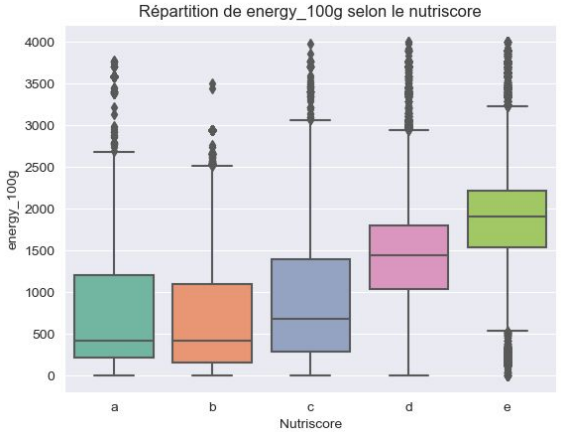


Analyse des données - Bivariée (Quali x Quanti)





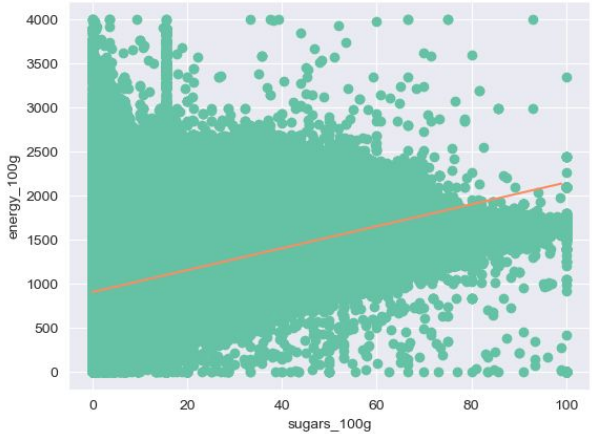
Eta-Squared = 0,53



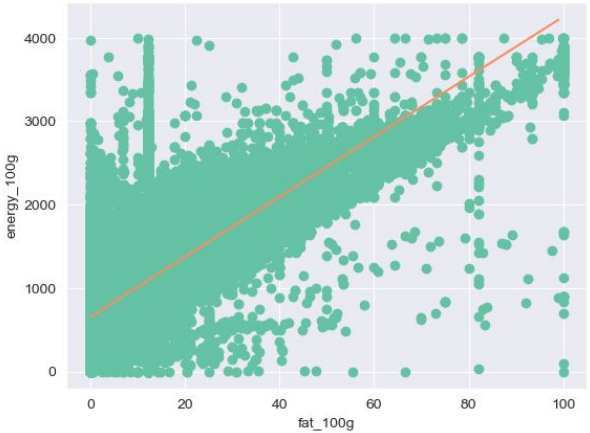
Eta-Squared = 0,35

> 0,14 : Effet de grande taille (Cohen)

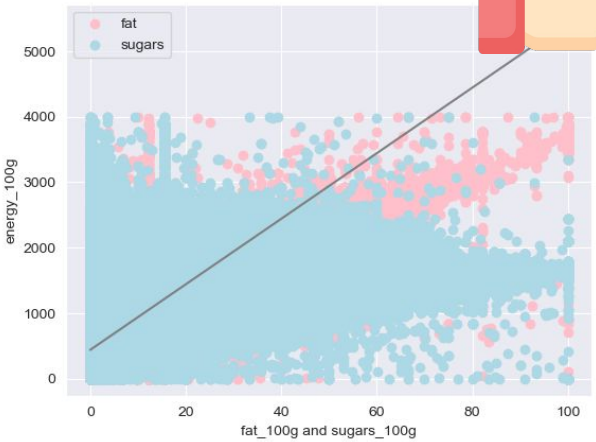
Analyse des données - Bivariée (Quanti x Quanti)



y = 12,4x + 903.7
r2 = 0,10



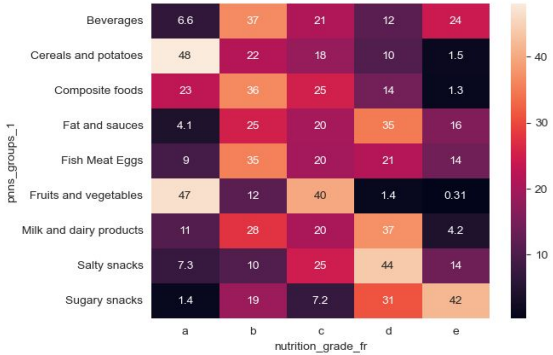
y = 36 x + 652
r2 = 0,58

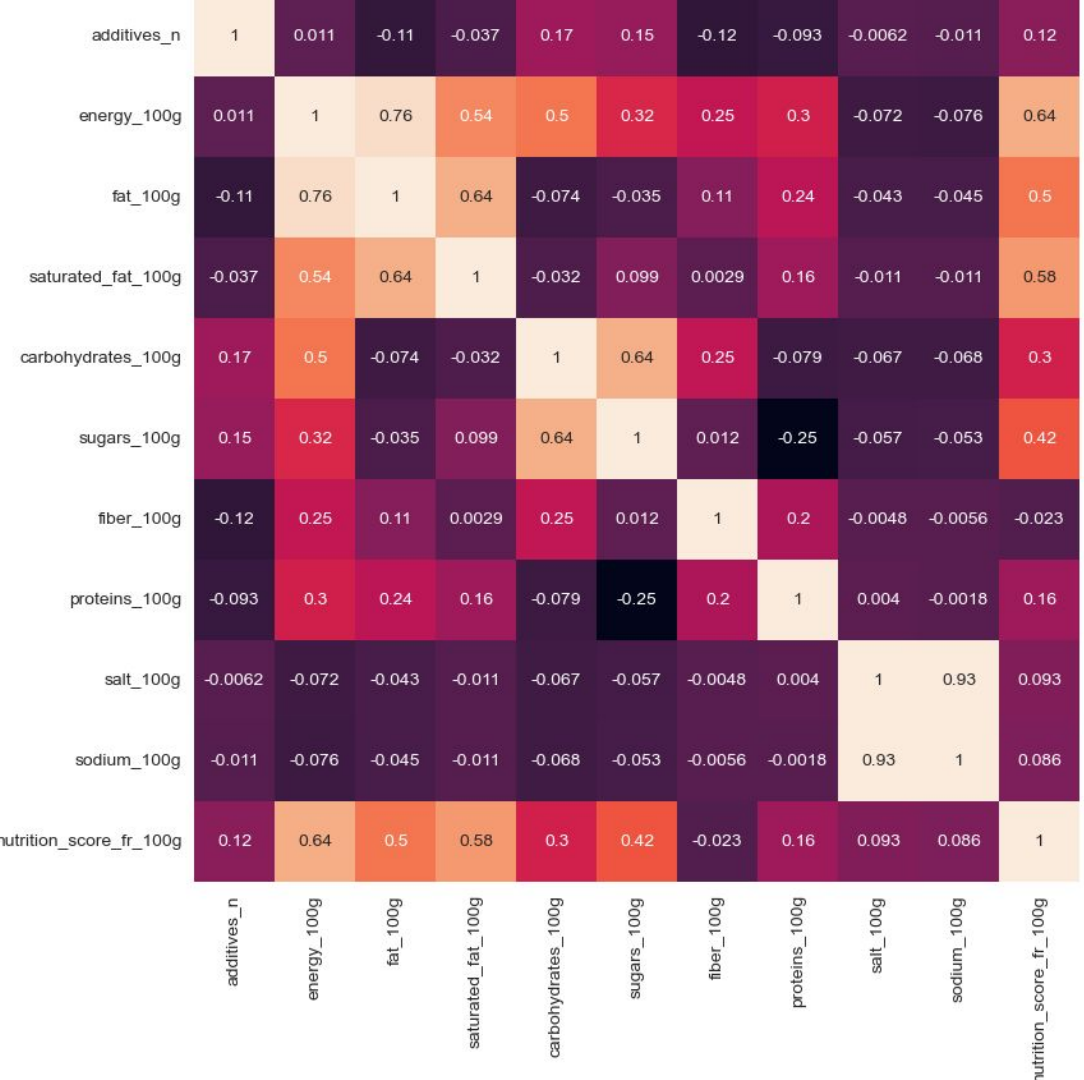


y = 36,6 x1 + 13,5 x2 + 436,7
r2 = 0,69

Analyse des données - Bivariée (Quali x Quali)

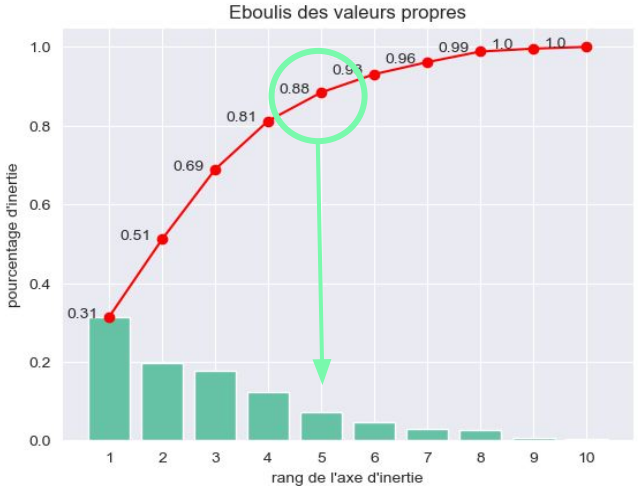
nutrition_grade_fr	a	b	c	d	e	TotalCateg
pnns_groups_1						
Beverages	6.61	36.55	21.19	11.52	24.13	100.0
Cereals and potatoes	48.08	21.66	18.28	10.49	1.49	100.0
Composite foods	22.81	36.39	25.16	14.32	1.32	100.0
Fat and sauces	4.09	25.14	19.99	34.94	15.84	100.0
Fish Meat Eggs	9.01	35.47	19.90	21.48	14.14	100.0
Fruits and vegetables	46.72	11.57	40.04	1.37	0.31	100.0
Milk and dairy products	10.70	27.65	20.40	37.08	4.17	100.0
Salty snacks	7.31	10.43	24.66	43.94	13.66	100.0
Sugary snacks	1.36	18.86	7.17	30.98	41.63	100.0

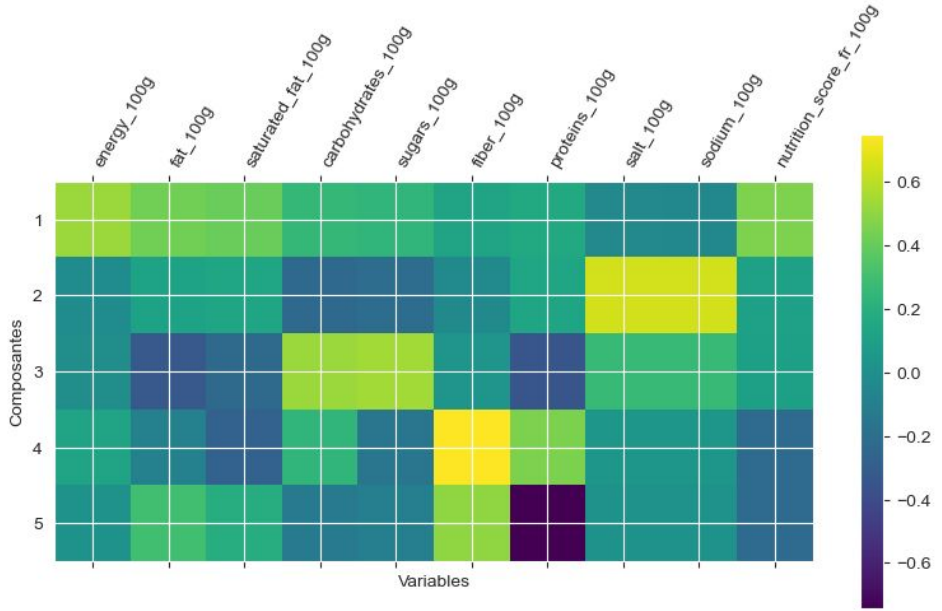




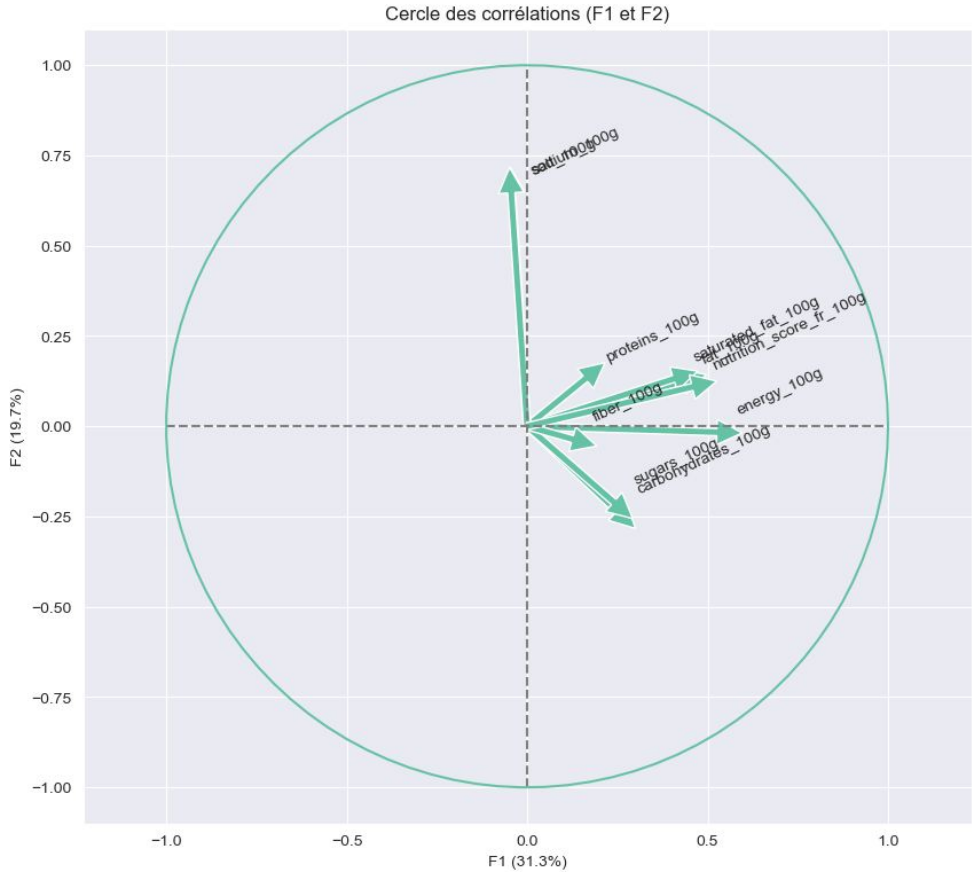
Corrélations

Analyse des données - Multivariées - ACP





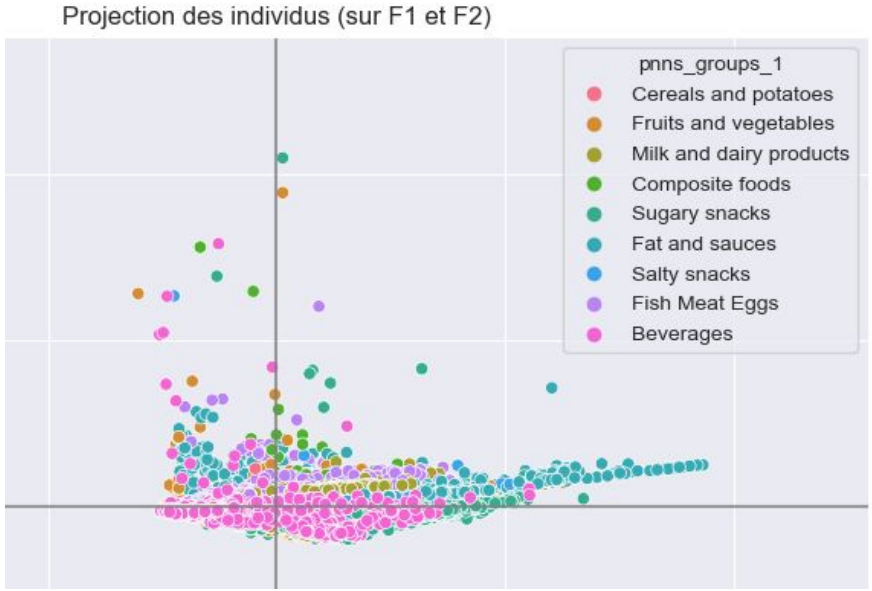
ACP - F1 & F2

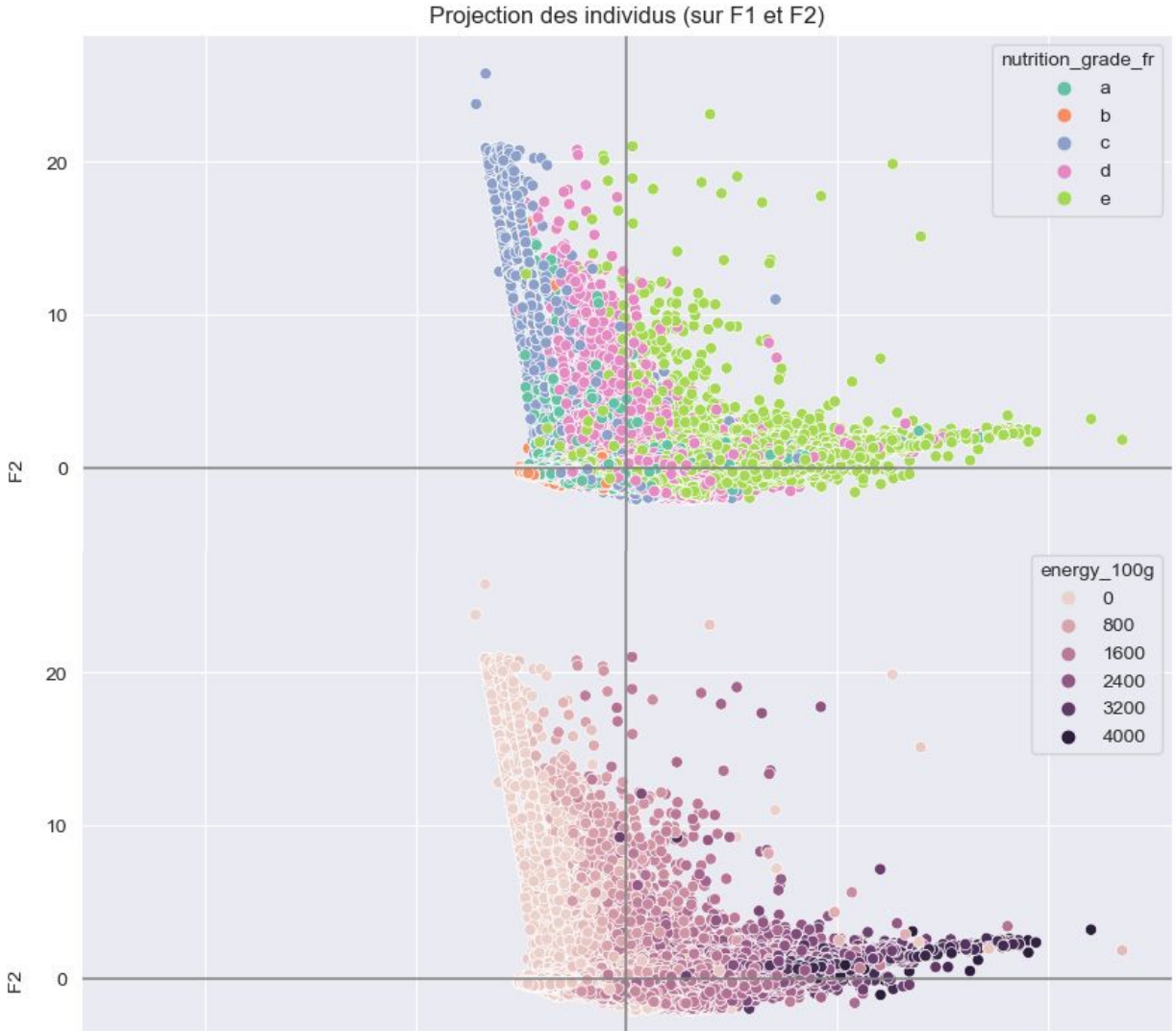


F1 : Aliments les plus caloriques et gras / mauvais à droite (Fat & Sauces →, fruits and vg ←)

F2 : Aliments salés en haut, aliments sucrés plus bas (salty snacks ↑, beverages & sugary snacks ↓)

	F1	F2
energy_100g	0.53	-0.02
fat_100g	0.43	0.12
saturated_fat_100g	0.41	0.13
carbohydrates_100g	0.25	-0.24
sugars_100g	0.24	-0.21
fiber_100g	0.13	-0.04
proteins_100g	0.16	0.13
salt_100g	-0.04	0.65
sodium_100g	-0.04	0.65
nutrition_score_fr_100g	0.46	0.11



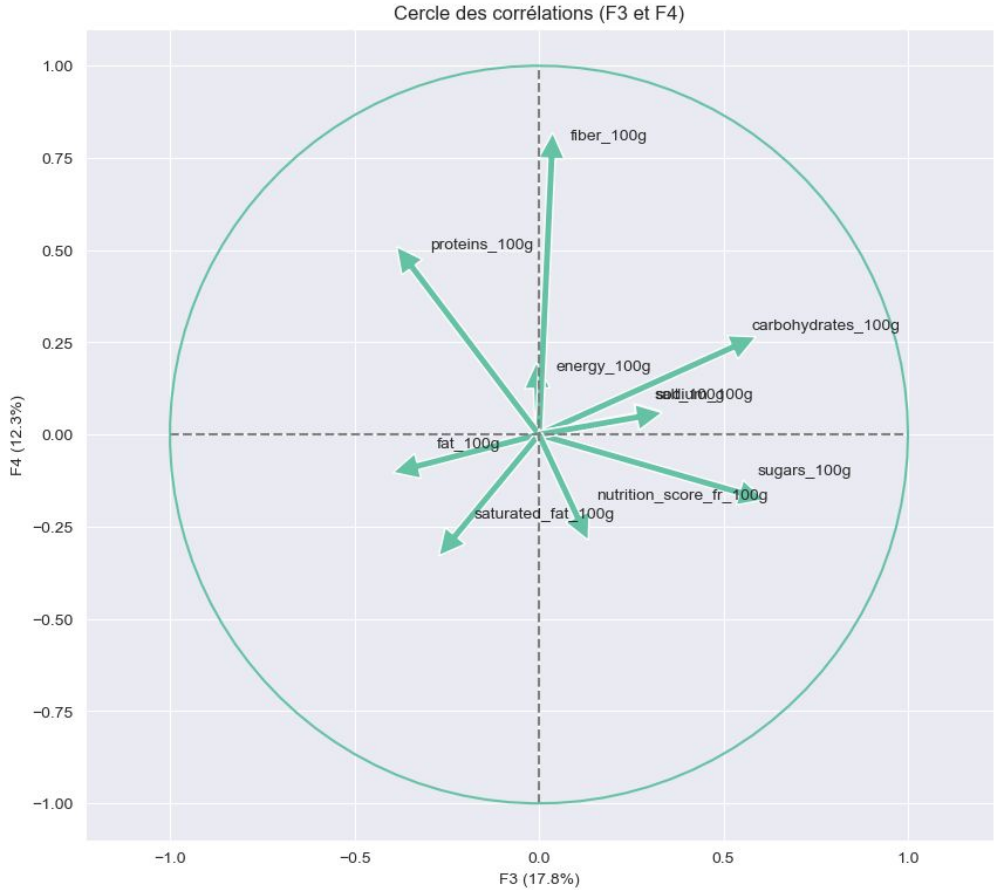


	F1	F2
energy_100g	0.53	-0.02
fat_100g	0.43	0.12
saturated_fat_100g	0.41	0.13
carbohydrates_100g	0.25	-0.24
sugars_100g	0.24	-0.21
fiber_100g	0.13	-0.04
proteins_100g	0.16	0.13
salt_100g	-0.04	0.65
sodium_100g	-0.04	0.65
nutrition_score_fr_100g	0.46	0.11

F1 : Aliments les plus caloriques et gras / mauvais à droite

F2 : Aliments salés en haut, aliments sucrés plus bas

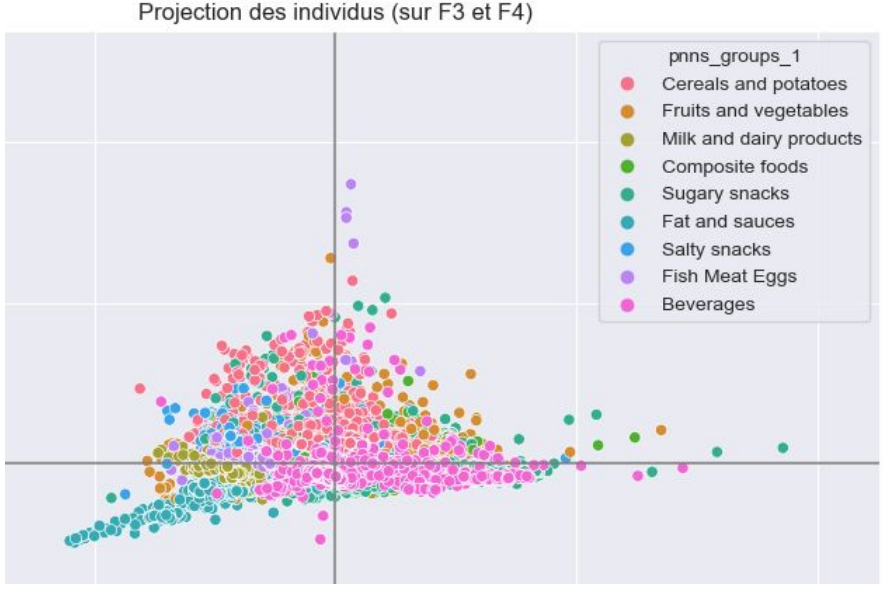
ACP - F3 & F4



F3 : Aliments les plus sucrés et avec des glucides , un peu de sel mais pas trop de gras ni de protéines (Beverages →,
Fat & Sauces, Salty Snacks, Milk Dairy, Fish Meat Eggs ←)

F4 : Fibres et protéines en haut, gras vers le bas (Fish Meat Eggs , Cereals↑,
Fat & Sauces ↓)

	F3	F4
energy_100g	-0.00	0.13
fat_100g	-0.33	-0.09
saturated_fat_100g	-0.23	-0.27
carbohydrates_100g	0.52	0.24
sugars_100g	0.54	-0.16
fiber_100g	0.03	0.75
proteins_100g	-0.34	0.45
salt_100g	0.26	0.05
sodium_100g	0.27	0.05
nutrition_score_fr_100g	0.10	-0.22

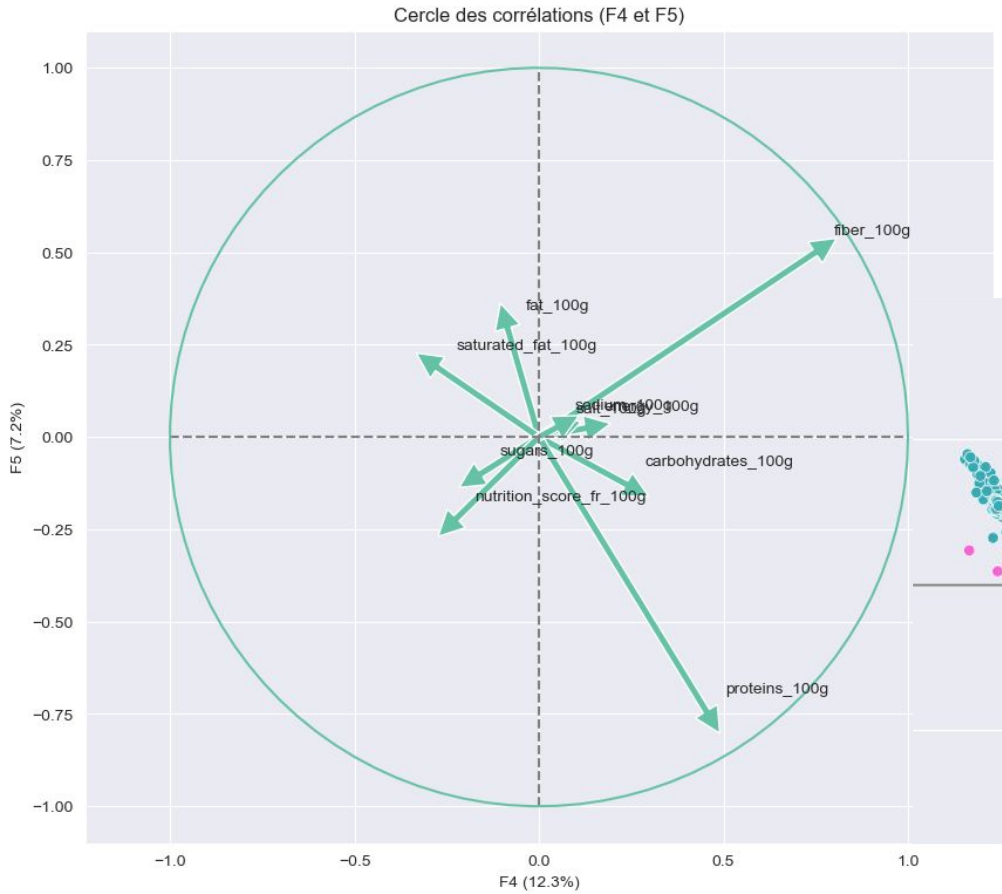


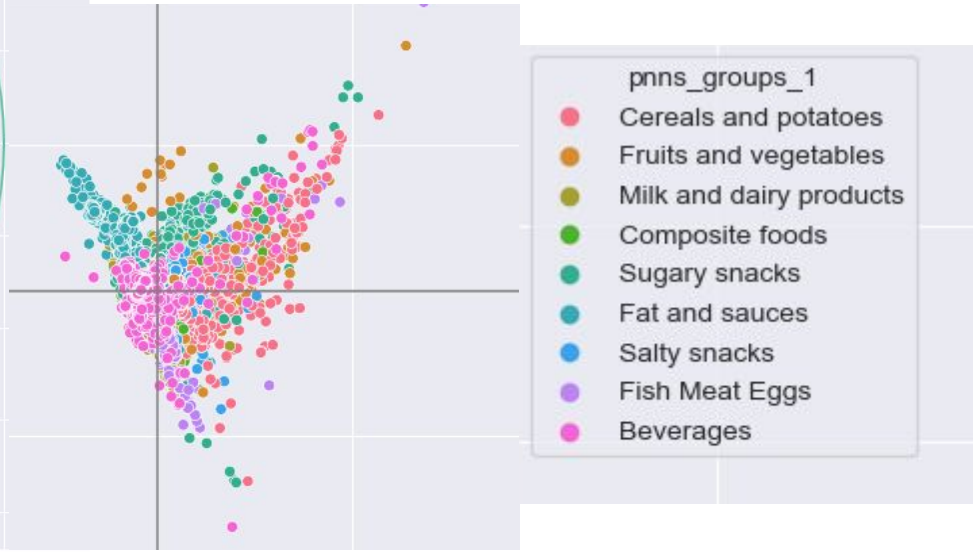
ACP - F4 & F5

F4 : Fibres et protéines a droite, gras à gauche
(Fish Meat Eggs , Cereals→, Fat & Sauces ←)

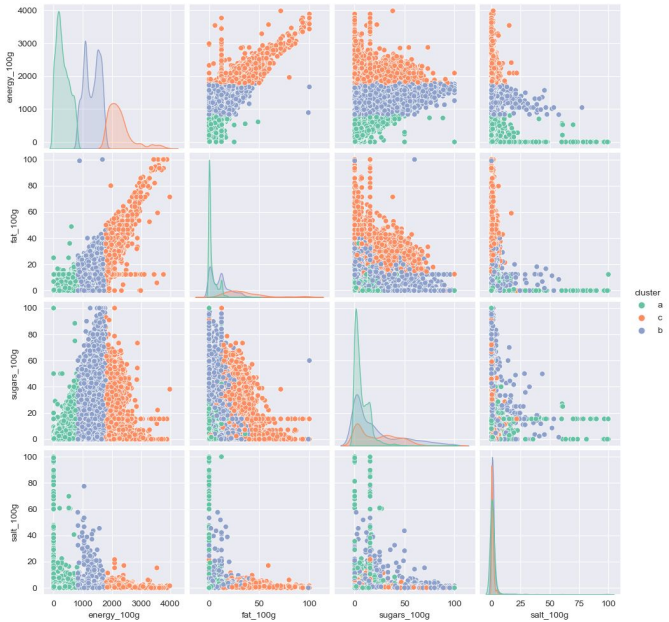
F5 : Fibres et gras en haut, protéines et sucre vers le bas (Fat & Sauces↑, Fish Meat Eggs ↓)

	F4	F5
energy_100g	0.13	0.02
fat_100g	-0.09	0.30
saturated_fat_100g	-0.27	0.19
carbohydrates_100g	0.24	-0.13
sugars_100g	-0.16	-0.10
fiber_100g	0.75	0.50
proteins_100g	0.45	-0.74
salt_100g	0.05	0.02
sodium_100g	0.05	0.03
nutrition_score_fr_100g	-0.22	-0.22

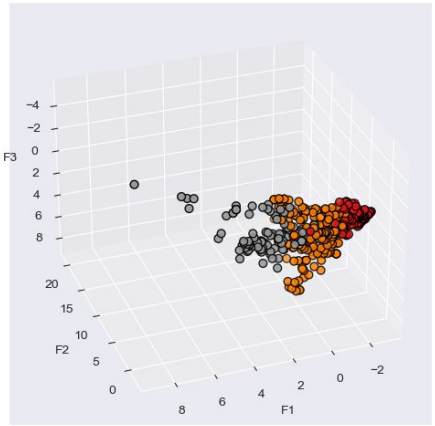




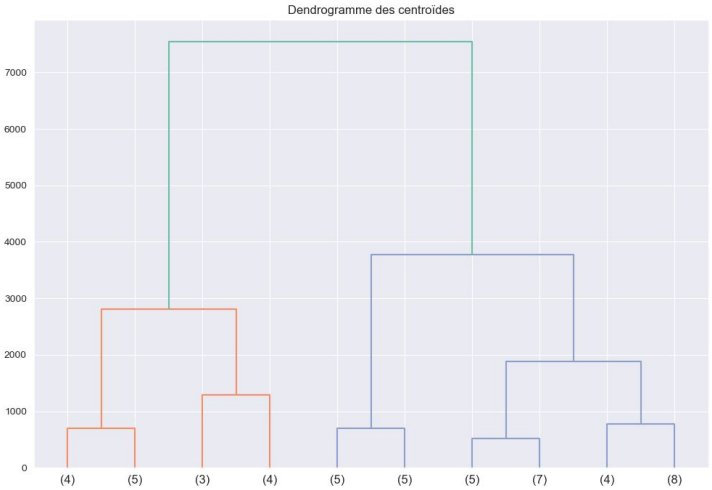
Analyse des données - Pour aller + loin



Clusters n_clust=3
⇒ Clustering x ACP



Clusters n_clust=50
⇒ Classification



LE PROJET

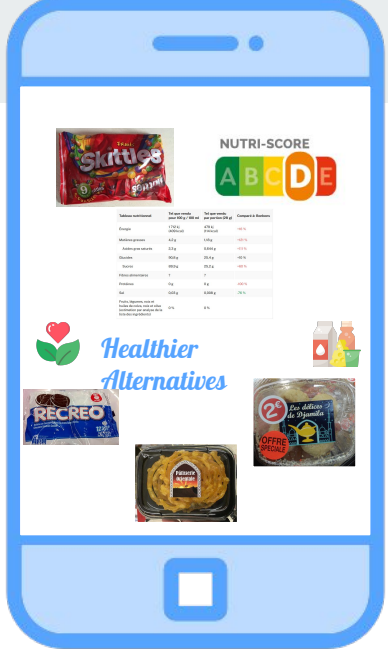
Le projet et sa faisabilité

```
def applicationbis(code):
    code = str(code)
    product = data[data['code'] == code]
    cols = ['code','url','product_name','nutrition_grade_fr','nutrition_score_fr_100g','pnns_groups_1','energy_100g',
       'fat_100g','sugars_100g','fiber_100g', 'proteins_100g', 'salt_100g']
    product_scanned = product[cols]

    if not pd.isnull(product_scanned['pnns_groups_1'].iloc[0]):
        category = product_scanned['pnns_groups_1'].iloc[0]
        similar_products = data[data['pnns_groups_1'] == category]

        features = similar_products[['nutrition_score_fr_100g','energy_100g',
       'fat_100g','sugars_100g','fiber_100g', 'proteins_100g', 'salt_100g']]
        kmeans = KMeans(n_clusters=3, random_state=0,n_init='auto').fit(features)
        closest_products = similar_products.iloc[kmeans.labels_==kmeans.predict(features)[0]]
        closest_products = closest_products.sort_values(by='nutrition_score_fr_100g', ascending=True)
        closest_products = closest_products[cols].head(3)
        return pd.concat([product_scanned, closest_products]).set_axis(['Scanned_Product', 'Similar_1', 'Similar_2', 'S
    else:
        return product_scanned
```

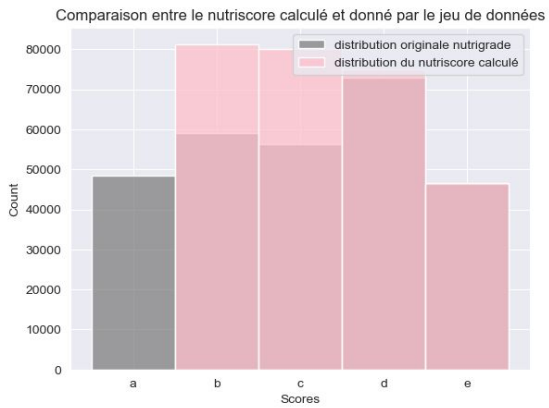
L'application : on scanne un produit, on obtient son nutri score

.

Si la catégorie est identifiée on obtient 3 alternatives avec un meilleur Nutri Score dans la même catégorie.



	code	url	product_name	nutrition_grade_fr	nutrition_score_fr_100g	pnns_groups_1	energy_100g	fat_100g	sugars_100g	fiber_100g	proteins_100g	salt_100g
Scanned_Product	5000159012041	http://world-fr.openfoodfacts.org/produit/5000159012041/skittles-fruits-wrigley-s	Skittles Fruits	d	17.0	Sugary snacks	1712.000000	4.2	89.9	1.1	0.0	0.03
Similar_1	7702025182305	http://world-fr.openfoodfacts.org/produit/7702025182305/recreo	Recreo	b	0.0	Sugary snacks	1903.438952	21.0	30.5	2.6	6.1	0.57
Similar_2	3582730001719	http://world-fr.openfoodfacts.org/produit/3582730001719/patisserie-orientale-zalabia-x2-sans-marque	Pâtisserie orientale (Zalabia x2)	b	0.0	Sugary snacks	1903.438952	21.0	30.5	2.6	6.1	0.57
Similar_3	3582739701887	http://world-fr.openfoodfacts.org/produit/3582739701887/mini-montecao-x4-les-delices-de-djamila	Mini Montecao x4	b	0.0	Sugary snacks	1903.438952	21.0	30.5	2.6	6.1	0.57



Conclusions & Remarques

Si c'était à refaire :

- Utiliser un iterative imputer pour toutes les données en 100g plutôt que la médiane du groupe, puisque je m'en sers dans l'application.
- Mettre à part les individus dont j'ai le nutriscore mais aucune info de composition (donc pas d'iterative imputer possible) pour les garder pour infos produits pour l'app mais pas pour les suggestions

Avec plus de connaissances :

- Essayer d'estimer les pnns_groups_1 & 2 inconnus à l'aide de la composition, du nom et des ingrédients
- concernant le nutrition grade, j'ai fait une fonction pour le calculer à partir du nutriscore estimé mais non concluant
- pas réussi à exploiter les pays

Le plus important : j'ai gagné en compétences