Université Paris 8 - Vincennes à Saint-Denis

Licence informatique & vidéoludisme

Rapport - Ingénierie des langues

Rayane MALIK

Date de rendu : le 16/04/2024

INDEX:

Introduction:

Il s'agit d'un projet d'ingénierie des langues , développé dans le cadre de la Licence Informatique de l'Université Paris 8, visant à extraire des données d'un site pour ensuite les utiliser dans le cas d'un apprentissage.

Pour ma part j'ai extrait les valeurs nutritionnelles de différents aliments que l'on peut retrouver sur le site "www.lanutrition.fr/" que j'ai classifié par type pour constituer une base de données plus solide.

Suite à ça, cette base de données est utilisée par l'algorithme K-MEANS pour représenter les aliments en fonction de leurs taux de protéines en fonction des calories.

Chapitre 1: Organisation des données

1.1: Scrapper.py

Permet premièrement de parcourir la page “les-aliments-a-la-loupe” du site “lanutrition.fr” pour récolter tous les liens présents sur cette page (get_links_from_page(start_url)) en les stockant dans un “set()”. Suite à cela nous parcourons le fameux “set()” pour scrapper les données qui nous intéressent sur chacune des pages (scraper(liens))

```
# URL de départ
start_url = "https://www.lanutrition.fr/les-aliments-a-la-loupe"

filename = "../data/scrapped.txt"
#with open(filename,"w") as file:
file = open(filename,"w")

# Obtenir les liens de la page de départ
links_from_start_page = get_links_from_page(start_url)
#print(links_from_start_page)
test = 0
#scrappe toutes les données nécéssaires
for liens in links_from_start_page:
    scraper(liens)
    test += 1
    print(test," | ",liens)
print("Toutes les données ont été extraites avec succès dans le fichier texte !")
```

1.2: Parser.py

Nous permet de récupérer le fichier texte précédemment créé et de le formater pour un enregistrement en xml, c’est dans cette partie que nous ajoutons un type aux aliments dans la balise <aliment> via l’utilisation d’un dictionnaire et de différentes règles.

```python
# Fonction pour attribuer un type à l'aliment en fonction de son nom
Codeium: Refactor | Explain | Generate Docstring | ×
def determiner_type_aliment(nom_aliment):
    types = {
        'fruit': ['pêche', 'fraise', 'banane', 'pomme', 'orange', 'raisin', 'cerise', 'poire', 'grenade', 'mangue',
                  'ananas', 'kiwi', 'melon', 'papaye', 'framboise', 'myrtille', 'groseille', 'mure', 'citron',
                  'abricot', 'figue', 'prune', 'cassis', 'litchi', 'nèfle', 'fruit', 'goyave', 'pamplemousse',
                  'canneberge', 'nectarine', 'kaki', 'avocat', 'pomelo', 'coing', 'datte', 'pastèque', 'olive',
                  'mandarine'],
        'spice': ['paprika', 'curry', 'cumin', 'cannelle', 'muscade', 'gingembre', 'safran', 'poivre', 'cardamome',
                  'coriandre', 'curcuma', 'anis', 'aneth', 'sel', 'romarin', 'graine', 'basilic', 'pétoncles',
                  'persil', 'câpre','piment', 'ail', 'thym', 'estragon','laurier', 'cari'],
        'sauce': ['mornay', 'béarnaise', 'hollandaise', 'tomate', 'barbecue', 'vinaigrette', 'mayonnaise', 'soja',
                  'pesto', 'sriracha', 'hoisin', 'sauce', 'vinaigre', 'tabasco', 'moutarde'],
```

```python
aliment.strip():

    # Recherche et écriture des données
    nom_aliment = re.search(r'Aliment: (.+)', aliment).group(1)
    type_aliment = determiner_type_aliment(nom_aliment)
    #print(type_aliment)
    nom_aliment = escape(nom_aliment)  # Remplacer l'apostrophe par &apos;
    f.write('    <aliment type="{}">\n'.format(type_aliment))
    f.write('        <nom>{}</nom>\n'.format(nom_aliment))
```

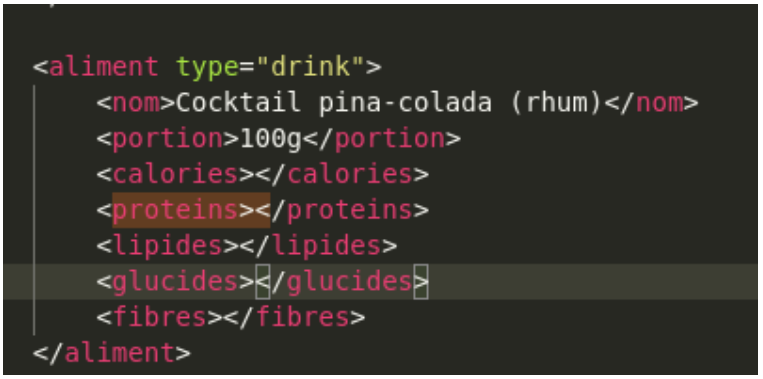
1.3: Analyse.py

Nous permet d’avoir une représentation plus lisible pour l'œil humain en prenant le fichier xml précédemment créé et en représentant les différents aliments par type dans un fichier texte.

```
vegetable:
	Chou gras cuit, Betterave, Haricots blancs petits (conserve), Rutabaga, Haricots rouges maison, Pois cassés
	Pois cajan, Gratin de pommes de terre, Cornichons sucrés, Artichaut cuit, Chou rouge cru, Fèves de soja cui
	Chou vert frisé cru, Courge, Échalote crue, Céleri-rave cru, Céleri-rave cuit, Laitue romaine, Chou-rave cr
	Chou de Chine cru, Céleri cru, Cresson de fontaine cru, Chou-rave cuit, Aubergine, Chou de Chine cuit, Fève
	Cresson cuit, Carotte crue, Pois pigeon (cajan), Choucroute, Petits pois et carottes (congelés), Fèves de s
	Graine de courge rôtie, Haricots beurre cuits, Purée de pommes de terre maison, Poireaux crus, Courgette cu
	Purée de pommes de terre, Épinard (conserve), Courgette crue, Haricots pinto (conserve), Pois mange-tout, H
	Champignon, Chou vert frisé congelé, Radis orientaux crus (daikons), Oignon cru, Pois mange-tout crus, Épin
	Haricots blancs maison, Navet, Haricots mungo germés cuits, Endive, Rhubarbe (compote), Patate douce bouill
meat:
	Roulé de poulet, Bœuf froid tranché (fin), Salami cuit, Autruche, Saucisse de dinde, Saucisse de bologne (b
	Ris de veau, Salami de dinde cuit, Rognon de bœuf, Lapin mijoté, Bouillon ou consommé de bœuf (déshydraté),
	Palette de veau, Veau haché, Saucisson de foie et de bacon, Veau maigre, Brochette d'agneau, Poulet frit (s
	Salami sec, Saucisse de bœuf, Poulet mijoté, Poivrons farcis (bœuf), Saucisse bratwurst, Oie rôtie avec la
	Saucisse de Morteau, Rognon d'agneau, Saucisse porc et bœuf, Cœur de veau, Jambon, Bœuf haché à point, Rôti
	Saucisse de Toulouse, Chipolata, Blanquette de veau, Gigot d'agneau, Foie de bœuf, Queue de porc, Saucisse
	Saucisse de Strasbourg, Pilon de poulet pané et frit, Saucisse fumée, Langue de veau, Saucisse de bière (bœ
	Burrito jambon et fromage, Tripes de boeuf, Blanc de dinde rôtie avec la peau, Cœur de bœuf, Epaule de porc
	Cervelle de bœuf, Poulet rôti (sans peau), Pieds de porc, Veau gras, Bœuf haché cru,
sauce:
	Spaghetti en sauce, Salade de légumes sans sauce, Vinaigrette maison, Graine de soja rôtie, Tomate rouge d'
	Sauce tartare, Tamari (sauce soja), Mayonnaise commerciale allégée, Sauce hollandaise, Sauce blanche, Sauce
	Moutarde jaune (graines), Sauce tomate, Mayonnaise commerciale, Vinaigrette commerciale, Tomate farcie,
```

1.4: To_csv.py

Prend le fichier xml en argument et crée un fichier en csv pour l'utilisation ultérieure de l'algorithme kmean, remplace aussi les champs vide par des 0 et retire les unités de mesure pour des donnés plus simples à traiter.




```
Haricots rouges maison,vegetable,100,124,9.13,0.09,22.41,9.3
Chili con carne (conserve),autre,100,121,7.09,5.26,11.43,3.9
Pois cassés,vegetable,100,118,8.34,0.39,21.1,2.92
Cocktail pina-colada (rhum),drink,100,0,0,0,0,0
Merlan (lieu noir),seafood,100,118,24.92,1.26,0,0
Salade de légumes sans sauce,sauce,100,16,1.25,0.07,3.22,0.74
```

Chapitre 2:Algorithme de clustering K-means

2.1 Kmean.py

Contient l'implémentation de l'algorithme de clustering K-means, permettant d'offrir les fonctionnalités nécessaires pour le clustering des données.
Voici les differentes fonctions

- ❖ tmp_centre(): génère aléatoirement k centres à partir des données fournies.
- ❖ euclidian_distance(): Calcule la distance euclidienne entre deux points de données.
- ❖ cluster_association(): Associe chaque point de données(aliment) à son centre le plus proche, formant ainsi des clusters.
- ❖ definitive_centers(): Détermine les centres des clusters en calculant la moyenne des points de données(aliments) dans chaque cluster.
- ❖ calcul_sse(): Calcule la somme des carrés des erreurs (SSE) pour évaluer la qualité du clustering. Plus celle-ci est basse et plus le clustering est qualitatif

2.2 Kmoyen.py

Nous permet de visualiser les aliments par des points de couleur sur un graphe indiquant leurs taux de protéines en fonction du nombre de calories présents dans une portion de 100g.

- ❖ Exécute l'algorithme kmeans via l'utilisation des donnés du csv créé lors du script to_csv.py en utilisant les fonctions présentes dans kmeans.py.
- ❖ Suite à celà; affiche les résultats obtenus lors de l'analyse via deux graphiques, l'un par cluster et l'autre par type d'aliment.
- ❖ En fonction du nombre de clusters choisi par l'utilisateur lors de l'exécution du programme; la vue du graphe par cluster est différente.

Via la vue par aliments nous pouvons voir que certaines tendances existent au niveau des aliments concernant le taux de protéines et le nombre de calories. Dans le cas d'une personne recherchant une diète basse en calorie et hautes en protéines nous pouvons analyser la vue par type pour voir si des tendances existent au niveau des aliments. Nous pouvons voir que les aliments de types "seafood" et "meat" proposent une quantité de protéines plutôt élevée par rapport à leurs taux de calories en comparaison avec les "nuts" qui eux contiennent aussi beaucoup de protéines mais en contrepartie sont très caloriques.

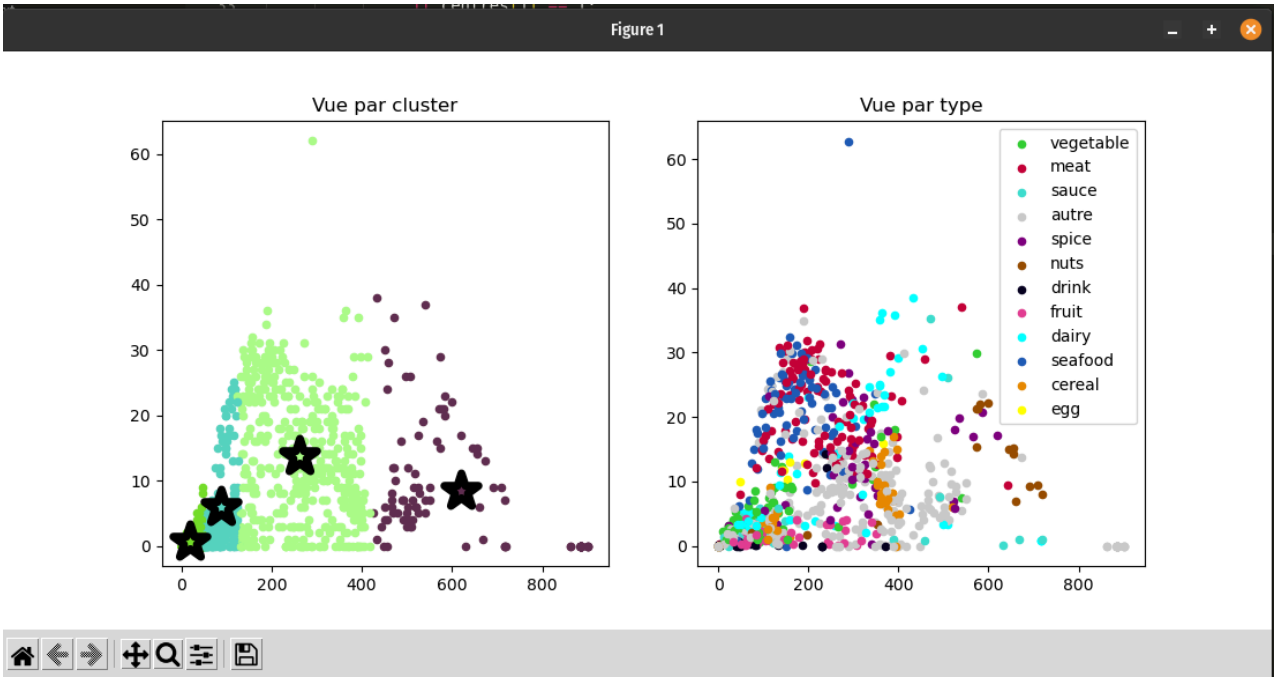


Figure 1
Vue par cluster
Vue par type
vegetable
meat
sauce
autre
spice
nuts
drink
fruit
dairy
seafood
cereal
egg
0
10
20
30
40
50
60
0
200
400
600
800