

# Préparez des données pour un organisme de santé publique

...

Fevrier 02, 2023

# Aperçu

L'objectif de ce second projet est:

- Savoir effectuer une analyse multivariée à partir d'un dataset portant sur les données de santé publique (incluant PCA, Anova)
- Interpréter et Présenter les résultats de cette EDA à une audience
- Proposer un smart product qui aura un réel intérêt/impact auprès de la clientèle cible (les agents de Santé Publique France).

# Les phases du projet

## Setup l'environnement

L'installation:

- Python
- Ide (editeur)
- Jupyter notebook
- Environnement virtuel python
- Libraries python

## Collecte des données

L'utilisation de pandas:

- Charger les données issues du fichier CSV
- Creation d'un DataFrame (tableau 2D)

## Nettoyage des données

Le nettoyage des données consiste à:

- Traitement des données invalides.
- Suppression des doublons.
- Remplacement de valeurs
- Detecte et filtre les outliers

## Analyse, visualisations et interpretations.

La finalité du projet consiste

- Proposer un smart produit
- Faire une analyse univariée/multivariée
- Utilisation PCA
- Utilisation de anova



# Python, Jupyter et les librairies

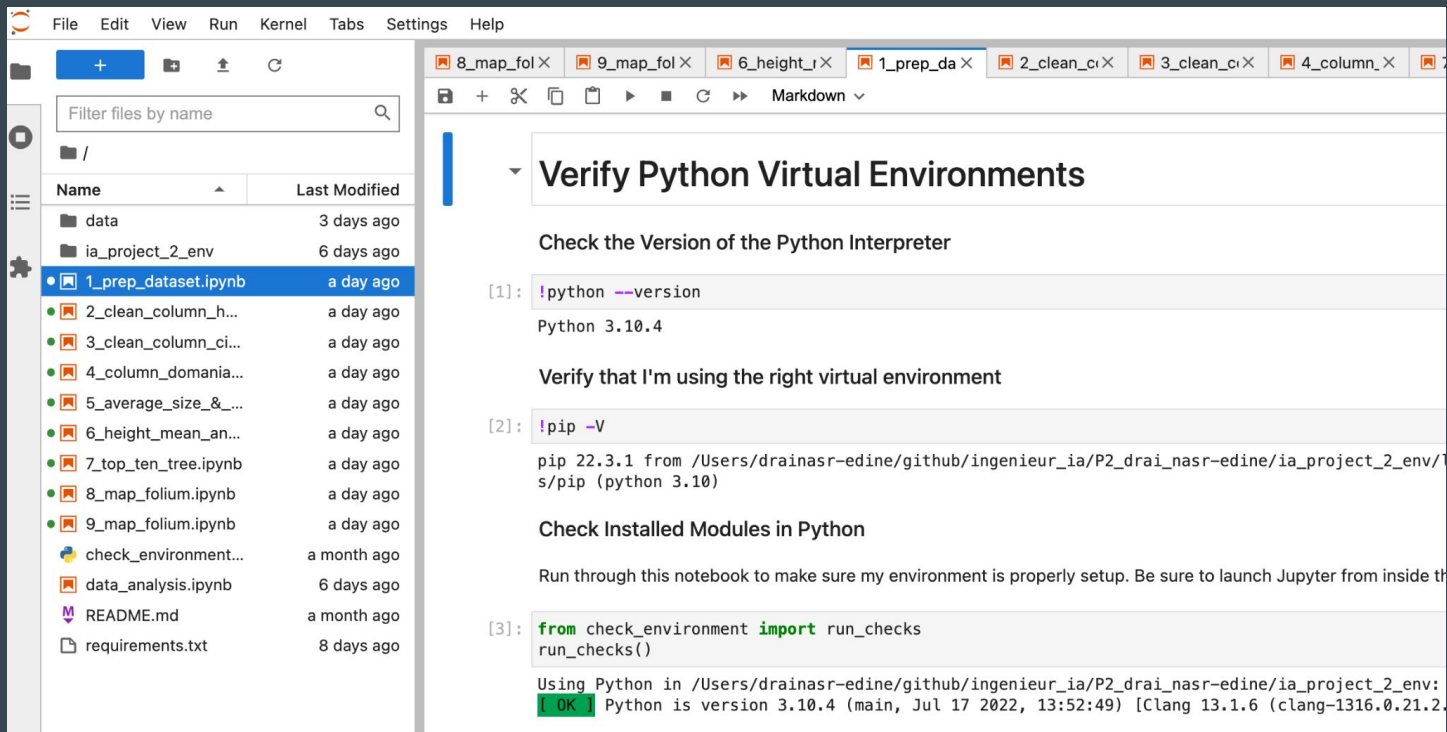


Python étant déjà installé sur mon poste avec une version récente (3.10.4). J'ai fait le choix d'utiliser PIP pour installer les modules nécessaires:

- Jupyterlab
- Pandas
- Matplotlib
- sklearn

---

# Jupyterlab



The screenshot displays the JupyterLab web interface. On the left is a file browser sidebar with a search bar and a list of files and folders. The main area on the right shows an open notebook titled 'Verify Python Virtual Environments'. The notebook contains three code cells:

- Cell [1]:** Checks the Python version using `python --version`, outputting `Python 3.10.4`.
- Cell [2]:** Verifies the virtual environment using `pip -V`, outputting `pip 22.3.1 from /Users/drainasr-edine/github/ingenieur_ia/P2_drai_nasr-edine/ia_project_2_env/.../pip (python 3.10)`.
- Cell [3]:** Checks installed modules using `from check_environment import run_checks; run_checks()`. The output shows the Python path and version: `Using Python in /Users/drainasr-edine/github/ingenieur_ia/P2_drai_nasr-edine/ia_project_2_env: [OK] Python is version 3.10.4 (main, Jul 17 2022, 13:52:49) [Clang 13.1.6 (clang-1316.0.21.2...]`

Jupyter Lab et la nouvelle génération d'interface notebook qui offre d'avantage de fonctionnalités que jupyter notebook classique

# Vérification de l'environnement

# Vérification de l'environnement

## Commande bash dans le notebook:

Jupyter offre la possibilité d'utiliser bash en precedent la commande du symbole '!'

Vérifier python et l'environnement virtuel:

Commande	resultat
<code>!python --version</code>	Python 3.10.4
<code>!pip -V</code>	Chemin du virtual env

```
Check the Version of the Python Interpreter

[1]: !python --version

Python 3.10.4

Verify that I'm using the right virtual environment

[2]: !pip -V

pip 22.3.1 from /Users/drainasr-edine/github/ingenieur_ia/P2_drai_nasr-edine/ia_project_2_env/lib/python3.10/site-packages/pip (python 3.10)

Check Installed Modules in Python

Run through this notebook to make sure my environment is properly setup. Be sure to launch Jupyter from inside the virtual environment.

[3]: from check_environment import run_checks
run_checks()

Using Python in /Users/drainasr-edine/github/ingenieur_ia/P2_drai_nasr-edine/ia_project_2_env:
[ OK ] Python is version 3.10.4 (main, Jul 17 2022, 13:52:49) [Clang 13.1.6 (clang-1316.0.21.2.5)]

[ OK ] jupyterlab
[ OK ] matplotlib
[ OK ] numpy
[ OK ] pandas
[ OK ] seaborn
[ OK ] statsmodels
[ OK ] folium
```

# Verification des dependances

Un script python me permet de vérifier que les dépendances sont bien installés sur mon environnement virtuel

```
# check the requirements
for pkg in requirements:
    try:
        mod = importlib.import_module(pkg)
        print(f"{OK} {mod.__name__}")
    except ImportError:
        print(f"{FAIL} {pkg} not installed.")
```

```
3]: from check_environment import run_checks()
```

Using Python in /Users/drainasr-

[ OK ] Python is version 3.10.4

[ OK ] jupyterlab

[ OK ] matplotlib

[ OK ] numpy

[ OK ] pandas

[ OK ] seaborn

[ OK ] statsmodels

[ OK ] folium



# Presentation des donnees

# Vue générale des données (statistiques)

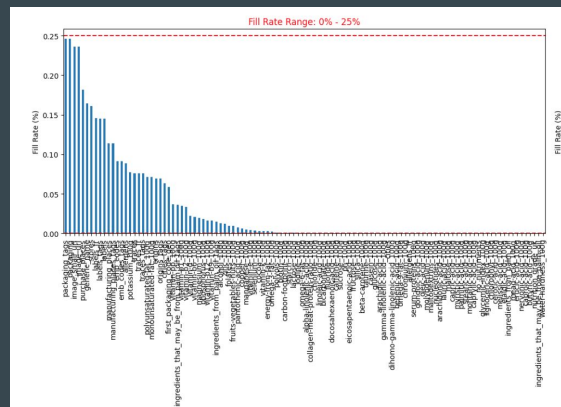
`df.shape`

- Récupérer les dimensions du tableau
- On a 162 colonnes et 320772 lignes

`df.dtypes`

- Récupérer le type des données
- On a des floats (34.6%) et objects (65.4%)

On constate un nombre de features dont le taux de remplissage est inférieur à 25% très important. Ici, dont le graphique, on comptabilise pas 112 features, soit 69% sous la barre de 25% de taux de remplissage



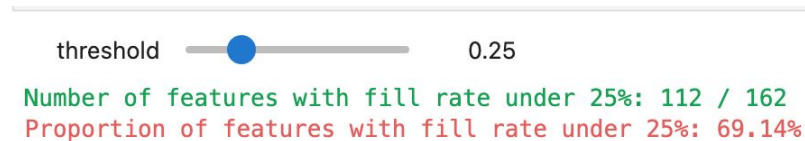
# Le nettoyage des données

# Suppression des colonnes en dessous de 25% de taux de remplissage

On a également effectuer les nettoyages suivantes:

- Rempli par "0" pour les valeurs nulles de "fibre"
- Supprime les lignes ou le nom ou code du produit n'est pas indiqué et ou il y a des doublons
- Supprime les lignes ou aucunes valeurs nutritionnelles n'est renseigne
- Supprime les lignes ou le maximum de calories possible pour 100g est dépassé
- Remplace par la médiane les lignes ou les valeurs nutritionnelles sont  $< 0$  ou  $> 100$

Ajout d'un slider pour avoir interactivement le taux de remplissage



Pour au plus 25% de remplissage, on a 69% de features concernées

# Traitement des outliers

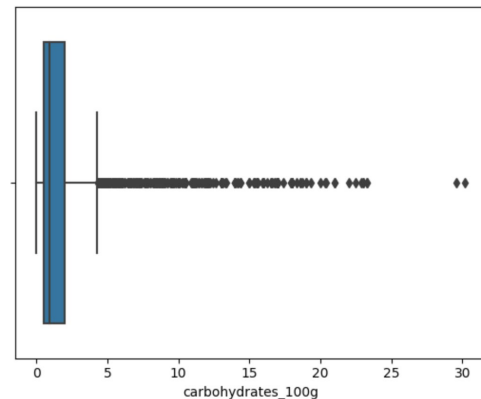
On a utilisé l'interquartile pour les détecter:

**1 étape:** on regroupe les produits selon la catégorie (pnns1) pour chaque feature. Soit dans l'exemple ci-contre, on:

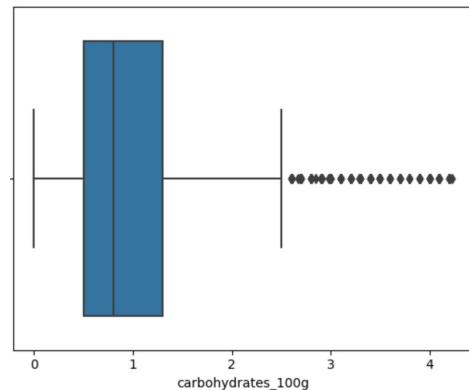
- Carbohydrates\_100g (feature)
- Fish Meat Eggs (groupe pnns1)

**2 étape:** On remplace les outliers par la médiane et du fait qu'on a regroupé par categorie de produit, rend la valeur médiane plus adapté

Avant:



Après:



## Remplacement valeurs nulles:

De la même façon que pour les outliers, j'ai remplacé les valeurs nulles par la médiane pour chaque feature, en faisant un regroupement par catégorie de produits. Soit en utilisant la colonnes (pnns 1)

Avant:

	Fill rate
energy_100g	0.000000
sugars_100g	0.000000
fat_100g	0.000000
carbohydrates_100g	0.000000
saturated-fat_100g	0.000000
fiber_100g	0.000000
proteins_100g	0.182044
salt_100g	0.583697
sodium_100g	0.586587
nutrition-score-fr_100g	1.522813
calcium_100g	95.437339
vitamin-c_100g	97.283787
iron_100g	97.598752
vitamin-a_100g	98.823937
cholesterol_100g	99.231369
trans-fat_100g	99.306499

Après:

	Fill rate
energy_100g	0.0
proteins_100g	0.0
salt_100g	0.0
sodium_100g	0.0
sugars_100g	0.0
fat_100g	0.0
carbohydrates_100g	0.0
saturated-fat_100g	0.0
nutrition-score-fr_100g	0.0
fiber_100g	0.0
cholesterol_100g	0.0
trans-fat_100g	0.0
calcium_100g	0.0
vitamin-c_100g	0.0
iron_100g	0.0
vitamin-a_100g	0.0

# Selection des donnees

On va récupérer d'une les donnees numerique et d'autres parts les données catégoriels:

- Parmi les données numériques, on récupère les informations nutritionnelles (\_100g)
- Parmi les données catégorielles, on aura besoin du nutri-grade, les références des produits (nom, code) ainsi que leurs catégories (pnns, catégories etc..). Enfin le pays (country)

On va se focaliser sur ce qui est pertinent dans un contexte français.

- Pour la feature “countries fr”, on filtre uniquement les produits francais
- On élimine 'nutrition-score-uk\_100g' qui n'est pas pertinent pour cette étude.

## Quantitative features

0	energy_100g
1	proteins_100g
2	salt_100g
3	sodium_100g
4	sugars_100g
5	fat_100g
6	carbohydrates_100g
7	saturated-fat_100g
8	nutrition-score-fr_100g
9	fiber_100g
10	cholesterol_100g
11	trans-fat_100g
12	calcium_100g
13	vitamin-c_100g
14	iron_100g
15	vitamin-a_100g

---

## Qualitative features

0	countries_fr
1	categories_fr
2	nutrition_grade_fr
3	main_category_fr
4	pnns_groups_1
5	pnns_groups_2
6	product_name
7	code

# Analyse exploratoire



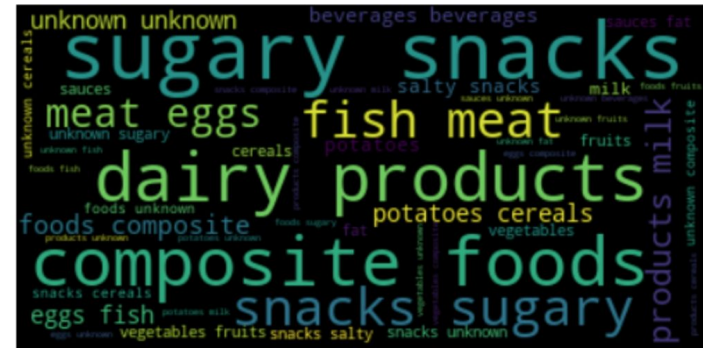
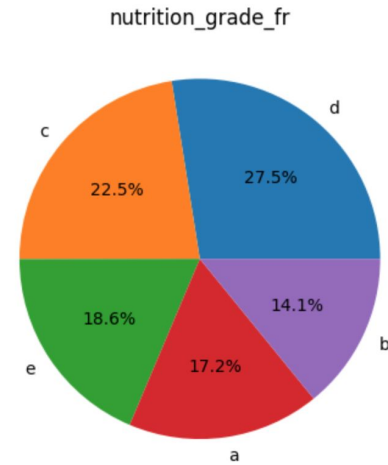
## Exemple d'analyse univariée

On constate nettement d'après ce graphique:

- Les nutri-score C, D et E sont totalisent les  $\frac{2}{3}$  des produits avoisinant les 70%. Tandis que A et B sont présent d'environ le  $\frac{1}{3}$  restant environ 30%

On a également utilise un WordCloud, qui laisse apparaître:

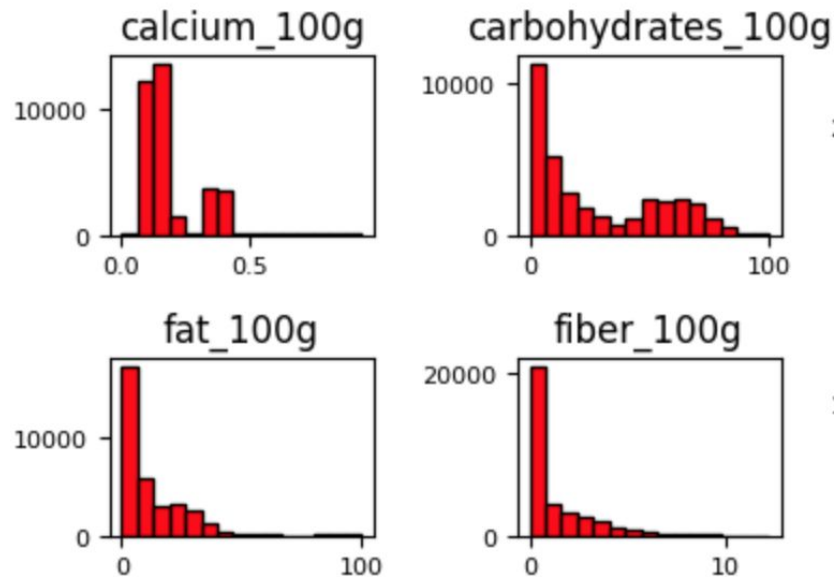
- Une forte présence des produits sucrés (sugary snacks)
- Une faible présence des fruits et légumes



## Exemple d'analyse univariée

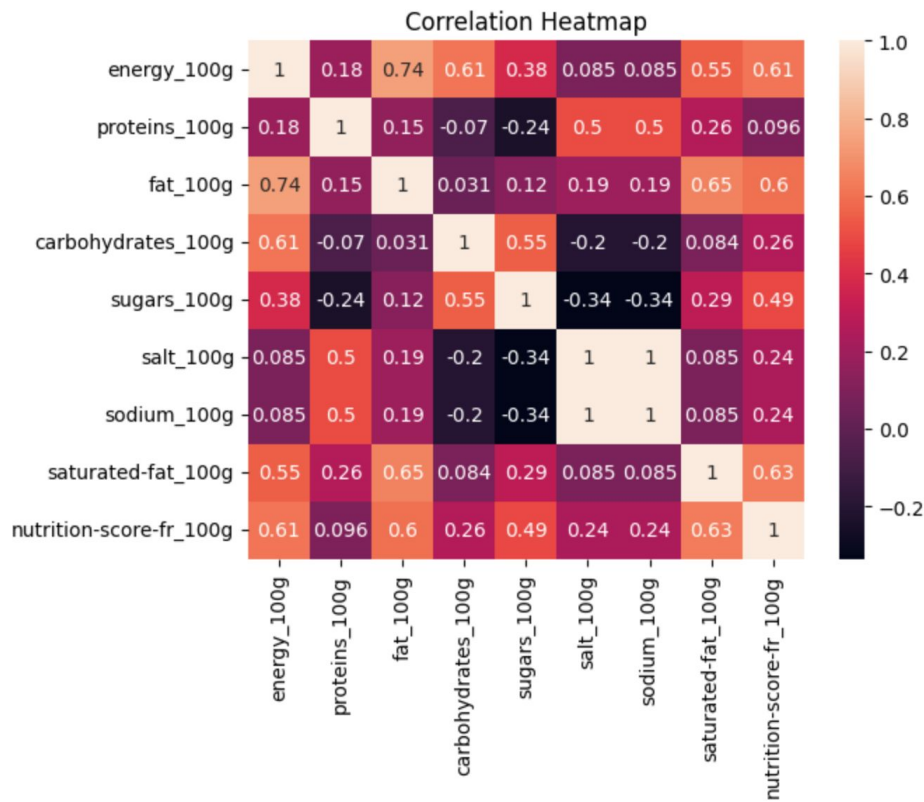
On constate d'après chaque feature nutritionnelle :

- Les macro-nutriments tel que le carbohydrate ou les lipides sont présent sur toute la plage qui va de 0 à 100g de manière décroissante
- Les micronutriments sont souvent proches de 0, présent en très faible quantité



## Exemple d'analyse bivariée

- Les indices de corrélation entre chaque variable numérique montrent qu'il existe une relation entre les graisses et les graisses saturées, les glucides et les sucres, et qu'il existe un ratio identique entre le sel et le sodium.
- De plus, l'énergie est fortement corrélée avec les graisses, ce qui est cohérent avec les informations que nous avons vues précédemment. Nous avons 9 calories par gramme pour les graisses tandis que pour les protéines et les glucides, nous n'avons que 4.



# Recherche sur internet des aliments les plus énergétiques

Checking internet for find maximum possible energy value per 100g

According to the [What are calorie-dense foods?](#) website, the highest calorie foods are...

## Animal Fats



Lard is a high calorie food.

Per 100 g	892 kcal
-----------	----------

Per Serving	116 kcal
-------------	----------

Serving Size	1 tbsp (13 g)
--------------	---------------

## Plant Oils



Plant oils are a high calorie food.

Per 100 g	850 kcal
-----------	----------

Per Serving	119 kcal
-------------	----------

Serving Size	1 tbsp (14 g)
--------------	---------------

## Butter and Margarine



Butter is a high calorie food.

Per 100 g	726 kcal
-----------	----------

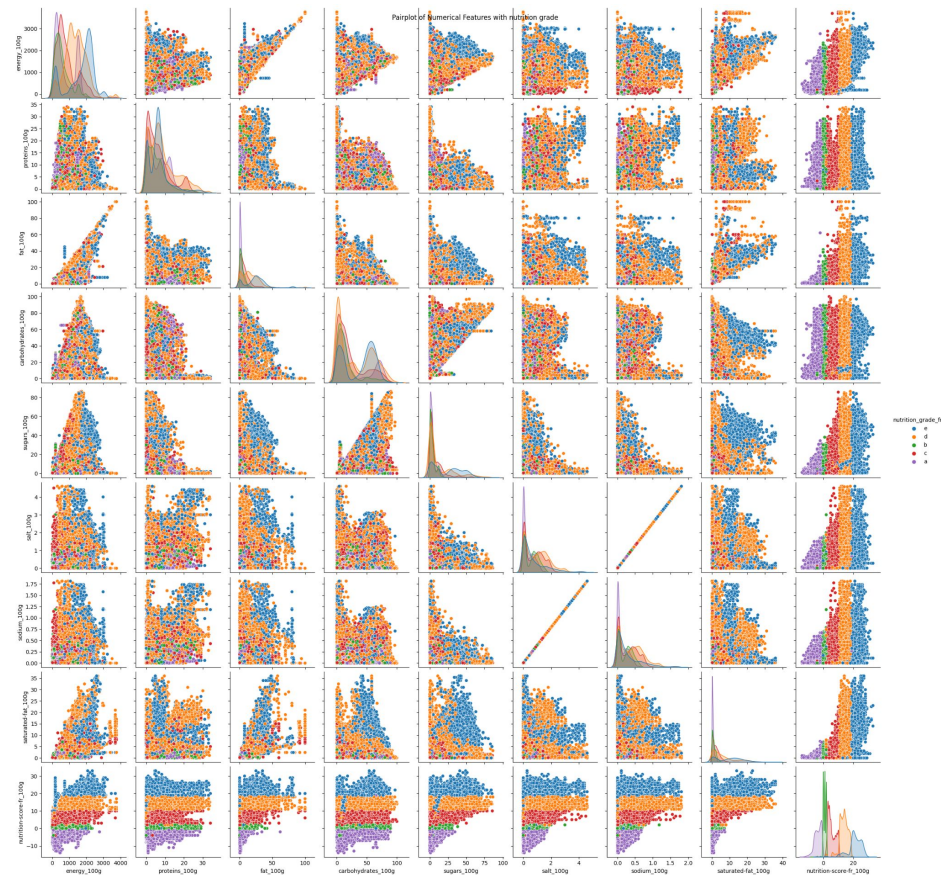
Per Serving	102 kcal
-------------	----------

Serving Size	1 tbsp (14 g)
--------------	---------------

# Pairplot

On constate plusieurs choses:

- Nutri-score numerique et nutriscore grades sont bien coherent, plus la valeur est basse (-10: 0) plus le score tend vers A.
- Correlation quasi parfaite entre sel et sodium
- Courbe ascendante entre energy et fat
- Courbe descendante entre sucre et sel



# PCA et Anova

# PCA

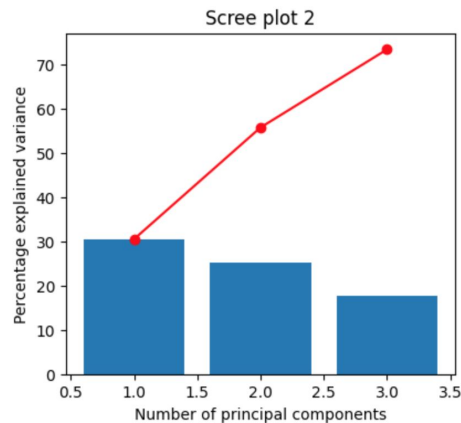
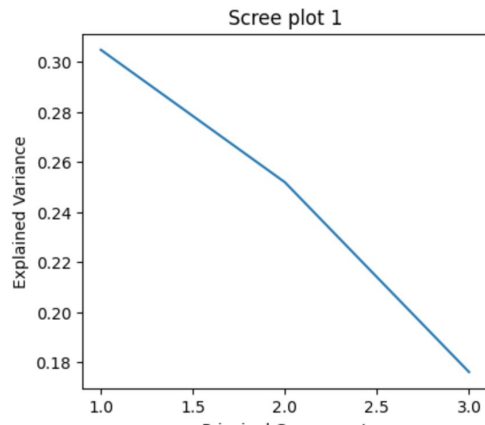
On constate d'après cette étude basée sur la méthode PCA:

- PC1 = 30%
- PC2 = 25%
- PC3 = 17%

Les 2 premières composantes expliquent environ 55% de la variance. Ce qui semble un nombre suffisant pour pouvoir continuer l'analyse

```
[8]: explained_variance = pca.explained_variance_ratio_  
     explained_variance
```

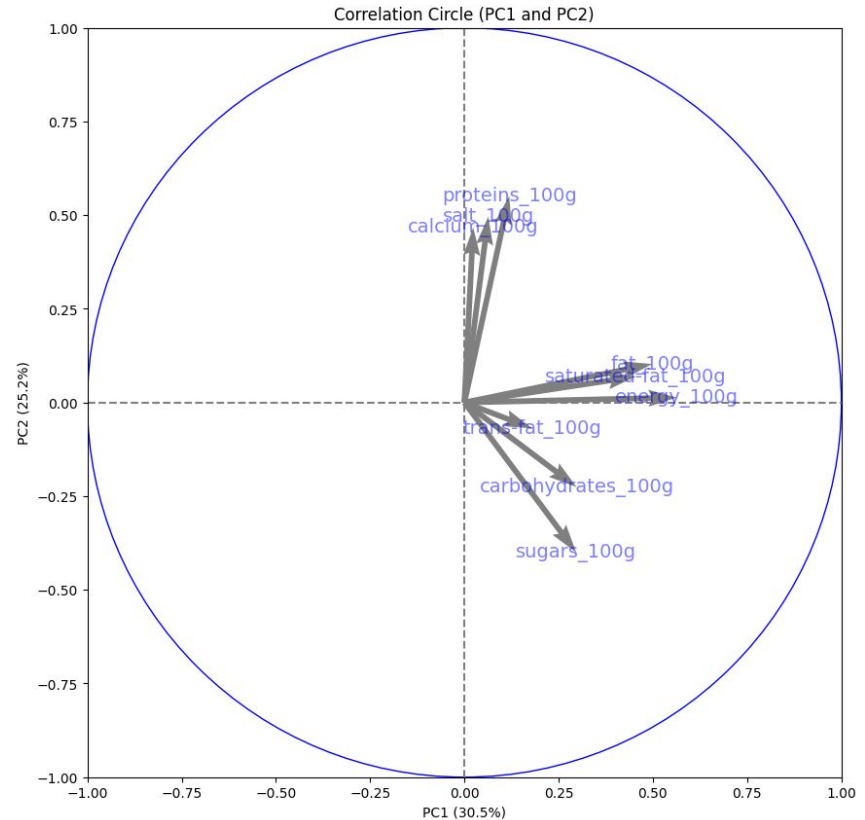
```
[8]: array([0.30483333, 0.25190673, 0.17609959])
```



# Cercle de correlation PC1 et PC2

On constate d'après ce cercle de correlation

- Énergie est fortement corrélé avec la première composante
- Les lipides sont corrélé à l'énergie car l'angle entre eux est très petit
- Les protéines sont plutôt corrélés à la composante 2
- L'angle très proche de 90 degrés entre protéine et energy suggère une faible corrélation entre les 2





# Cercle de corrélation PC2 et PC3

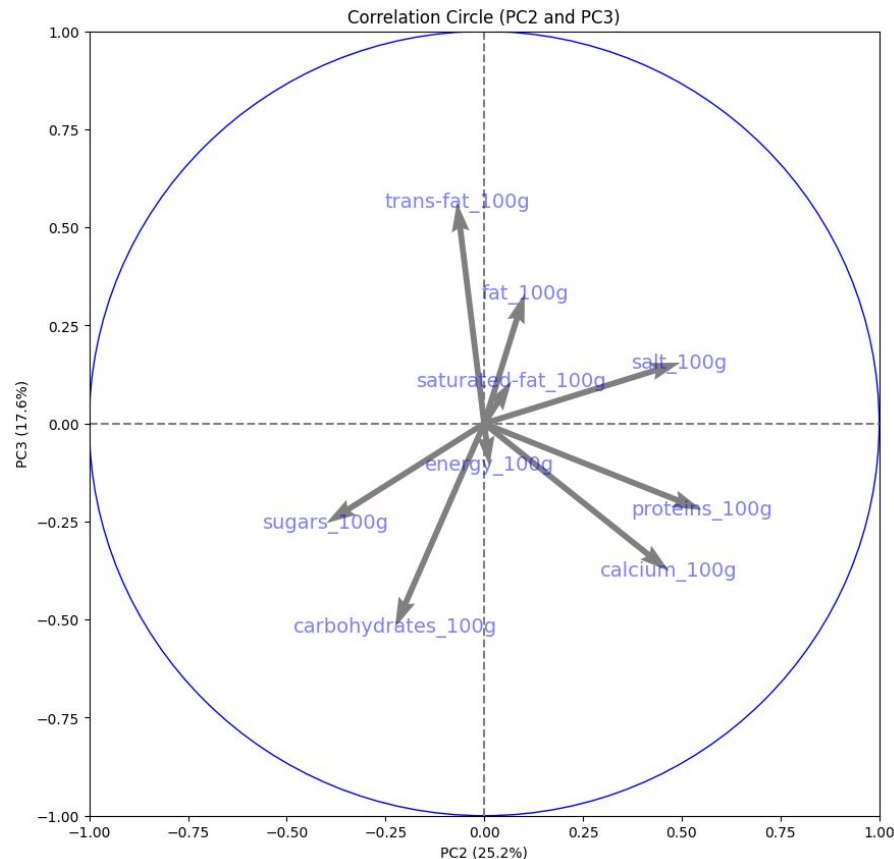
On constate d'après ce cercle de corrélation

On peut voir avec les composants 2 et 3 : plusieurs oppositions. Notamment entre

- le sucre/sel
- les graisses et les glucides.

il existe une relation inverse entre les deux variables, de sorte qu'une augmentation d'une est associée à une diminution de l'autre.

Concernant la prevalence de ces featuressur une compostante plutôt que l'autre, on ne peut pas se prononcer.



# Anova

Pour l'utilisation d'anova, on a choisit le nutri-score et comme variable quantitative les proteines.

On a utilise la fonction `f_oneway` de la librairie `scipy` qui permet d'obtenir la Valeur F et P

Ce p-value est incroyablement petit, ce qui signifie que nous pouvons rejeter  $H_0$  et conclure que la vraie moyenne des protéines n'est pas la même pour les cinq niveaux de nutrition.

```
from scipy import stats
F, p = stats.f_oneway(df[df['nutrition_grade_fr'] == 'a']['proteins_100g'],
df[df['nutrition_grade_fr'] == 'b']['proteins_100g'],
df[df['nutrition_grade_fr'] == 'c']['proteins_100g'],
df[df['nutrition_grade_fr'] == 'd']['proteins_100g'],
df[df['nutrition_grade_fr'] == 'e']['proteins_100g'])
print('ANOVA test for mean proteins')
print('F Statistic:', F, '\tp-value:', p)
```

ANOVA test for mean proteins

F Statistic: 182.71131504741737

p-value: 3.401964690233302e-155

# Conclusion

# Conclusion de l'analyse de données

## Points important

Cette analyse nous a permis de mieux comprendre comment fonctionne le nutri-score, l'apport énergétique des aliments et quels sont les liens entre les différents types de nutriments.

- On peut utiliser le dataset qui a été nettoyé comme outils de comparaisons entre plusieurs produits par exemple pour savoir quels sont les produits les plus énergétiques pour un athlète.
- On peut également utiliser ce dataset pour prédire le nutri-score en se basant sur les valeurs nutritionnelles fournies.

# L'auteur



Nasr-edine Draï, étudiant  
ingénieur IA

---

J'ai obtenu un diplôme de  
développeur d'applications  
python OpenClassrooms et je  
poursuis mes études pour  
devenir ingénieur IA