CONCEVEZ UNE APPLICATION AU SERVICE DE LA SANTÉ PUBLIQUE

MARWA EL HOURI

APPEL A PROJET

- Objectif: Trouver des idées innovantes d'applications en lien avec l'alimentation.
- Une application pour une alimentation plus saine.

UNE APPLICATION POUR UNE ALIMENTATION PLUS SAINE

- Pour vivre mieux il faut manger mieux
- Je propose une application qui nous aide a trouver des produits meilleur pour la sante.
- L'application permet de recommander des produits avec moins de sucre et matières grasses
- Avec l'option de voir si le produit est riche en glucides et\ou protéines pour les plus sportifs!

PLAN DE TRAVAIL

Etape 1 :

Nettoyage des données

Etape 2 :

Exploration des données

Etape 3 :

Présentation de la pertinence de l'application

PLAN DE TRAVAIL

Etape 1 :

Nettoyage des données

Etape 2 :

Exploration des données

Etape 3 :

Présentation de la pertinence de l'application

NETTOYAGE DES DONNÉES

1. Présentation du jeu de données
2. Réduction du nombre de colonnes
3. Exploration des valeurs manquantes et dupliquées
4. Exploration des catégories de produits
5. Sélection et nettoyage des variables quantitatives
6. Traitement des valeurs manquantes par imputation

1-PRÉSENTATION DU JEU DE DONNÉES

- Source : « Open Food Facts »
- Table (320 772, 162)
- 56 colonnes de types ‘objet’
- 106 colonnes de type ‘float64’

	code	url	creator	created_t	created_datetime	last_modified_t	last_modified_datetime
0	3087	http://world-fr.openfoodfacts.org/produit/0000...	openfoodfacts-contributors	1474103866	2016-09-17T09:17:46Z	1474103893	2016-09-17T09:18:13Z
1	4530	http://world-fr.openfoodfacts.org/produit/0000...	usda-ndb-import	1489069957	2017-03-09T14:32:37Z	1489069957	2017-03-09T14:32:37Z
2	4559	http://world-fr.openfoodfacts.org/produit/0000...	usda-ndb-import	1489069957	2017-03-09T14:32:37Z	1489069957	2017-03-09T14:32:37Z
3	16087	http://world-fr.openfoodfacts.org/produit/0000...	usda-ndb-import	1489055731	2017-03-09T10:35:31Z	1489055731	2017-03-09T10:35:31Z
4	16094	http://world-fr.openfoodfacts.org/produit/0000...	usda-ndb-import	1489055653	2017-03-09T10:34:13Z	1489055653	2017-03-09T10:34:13Z

2- RÉDUCTION DU NOMBRE DE COLONNES

- Enlever les colonnes avec un taux de remplissage inferieur a 20% (plus de 80% de valeurs manquantes)

```
1  na_values=df.isna().mean()
2  na_values=na_values[na_values>0.8]
3  len(na_values)
```

```
108
```

- Enlever les colonnes non-numériques et non-pertinentes

```
df.drop(columns=['creator', 'created_t', 'created_datetime',
       'last_modified_t', 'last_modified_datetime'], inplace=True)
```

- Résultat : 49 colonnes restantes (20 'float64', 29 'objects')

3- EXPLORATION DES VALEURS MANQUANTES ET DUPLIQUÉES

- Enlever les produit avec « code » ou « nom de produit » manquants
- Enlever les produits dupliqués
  - Par code
  - Par nom de produit et valeurs nutritionnelles
  - Par toutes les valeurs nutritionnelles
- Résultat : (196 225, 45)

4- EXPLORATION DES VARIABLES CATÉGORIES

- 7 variables contiennent des informations (détaillées ou non) sur les catégories des produits

```
1  categ=['categories','categories_tags','categories_fr',
2         'pnns_groups_1','pnns_groups_2',
3         'main_category','main_category_fr']
```

```
1  df[categ].isna().mean().sort_values()
```

```
pnns_groups_2       0.687104
pnns_groups_1       0.687996
categories          0.707468
categories_fr       0.707468
categories_tags     0.707474
main_category       0.707474
main_category_fr    0.707474
dtype: float64
```

4- EXPLORATION DES VARIABLES CATÉGORIES

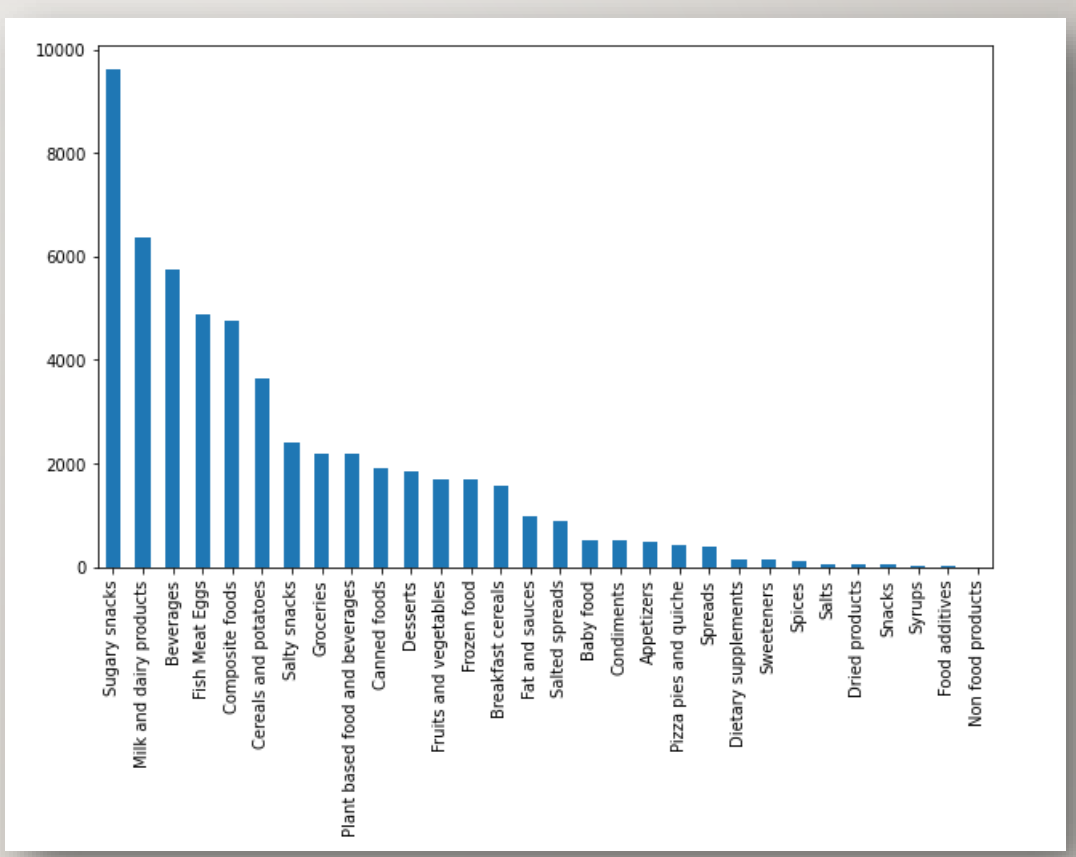
- Regroupement des informations sur les catégories dans la variable « pnns_groups_1 » ayant le moins de catégories
  - Homogénéisation des entrées (français/ anglais, majuscule/minuscule)
  - Remplacement des valeurs NaN par « unknown »
  - Récupération des entrées manquantes a partir des variables « pnns_groups_2 »
  - Regroupement par mots clé dans la variable « main_category »

```
1  df['pnns_groups_1'].value_counts()
```

```
unknown                  11693
Sugary snacks             9364
Milk and dairy products   6647
Cereals and potatoes      6449
Composite foods           5871
Beverages                 5630
Fish Meat Eggs            5110
Fruits and vegetables     3641
Fat and sauces            3448
Salty snacks              2251
fruits-and-vegetables      714
sugary-snacks              389
cereals-and-potatoes        15
salty-snacks                 1
Name: pnns_groups_1, dtype: int64
```

4- EXPLORATION DES VARIABLES CATÉGORIES

- Apres cette première catégorisation:
  - Unknown restants : 149 911



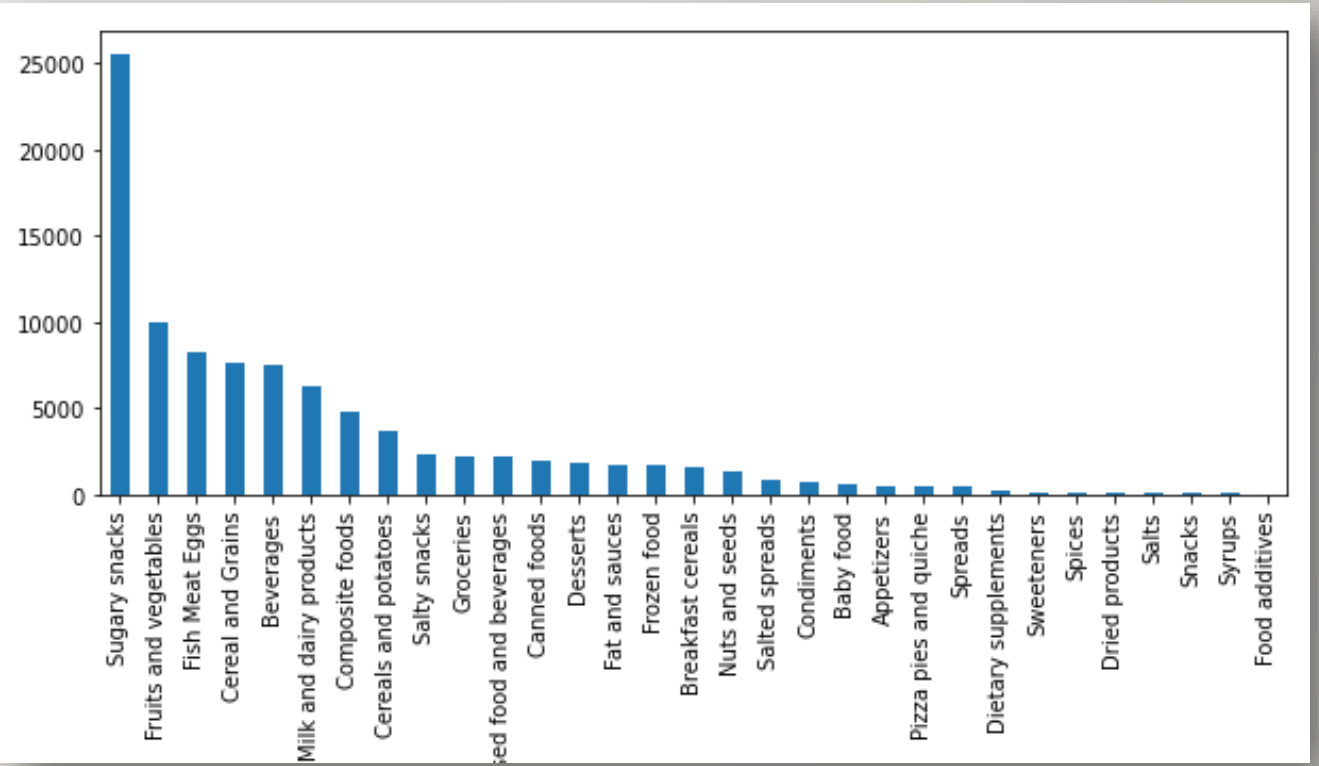
4- EXPLORATION DES VARIABLES CATÉGORIES

- Une seconde catégorisation de produits par mot-clé et valeurs nutritionnelles permet de catégoriser un peu plus de produits

```
nuts=['nut', 'almond', 'cashew','Cajun', 'hazelnut','Mixed Nuts',
      'mix', 'macadamia', 'pecan', 'pine nuts', 'pistachio', 'walnut',
      'peanuts','Pecan', 'Nut Mix', 'pumpkin seed', 'flax seed', 'sesame', 'poppy seed', 'sunflower seed',
      'psyllium seed', 'chia seed', 'kernel', 'seeds']
for n in nuts:
    Indicateurs.loc[(Indicateurs['fat_100g']>50)
                    &(Indicateurs['product_name'].str.contains(n, case=False)), 'pnns_groups_1']='Nuts and seeds'
```

4- EXPLORATION DES VARIABLES CATÉGORIES

- Apres cette catégorisation:
  - Unknown restants : 100 805



5- SÉLECTION ET NETTOYAGE DES VARIABLES QUANTITATIVES

1. Sélection des variables quantitatives
2. Nettoyage des variables quantitatives
3. Traitement des valeurs aberrantes

5- SÉLECTION DES VARIABLES QUANTITATIVES

```
1 quant_df.isna().mean().sort_values(ascending=False)
```

vitamin-a_100g	0.497085
trans-fat_100g	0.486755
vitamin-c_100g	0.485272
iron_100g	0.484859
calcium_100g	0.481740
cholesterol_100g	0.479936
fiber_100g	0.212295
nutrition-score-fr_100g	0.125399
nutrition-score-uk_100g	0.125399
ingredients_from_palm_oil_n	0.121194
ingredients_that_may_be_from_palm_oil_n	0.121194
additives_n	0.121194
saturated-fat_100g	0.098796
carbohydrates_100g	0.089353
fat_100g	0.089225
sugars_100g	0.054474
sodium_100g	0.030063
salt_100g	0.029895
proteins_100g	0.011523
energy_100g	0.005937

dtype: float64

```
1 quant_df.median().sort_values()
```

ingredients_from_palm_oil_n	0.00000
ingredients_that_may_be_from_palm_oil_n	0.00000
trans-fat_100g	0.00000
cholesterol_100g	0.00000
vitamin-c_100g	0.00000
vitamin-a_100g	0.00000
iron_100g	0.00109
calcium_100g	0.03800
sodium_100g	0.25700
salt_100g	0.65278
additives_n	1.00000
fiber_100g	1.50000
saturated-fat_100g	2.00000
proteins_100g	5.17000
sugars_100g	6.15000
fat_100g	6.30000
nutrition-score-fr_100g	10.00000
nutrition-score-uk_100g	10.00000
carbohydrates_100g	23.08000
energy_100g	1084.00000

dtype: float64

```
1 quant_df.var().sort_values()
```

vitamin-a_100g	7.484673e-03
ingredients_from_palm_oil_n	2.416810e-02
iron_100g	3.892513e-02
ingredients_that_may_be_from_palm_oil_n	8.773385e-02
cholesterol_100g	1.314971e-01
trans-fat_100g	2.887717e+00
vitamin-c_100g	6.699610e+00
additives_n	6.802288e+00
calcium_100g	1.522541e+01
saturated-fat_100g	5.879211e+01
proteins_100g	6.857830e+01
nutrition-score-fr_100g	7.965221e+01
nutrition-score-uk_100g	8.216624e+01
fiber_100g	2.090230e+02
fat_100g	2.449405e+02
sugars_100g	4.648455e+02
carbohydrates_100g	8.286204e+02
sodium_100g	3.418474e+03
salt_100g	2.205080e+04
energy_100g	5.529991e+07

dtype: float64

5-1 SÉLECTION DES VARIABLES QUANTITATIVES

- En prenant en compte le taux de remplissage, la médiane et la variance des indicateurs je choisis de garder :
  - 'energy_100g',
  - 'fat_100g',
  - 'saturated-fat_100g',
  - 'cholesterol_100g', (Je le garde pour sa pertinence)
  - 'carbohydrates_100g',
  - 'sugars_100g',
  - 'fiber_100g',
  - 'proteins_100g',
  - 'sodium_100g',
  - 'nutrition-score-fr_100g'

5-2 NETTOYAGE DES VARIABLES QUANTITATIVES

1. Remplacement des valeurs aberrantes par NaN
2. Remplacement des valeurs erronées « energie » par la valeur calculée
3. Utilisation d'une méthode d'imputation pour remplacer les valeurs manquantes restantes

	count	mean	std	min	25%	50%	75%	max
energy_100g	195057.0	1146.909378	7436.390648	0.00	418.00000	1084.000	1674.000000	3251373.000
fat_100g	178714.0	12.454227	15.650576	0.00	0.50000	6.300	20.000000	714.290
saturated-fat_100g	176836.0	5.031914	7.667601	0.00	0.00000	2.000	7.140000	550.000
cholesterol_100g	102048.0	0.019762	0.362625	0.00	0.00000	0.000	0.023000	95.238
carbohydrates_100g	178689.0	32.413795	28.785768	0.00	7.00000	23.080	57.140000	2916.670
sugars_100g	185533.0	15.632611	21.560277	-17.86	1.50000	6.150	24.000000	3520.000
fiber_100g	154565.0	2.895157	14.457627	-6.70	0.00000	1.500	3.600000	5380.000
proteins_100g	193961.0	7.239117	8.281202	-800.00	1.20000	5.170	10.000000	430.000
sodium_100g	190323.0	0.833538	58.467714	0.00	0.03937	0.257	0.551181	25320.000
nutrition-score-fr_100g	171616.0	9.311923	8.924808	-15.00	1.00000	10.000	16.000000	40.000

5-3 TRAITEMENT DES VALEURS ABERRANTES

- Remplacement par NaN pour préparer a l'imputation

```
Indicateurs.loc[Indicateurs['carbohydrates_100g']>110,'carbohydrates_100g']=np.nan
```

```
Indicateurs.loc[Indicateurs['fiber_100g']<0, 'fiber_100g']=np.nan
```

5-3 TRAITEMENT DES VALEURS ABERRANTES

- Remplacement pour une valeur calculée (dans le cas de « energy »)
- Energy=4.18(4*carbohydrates+9*fat+4*proteins)

Remplacement par NaN

```
Indicateurs.loc[(Indicateurs['difference']>300) & (Indicateurs['energy_100g']>5000)
                & (Indicateurs['fat_100g'].isna()),'energy_100g']=np.nan
Indicateurs.loc[(Indicateurs['difference']>300) & (Indicateurs['energy_100g']>5000)
                & (Indicateurs['fat_100g']<10),'energy_100g']=np.nan
```

- Remplacement basé sur
  - la différence entre les 2 valeurs et
  - la valeur des indicateurs impliqués dans le calcul de l'énergie)

Remplacement par la valeur calculée

```
Indicateurs.loc[(Indicateurs['energy_100g']>3500) & (abs(Indicateurs['difference'])>300)
                & (Indicateurs['fat_100g']<70),'energy_100g']=Indicateurs.loc[(Indicateurs['energy_100g']>3500)
                & (abs(Indicateurs['difference'])>300) & (Indicateurs['fat_100g']<70),'energie-calculee']
```

5-3 TRAITEMENT DES VALEURS ABERRANTES

	count	mean	std	min	25%	50%	75%	max
energy_100g	194964.0	1128.005186	757.744682	0.0	420.00000	1096.000	1674.000000	6001.191648
fat_100g	178014.0	12.488652	15.531231	0.0	0.50000	6.430	20.000000	100.000000
saturated-fat_100g	176830.0	5.011918	7.490542	0.0	0.00000	2.000	7.140000	100.000000
cholesterol_100g	102045.0	0.017923	0.035964	0.0	0.00000	0.000	0.023000	0.996000
carbohydrates_100g	178531.0	32.336100	27.897840	0.0	7.00000	23.080	57.140000	100.000000
sugars_100g	185518.0	15.612318	19.959741	0.0	1.50000	6.150	24.000000	100.000000
fiber_100g	154462.0	2.815293	4.271946	0.0	0.00000	1.500	3.600000	50.000000
proteins_100g	193304.0	7.267385	7.935137	0.0	1.27000	5.200	10.000000	100.000000
sodium_100g	190286.0	0.635242	2.390449	0.0	0.03937	0.257	0.551181	100.000000
nutrition-score-fr_100g	171611.0	9.311478	8.924450	-15.0	1.00000	10.000	16.000000	40.000000

Description des indicateurs après nettoyage des valeurs aberrantes

6- TRAITEMENT DES VALEURS MANQUANTES PAR IMPUTATION

- Méthode d'imputation multiple pour remplacer les valeurs manquantes
- Nous utiliserons deux méthodes d'imputation multiple
  1. **Iterative Imputer** sur les variables quantitatives corrélées
  2. **Knn Imputer** sur toutes les variables quantitatives

6- TRAITEMENT DES VALEURS MANQUANTES PAR IMPUTATION

- Avant de faire l'imputation de divise mon dataset en « training set » et « testing set » pour réduire le biais entre les deux data sets.
  - Sur le « training set » je fais un fit_transfrom
  - Sur le « testing set » je fais uniquement un transform

```
1 X_train, X_test = train_test_split(Indicateurs, test_size=0.2, random_state=42)
```

```
1 X_train.shape
```

```
(156171, 14)
```

```
1 X_test.shape
```

```
(39043, 14)
```

6-1 ITERATIVE IMPUTER

	energy_100g	fat_100g	saturated-fat_100g	cholesterol_100g	carbohydrates_100g	sugars_100g	fiber_100g	proteins_100g	sodium_100g	nutrition-score-fr_100g
energy_100g	1.000000	0.780781	0.575734	0.045333	0.578599	0.354985	0.312905	0.298115	-0.072916	0.645867
fat_100g	0.780781	1.000000	0.722928	0.194249	-0.025716	0.003187	0.162664	0.260424	-0.049477	0.586060
urated-fat_100g	0.575734	0.722928	1.000000	0.317476	0.006377	0.148850	0.019813	0.146560	-0.028970	0.630856
holesterol_100g	0.045333	0.194249	0.317476	1.000000	-0.290915	-0.126687	-0.227829	0.368245	0.010179	0.250898
ohydrates_100g	0.578599	-0.025716	0.006377	-0.290915	1.000000	0.659702	0.276242	-0.096356	-0.069440	0.307511
sugars_100g	0.354985	0.003187	0.148850	-0.126687	0.659702	1.000000	-0.002297	-0.251773	-0.084559	0.488612
fiber_100g	0.312905	0.162664	0.019813	-0.227829	0.276242	-0.002297	1.000000	0.247958	-0.018928	-0.133274
proteins_100g	0.298115	0.260424	0.146560	0.368245	-0.096356	-0.251773	0.247958	1.000000	0.005639	0.092192
sodium_100g	-0.072916	-0.049477	-0.028970	0.010179	-0.069440	-0.084559	-0.018928	0.005639	1.000000	0.114238
nutrition-score-fr_100g	0.645867	0.586060	0.630856	0.250898	0.307511	0.488612	-0.133274	0.092192	0.114238	1.000000

Matrice de corrélation des indicateurs

6-1 ITERATIVE IMPUTER

- Imputation sur les variables corrélées
  - 'energy_100g',
  - 'fat_100g',
  - 'saturated-fat_100g',
  - 'carbohydrates_100g',
  - 'sugars_100g',
  - 'fiber_100g',
  - 'proteins_100g',
  - 'nutrition-score-fr_100g'

	count	mean	std	min	25%	50%	75%	max
energy_100g	156172.0	1129.437597	757.865485	0.000000	423.000000	1100.00	1674.00	5128.830000
fat_100g	156172.0	12.504407	15.108941	-24.793716	0.770000	7.00	20.00	103.562988
saturated-fat_100g	156172.0	4.719132	7.266025	-13.449631	0.000000	1.79	6.67	100.000000
carbohydrates_100g	156172.0	32.194666	28.074399	-116.632024	6.780000	23.00	57.14	142.438933
sugars_100g	156172.0	15.289146	19.723701	-57.635264	1.500000	6.00	23.60	100.000000
fiber_100g	156172.0	2.566674	3.916276	-11.151473	0.000000	1.50	3.40	50.000000
proteins_100g	156172.0	7.271051	7.947205	-100.268801	1.297979	5.26	10.00	100.000000
nutrition-score-fr_100g	156172.0	9.065206	8.799328	-33.933867	1.497078	9.00	16.00	70.627205

Résultat: Des valeurs aberrantes dans toutes les variables !

6-2 KNN IMPUTER

- Imputation multiple
  - Sur toutes les variables quantitatives
- Conclusion:
  - On utilise ce jeu de donnée pour l'exploration des données de l'application

	count	mean	std	min	25%	50%	75%	max
energy_100g	155618.0	1129.310812	757.788131	0.0	423.00000	1100.000	1674.000000	5128.830
fat_100g	142342.0	12.491295	15.520257	0.0	0.50000	6.400	20.000000	100.000
saturated-fat_100g	141437.0	5.019219	7.485557	0.0	0.00000	2.000	7.140000	100.000
cholesterol_100g	81729.0	0.017850	0.035494	0.0	0.00000	0.000	0.023000	0.975
carbohydrates_100g	142765.0	32.354439	27.894113	0.0	7.00000	23.080	57.140000	100.000
sugars_100g	148356.0	15.610233	19.937635	0.0	1.50000	6.200	24.000000	100.000
fiber_100g	123546.0	2.814450	4.266660	0.0	0.00000	1.500	3.600000	50.000
proteins_100g	154637.0	7.274192	7.937764	0.0	1.27000	5.200	10.000000	100.000
sodium_100g	152110.0	0.634868	2.378554	0.0	0.03937	0.257	0.551181	100.000
nutrition-score-fr_100g	137215.0	9.322253	8.928824	-15.0	1.00000	10.000	16.000000	40.000

Résultat: Des valeurs statistiquement proches des valeurs initiales

RÉSULTAT DU NETTOYAGE DES DONNÉES

- Un train test
  - Dimensions (156 171, 14)
  - Pas de valeurs manquantes
  - Pas de valeurs aberrantes
- Un test set
  - Dimensions (39 043, 14)
  - Pas de valeurs manquantes
  - Pas de valeurs aberrantes

PLAN DE TRAVAIL

Etape 1 :

Nettoyage des données

Etape 2 :

Exploration des données

Etape 3 :

Présentation de la pertinence de l'application

EXPLORATION DES DONNÉES

- Objectif: Classification des produits en groupes pour identifier les produits a forte ou faible contenance en « mauvais nutriments »
- Plan d'étude:
  1. Choix des indicateurs a effet importants pour cette classification
  2. Classification non-supervise (Kmeans)
  3. Validation par des méthodes de classifications ensemblistes supervises
  4. ACP pour Visualiser la classification

1- CHOIX DES INDICATEURS POUR LA CLASSIFICATION

- Indicateurs de base:
  - Energy
  - Fat
  - Saturated fat
  - Cholesterol
  - Carbohydrates
  - Sugars
  - Fiber
  - Proteins
  - Sodium
  - Nutrition score

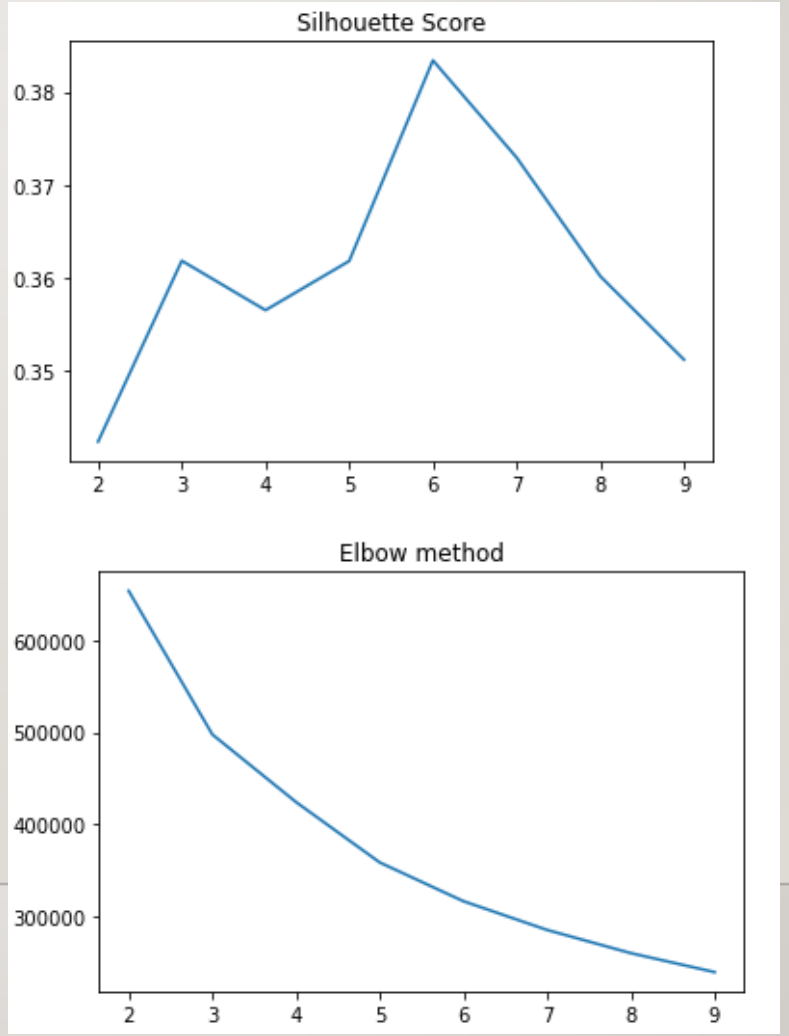
- Feature selection : **VarianceThreshold**
  - **Energy**
  - **Fat**
  - **Saturated fat**
  - ~~Cholesterol~~ (faible variance)
  - **Carbohydrates**
  - **Sugars**
  - ~~Fiber~~ (faible variance)
  - **Proteins**
  - ~~Sodium~~ (faible variance)
  - Nutrition score (je ne l'inclus pas par choix)

2- CLASSIFICATION NON - SUPERVISÉE

- Méthode : Kmeans
- Préparation des données:
  - Normalisé les données pour prendre en compte la différence de grandeur entre les différents indicateurs
  - Scaler: StandardScaler()
    - Apprentissage et transformation du train set
    - Transformation du test set

2- CLASSIFICATION NON - SUPERVISÉE

- Choix du nombre de clusters
  - Méthode Silhouette
  - Méthode du coude
- Nombre de cluster optimal: 6
- Score silhouette : 0.38



3- VALIDATION DE LA CLASSIFICATION

- Estimateurs:
  - Decision Tree Classifier
  - Random Forest Classifier
  - Xgboost Classifier
- Conclusion
  - Les indicateurs les plus influents:

    (‘energy’, ‘sugars’ , ‘proteins’ et ‘carbohydrates)

- Feature Importances

	Features	Decision Tree	Random Forest	Xgboost
0	energy_100g	0.288750	0.236659	0.246509
1	fat_100g	0.110713	0.120074	0.131350
2	saturated-fat_100g	0.147547	0.113709	0.118678
3	carbohydrates_100g	0.126233	0.182712	0.136760
4	sugars_100g	0.148683	0.160258	0.210681
5	proteins_100g	0.178074	0.186588	0.156021

3- VALIDATION DE LA CLASSIFICATION

- Score de la validation croisée sur le train test :

	Decision Tree	Random Forest	Xgboost
0	0.978000	0.988316	0.991292
1	0.978228	0.988665	0.991196
2	0.978678	0.988697	0.991868
3	0.978199	0.987416	0.990490
4	0.978283	0.987864	0.991387

- Les résultats sont très optimiste (overfitting?)
- Meilleur classificateur : Xgboost Classifier

- F1 score sur le test

	Decision tree	Random Forest	XGboost
0	0.979078	0.988498	0.992084

- F1 score sur le test par groupe :

	Decision tree	Random Forest	XGboost
0	0.970260	0.984388	0.990579
1	0.991149	0.995560	0.996274
2	0.962734	0.980969	0.987619
3	0.966503	0.978623	0.984525
4	0.981818	0.988761	0.992654
5	0.975080	0.984900	0.989031

3- VALIDATION DE LA CLASSIFICATION

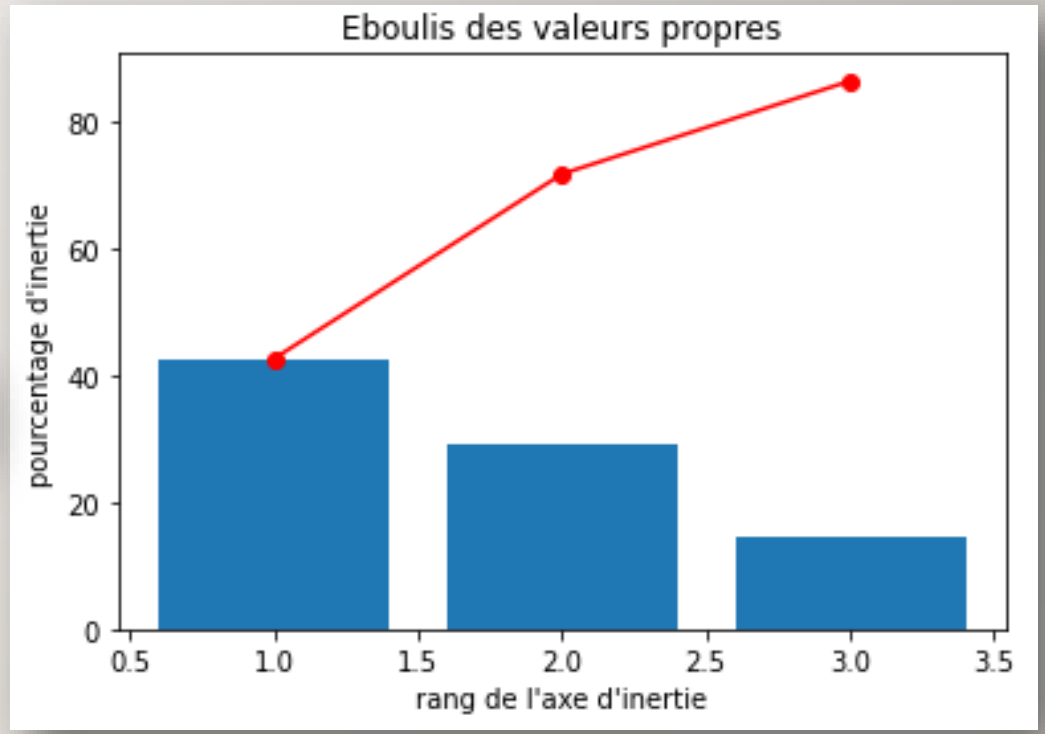
- Score sur le Test set

4- ACP POUR VISUALISER LA CLASSIFICATION

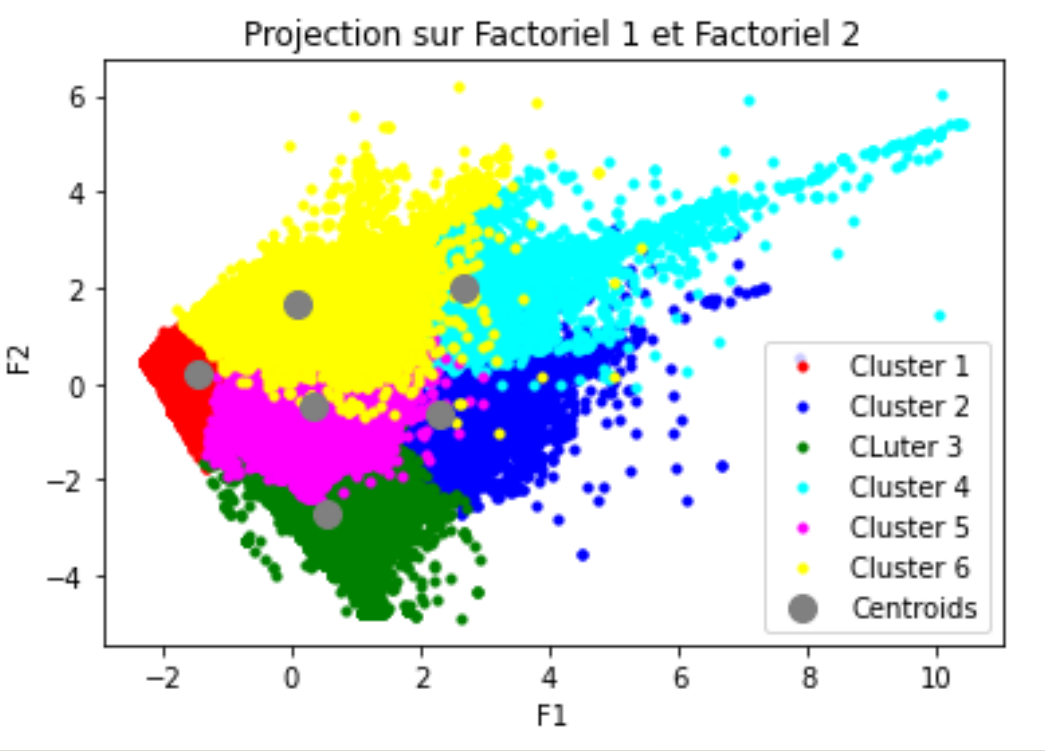
- Reduction de dimension en utilisant l'ACP avec une représentation minimal de la variance a 80%
- Somme cumulative de la variance expliquée:

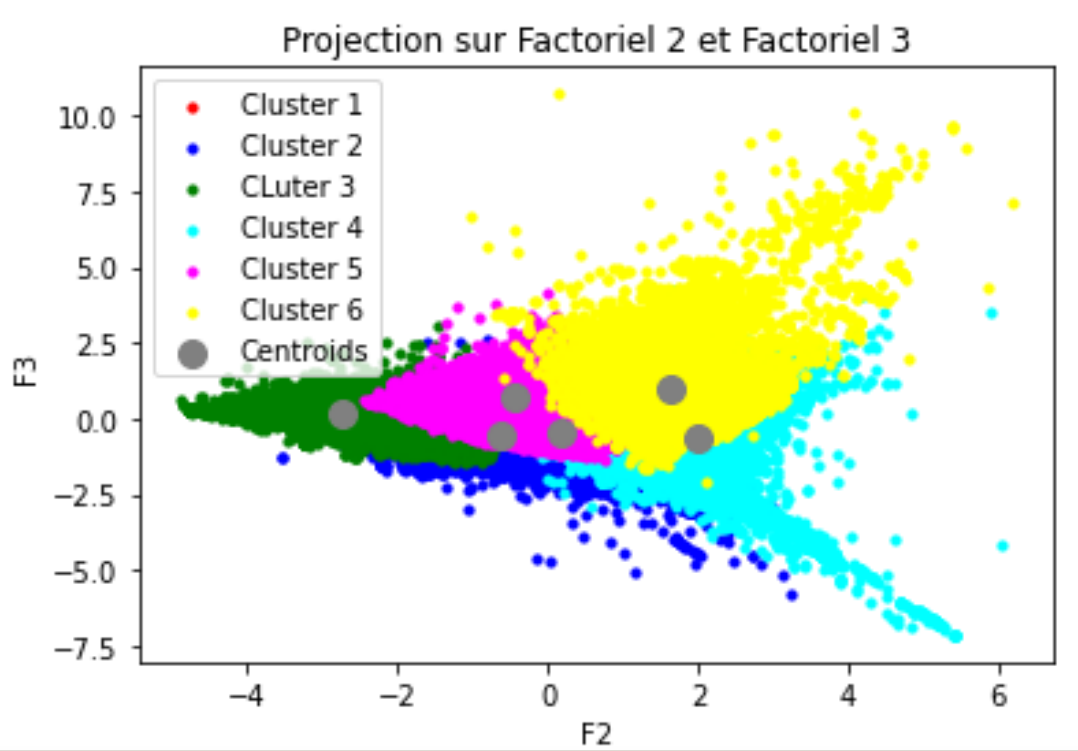
```
array([0.42633726, 0.71773525, 0.86441144])
```

- Une projection avec 3 composantes permet donc d'explique plus de 86% de la variance



4- ACP POUR VISUALISER LA CLASSIFICATION

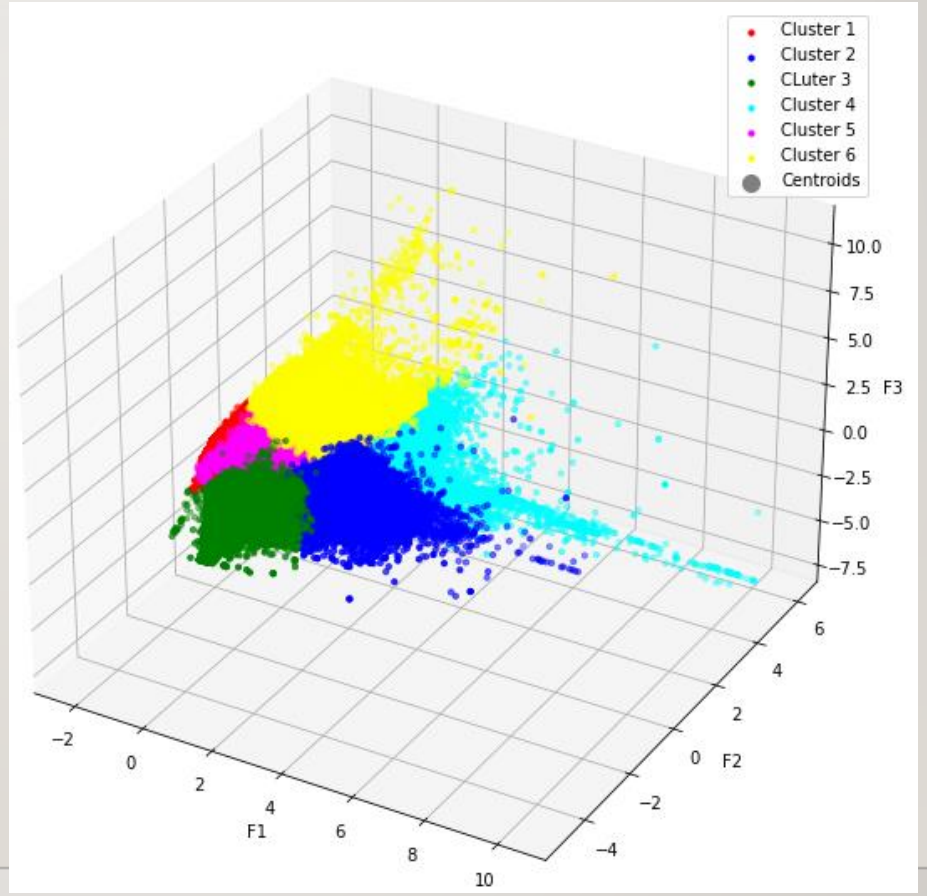




Visualisation par pair de plans factoriels

4- ACP POUR VISUALISER LA CLASSIFICATION

Visualisation 3d sur les trois plans factoriels



PLAN DE TRAVAIL

Etape 1 :

Nettoyage des données

Etape 2 :

Exploration des données

Etape 3 :

Présentation de la pertinence de l'application

ANALYSE DE LA CLASSIFICATION

1. Anova
2. Distribution des Indicateurs par groupe
3. Classification et recommandation
4. Présentation de l'application

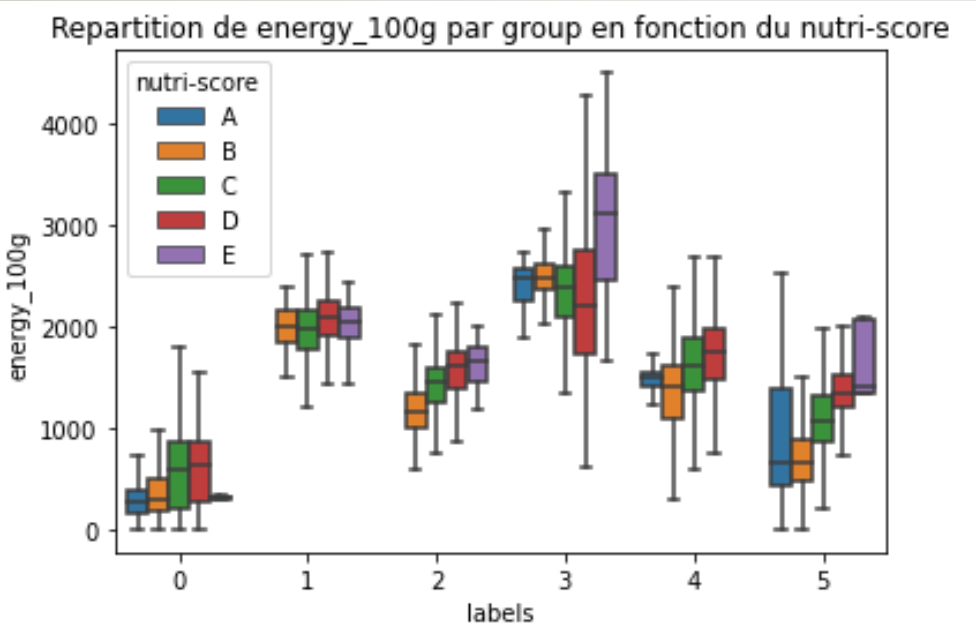
I- ANOVA

- Objectif : Vérifier l'hypothèse de la différence des moyennes d'indicateurs par groupe
- ANOVA entre la variable 'labels' et la variable 'indicateur' pour les indicateurs
  - Conclusion Les p-values de tous les indicateurs sont très faible impliquant une différence significative des moyennes par groupe, ce qui signifie que la classification est acceptable
- Le rapport de corrélations pour ces indicateurs:
  - Les rapports de corrélation varient en fonction de l'importance de l'indicateur dans la classification

	energy_100g	saturated_fat_100g	fat_100g	cholesterol_100g	carbohydrates_100g	sugars_100g	fiber_100g	proteins_100g	sodium_100g
0	0.764695	0.485388	0.660137	0.224322	0.754397	0.741458	0.091418	0.569817	0.006438

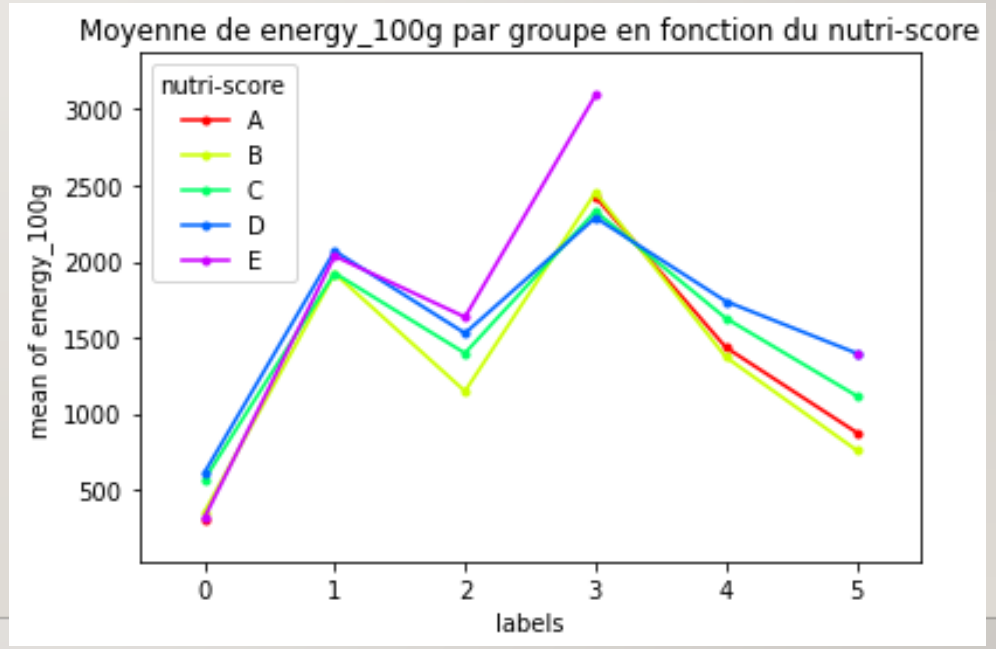
Rapport de corrélation

I- ANOVA - (ENERGY~LABELS)

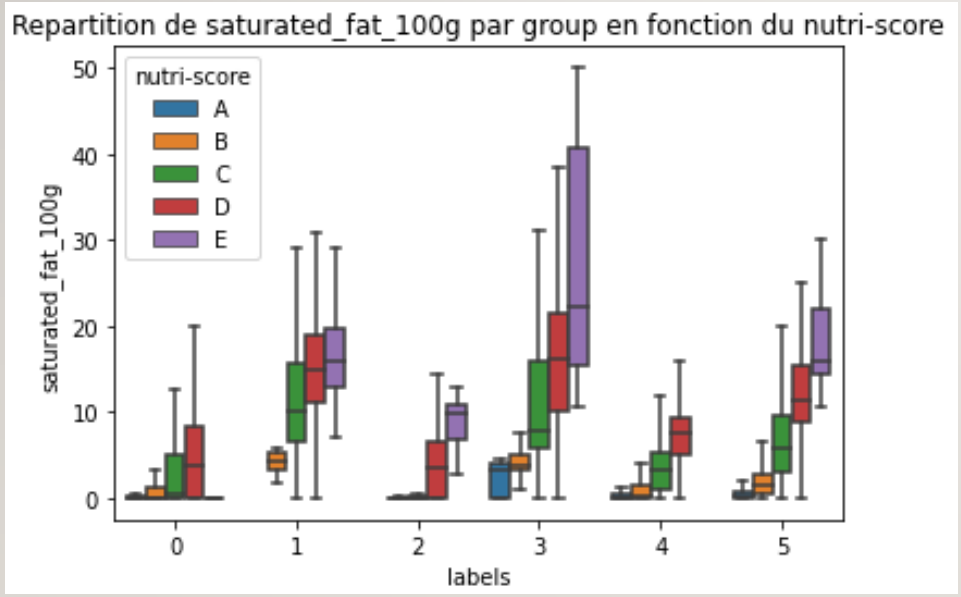


	sum_sq	df	F	PR(>F)
C(labels)	6.850876e+10	5.0	101501.74948	0.0
Residual	2.108090e+10	156166.0	NaN	NaN

Effect size= 0.7646949795640119

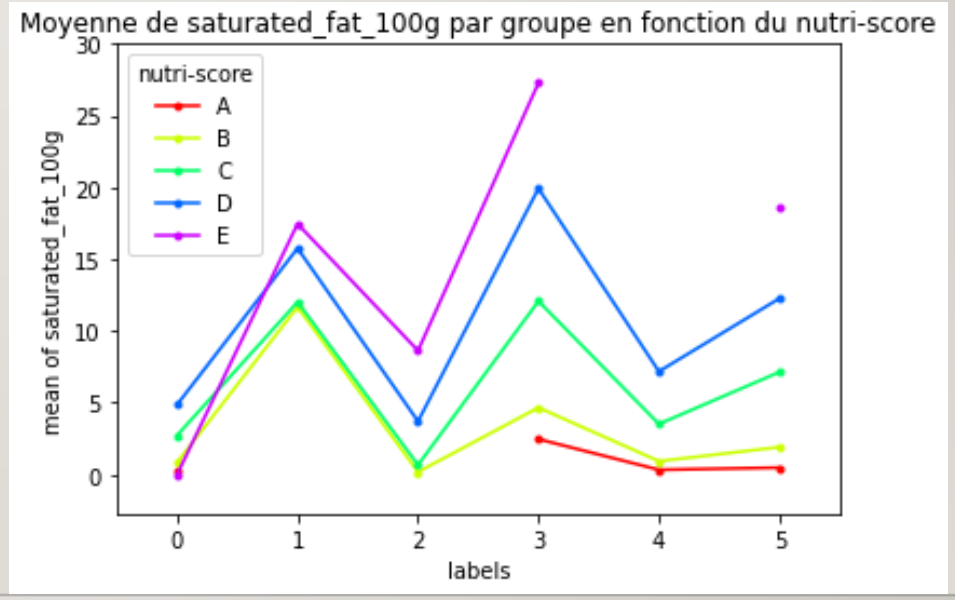


2- ANOVA - (SATURATED-FAT~LABELS)

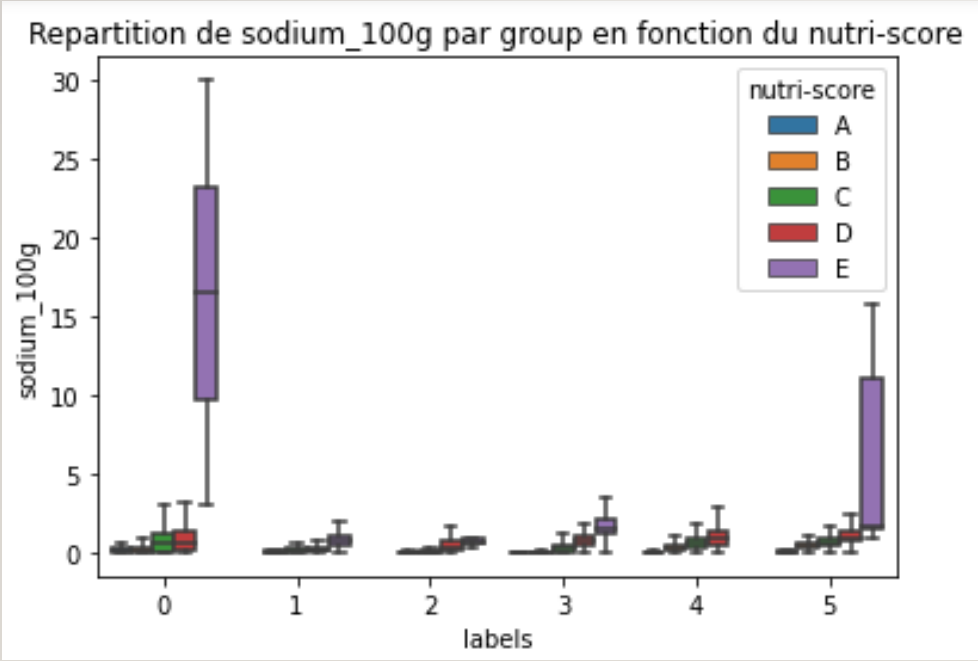


	sum_sq	df	F	PR(>F)
C(labels)	4.074434e+06	5.0	29459.474826	0.0
Residual	4.319751e+06	156166.0	NaN	NaN

Effect size= 0.48538765032573505

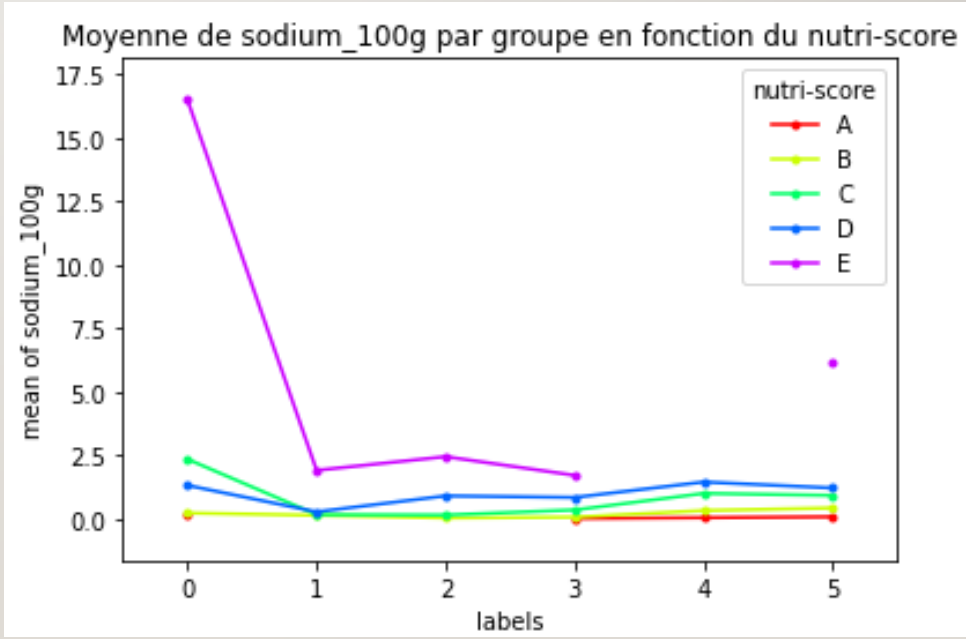


2- ANOVA - (SODIUM~LABELS)

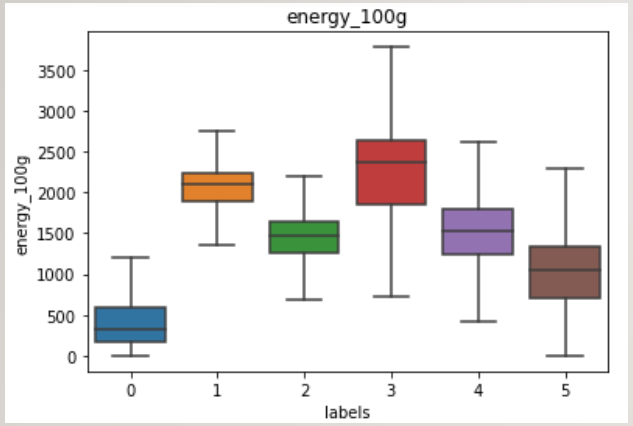


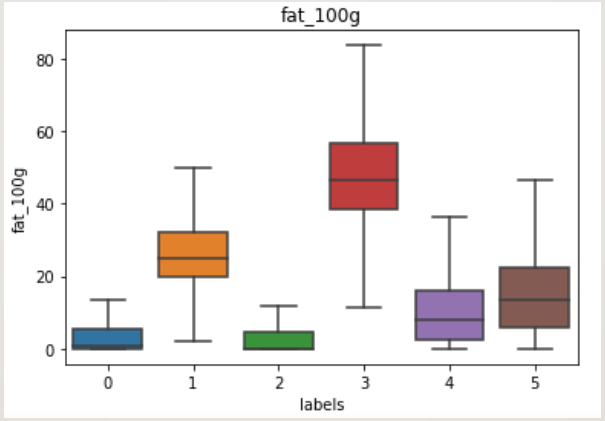
	sum_sq	df	F	PR(>F)
C(labels)	5589.585924	5.0	202.394507	7.790196e-216
Residual	862576.051998	156166.0	NaN	NaN

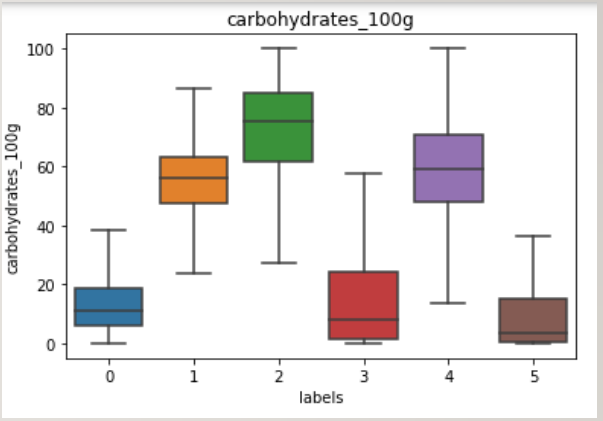
Effect size= 0.006438386501663231

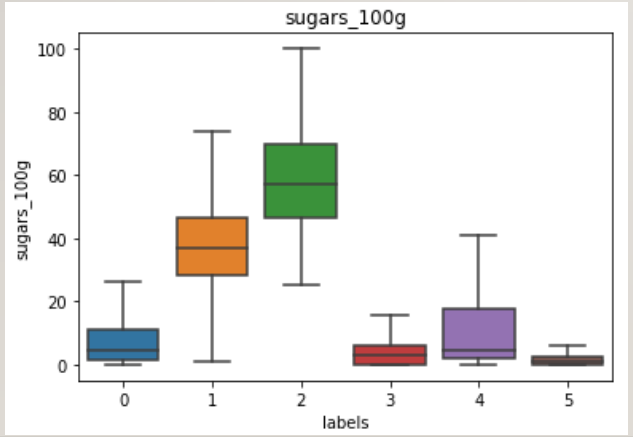


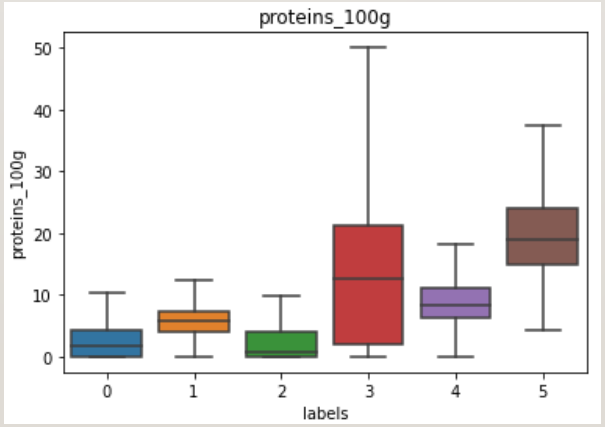
2- DISTRIBUTION DES INDICATEURS PAR GROUPE

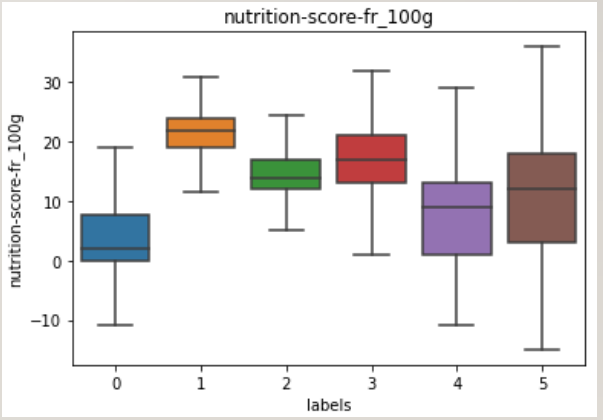




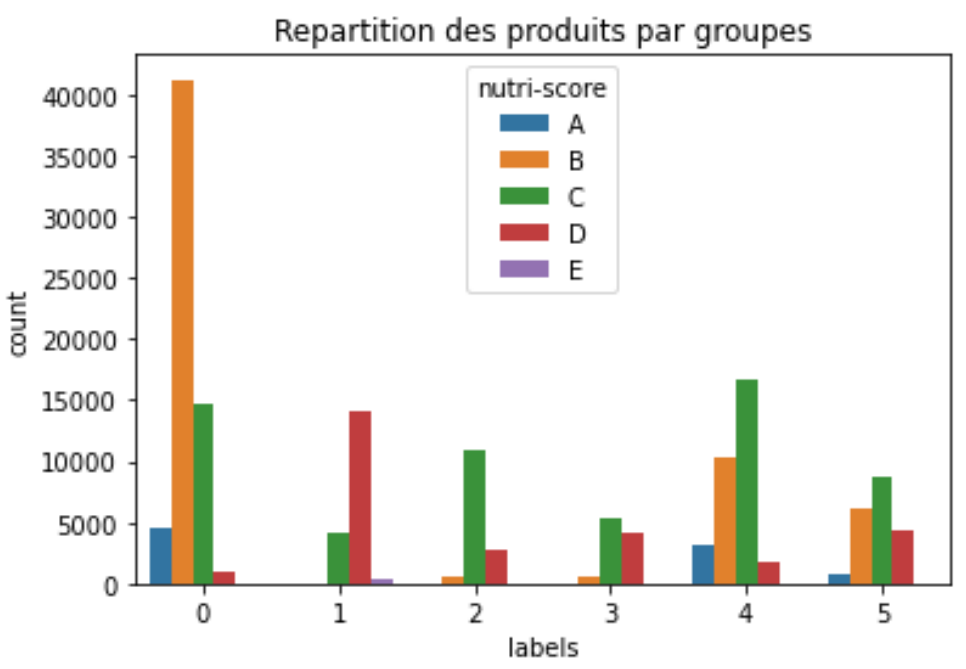


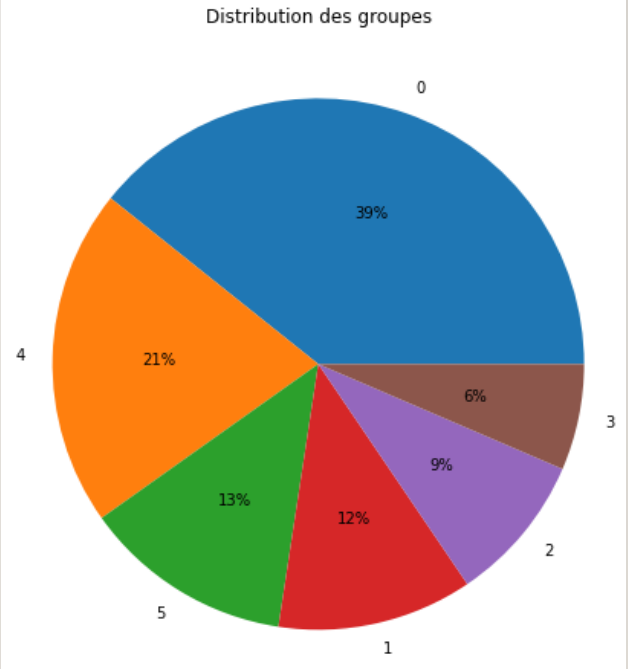






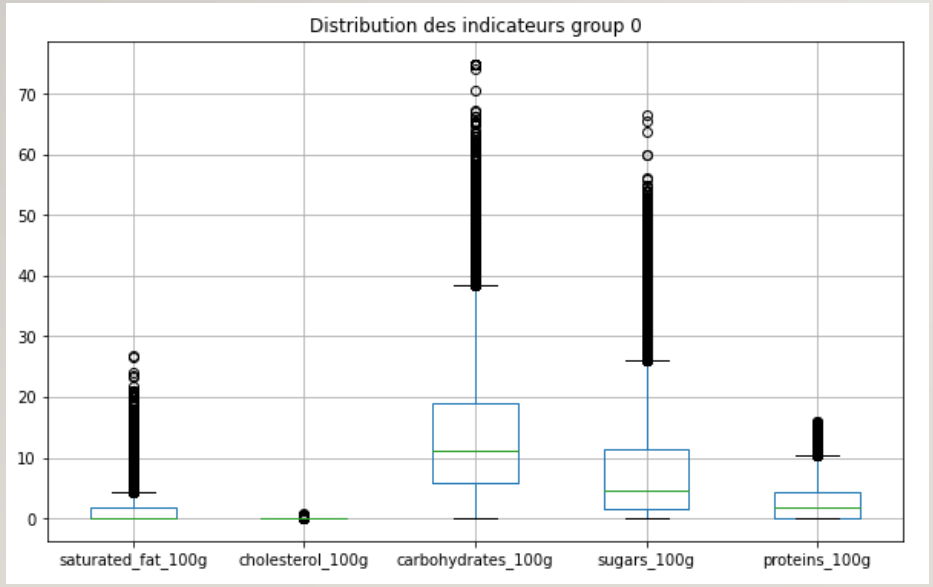
3- CLASSIFICATION ET RECOMMANDATION

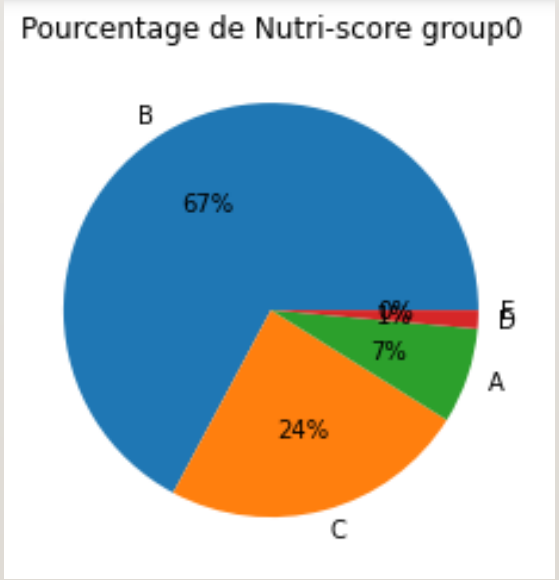


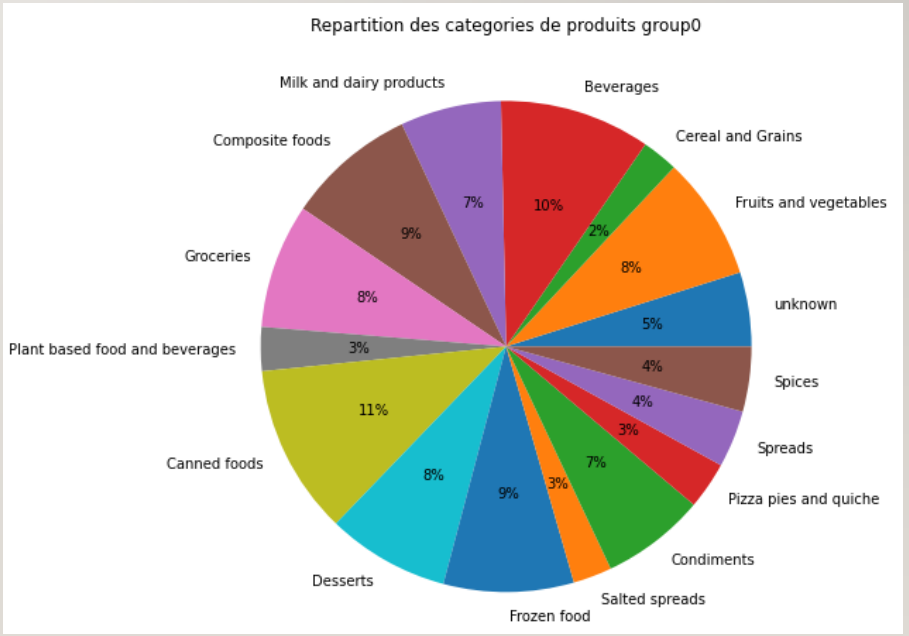


3-1 GROUPES RECOMMANDÉS

- Groupe 0 : Meilleur groupe, bas taux de matières grasses et sucres, produits diversifies (mais faibles en protéines)

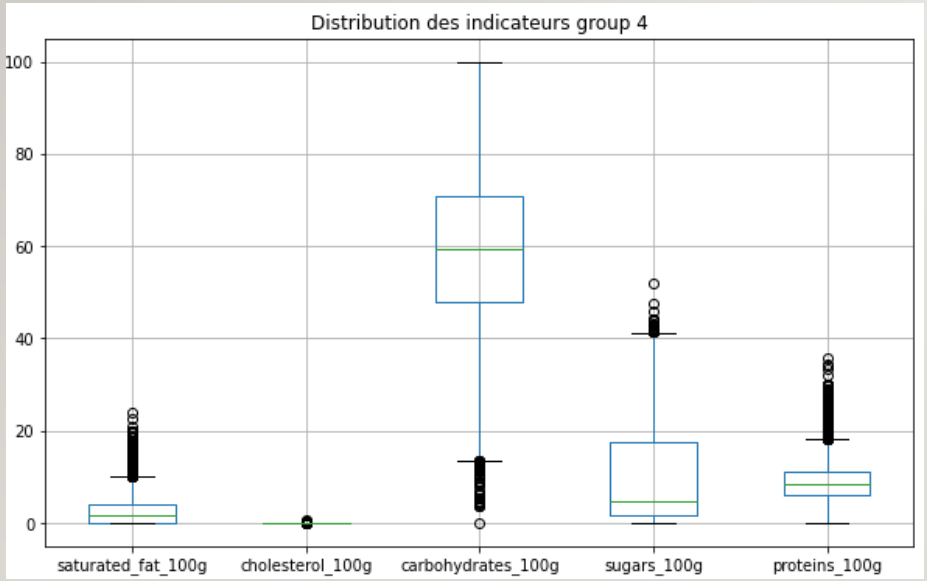


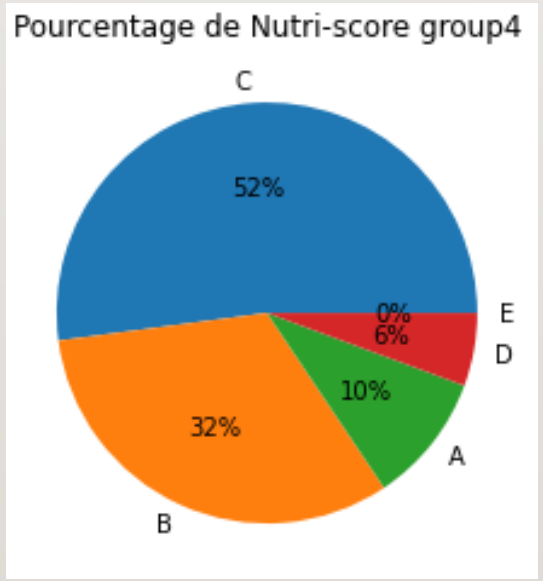


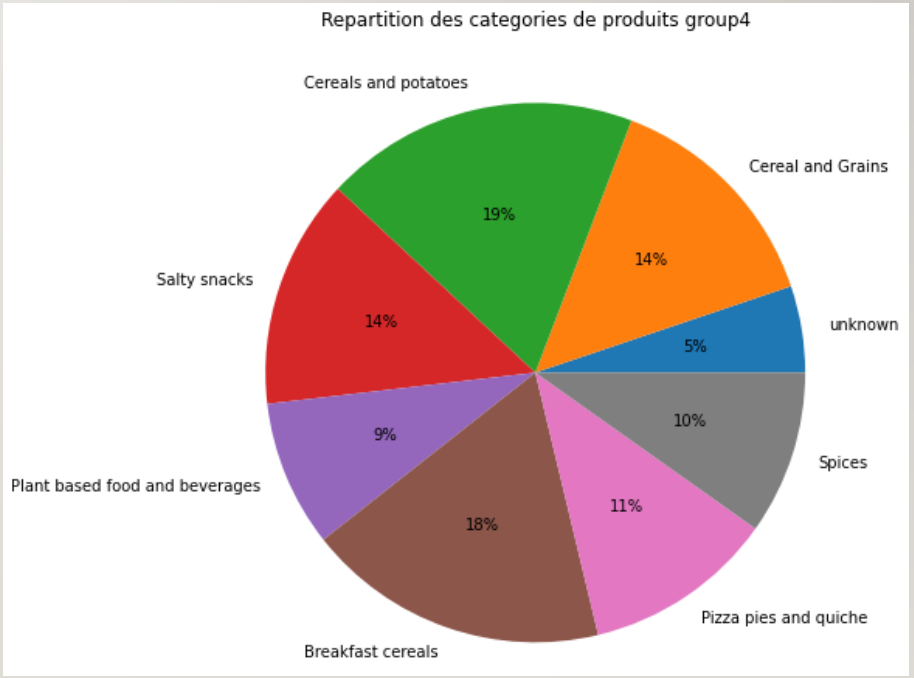


3-1 GROUPES RECOMMANDÉS

- Groupe 4 : riche en glucide (mais pauvre en gras et sucre) recommandé

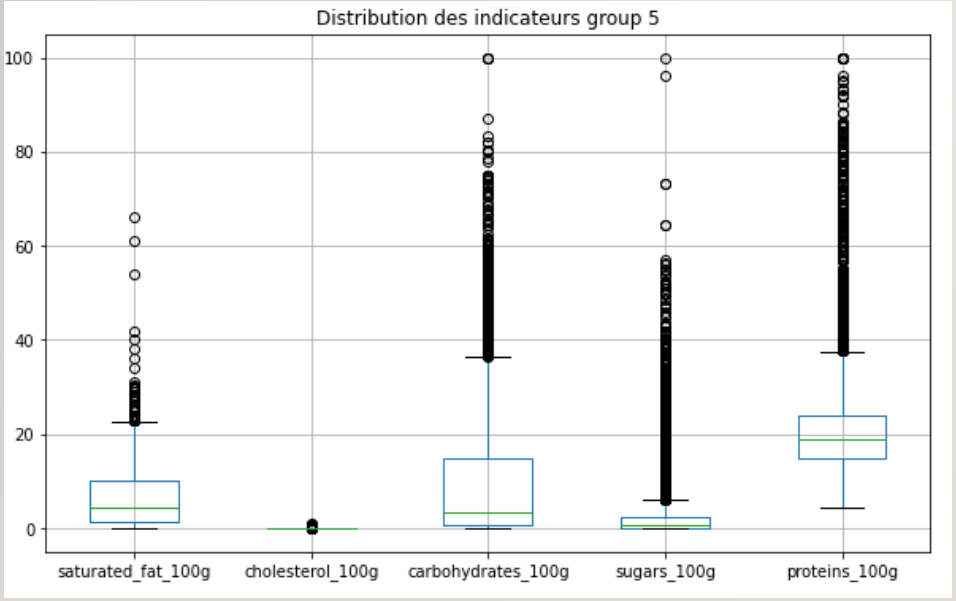


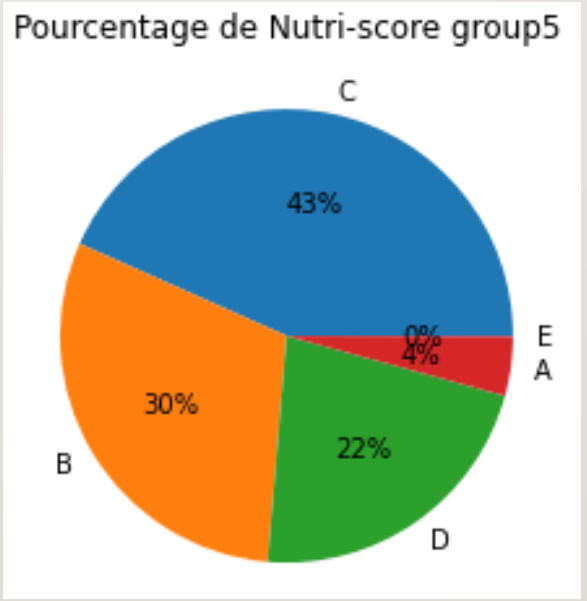


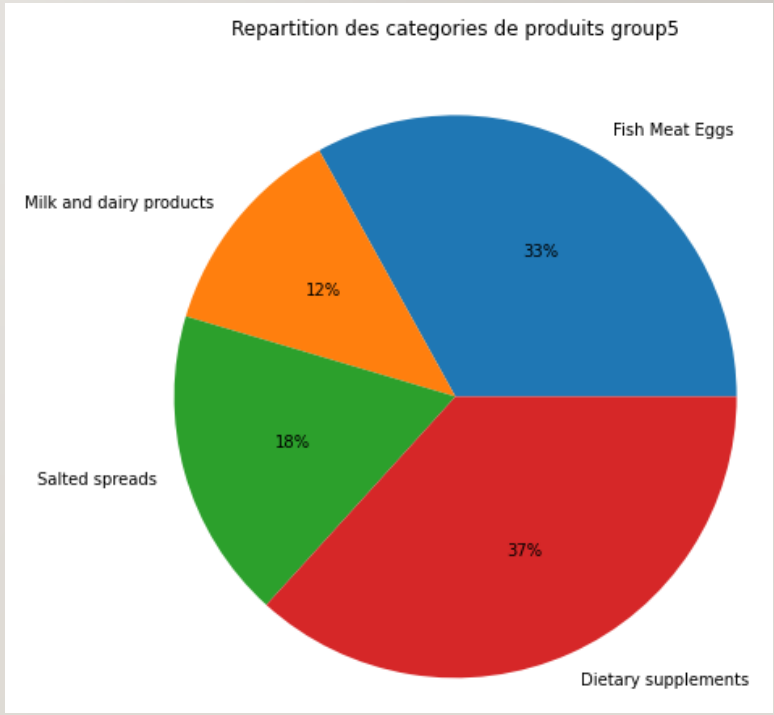


3-1 GROUPES RECOMMANDES

- Group 5 : Moyennement recommande, faible en sucre, riche en protéine

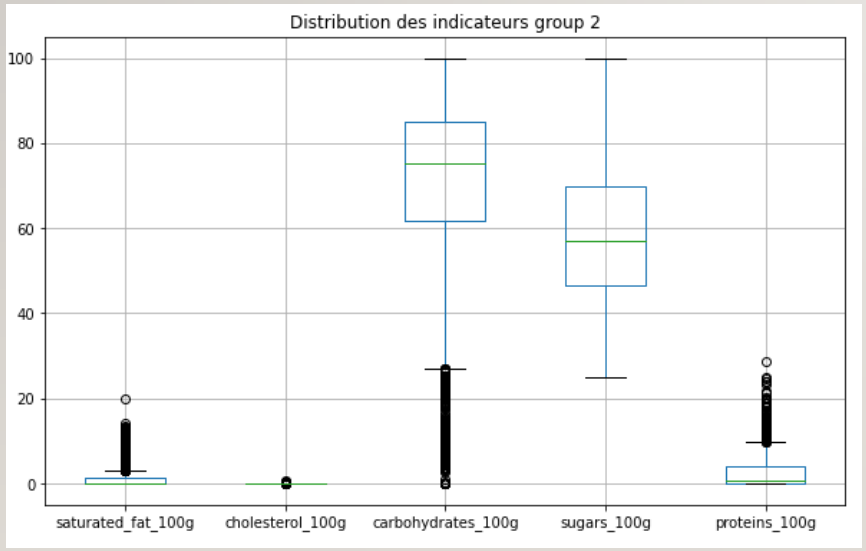


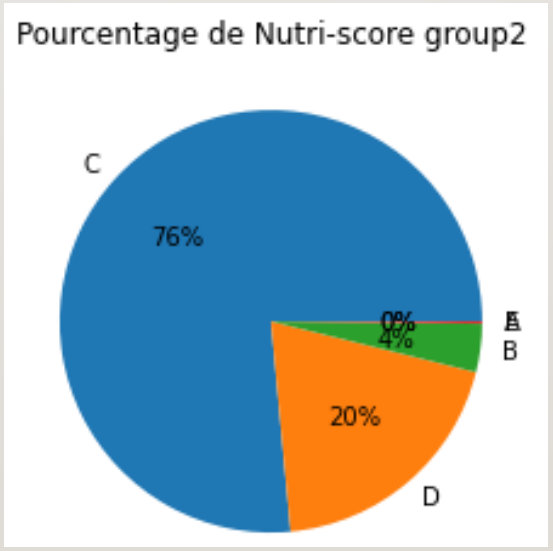


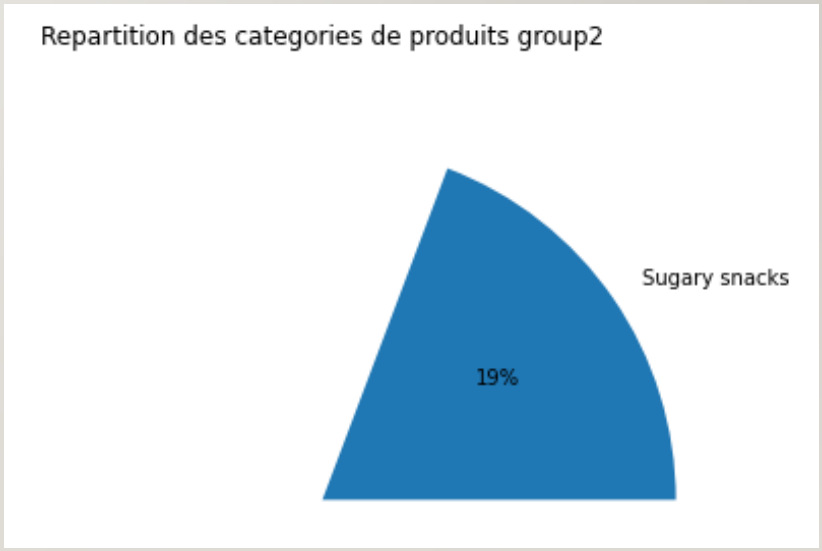


3-2 GROUPES NON- RECOMMANDÉS

- Group 2 : très riche en sucre et glucides (non-recommandé)

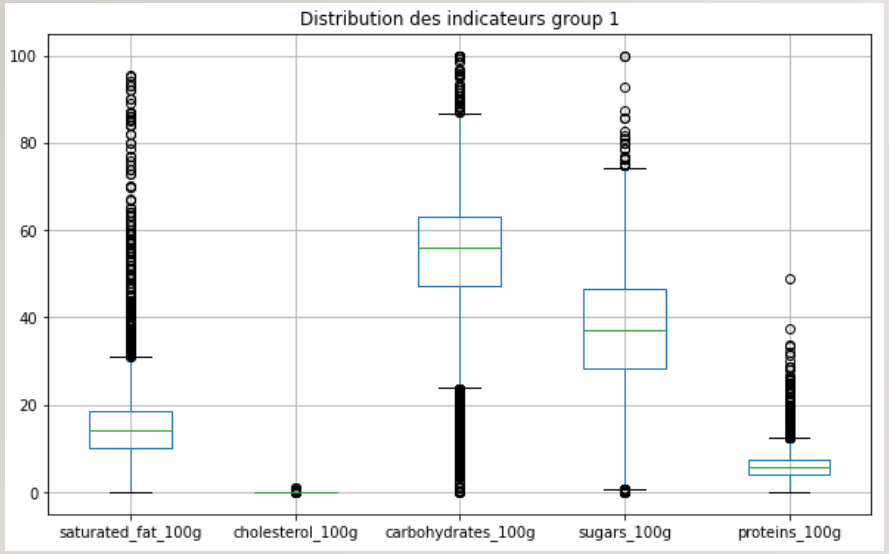


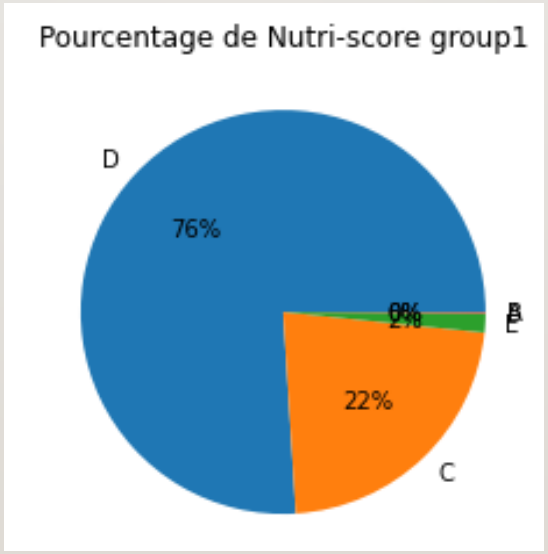




3-2 GROUPES NON- RECOMMANDÉS

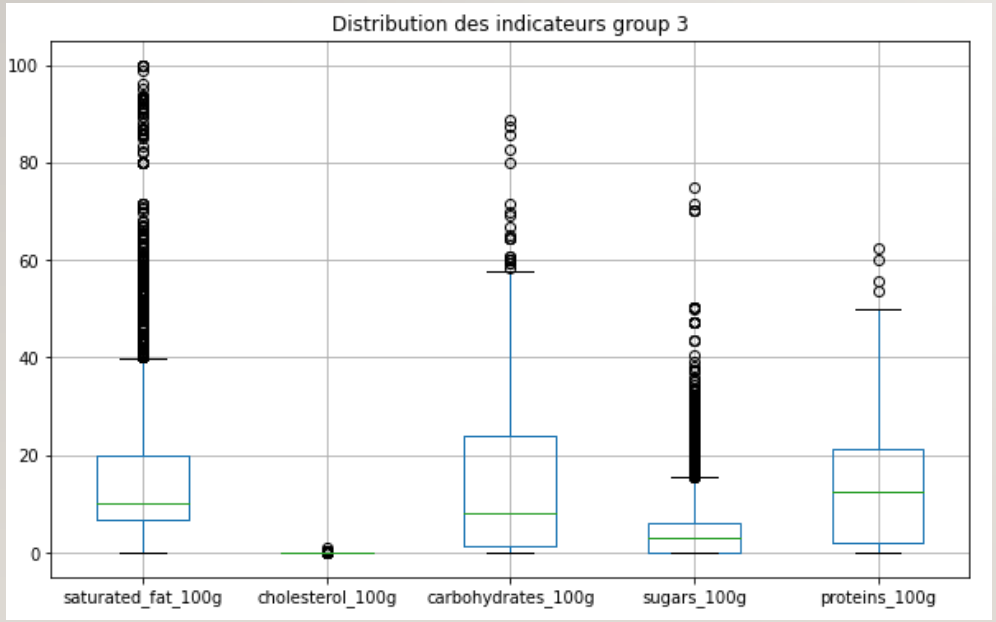
- Group 1 Très riche en glucides et sucres (non-recommandé)

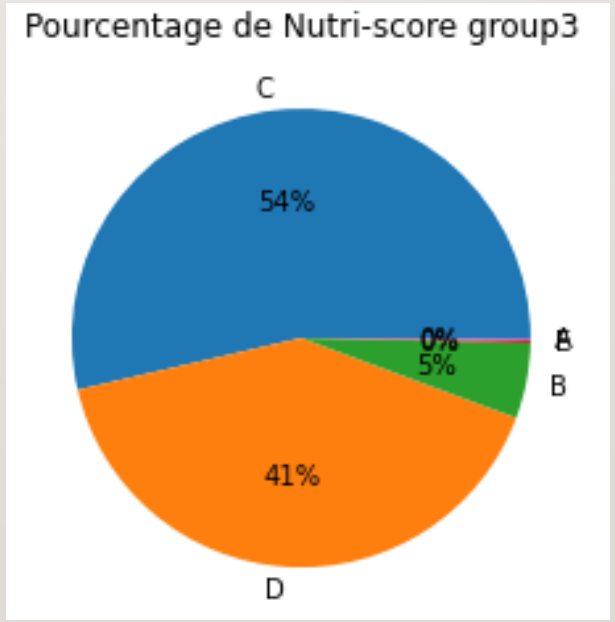


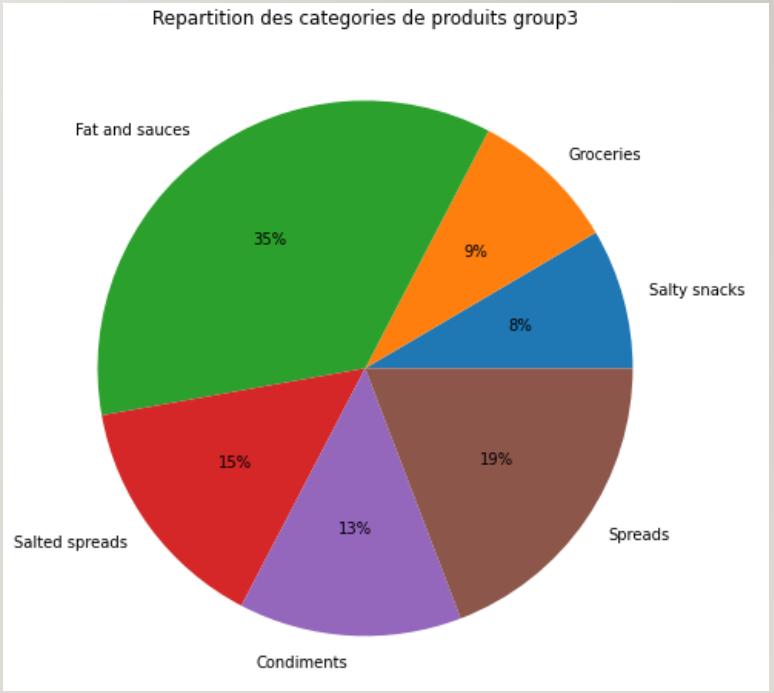


3-2 GROUPES NON-RECOMMANDÉS

- Group 3 très riche en matières grasses et glucides (non-recommande)







4- PRÉSENTATION DE L'APPLICATION

- Permet de faire une recherche par code produit
- Donne les recommandations pour ce produit
  - recommandé,
  - moyennement recommandé (faible en sucre et matières grasses)
    - riche en glucide ou
    - riche en protéine
  - non- recommandé
- Donne une liste de produit de la même catégorie si le produit est
  - moyennement recommandé,
  - non- recommandé

4- PRÉSENTATION DE L'APPLICATION

```
Entrer le code du produit
```

Test produit label 0:

```
1 recomandproduit()

Entrer le code du produit3021760285220
Ce produit est bon pour vous
```

Exemple d'un produit recommandé

Test produit label 2

```
1 recomandproduit()

Entrer le code du produit3045320001525
Attention ce produit n'est pas du tout recommende!
Essayer un produit de la liste suivante
10                Breaktime, Chocolate Chip Cookies
41                                  Shortcake Cookies
45                     Lindt Les Grandes 32% Amandes
```

Exemple d'un produit moyennement recommandé

Test produit label 4:

```
1 recomandproduit()
2

Entrer le code du produit3250390008354
Ce produit est bon mais riche en glucide
Si vous n'etes pas tres actif, essayez un produit de cette liste:
637                                    SNICKERS
757                                 Mini Almond
915                  Helado de Fresa Natural
1909                      Smarties fun cones
2169    Mousse au chocolat aux œufs frais
```

Exemple d'un produit non recommandé

MISE A JOUR DE LA CLASSIFICATION

- Pour les nouveaux produits :
  - Entrainement d'un classificateur ensembliste supervisé sur le jeu de données avec les 'labels'
  - Prédiction des 'labels' pour les nouveaux produits

Merci!