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

Présenté par :
Bourama FANE
Etudiant Data Scientist

Dirigé par :
Babou M'BAYE
Mentor chez OpenClassrooms

24 février 2023

Sommaire

Plan de la présentation

Problématique

L'agence **Santé publique France** a lancé un appel à projets pour trouver des idées innovantes d'applications en lien avec l'alimentation.

Vous souhaitez y **participer et proposer une idée d'application**.

Sources de données

Il s'agit du jeu de données **Open Food Fact**.

Mission

Nettoyer et analyser le jeu de données **Open Food Fact**, tout en reflechissant à une idée d'application.

Plan de la présentation

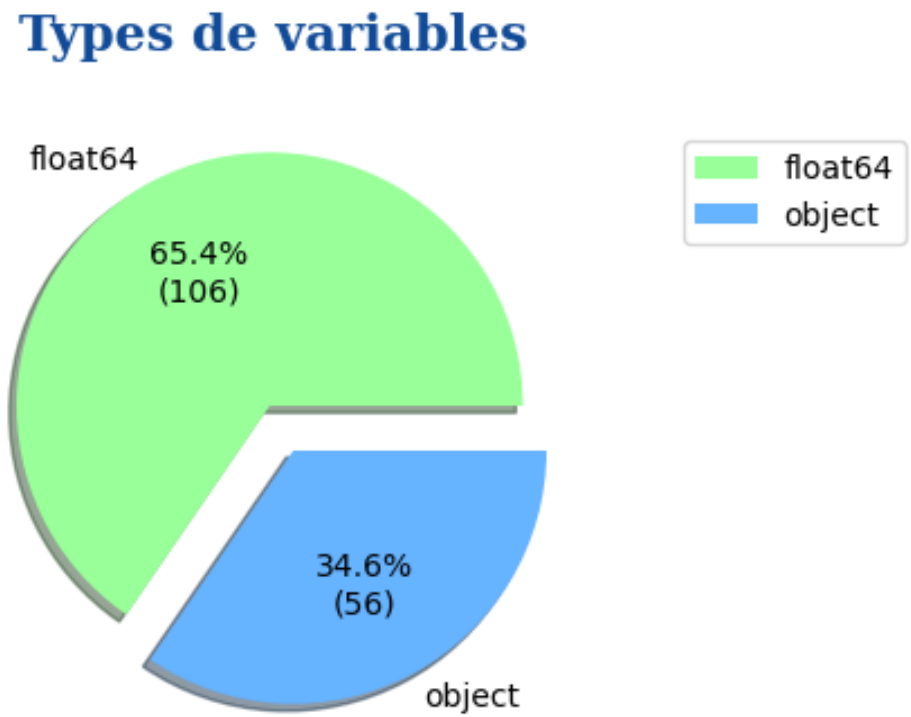
Description de la base

Les données sont organisées en 5 sections listées ci dessous :

- ☞ 1- informations générales sur le produit
- ☞ 2- ensemble de tags sur le produit et sa provenance
- ☞ 3- ingrédients et allergènes
- ☞ 4- informations diverses
- ☞ 5- informations nutritionnelles.

Description de la base

	Variable	nombre
0	lignes	320772
1	colonnes	162

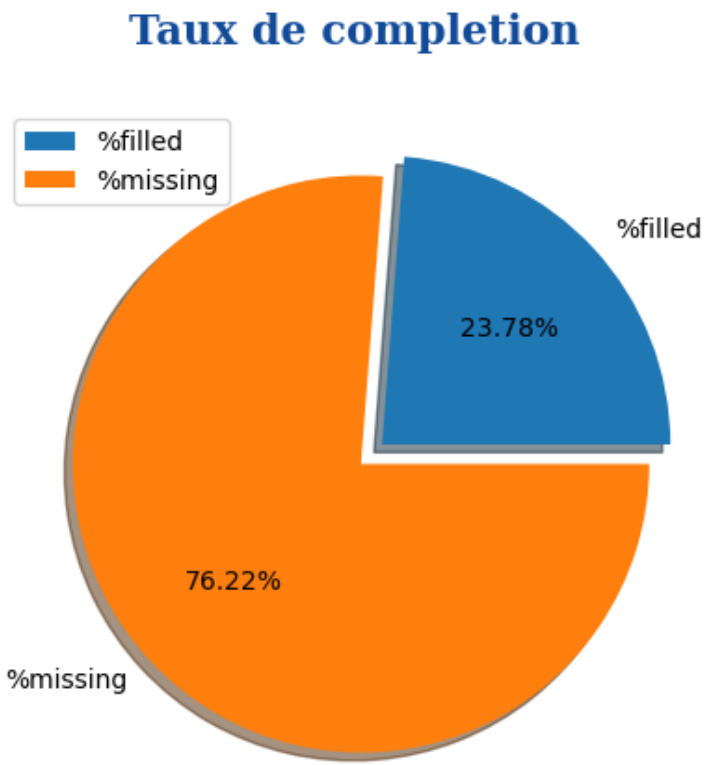


- ☞ La base de données contient **320 772 lignes**, contre **162 variables** ;
- ☞ Nous distinguons deux types de variables (**object** et **float**).

Description de la base

Les données présentent beaucoup de valeurs manquantes.

- Taux de missings de 76% .
- Certaines variables sont pratiquement vides.



Filtre sur le pays

- **countries**, le nom du pays est traduit dans différentes langues, souvent il affiche les initiales du nom du pays (ou région) ;
- **countries_tags** : il contient aussi souvent de la traduction anglaise du nom du pays ou région
- Nous allons donc utiliser **countries_fr** et supprimer les deux autres.

```
regionFrance = ['fr','france','Française', 'Réunion', 'Nouvelle-Calédonie', 'Martinique', 'Guadeloupe',
                'Mayotte', 'Guyane']

def filterColumns(data, cols, motsCles,negation=False):
    if negation==False:
        data=data.loc[data[cols].str.contains("|".join(motsCles), na = False, case=False),:]
    else:
        data=data.loc[~data[cols].str.contains("|".join(motsCles), na = False, case=False),:]
    return data

dfood=filterColumns(dfood, 'countries_fr', regionFrance)
```

Filtre sur la catégorie

- **categories_tags** est une traduction de la catégorie dans plusieurs langues.
- **categories**, elle fournit la meme information que catégories_fr ou fournit le sous groupe (preciser la catégorie)
- **difference entre categories_fr et categories :** il y a des tirets (-) dans les modalités de catégories_fr.

```
NonAlimentaires=['Non alimentaire','Open Beauty Facts','Nourriture pour animaux','Dentifrices']
dfNonAlimentaires=filterColumns(dfood, 'categories_fr', NonAlimentaires)
```

Filtre sur le nom du produit

```
dfood.dropna(subset=['product_name'], inplace=True)
dfood.product_name.isna().sum()
```

Nous avons supprimé les doublons au niveau du nom du produit.

Colonnes redondantes

Certaines colonnes du jeu de données semblent **dupliquées**.

Les noms des variables se ressemblent à la différence qu'une préfixe qui s'ajoute (**_fr**, **_tags**).

> Après examen de toutes ces variables, nous avons identifié une liste de **variables à supprimer**.

```
listeSupprimer=['additives','additives_tags','additives_fr','brands_tags','cities','labels','labels_tags',
                'allergens_fr','brands_tags', 'cities','emb_codes_tags','ingredients_from_palm_oil',
                'ingredients_from_palm_oil_tags','ingredients_that_may_be_from_palm_oil',
                'ingredients_that_may_be_from_palm_oil_tags','url', 'manufacturing_places_tags',
                'origins_tags', 'packaging_tags','states', 'states_tags', 'traces_tags','traces_fr']

dfood=dfood.drop(listeSupprimer,axis=1)
dfood.head()
```

Catégories de produits

Nous avons cinq variables relatives à la catégorie du produit.

'categories','pnns_groups_1','pnns_groups_2','main_category',
'main_category_fr'.

```
def ImputeModalite(data,col1, col2,modaliteToImpute="unknown",Varlist=''):
    if Varlist=='':
        Varlist=[col1,col2]
    ListModalite=data[(data[col1]==modaliteToImpute)][col2].unique().tolist()
    for modalite in ListModalite:
        df=data[(data[col2]==modalite)][Varlist]
        df=df[col1].value_counts(ascending=False).reset_index().rename(columns={'index':col1,col1:'nombre'})

        if df.iloc[0,0]!=modaliteToImpute:
            data.loc[(data[col1]==modaliteToImpute)&(data[col2]==modalite),col1]=df.iloc[0,0]
        elif df.shape[0]>1:
            data.loc[(data[col1]==modaliteToImpute)&(data[col2]==modalite),col1]=df.iloc[1,0]
        else:
            pass
    return data
```

```
ImputeModalite(dfood,'pnns_groups_1', 'main_category_fr',modaliteToImpute="unknown",Varlist=varCategory)
ImputeModalite(dfood,'pnns_groups_2', 'main_category_fr',modaliteToImpute="unknown",Varlist=varCategory)
```

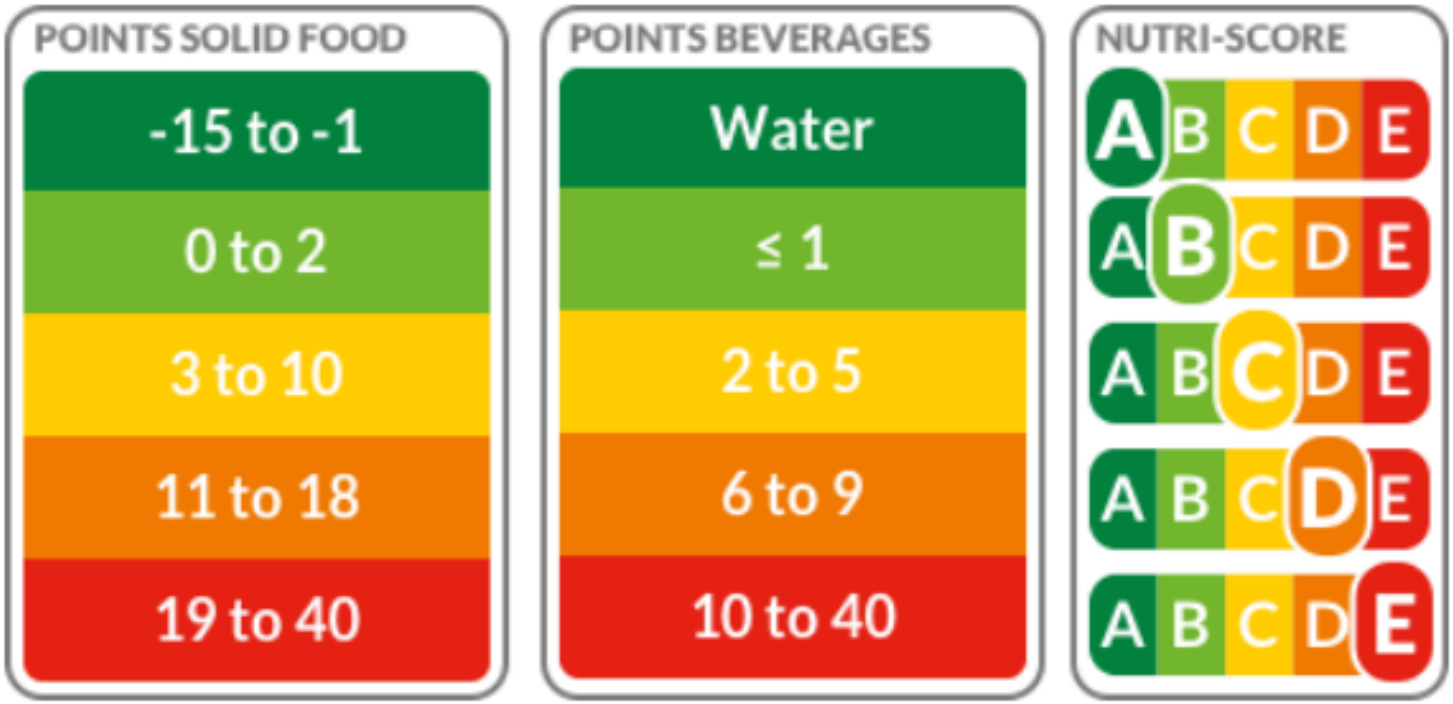
Nous avons supprimé les produits non identifiables à travers la catégorie.

Variables nutritionnelles & Autres

- ☞ Variables nutritionnelles (**en g**), les valeurs devront être comprises entre 0 et 100.
- ☞ Excepté la variable **energy_100g** (3700 kJ pour 100g ou 900 kcal pour 100g) ;
- ☞ Suppression des lignes pour lesquelles **toutes les variables nutriments sont vides** (NaN) ;
- ☞ Suppression des colonnes ayant un taux de completion $<60\%$.

Imputation du nutriscore

- ☞ la méthode KNN Imputer est utilisée pour imputer le nutriscore ;
- ☞ Ensuite, le nutriscore est utilisé pour imputer les nutrigrades



Outliers

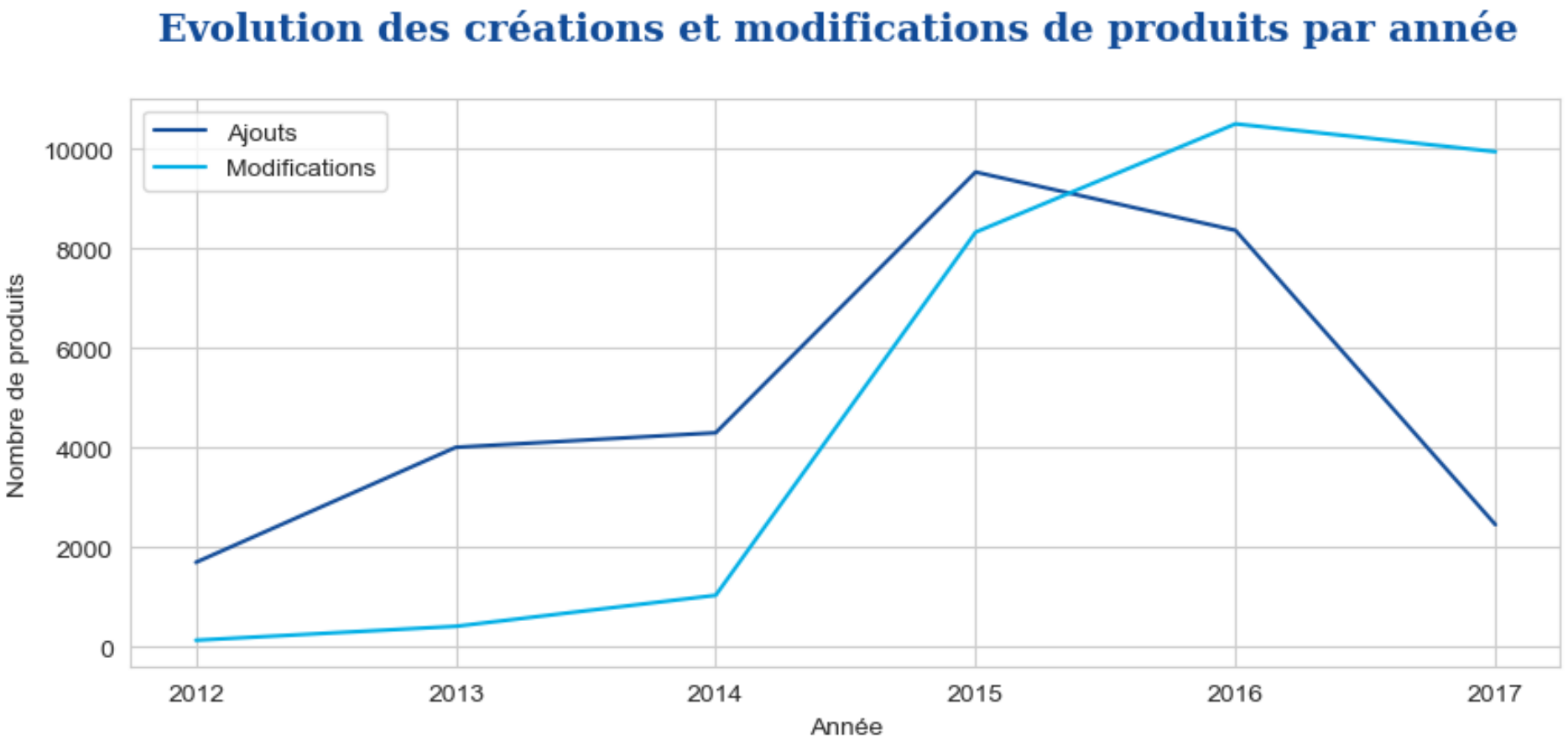
☞ la méthode IQR est utilisée pour la detection des outliers ;

```
# Creer un dataframe avec les outliers
def findOutliers(df, col):
    q1=df[col].quantile(0.25)
    q3=df[col].quantile(0.75)
    IQR=q3-q1
    outliers = df[((df[col]<(q1-1.5*IQR)) | (df[col]>(q3+1.5*IQR)))]
    return outliers

# Creer un dataframe en supprimant les Outliers
def deleteOutliers(df, col):
    q1=df[col].quantile(0.25)
    q3=df[col].quantile(0.75)
    IQR=q3-q1
    dfWithOut = df.loc[~((df[col]<(q1-1.5*IQR)) | (df[col]>(q3+1.5*IQR))),:]
    return dfWithOut
```

Plan de la présentation

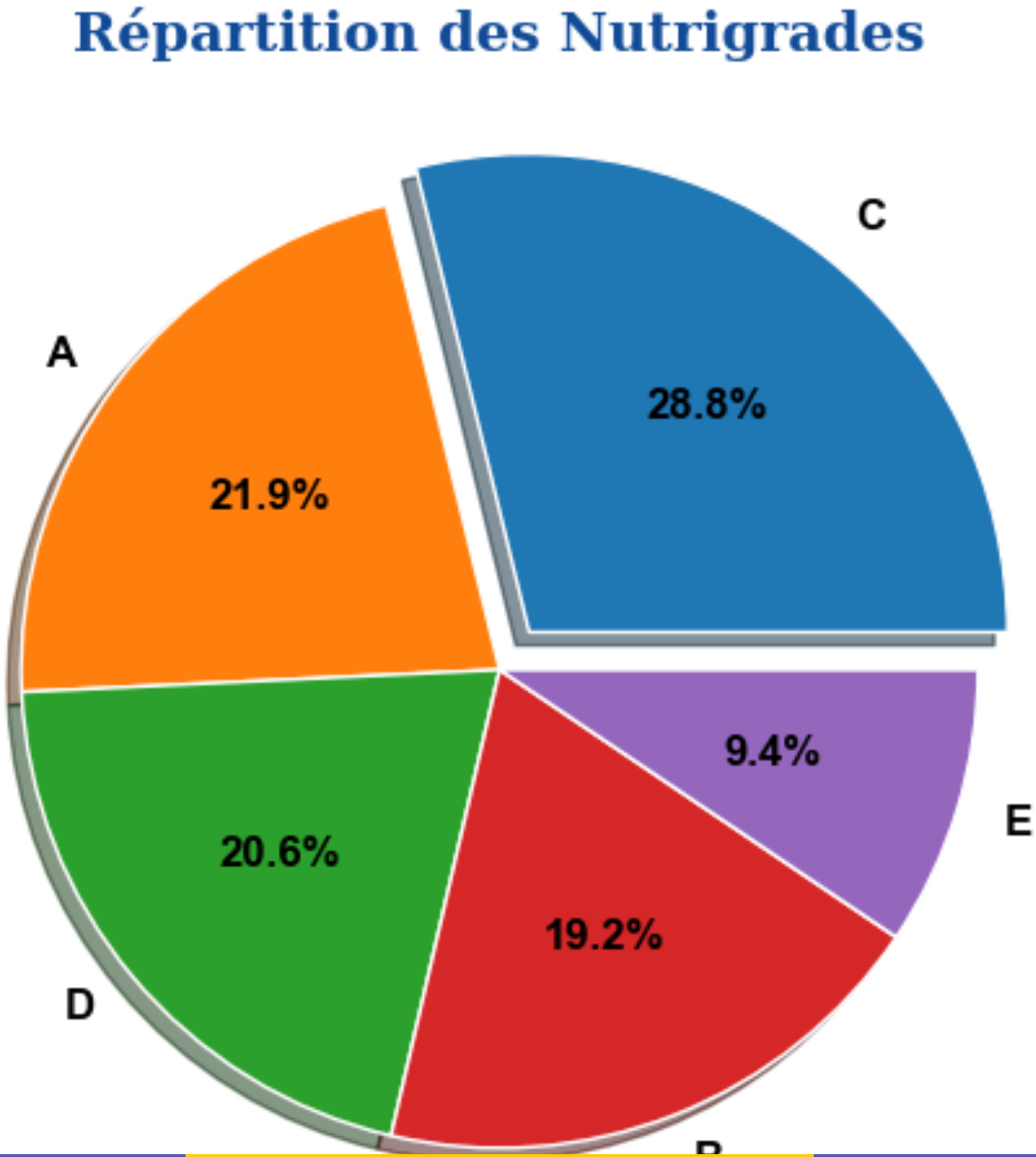
Ajout et modification des produits



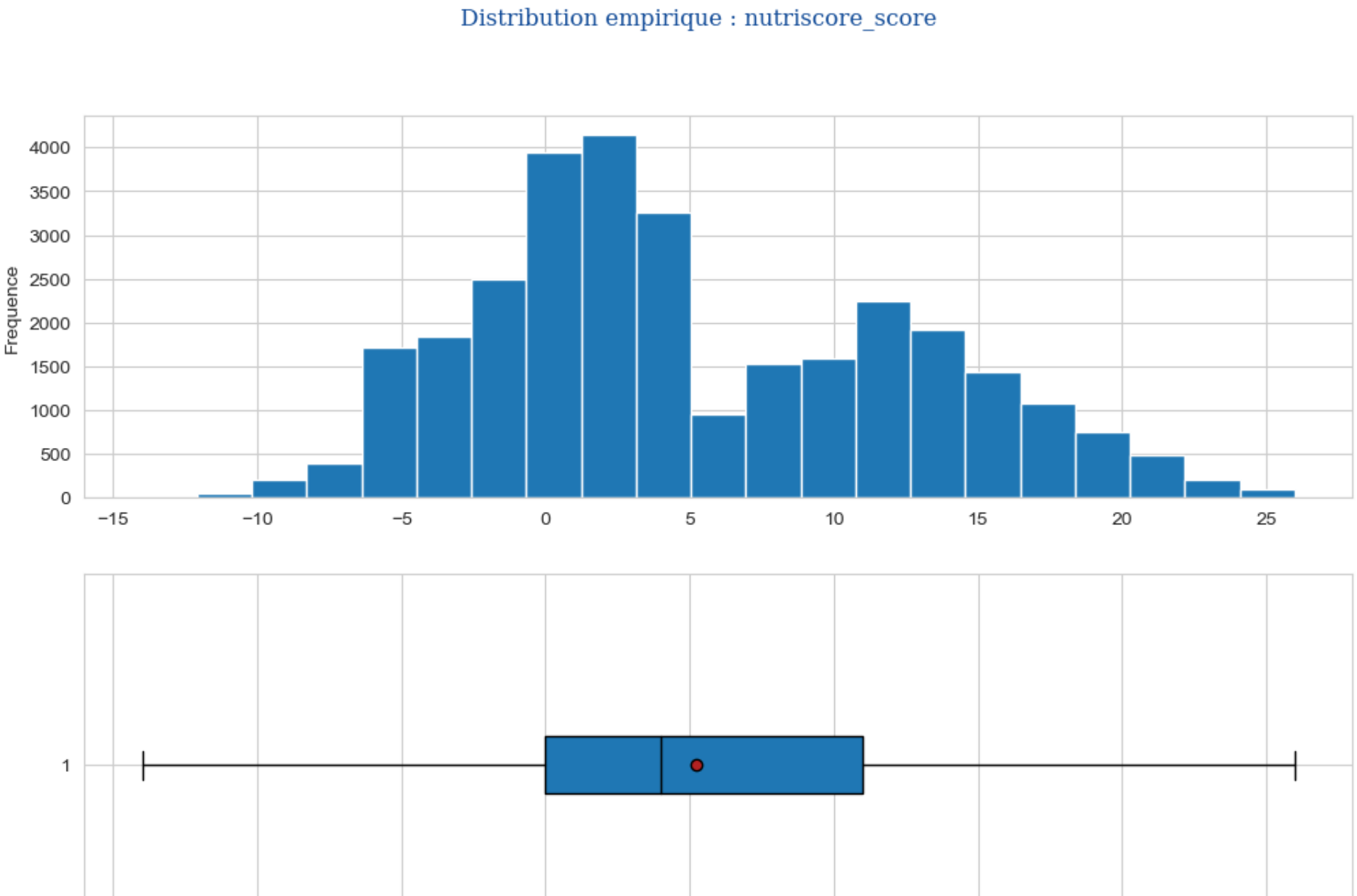
Nous constatons une ***intensification*** des modifications à a partir de 2015.

Les créations connaissent une croissance sur la meme période. Cependant, elles ***chutent*** en 2017.

Repartition du nutrigrade



Distribution du nutriscore



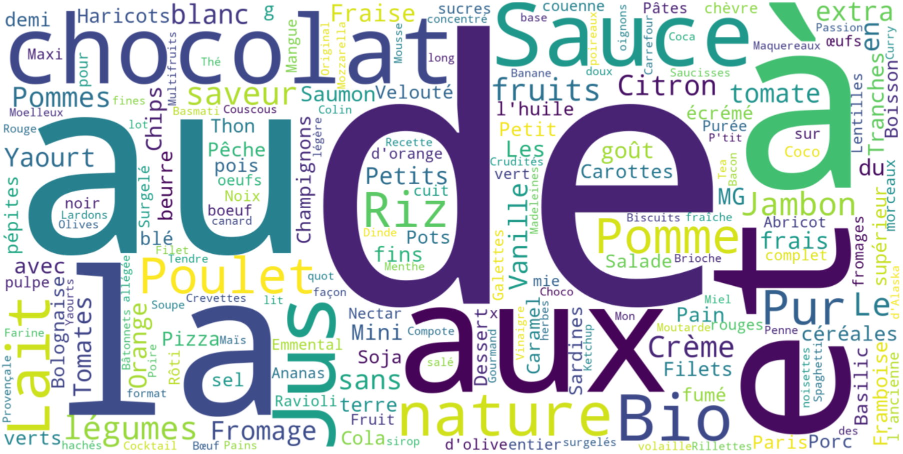
Test de normalité

```
varList=['energy_100g', 'fat_100g', 'saturated-fat_100g', 'carbohydrates_100g',
         'sugars_100g','fiber_100g', 'proteins_100g', 'salt_100g', 'sodium_100g']

for var in varList:
    test_AndersonDarling(dfood, var)
```

```
Distribution normale energy_100g : False
Distribution normale fat_100g : False
Distribution normale saturated-fat_100g : False
Distribution normale carbohydrates_100g : False
Distribution normale sugars_100g : False
Distribution normale fiber_100g : False
Distribution normale proteins_100g : False
Distribution normale salt_100g : False
Distribution normale sodium_100g : False
```

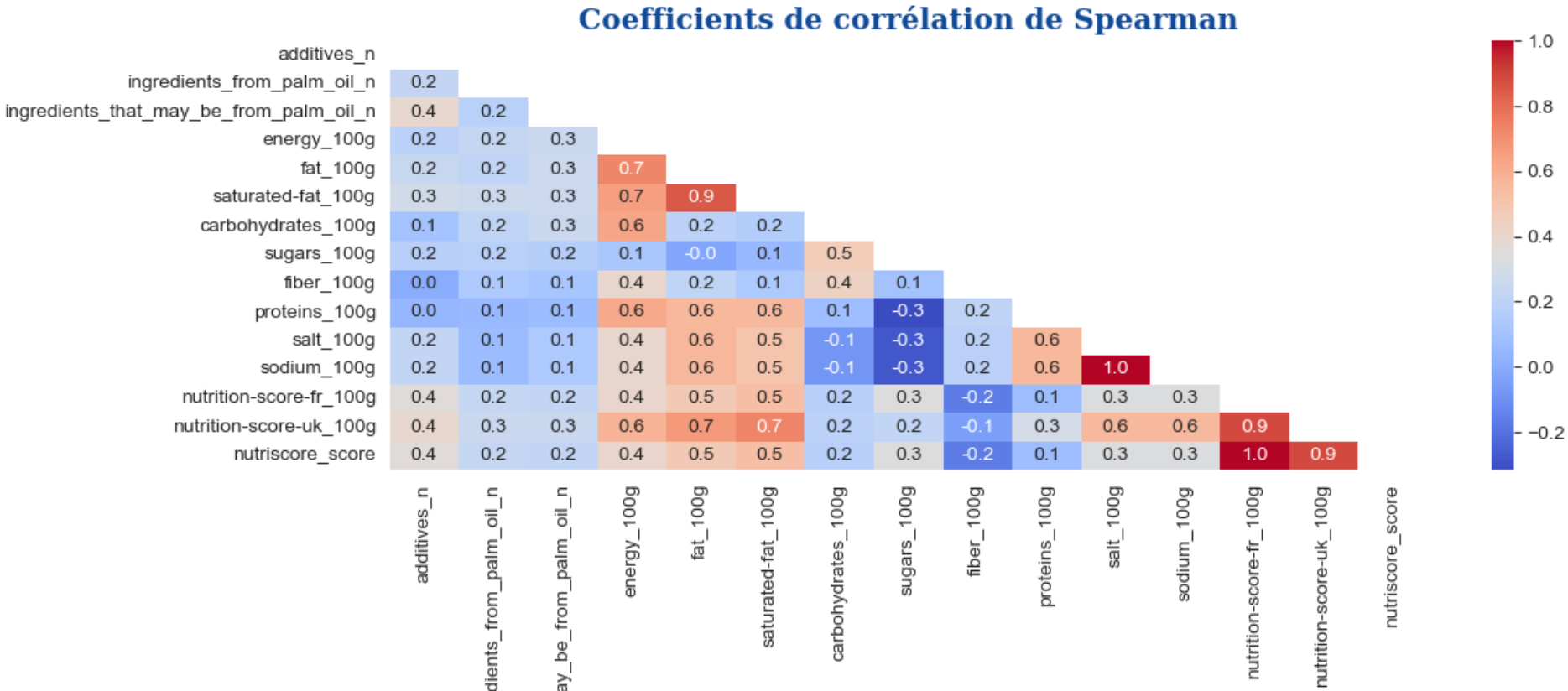
Produits


demi Haricots blanc g Fraise sucres concentré couenne base Pâtes chèvre extra
chocolat Sauce à de au la aux et
Maxi Thé pour Multifruits Mangue Original Mozzarella Mousse long oignons poireaux Carrefour Coca Passion Maquereaux œufs en Curry
Pommes fines Chips saveur Saumon Velouté Banane doux fruits Citron Saucisses tomate Lentilles Tranches Boisson
Moelleux Basmati Couscous Colin l'huile écrémé
Rouge lot Thon légère Petit Purée P'tit sur
Yaourt Pêche Recette d'orange Les Crudités goût Coco
pois oeufs Champignons Petits vert Carottes Tea Bacon du morceaux
pépites noir Lardons Olives Surgelé beurre Noix boeuf canard cuit Riz Vanille Madeleines MG Jambon
Biscuits fraîche Abricot fromages supérieur
blé Dinde Pots Pomme frais
Filet Tendre fins Menthe Salade Brioche complet
avec pulpe Poulet quot Galettes mie Choco
Bolognaise Yaourts allégée façon Miel Pur Le d'Alaska
Lait Farine Tomates Soupe Crevettes lit Nectar Mon Pain
Provençale Bâtonnets Orange Poire Rôti Pizza Maïs Emmental Mini Compote Dessert Gourmand Vinaigre Caramel herbes Sardines Ketchup Moutarde rouges Penne céréales
Soja salé Crème noisettes Spaghetti Basilic Framboise l'ancienne
format sel Ananas Jus sans Filets
Ravioli terre fumé
verts légumes Fromage Fruit nature Bio
hachés Cocktail Bœuf Pains Cola sirop d'olive entier surgelés volaille Rillettes Paris Porc des

les groupes de produits : pnns_groups_1

les groupes de produits : pnns_groups_2

Matrice de corrélation Spearman

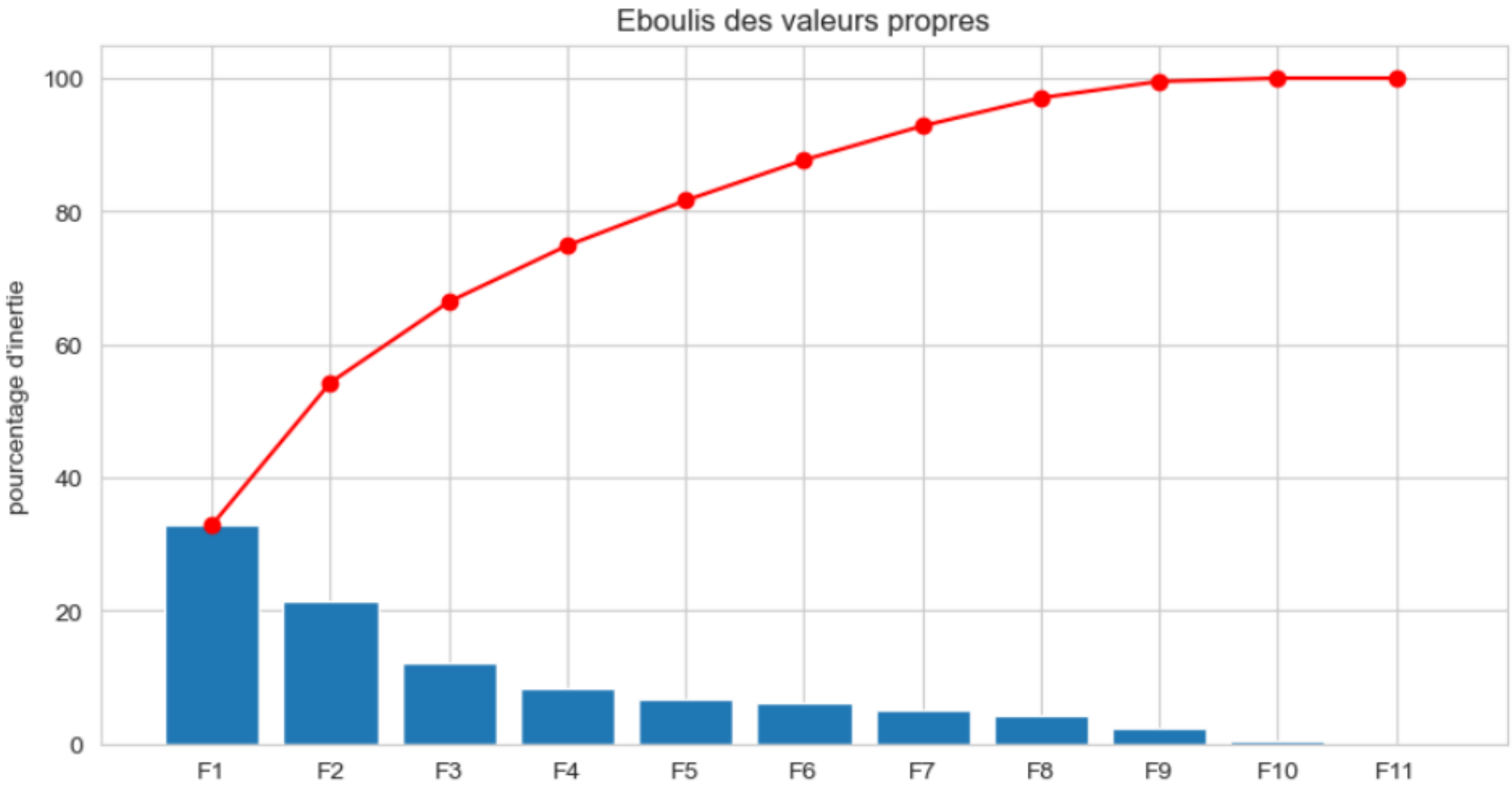


Analyse de la variance

	beverages	cereals and potatoes	composite foods	fat and sauces	fish meat eggs	fruits and vegetables	milk and dairy products	salty snacks	sugary snacks
beverages	1.000000	0.000000	0.000000	0.111397	0.000001	0.000000	0.000000	0.000000	0.000000
cereals and potatoes	0.000000	1.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
composite foods	0.000000	0.000000	1.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
fat and sauces	0.111397	0.000000	0.000000	1.000000	0.000000	0.000000	0.000000	0.000000	0.000000
fish meat eggs	0.000001	0.000000	0.000000	0.000000	1.000000	0.000000	0.000000	0.000000	0.000000
fruits and vegetables	0.000000	0.000000	0.000000	0.000000	0.000000	1.000000	0.000000	0.000000	0.000000
milk and dairy products	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	1.000000	0.000000	0.000000
salty snacks	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	1.000000	0.000000
sugary snacks	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	1.000000

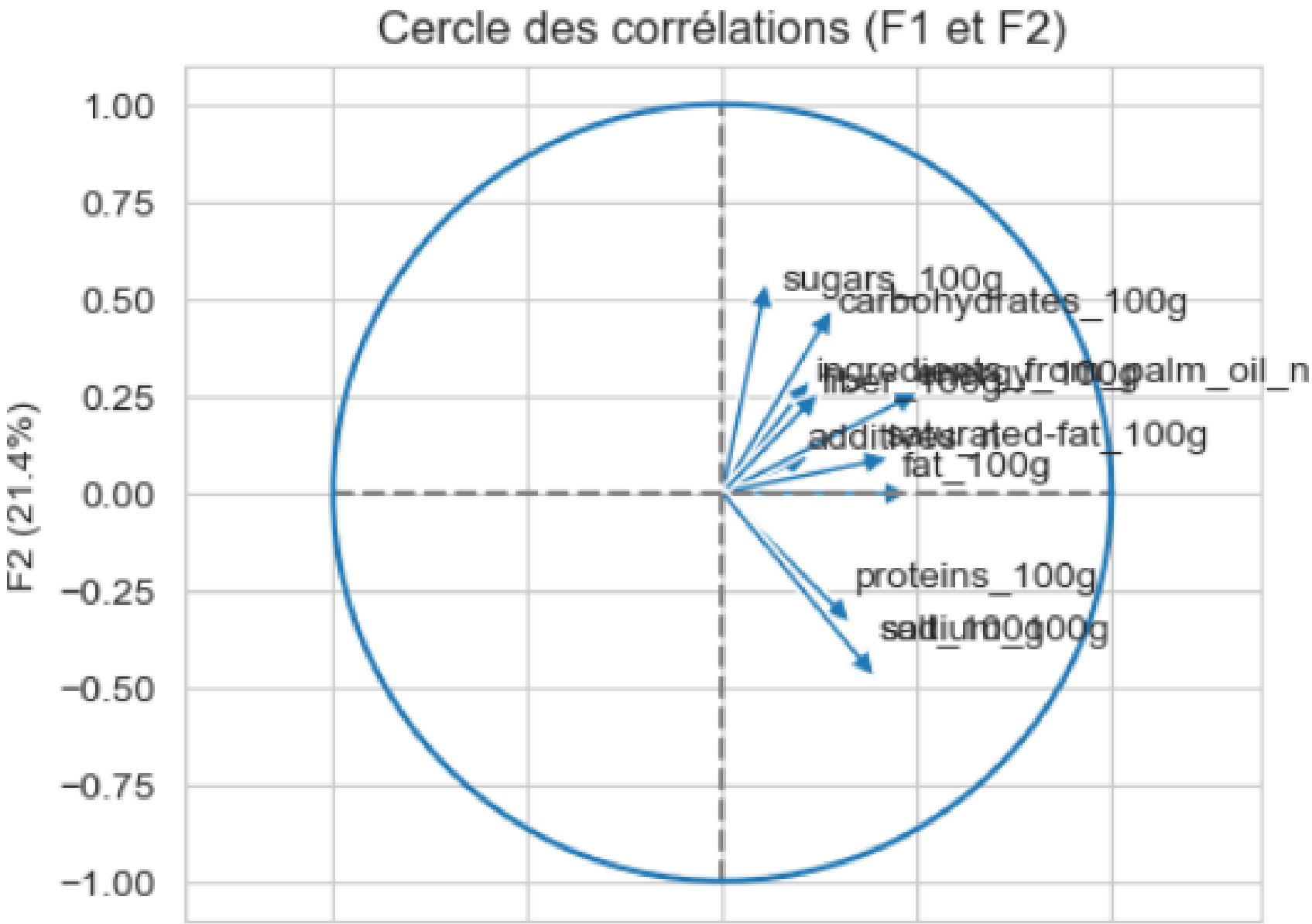
Le test ***Post Hoc (bonferroni)*** permet de dire que les groupes ***fat and sauces*** et ***beverages*** sont presques identiques et diffèrent, cependant, des autres groupes.

Eboulis des valeurs propres

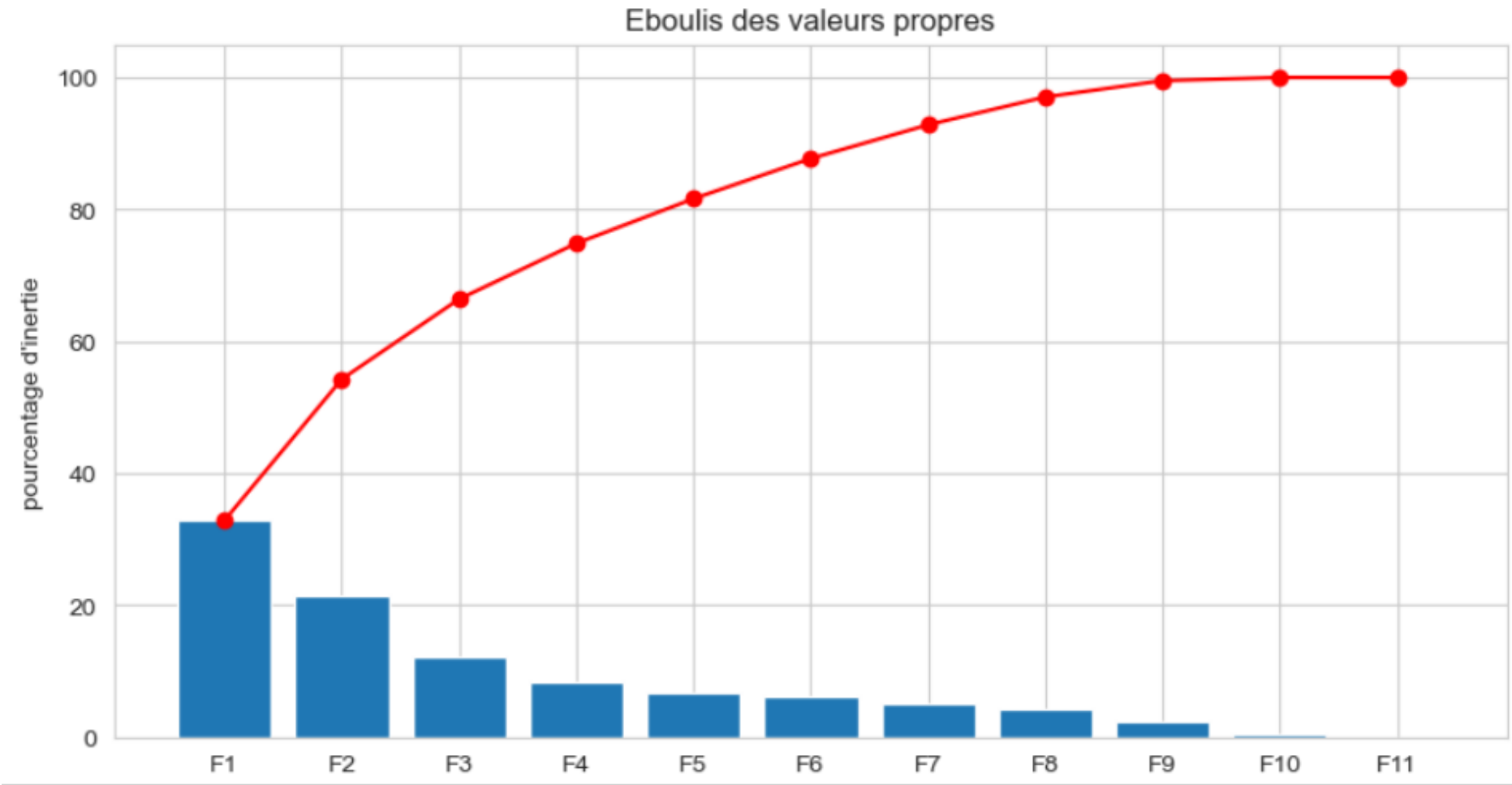


Avec les proportions de variance expliquée, nous notons que les ***huits(8) premiers facteurs restituent 97% de la variabilité***.

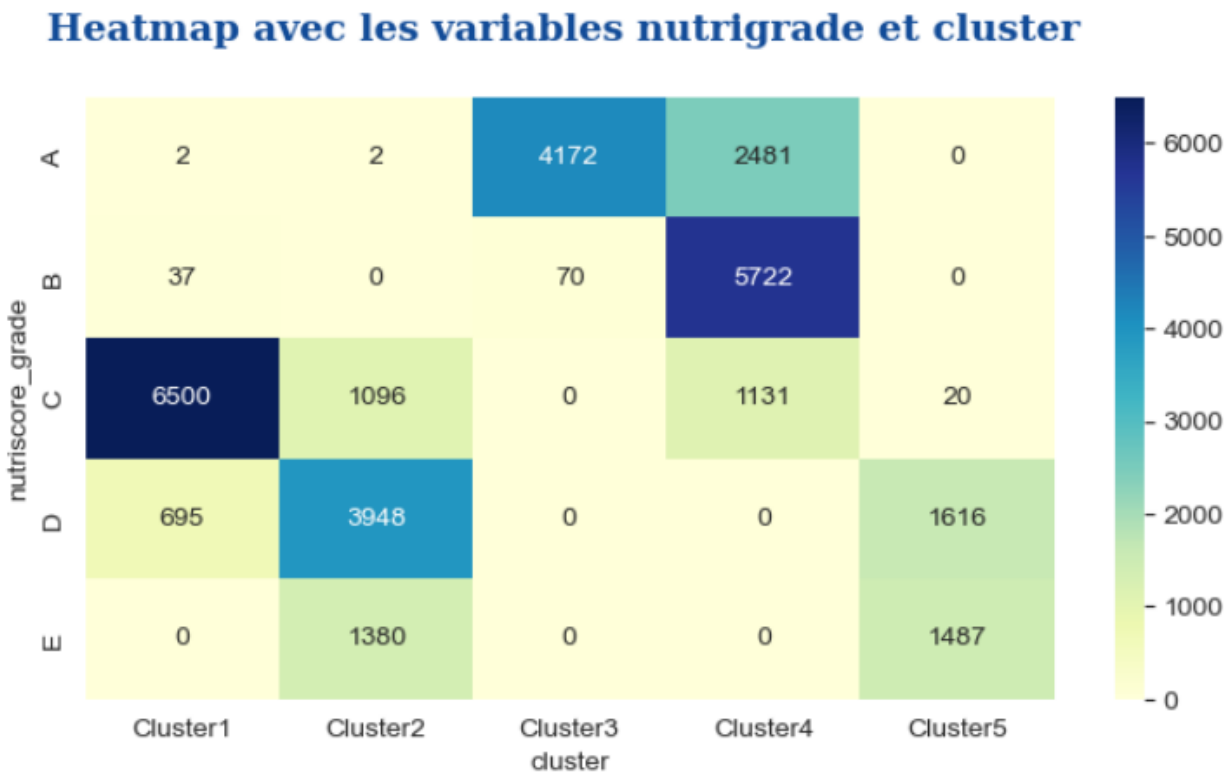
Graphique des variables



K-Means : Nombre de k ?



clusters et les nutrigrades ?



- Le **cluster 3** est associé au **nutrigrade A**
- Le **cluster 4** est associé au **nutrigrade B**
- Le **cluster 1** est associé au **nutrigrade C**
- Les **cluster 2 & 5** sont associés aux **nutrigrade D & E**

Plan de la présentation

Idée d'Application & Conclusion

- ☞ Nutrigrade
- ☞ **Variables d'entrées** : description des produits, nutriments ;
- ☞ Algorithme K-means (clustering, algorihme non supervisé) fournira le nutrigrade du produit.

MERCI POUR VOTRE AIMABLE ATTENTION