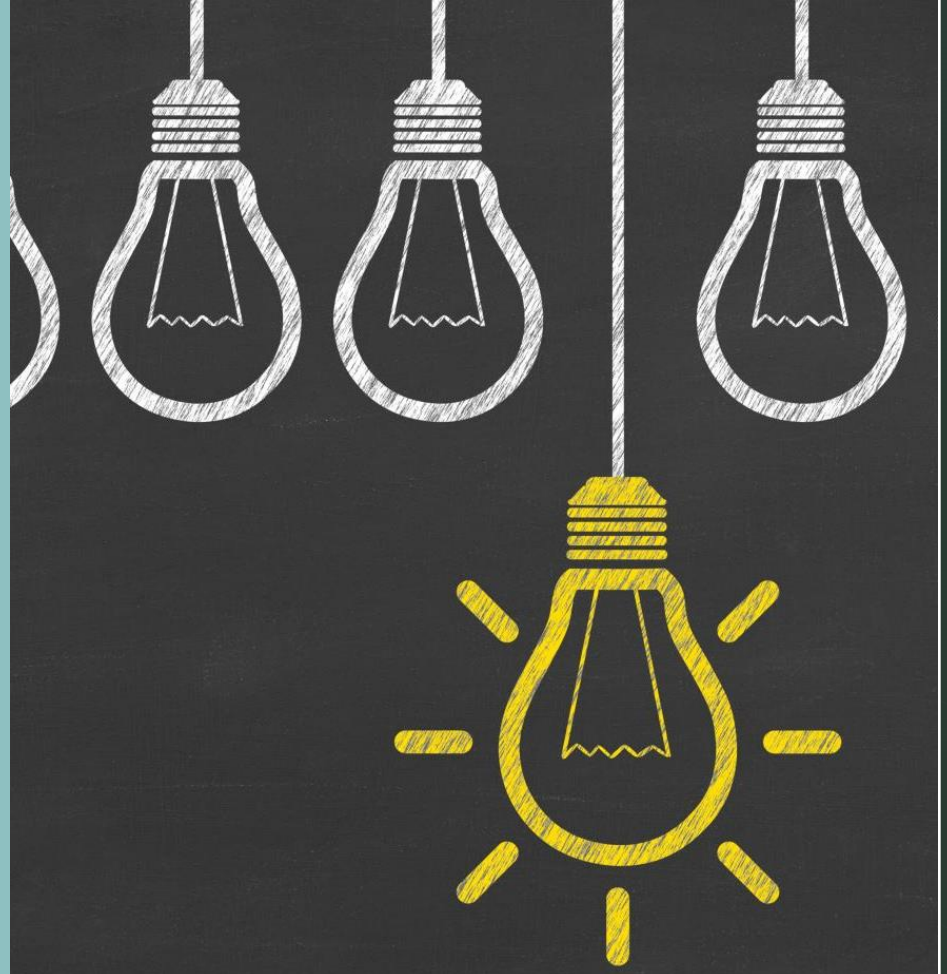
Projet n°3: Conception d'une application au service de la santé publique.

Présenter par: Maidoumi Mohamed
Date: 15/08/2021



- *Sommaire:*

I-Introduction et présentation du jeu de données

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

Objectif du projet:

Proposer une idée pour le calcul du nutriscore en 3 étapes: nettoyage du données, une analyse exploratoire des données et la proposition d'un cadre générale pour l'implémentation d'un calcul du nutriscore.

Environnement de développement:

- Python 3.6
- Windows 10
- Environnement virtuel
- Librairie: Pandas, Numpy, Matplotlib, seaborn,sklearn..

Présentation du Data set:

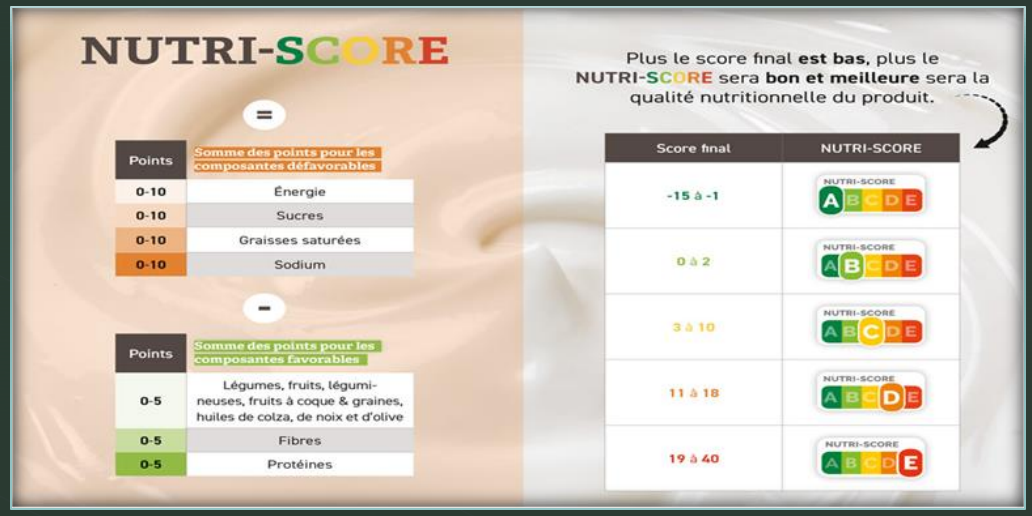
Le codage du fichier est Unicode UTF-8. Le caractère qui sépare les champs est la tabulation et l'architecture des données est la suivante (2.6 Go, `Index: 1911496, columns: 186`):

- Lignes: les produits ajouté par les bénévoles dans l'application Open Food.
- Colonnes se décompose: informations génerales (date de création, date de modification, url image…), étiquettes, ingredients(additifs, nb_additives...), données diverses et variées, nutriments(sodium,proteines,…).

- *Idée d'application:*

Idée d'application:

- Calcule du nutriscore à partir de données réduits présents dans la base de données (nutriments, catégories, …).
- Le nutriscore permets de valoriser les produits avec une bonne valeurs nutritionnelles.
- La note du nutriscore varie entre la note A comme la meilleure catégorie et E comme la moins bonne.



Idée d'application: calcul du nutriscore demande 3 types de features:

- 1- Nutriments défavorables: sucre, Graisse saturés, énergie, sodium.
- 2- Nutriments favorable: fibres, fruits légumes, protiéne.
- 3- Exceptions: café, thé , boisson alcolisées….
- Explorons la disponibilité et la qualité / équivalence de ces facteurs dans notre base de données.

Nettoyage de données:

Nettoyage de données/ Correction de types:

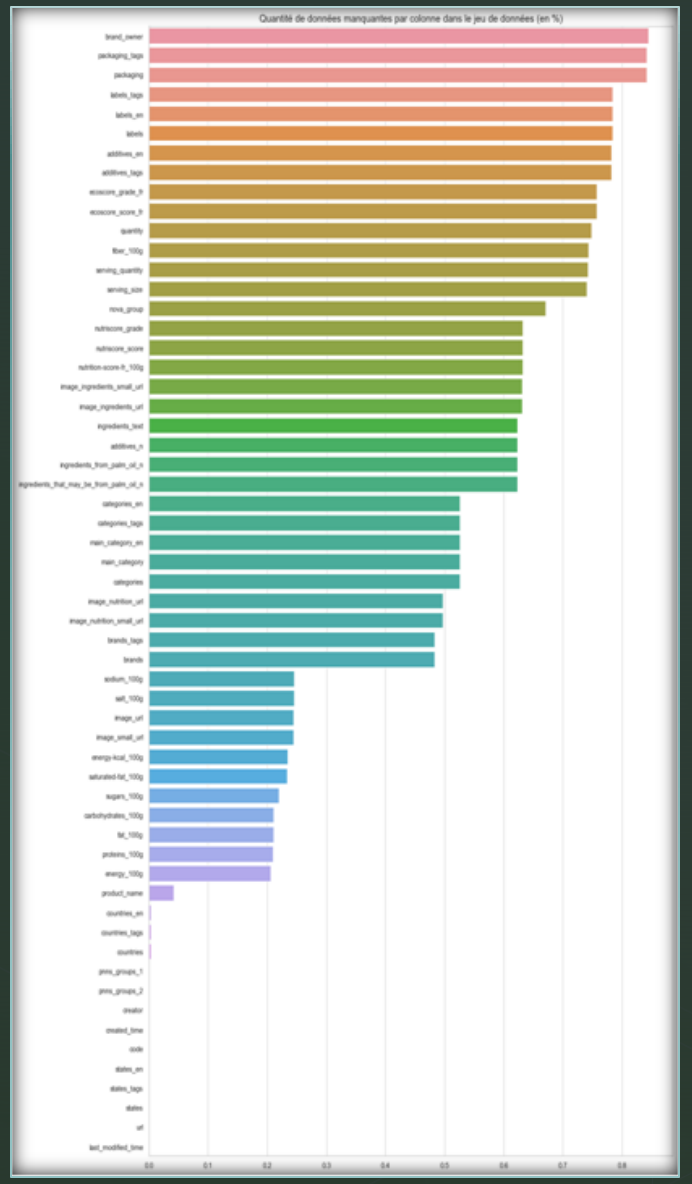
- **Correction des types / format des dates:**
- **- date de création: de objet à datetime.**
- **- date de modification: de int64 à datetime.**

```
# explorer datetime de création ??
data['created_time']=pd.to_datetime(data['created_t'],unit='s')
data['last_modified_time']=pd.to_datetime(data['last_modified_t'],unit='s')
#data['created_datetime_']=pd.to_datetime(data['created_datetime'])
#data['last_modified_datetime_']=pd.to_datetime(data['last_modified_datetime'])
data.drop(['created_t','last_modified_t','created_datetime','last_modified_datetime'],axis=1,inplace=True)
```

Suppresion des colonnes avec beacoup de NaN:

- Pays d'origine: France uniquement.
- Suppression des exceptions café, thé, boisson alcolisées dans la colonne
- Traitement des NaN:

–Suppression des colonnes avec plus de 85% de NaN.



Suppression des valeurs abérantes :

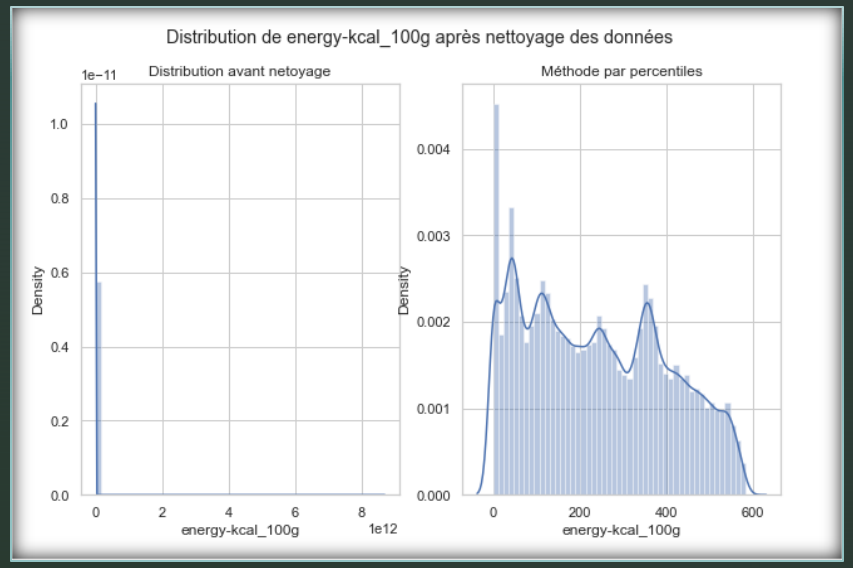
- La suppression de 1% des valeurs abérantes nous permet d'avoir des résultats coerrents dans notre analyse univariée.
- Résultat de nettoyage: (676612 lignes, 56 colonnes).

Supression des outlier.

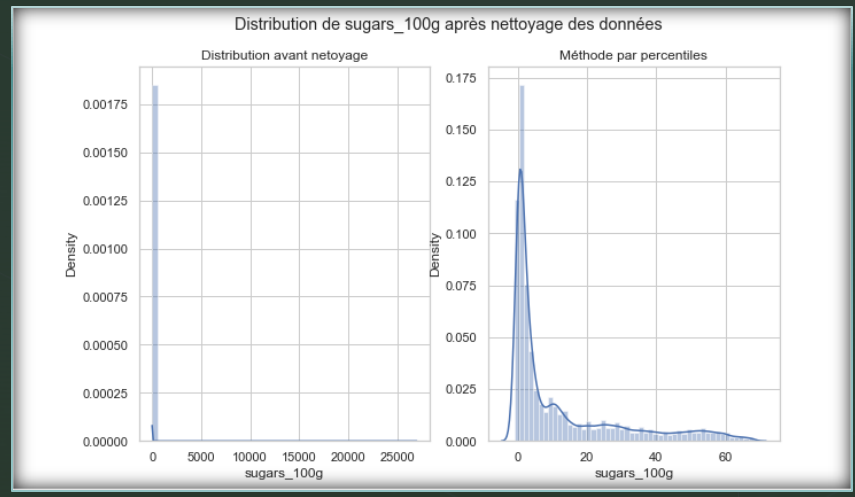
In [82]:

```
def outliers_percentile(data_frame):
    #for column in data_frame.select_dtypes(include=['int32','float64'])
    for column in float_ind: # supprimer les outlier selon les centiles
        data_frame.loc[data_frame[column]>data_frame[column].quantile(0.99)]=np.nan
        #data_frame.loc[data_[column] <data_[column].quantile(0.005)]=np.nan
        #data_frame.loc[data_[column] < 0]=np.nan
    return data_frame
data_outliers= outliers_percentile(data_)
```

Nutriments défavorables/analyse univariée:

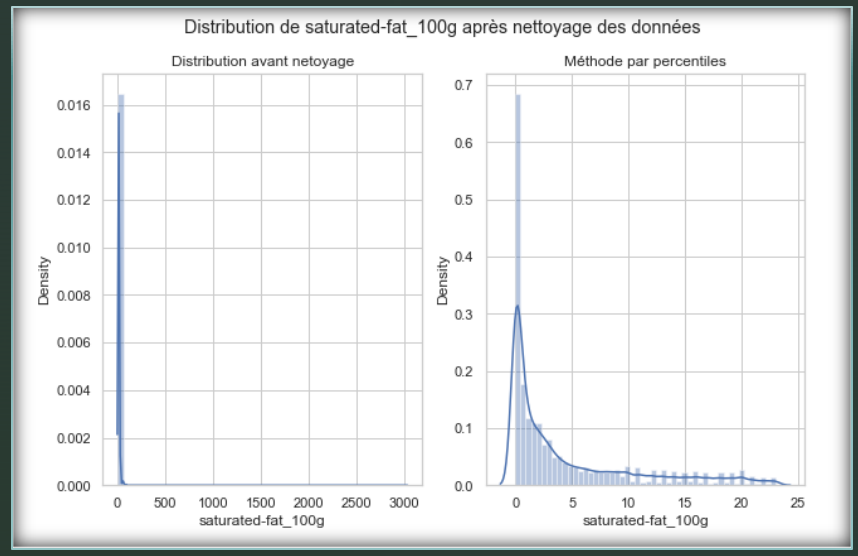


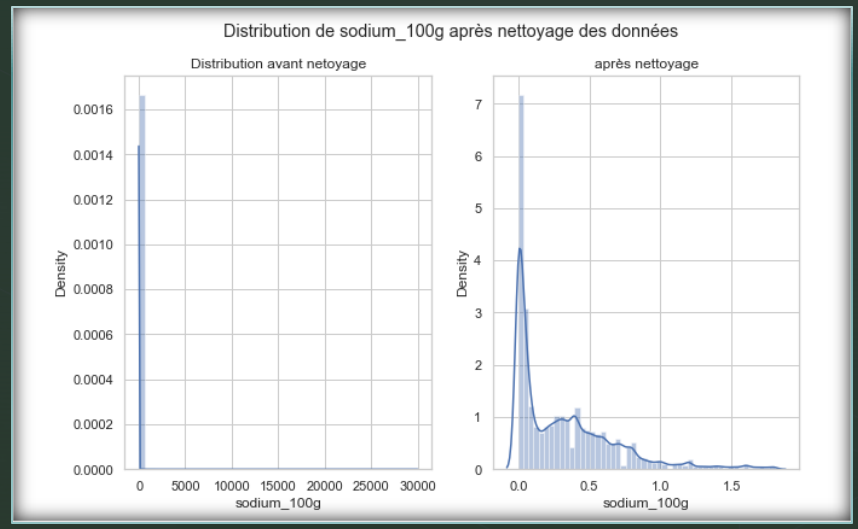
- *Sucre,energie_100g (distribution avant et après le nettoyage de données).*



Nutriments défavorables/analyse univariée:

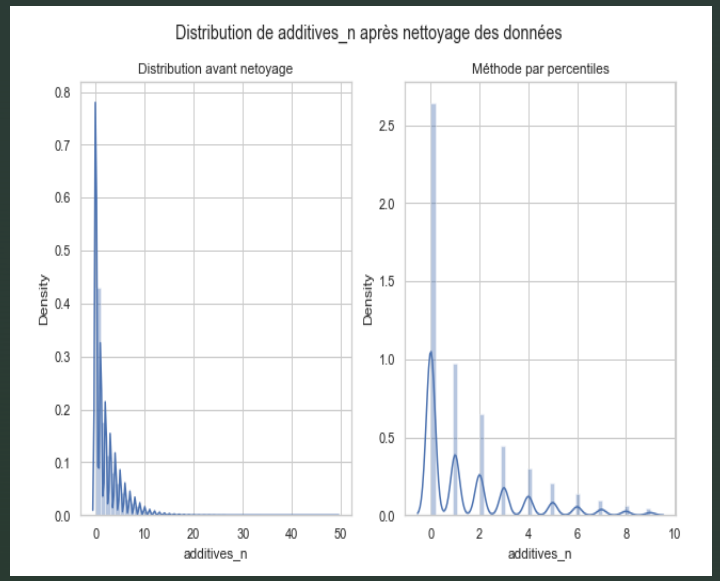
- *Graisse saturée, sodium (distribution avant et après le nettoyage de données).*

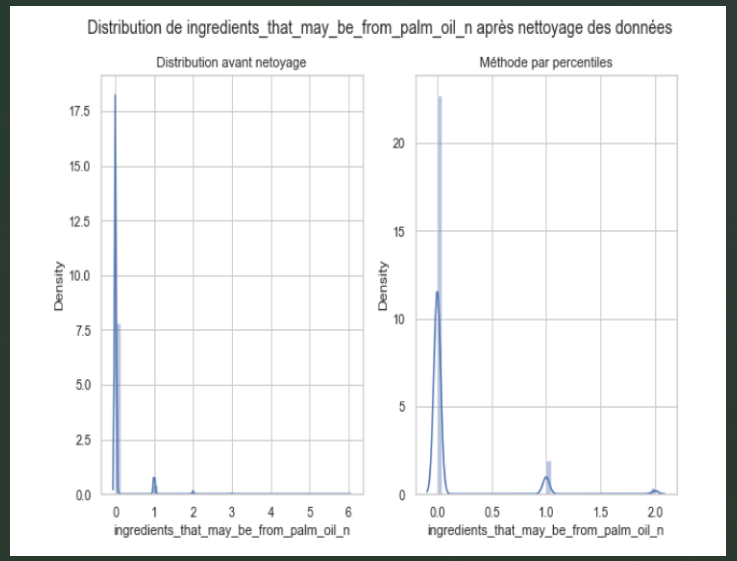




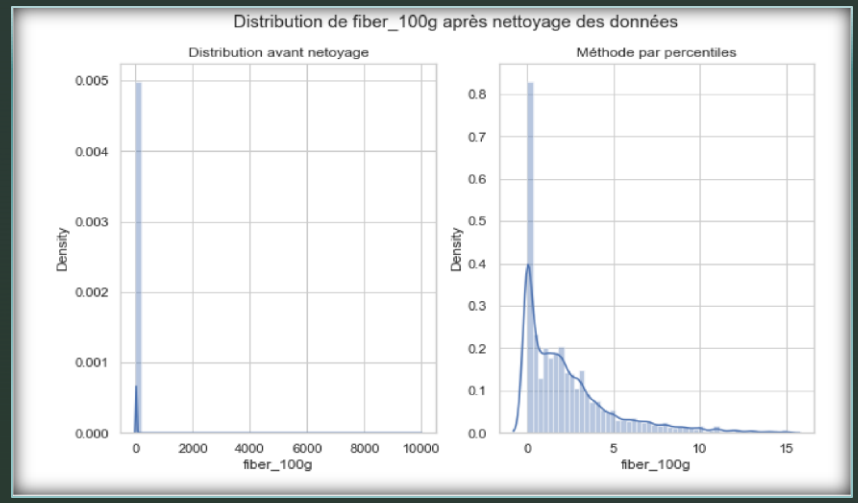
Nutriments défavorables/analyse univariée:

Produit avec huile de palme et nombre d'additives par produit sont des variables discrètes (distribution avant et après le nettoyage de données).

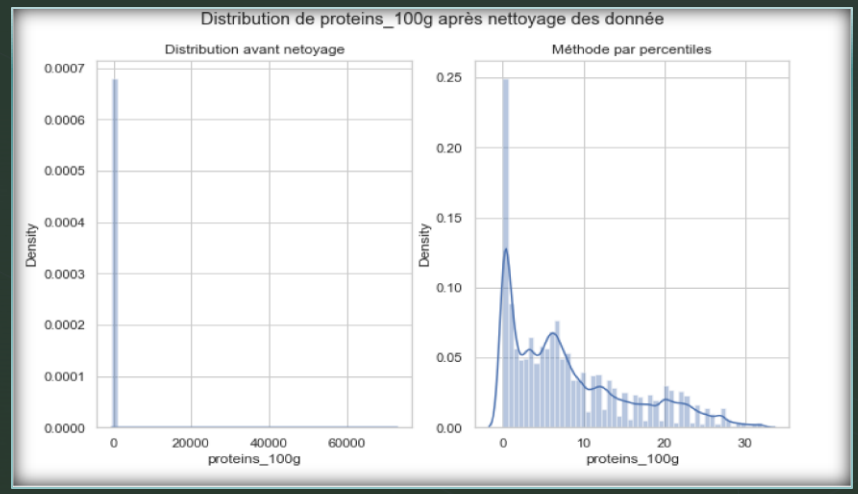




Nutriment favorable/analyse univariée:



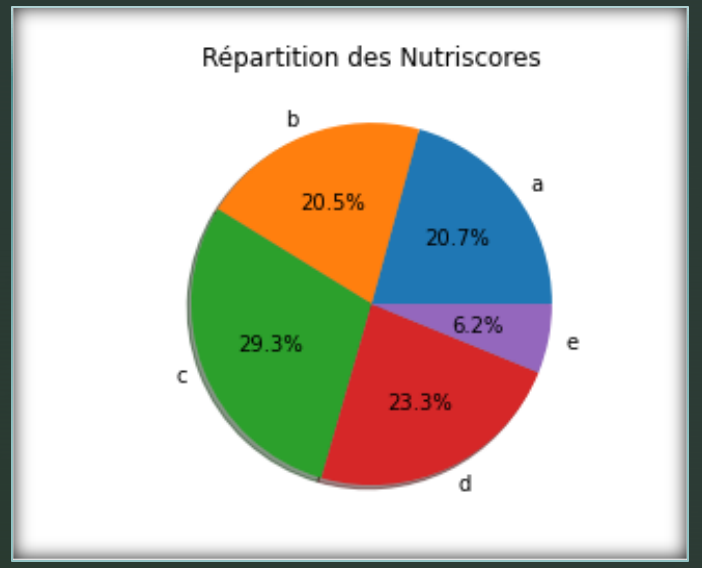
- *Fiber, proteine (distribution avant et après le nettoyage de données).*

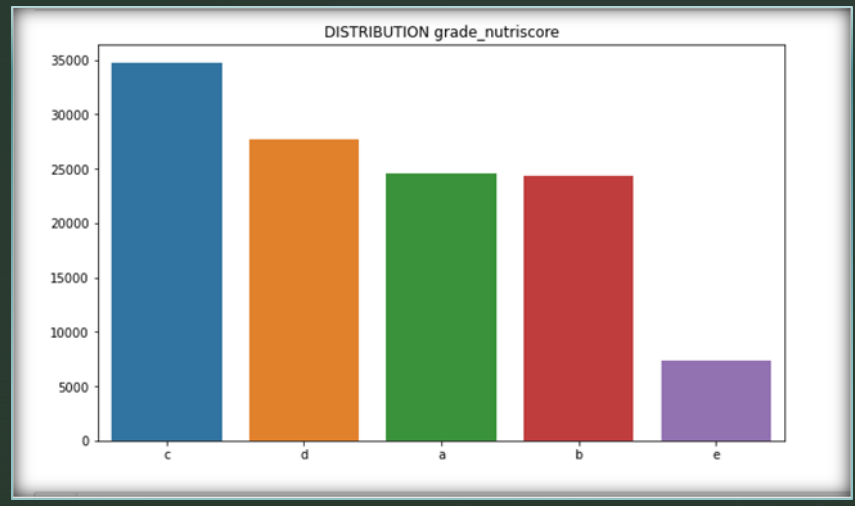


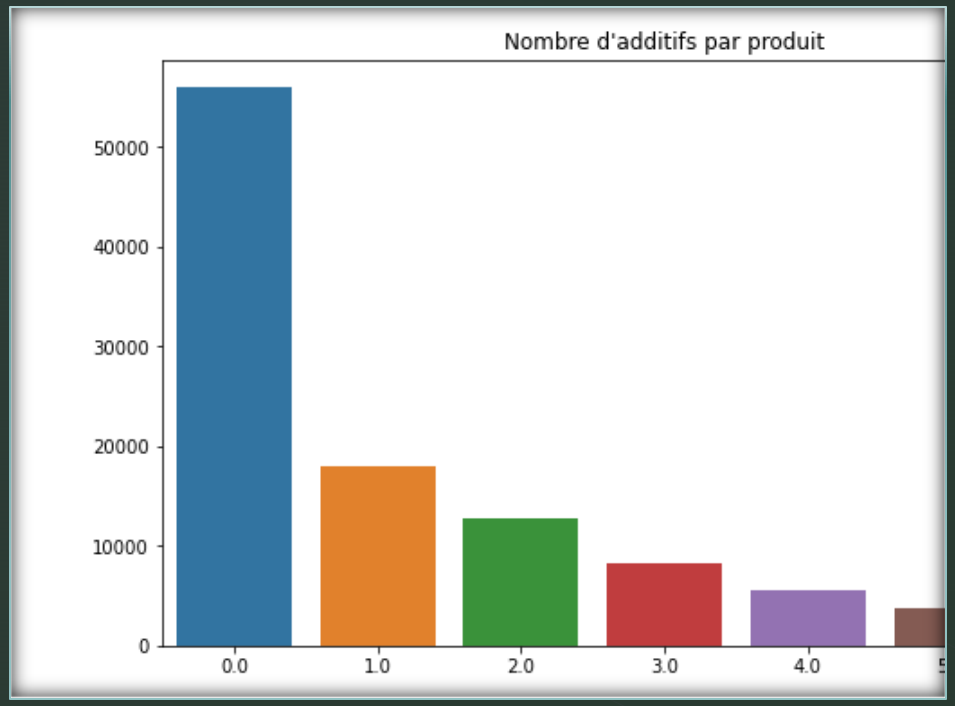
Exploration de données:

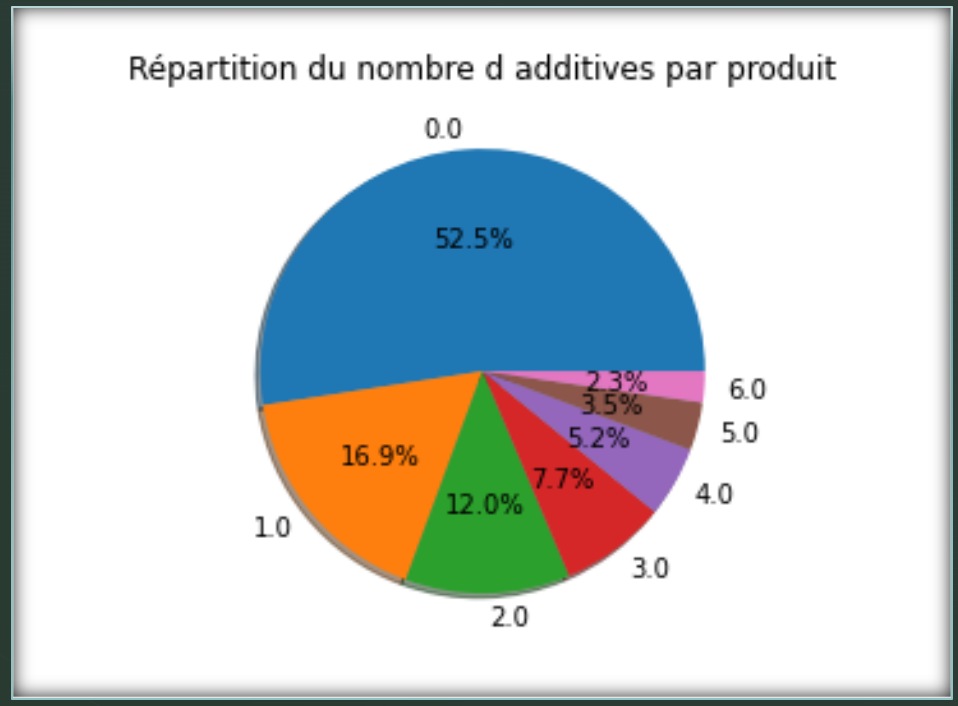
Analyse univariée/ grade nutriscore:

- Les produits de grade *c* i.e un nutriscore entre 3 et 10 sont les plus présents dans notre base de données suivit par les produits de grade *d, puis a*.





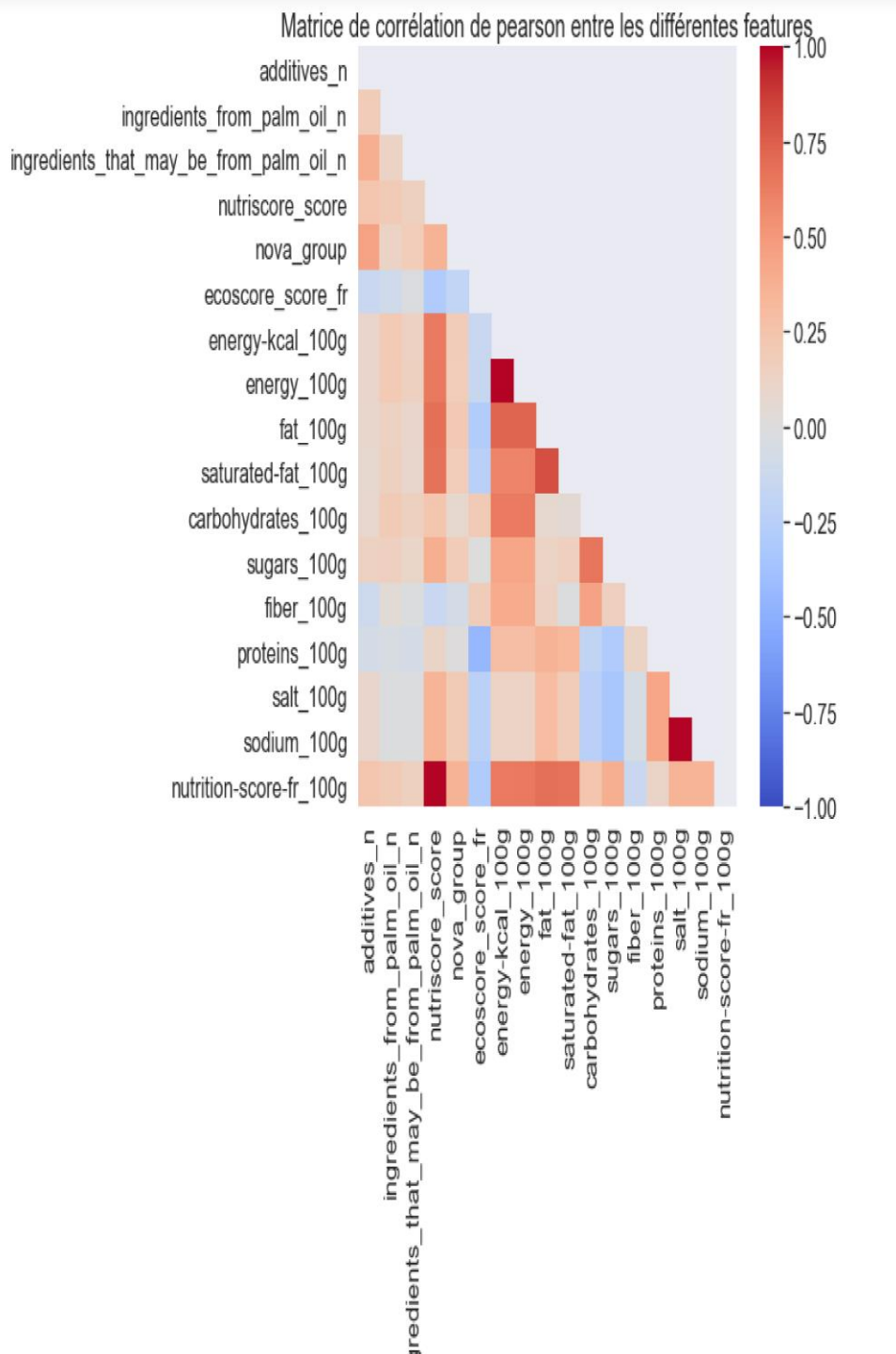




Analyse univariée/ nombre additives par produit

Exploration des données: analyse multivariée/Matrice de corrélation:

- *nutrition-score-fr_100g représente:*
  - *une forte corrélation positiv avec(autour de 75%): 1-saturated_fat_100g, 2-fat_100g, 3-energy_100g, 4-energy_kcal-100g.*
  - *Une corrélation moyenne avec(autour de 50%) 5- sodium_100g, 6-salt_100g, 7- sugars_100g, 8-nova_group.*
  - *Négativement corrélé avec(autour de -20%): 6-fiber_100g, 7- ecoscore_score_fr.*
  - *Corrélation faible avec(25%): 8- carbohydrates_100g, 9- ingredients_that_may_be_from_palm_oil_n, 10- additives_n, 11- ingredients_from_palm_oil_n, 12- proteins_100g.*
  - *Remarque: des corrélation quasiment parfaite avec nutriscore_score et ecoscore_score_fr représente deux variables de la même grandeur nutriscore.*
  - *Quelques analyse de la corrélation des variable corrélées avec nutrition-score-fr_100g (+50%):*
    - *saturated_fat_100g représente:*
      - *une forte corrélation (+75%) avec :1- fat_100g.*
      - *Une corrélation moyenne (+50%) avec: 2-energy_kcal-100g, 3-energy_100g.*
    - *energy_100g représente:*
      - *Parfaite corrélation (100%) avec energy_kcal-100g.*
    - *Nova_group représente:*
      - *correlation moyenne (+50%) avec additives_n.*
      - *Une faible corrélation (+25%): ingredients_from_palm_oil_n, ingredients_that_may_be_from_palm_oil_n,*
    - *sugars_100g repésente:*
      - *forte correlation(75%) avec carbohydrates_100g.*
      - *Une corrélation moyenne (50%) avec energy_100g et energy_kcal-100g.*
    - *sodium_100g représente:*
      - *correlation parfaite (100%) avec salt_100g.*
      - *Une corrélation moyenne (50%) avec proteine.*



Exploration des données/ analyse multivariée/Matrice de corrélation (2):

- D'après les tendances que nous avons découvertes dans la matrice de corrélations on cherche les variables les plus corrélées avec le "*nutrition-score-fr_100g"* et les moins corrélées entre eux:
  - Si deux variables sont bien corrélé avec "*nutrition-score-fr_100g" et en même temps* elles sont fortement corrélées entre eux, nous choisissons la variable qui impacte le plus la variable cible "*nutrition-score-fr_100g"* , dans le cas où ces variables sont équivalent on choisit celle qui représente moins de valeurs manquantes, dans le cas écheant on favorise la variable qui représente plus de visibilité en terme d'unité de calcul, un impacte plus inuitive sur le nutriscore.....
  - Exemple 1: entre *energy_kcal-100g et energy_100g qui sont parfaitement corrélé et très proche en terme d'impacte sur "nutrition-score-fr_100g" et on remarque un légère avantage pour "energy_100g" en terme de valeurs manquantes. Finalement on choisit "energy_kcal-100g" qui présente plus de lisibilité d'unité de calcul "kcal"*.
  - Exemple 2: les deux variables "*saturated_fat_100g" et "fat_100g" ont le même impacte sur le "nutrition-score-fr_100g" et on remarque une forte corrélation (0.88) entre eux et la même niveau de valeurs manquantes. On choisit pour notre sélection de variable "saturated_fat_100g" la variable qui représente la graisse saturée et qui logiquement impacte négativement le grade nutritionnelle.*
- A partir des régles précédemment citées notre première sélection de variables est la suivente: 1-*saturated_fat_100g, 2-energy_kcal-100g, 3-additives_n 4- sugars_100g. 5-fiber_100g, 6-ingredients_that_may_be_from_palm_oil_n.*

Analyse mutivariée/ Test du Chi 2 - Indépendance des variables

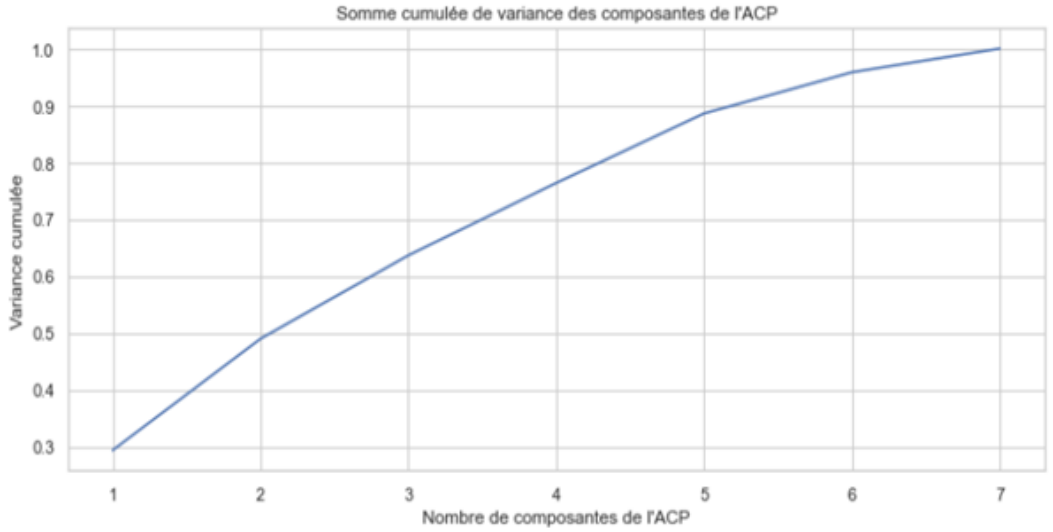
```
H0 non rejetée car p = 1.0 >= alpha = 0.03
test d'indépendance nutriscore / additives_n
chi2 : 8206.15760,
p : 8206.15760,
dof : 8206.15760

Variables non indépendantes (H0 rejetée) car p = 0.0 <= alpha = 0.03
test d'indépendance nutriscore / ingredients_from_palm_oil_n
chi2 : 4874.68198,
p : 4874.68198,
dof : 4874.68198

Variables non indépendantes (H0 rejetée) car p = 0.0 <= alpha = 0.03
test d'indépendance nutriscore / ingredients_that_may_be_from_palm_oil_n
chi2 : 2733.19824,
p : 2733.19824,
dof : 2733.19824
```

In [94]:

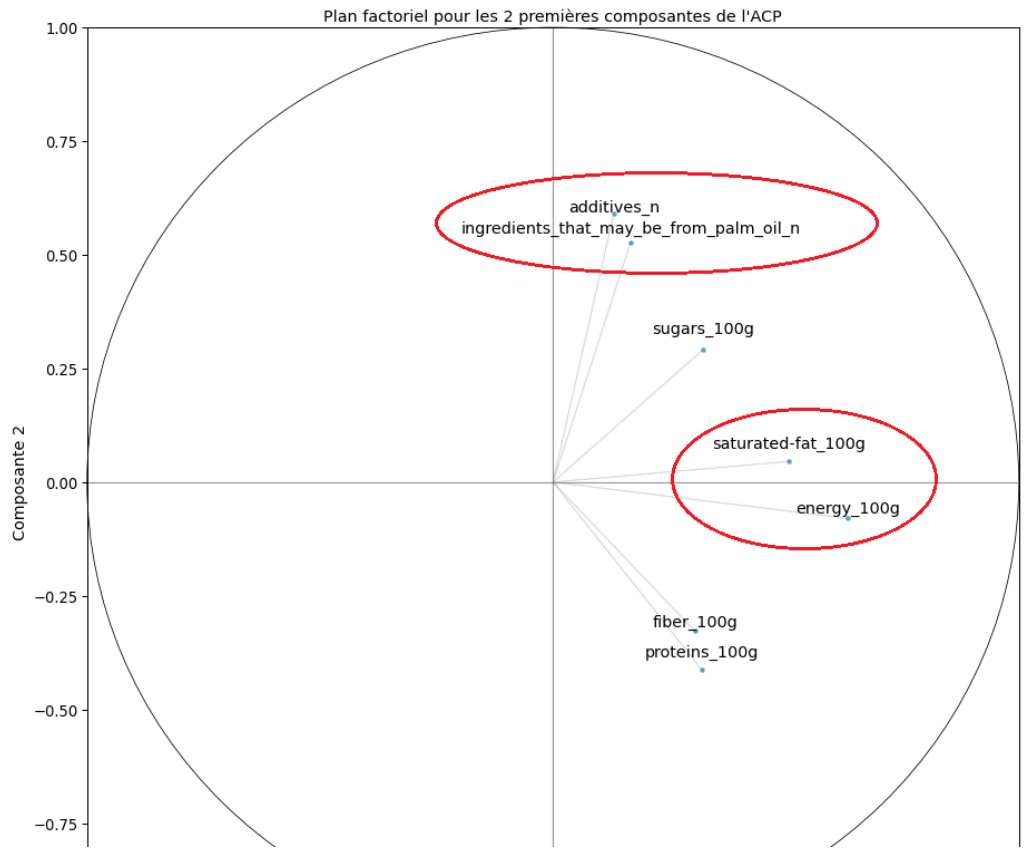
```python
from scipy.stats import chi2_contingency
from scipy.stats import chi2

def test_chi2(serie1, serie2):
    alpha = 0.03   #seuil de test
    #H0 : les variables sont indépendantes

    tab_contingence = pd.crosstab(serie1.array, serie2.array)
    stat_chi2, p, dof, expected_table = chi2_contingency(tab_contingence.values)
    print('chi2 : {0:.5f},\np : {0:.5f},\ndof : {0:.5f}\n'.format(stat_chi2, p, dof))
    critical = chi2.ppf(1-alpha, dof)
    if p <= alpha:
        print('Variables non indépendantes (H0 rejetée) car p = {} <= alpha = {}'.format(p, alpha))
        return False

    else:
        print('H0 non rejetée car p = {} >= alpha = {}'.format(p, alpha))
        return True
```

Reduction de dimension par ACP: Somme cumulée de variance des composantes de l'ACP.

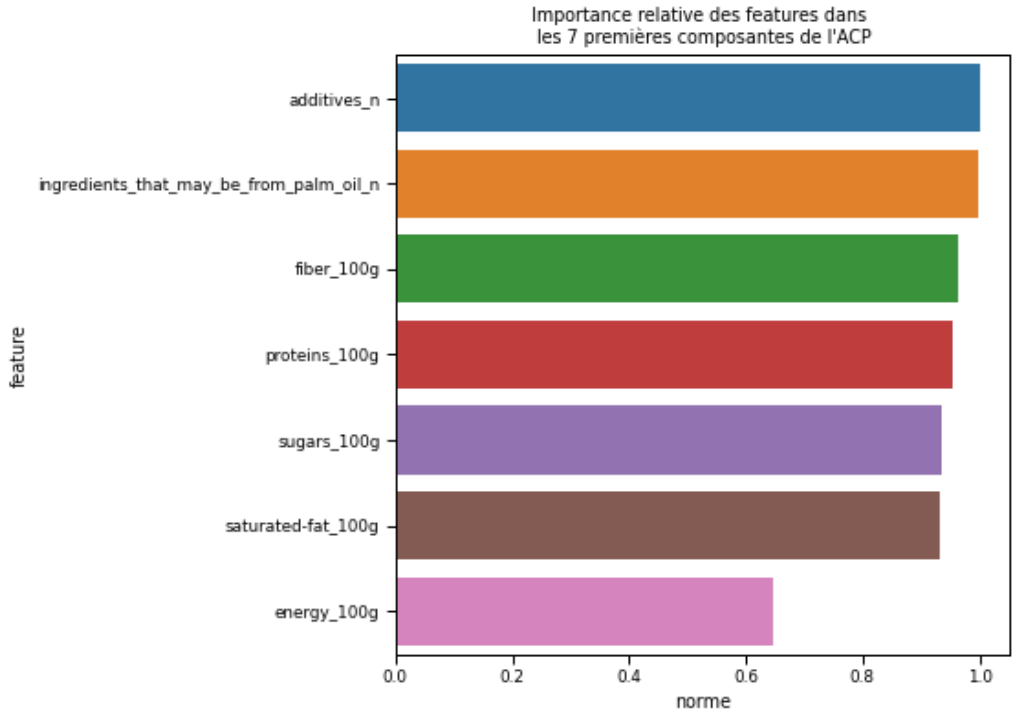
- On remarque qu'à partir de 6 features on a une variance cumulée de plus de 95 %. On pourrait donc réduire notre jeu de données à 6 dimensions.



ACP: Plan factoriel pour les 2 premières composantes de l'ACP



Importance relative des variables dans les 7 premières composantes de l'ACP.



Perspéctives de développement:

- Alimenter d'avantage le data par les colonnes avec beaucoup de valeurs manquantes et qui pourraient impacter significativement le nutriscore comme: fruits-vegetables-nuts_100g, chlorophyl_100g, cocoa_100g....
- Vérifier la qualité de données avant d'alimenter les bases de données: format des variables, éliminer valeurs aberantes... dans notre processus de nettoyage de données on a passé de la dimension (1911496, 184) à (676612, 56).
- Créer une base de données relationnelle pour bien structurer les données au lieu d'empiler les produits sur un grand simple tableau de taille (2.6 Go) cela nous permettera d'optimiser le temps de requêtage et l'accès directe aux valeurs de variables par type d'informations.

Conclusion: