

de vérifier les identifiants. Si l'authentification réussit, l'utilisateur est redirigé vers la page d'historique de produits scannés et les données d'authentification temporaires sont détruites. En cas d'échec, un message d'erreur est affiché à l'utilisateur. Ce processus assure la vérification et la gestion sécurisée des informations d'identification.

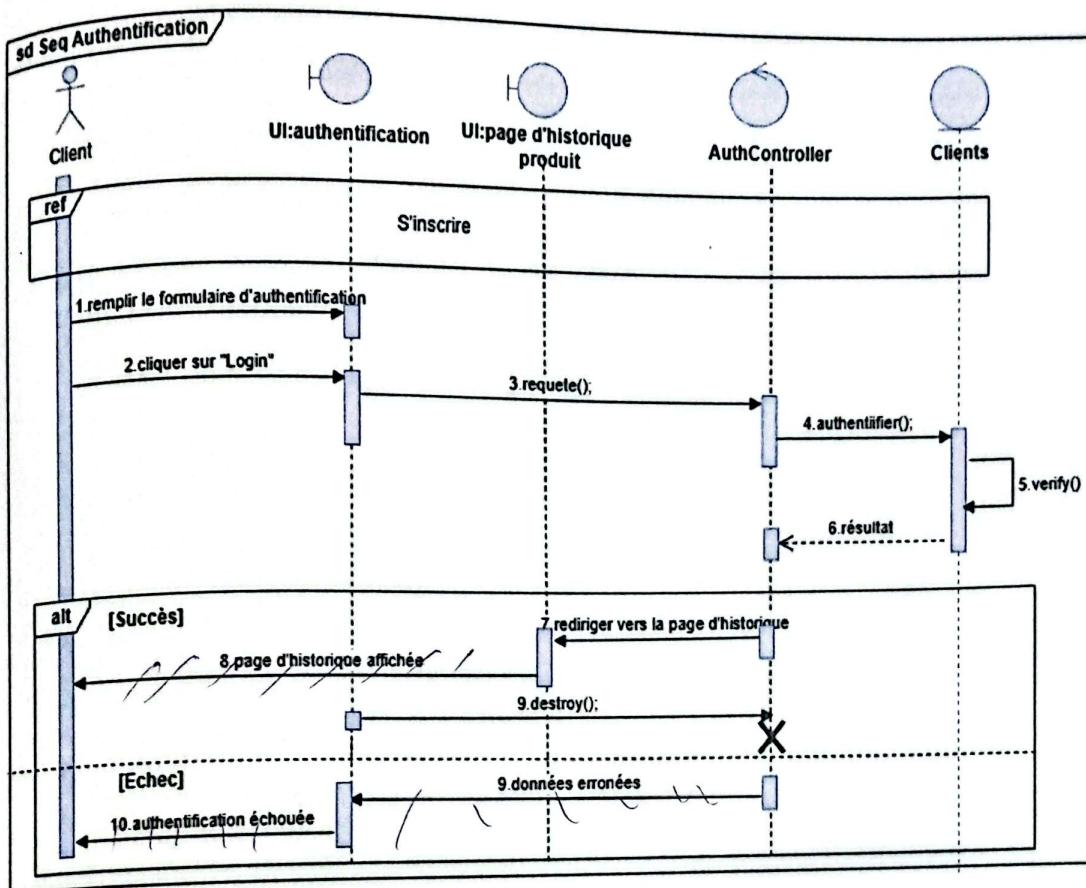


Figure 2.16: Diagramme de séquence du cas d'utilisation « s'authentifier »

2.4.2 Diagramme de séquence du cas d'utilisation « scanner un produit »

Le client doit scanner le produit pour consulter ses détails et obtenir des recommandations personnalisées. Il clique sur l'icône "scan", puis sur le bouton "scan your product", et scanne le code-barres. Une fenêtre modale affiche alors le produit scanné avec des produits similaires recommandés. Le client peut cliquer sur la photo du produit pour en voir les détails. Si le produit n'est pas trouvé, un popup indique "produit non trouvé" et propose d'ajouter le produit en cliquant sur un bouton "ajouter produit".

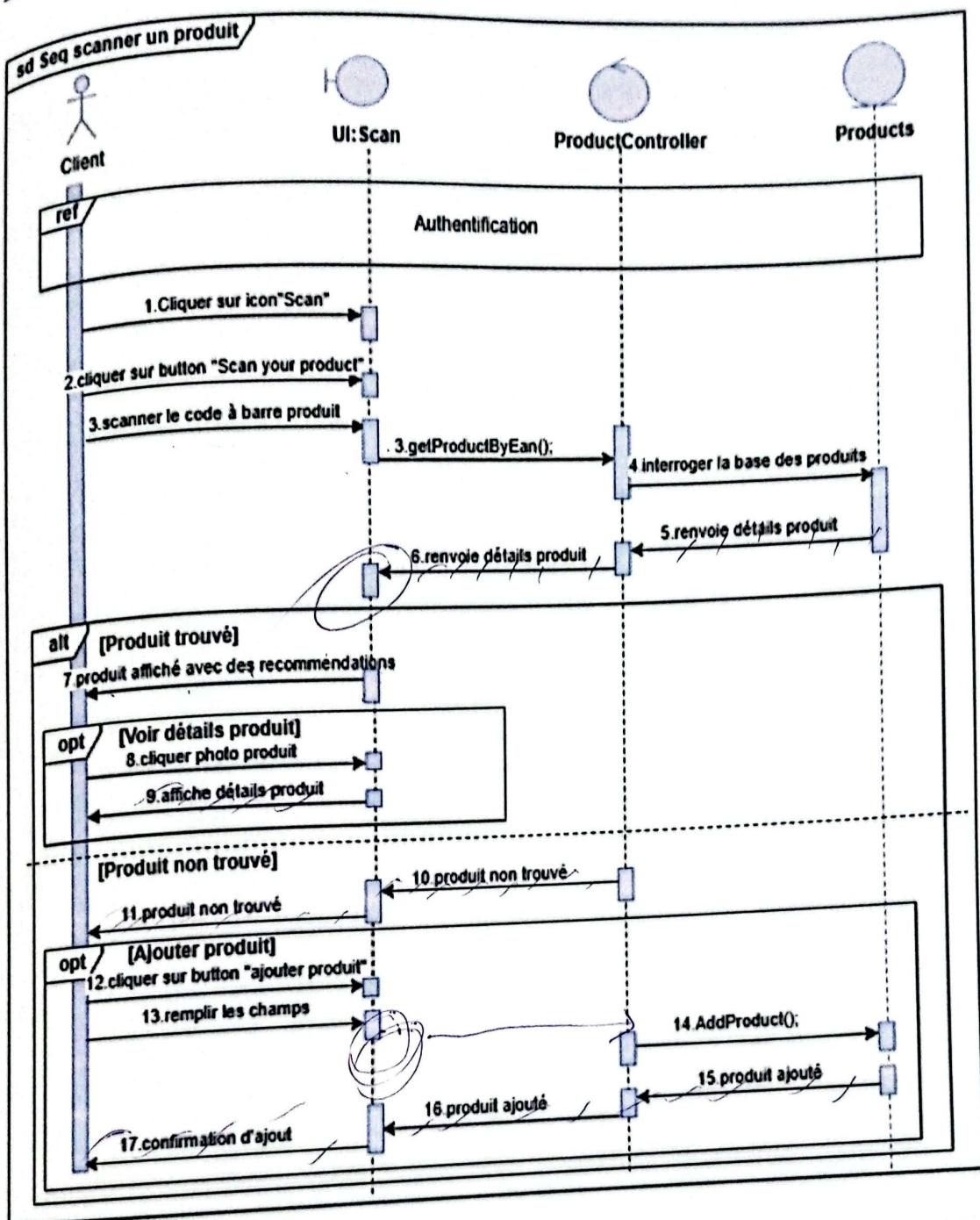


Figure 2.17: Diagramme de séquence du cas d'utilisation « scanner un produit »

Conclusion

Dans ce chapitre, nous avons mis l'accent sur la conception de notre solution. Nous avons présenté le diagramme de cas d'utilisation général et raffiné pour chaque acteur, ainsi que la description de l'aspect statique à l'aide du diagramme de classes et l'aspect dynamique à l'aide

SYSTEME DE RECOMMENDATION

Contents

3.1	Présentation de la solution	34
3.2	Choix Méthodologique	34
3.3	Étapes de la Méthodologie CRISP	35
3.3.1	Compréhension du Problème	35
3.3.2	Connaissance des Données	36
3.3.3	Préparation des Données	36
3.3.4	Modélisation	43
3.3.5	Evaluation	46
3.3.6	Déploiement	50

Introduction

Dans ce chapitre, nous avons développé un système de recommandation alimentaire intelligent basé sur les préférences nutritionnelles des utilisateurs. Pour ce faire, nous avons utilisé la méthodologie CRISP, la bibliothèque scikit-surprise et des algorithmes Machine Learning pour construire notre modèle.

3.1 Présentation de la solution

Notre solution consiste à développer un système de recommandation alimentaire intelligent qui propose des produits alimentaires en fonction des préférences nutritionnelles et du profil de santé de l'utilisateur, tels que normal, diabétique ou hypertendu. En utilisant la bibliothèque scikit-surprise et des algorithmes machine learning, nous avons conçu un modèle capable de personnaliser les recommandations pour chaque utilisateur. Les données nécessaires pour entraîner ce modèle ont été obtenues par web scraping à partir du site de Carrefour France, incluant des informations détaillées sur les valeurs nutritionnelles des produits. Le système analyse les caractéristiques nutritionnelles des produits ainsi que les préférences et les besoins spécifiques des utilisateurs pour générer des recommandations précises et pertinentes. Grâce à cette approche, nous visons à aider les utilisateurs à faire des choix alimentaires plus sains et adaptés à leur condition médicale, améliorant ainsi leur bien-être général.

3.2 Choix Méthodologique

La méthodologie **CRISP** est un processus standardisé utilisé pour la réalisation de projets de data mining et d'analyse prédictive. Cette méthodologie offre une approche systématique et structurée pour traiter les problèmes complexes de data mining, en guidant les praticiens à travers différentes phases, de la compréhension du problème à la mise en œuvre de solutions. Son importance réside dans sa capacité à fournir un cadre robuste pour gérer l'ensemble du

processus de développement, en favorisant la répétabilité, la transparence et la rigueur dans la démarche analytique[3].

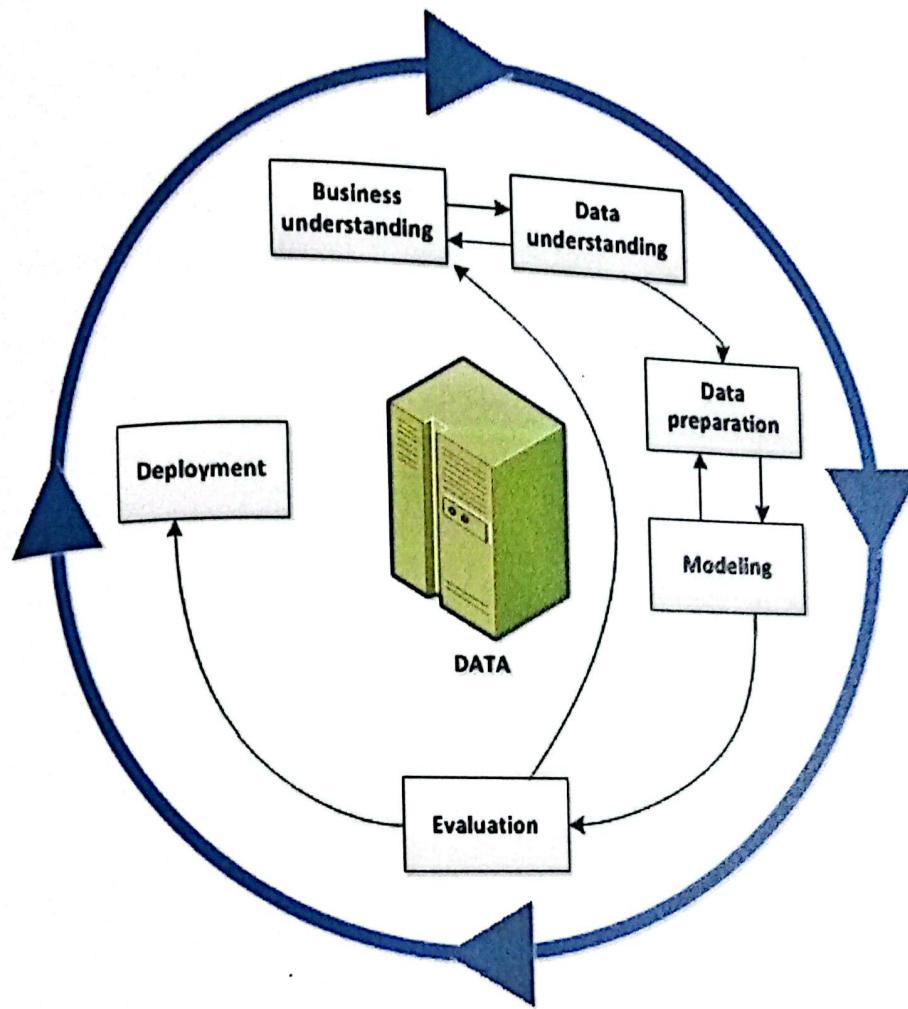


Figure 3.1: La méthode CRISP

3.3 Étapes de la Méthodologie CRISP

La méthodologie CRISP se compose de six étapes principales :

3.3.1 Compréhension du Problème

Cette étape consiste à définir clairement les objectifs du projet et à identifier les problèmes que le système de recommandation alimentaire intelligent doit résoudre. Il s'agit de comprendre les

bsu import beautifulsoup
request.

Octoparse

Web scraper + category graph.

{ workflow : sitemaps.

besoins des utilisateurs finaux et les exigences métier, notamment la nécessité de recommander des produits alimentaires adaptés aux profils nutritionnels spécifiques (normal, diabète, hypertension).

3.3.2 Connaissance des Données

Cette étape consiste à collecter et préparer les données qui seront utilisées pour entraîner le système de recommandation. Pour notre projet, les données ont été scrapées à partir du site de Carrefour France via des techniques de Web scraping. Les données sont réelles, ils comprennent des informations détaillées sur les produits, y compris les valeurs nutritionnelles telles que les calories, les glucides, les lipides, les protéines, ainsi que des ingrédients et les allergènes. La figure 3.2 présente les fichiers json de données de scrapées.

```

{
  "ean": "3560070325006",
  "brand": "CARREFOUR",
  "business_type": "food",
  "categories": [],
  "origin": [],
  "unit_of_measure": "kg",
  "freshness": [],
  "market": [
    {
      "name": "carrefour",
      "country": "FR"
    }
  ],
  "lang_desc": [],
  "evolutions": [],
  "label": [],
  "created_at": "2024-03-09T10:07",
  "updated_at": "2024-03-09T10:07"
}

{
  "ean": "3560070198504",
  "brand": "CARREFOUR",
  "business_type": "food",
  "categories": [],
  "origin": []
}

```

Figure 3.2: Les données scrapées de site Carrefour France

3.3.3 Préparation des Données

Une fois les données collectées et comprises, il est nécessaire de passer à l'étape de préparation des données. Cette phase comprend toutes les activités nécessaires à la préparation des données

merged Island
mapping (lefton, right)

Filtrage Collaboratif \rightarrow interactie user profiel
met de profiel similar
percentage \Rightarrow en considerat le nombre
de profiel.

SYSTEME DE RECOMMANDATION

qui seront utilisées dans la phase de modélisation, telles que le nettoyage des données, la gestion des valeurs manquantes et la normalisation des caractéristiques nutritionnelles.

• Nettoyage des données

Le scraper nous a fourni des informations complètes sur les produits, incluant de nombreuses colonnes non pertinentes. Nous allons extraire uniquement les colonnes pertinentes pour les produits afin de simplifier et de focaliser notre analyse.

	eан	brand	business_type	categories	origin	unit_of_measure	freshness	market	long_desc	evolutions	label	created_at	updated_at
0	8710438123937	LU	food	[{"id": "2074", "label": "Surgelés"}, {"id": "..."}]	(ean: [N/A])	KG	(value: 3, period: week)	(name: carrefour, country: FR)	{fr (title: Frites golden numero MCCAN)}	[{"purchasing_date": 2024-03-08, format...}	(frozen: True)	2024-03-08 07:04:00	2024-03-08 07:04:00
1	8710438110012	MCCAIN	food	[{"id": "2074", "label": "Surgelés"}, {"id": "..."}]	(ean: [N/A], parsel: FR)	KG	(value: 3, period: week)	(name: carrefour, country: FR)	{fr (title: Frites du rond MCCAIN)}	[{"purchasing_date": 2024-03-08, format...}	(frozen: True)	2024-03-08 07:04:00	2024-03-08 07:04:00
2	8710438107975	MCCAIN	food	[{"id": "2074", "label": "Surgelés"}, {"id": "..."}]	(ean: [N/A])	KG	(value: 3, period: week)	(name: carrefour, country: FR)	{fr (title: Frites côte courante MCCAIN)}	[{"purchasing_date": 2024-03-08, format...}	(frozen: True)	2024-03-08 07:04:00	2024-03-08 07:04:00
3	8710438123876	TRADITION	food	[{"id": "2074", "label": "Surgelés"}, {"id": "..."}]	(ean: [N/A])	KG	(value: 3, period: week)	(name: carrefour, country: FR)	{fr (title: Frites tradition MCCAIN)}	[{"purchasing_date": 2024-03-08, format...}	(frozen: True)	2024-03-08 07:04:00	2024-03-08 07:04:00
4	8710438123272	MCCAIN	food	[{"id": "2074", "label": "Surgelés"}, {"id": "..."}]	(ean: [N/A])	KG	(value: 3, period: week)	(name: carrefour, country: FR)	{fr (title: Frites au four et fromage L)}	[{"purchasing_date": 2024-03-08, format...}	(frozen: True)	2024-03-08 07:05:00	2024-03-08 07:05:00
-	-	-	-	-	-	-	-	-	-	-	-	-	-
6	3523480457454	CARREFOUR	food	[{"id": "1952", "label": "Pain et Pâtisseries..."}, {"id": "..."}]	(ean: [FR, MC])	KG	0	(name: carrefour, country: FR)	{fr (title: Broche rôti au nature CARREFOUR)}	[{"purchasing_date": 2024-03-08, format...}	0	2024-03-08 10:51:00	2024-03-08 10:51:00
7	3523480457492	CARREFOUR	food	[{"id": "1952", "label": "Pain et Pâtisseries..."}, {"id": "..."}]	(ean: [FR, MC])	KG	0	(name: carrefour, country: FR)	{fr (title: Broche à l'assaisonnement CARREFOUR)}	[{"purchasing_date": 2024-03-08, format...}	0	2024-03-08 10:51:00	2024-03-08 10:51:00
8	3276550577195	CARREFOUR	food	[{"id": "1952", "label": "Pain et Pâtisseries..."}, {"id": "..."}]	(ean: [FR, MC])	KG	0	(name: carrefour, country: FR)	{fr (title: Galette des rois framboise)}	[{"purchasing_date": 2024-03-08, format...}	0	2024-03-08 10:51:00	2024-03-08 10:51:00
9	3307483011125	CASTELANE	food	[{"id": "1952", "label": "Pain et Pâtisseries..."}, {"id": "..."}]	(ean: [FR, MC])	KG	(value: 5, period: day)	(name: carrefour, country: FR)	{fr (title: Galette pomme à l'huile d'olive)}	[{"purchasing_date": 2024-03-08, format...}, {"fr_grocery": True}]	0	2024-03-08 10:51:00	2024-03-08 10:51:00
10	3276550598268	CARREFOUR	food	[{"id": "1952", "label": "Pain et Pâtisseries..."}, {"id": "..."}]	(ean: [FR, MC])	U	0	(name: carrefour, country: FR)	{fr (title: Galette des rois pomme vanille)}	[{"purchasing_date": 2024-03-08, format...}]	0	2024-03-08 10:52:00	2024-03-08 10:52:00

Figure 3.3: Les données initiales

Nous avons procédé à un nettoyage approfondi. Pour ce faire, nous avons supprimé les colonnes contenant des produits avec des codes EAN dupliqués afin d'éliminer les redondances et d'assurer la qualité des données.

Nous avons également décidé aussi de supprimer les colonnes où les valeurs nutritionnelles sont vides. Les informations nutritionnelles étaient initialement situées sous la colonne principale "évolutions". Nous avons extrait ces informations et les avons réorganisées dans une colonne distincte "nutrition", sous forme de dictionnaire.

SYSTEME DE RECOMMANDATION

	code	produit_name	catégories	catégories_1	catégories_2	catégories_3	catégories_4	ingrédients	allergens	nutrition	nut_label
1	87104581122937	Frites poivreé MUSSEAU MCCAIN	Burgétois	Frites et Pommes de terre	Frites et Pommes de terre	Frites pour friture	Frites pour friture	Pommes de terre (98%), huile de tournesol (1%)		Valeur énergétique (kJ)	A
1	8710458110512	Frites du nord MCCAIN	Burgétois	Frites et Pommes de terre	Frites pour friture	Frites pour friture	Frites pour friture	Pommes de terre (98,5%), huile de tournesol (1%)		Valeur énergétique (kJ)	A
1	8710458107978	Frites côté couronne MCCAIN	Burgétois	Frites et Pommes de terre	Frites pour friture	Frites pour friture	Frites pour friture	Pommes de terre (98%), huile de tournesol, sel		Valeur énergétique (kJ)	A
1	8710458123878	Frites tradition MCCAIN	Burgétois	Frites et Pommes de terre	Frites pour friture	Frites pour friture	Frites pour friture	Pommes de terre (97%), huile de tournesol, sel		Valeur énergétique (kJ)	A
1	8710458125272	Frites au Four et Sérvielle la Rote classique	Burgétois	Frites et Pommes de terre	Frites au four	Frites pour friture	Frites au four	Pommes de terre (98,5%) huile de tournesol (1,5%)		Valeur énergétique (kJ)	A
10635	5123620441293	Rôti ferci saumon et St-Jacques CARREFOUR LE M.	Viandes et Poissons	Poissonnerie	Poissons panés et cuiturés			SALMON atlantique 33% (filets 30%, poitrine 21%, e.		Valeur énergétique (kJ)	A
10636	8436648070442	Saumon au curry	Viandes et Poissons	Poissonnerie	Poissons panés et cuiturés			Saumon (POISSON) 80%, huile de tournesol, sel		Valeur énergétique (kJ)	A
10637	3523680429429	Tartare de saumon saupoudré vertes et cépées CA	Viandes et Poissons	Poissonnerie	Poissons panés et cuiturés			Tartare de SALMON 75%		Valeur énergétique (kJ)	A
10638	5123620470992	Brochettes crevettes sauce provençale CARREFOUR	Viandes et Poissons	Poissonnerie	Poissons panés et cuiturés			(SALMON atlantique 68%, CREVETTES déshéve 20%		Valeur énergétique (kJ)	A
10639	327655555488	Loin de thon allumette débarrassée	Viandes et Poissons	Poissonnerie	Poissonnerie à la coupe	Filets et pavés	THON ALBACORE (85%), huile végétale (1,5%) sucre de		Valeur énergétique (kJ)	A	
10632 (nouvelles + 10 colonnes)											

Figure 3.4: Données après suppression duplication

- Normalisation et transformation de la colonne "nutrition"

Nous avons dû traiter la colonne "nutrition" de notre base de données vu qu'elle comprenait des informations nutritionnelles détaillées telles que la valeur énergétique, les matières grasses, les glucides, les protéines, et le sel pour chaque produit. Par exemple, voici une entrée de la colonne "nutrition".

```
{'valeur_energetique_kj': '602 kJ / 100 g',
 'valeur_energetique_kcal': '143 kcal / 100 g',
 'matieres_grasses': '4 g / 100 g',
 'acides_gras_satures': '0.4 g / 100 g',
 'acides_gras_mono-insatures': '1.6 g / 100 g',
 'glucides': '23 g / 100 g',
 'sucres': '0.5 g / 100 g',
 'fibres_alimentaires': '2.5 g / 100 g',
 'proteines': '2.5 g / 100 g',
 'sel': '0.1 g / 100 g'}
```

Figure 3.5: Exemple d'une colonne nutrition

Il était nécessaire de normaliser ces données pour garantir qu'elles soient sur la même échelle. La normalisation permet de redimensionner les valeurs des caractéristiques pour qu'elles se situent dans une plage commune. Cela est particulièrement important pour

SYSTEME DE RECOMMENDATION

les algorithmes de recommandation qui sont sensibles aux écarts de grandeurs entre les caractéristiques.

La transformation de la colonne "nutrition" était essentielle pour rendre les données exploitables. Initialement, les informations nutritionnelles étaient stockées sous forme de dictionnaires imbriqués, ce qui compliquait leur utilisation directe dans les modèles de recommandation. Pour résoudre ce problème, nous avons extrait chaque clé du dictionnaire et l'avons convertie en une colonne distincte.

	valore_energetique_kj	valore_energetique_kcal	nutriments_grasses	acides_gras_saturati	acides_gras_insaturati	vitamines	mineraux	sodium_sacres	fibres_alimentariques	proteines	sel	...	ter	valore_energetique_kj
0	602.0	143.0	4.0	2.40	1.4	22.0	0.0	2.0	2.5	8.0	0.0	...	0.0	602.0
1	555.0	132.0	5.0	2.40	1.4	21.0	0.0	3.0	2.5	8.0	0.0	...	0.0	555.0
2	736.0	176.0	6.0	2.70	0.0	26.0	0.0	4.0	2.5	8.0	0.0	...	0.0	736.0
3	569.0	135.0	3.0	2.30	1.2	23.0	0.0	2.0	2.5	8.0	0.0	...	0.0	569.0
4	583.0	139.0	5.0	2.40	1.4	23.0	0.0	2.0	2.5	8.0	0.0	...	0.0	583.0
...
10655	704.0	169.0	11.0	1.60	0.0	2.0	2.5	0.0	15.0	44.0	0.0	...	0.0	704.0
10656	660.0	158.0	6.0	1.20	0.0	0.0	0.0	0.0	18.0	3.00	0.0	...	0.0	660.0
10657	769.0	185.0	13.0	1.80	0.0	1.0	1.0	0.0	15.0	3.00	0.0	...	0.0	769.0
10658	633.0	154.0	6.0	1.40	0.0	1.0	0.0	0.0	16.0	1.50	0.0	...	0.0	633.0
10659	432.0	102.0	1.2	0.40	0.0	0.0	0.0	0.0	22.0	1.70	0.0	...	0.0	432.0
10660 rows × 29 columns														

Figure 3.6: Normalisation et transformation de la colonne "nutrition"

Nous utilisons standardisation.

• Prediction allergénés

Nous avons constaté que la colonne "allergens" contient de nombreuses listes vides. Pour remédier à cela, avons mis en place un modèle de machine learning capable de prédire les allergènes d'un produit en se basant sur son nom et ses ingrédients.

SYSTEME DE RECOMMANDATION

Figure 3.7: Données après nettoyage la colonne nutrition

Nous avons d'abord nettoyé les colonnes des ingrédients en supprimant les espaces superflus et en convertissant toutes les lettres en minuscules. Cette normalisation permet d'uniformiser les données et de faciliter leur traitement.

✓ Cleaning ingredients column

```
original_expanded_data['ingredients'] = original_expanded_data['ingredients'].apply(lambda x: x.replace('*', '')).replace('\n', ' ')
```

Figure 3.8: Nettoyage de colonne des ingrédients

Chaque liste d'ingrédients a été convertie en une seule chaîne de caractères. Pour ce faire, nous avons utilisé le `TfidfVectorizer`, une méthode qui transforme les textes en vecteurs de caractéristiques en tenant compte de la fréquence des termes et de leur importance dans le corpus. Cette étape est cruciale pour convertir les données textuelles en une forme exploitable par les algorithmes de machine learning.

Nous avons choisi d'utiliser un algorithme d'apprentissage supervisé, à savoir l'arbre de décision. Ce choix est motivé par la capacité des arbres de décision à gérer des données complexes et à fournir des résultats interprétables.

SYSTEME DE RECOMMENDATION

Les données ont été divisées en ensembles d'entraînement et de test afin d'évaluer les performances du modèle. Cette division permet de s'assurer que le modèle peut généraliser les connaissances acquises à de nouvelles données non vues pendant l'entraînement.

Nous avons entraîné l'arbre de décision sur l'ensemble d'entraînement et évalué ses performances sur l'ensemble de test. Les métriques utilisées pour évaluer le modèle incluent la précision, le rappel, le score F1 et le support. Ces métriques fournissent une vue d'ensemble de la performance du modèle en termes de classification des allergènes.

Ci-dessous, la figure montre la performance du modèle en fonction des métriques utilisées.

	precision	recall	f1-score	support
blé	1.00	1.00	1.00	2019
gluten	0.99	0.97	0.98	78
gluten,blé	0.99	1.00	0.99	78
gluten,blé,mollusques	1.00	0.96	0.98	23
gluten,blé,noisette	1.00	1.00	1.00	1
gluten,mollusques	0.00	0.00	0.00	1
gluten,noisettes,noisette	1.00	1.00	1.00	1
lait	1.00	1.00	1.00	569
lait,blé	0.97	1.00	0.99	39
lait,blé,mollusques	0.00	0.00	0.00	1
lait,blé,noisettes,noisette	1.00	0.75	0.86	4
lait,gluten	0.98	1.00	0.99	52
lait,gluten,blé	1.00	1.00	1.00	19
lait,gluten,mollusques	1.00	1.00	1.00	1
lait,gluten,noisette	0.00	0.00	0.00	1
lait,gluten,noisettes,noisette	1.00	1.00	1.00	14
lait,laits	1.00	1.00	1.00	3
lait,milk	0.00	0.00	0.00	1
lait,mollusques	1.00	1.00	1.00	3
lait,noisette	1.00	1.00	1.00	5
lait,noisettes,noisette	0.93	1.00	0.96	13
accuracy				0.98
macro avg	0.69	0.70	0.69	4066
weighted avg	0.98	0.98	0.98	4066

accuracy is 98.20462370890472

Figure 3.9: Performance du modèle de prédiction des allergènes

• Calcul et mise à jour du Nutriscore

La dernière étape consiste à calculer le Nutriscore de chaque produit sur la base de critères nutritionnels spécifiques. Les points sont attribués aux différents composants nutritionnels tels que l'énergie, le sucre, les graisses saturées, le sodium, les protéines et les fibres alimentaires en fonction de seuils prédéfinis. Ces points sont utilisés pour calculer un Nutriscore final, qui est ensuite mis à jour dans l'ensemble de données.

SYSTEME DE RECOMMENDATION

	pts_kj	pts_glus	pts_agr	pts_na	pts_prot	pts_fib	pts_fln	pts_product	score	nutri_score
18640	3	0	5	10	5	0	0	18	18	D
18641	2	0	0	2	5	1	0	4	-2	A
18642	1	0	0	4	5	0	0	5	0	B
18643	2	0	0	5	5	1	0	5	0	B
18644	3	0	4	10	5	0	0	7	1	B
18645	2	0	1	3	5	1	0	17	17	D
18646	3	0	3	3	5	1	0	6	0	B
18647	2	0	1	4	5	0	0	9	4	C
18648	2	0	1	3	5	1	0	7	1	B
18649	2	0	0	1	5	0	0	6	1	B
18650	2	0	0	4	5	1	0	3	-2	A
18651	2	0	0	4	5	1	0	6	0	B
18652	2	0	3	3	5	0	0	8	3	C
18653	2	0	1	4	5	1	0	7	1	B
18654	2	0	1	3	5	0	0	6	1	B
18655	2	0	1	2	5	0	0	5	0	B
18656	1	0	1	3	5	0	0	5	0	B
18657	2	0	1	8	5	0	0	11	11	D
18658	1	0	1	6	5	0	0	8	3	C
18659	1	0	0	7	5	0	0	8	3	C

Figure 3.10: Calcul et mise à jour du Nutriscore

- Sélection des caractéristiques

La sélection des caractéristiques consiste à créer de nouvelles caractéristiques ou à modifier les caractéristiques existantes afin d'améliorer les performances d'un modèle d'apprentissage automatique. performance d'un modèle d'apprentissage automatique. Ici, nous avons combiné les catégories de produits et les ingrédients en une seule colonne afin d'optimiser les résultats de notre modèle de filtrage basé sur le contenu.

```
[ ] # Content-based filtering using product attributes (categories and ingredients)
[ ] Assuming you have already preprocessed categories and ingredients
[ ] Concatenate categories and ingredients into a single text column
products_df['features'] = products_df['categories'] + ' ' + products_df['ingredients']
```

Figure 3.11: Combinaison entre les colonnes catégories et ingrédients

3.3.4 Modélisation

Dans cette partie, nous nous concentrerons sur le développement et la construction des modèles utilisés dans le système de recommandation alimentaire. Nous commencerons par discuter de la sélection d'algorithmes appropriés pour le filtrage collaboratif et le filtrage basé sur le contenu, en soulignant les points forts et l'adéquation de chaque approche pour notre cas d'utilisation spécifique.

- **Entrainement des modeles**

a. **Le filtrage collaboratif:** Le filtrage collaboratif regroupe l'ensemble des méthodes qui visent à construire des systèmes de recommandation utilisant les opinions et évaluations d'un groupe pour aider l'individu. Il recherche les similitudes entre utilisateurs ou entre éléments pour formuler des recommandations basées sur ces similitudes.

Avant de passer à la modélisation, nous devons diviser nos données à l'aide de la technique train-test pour évaluer les performances d'un algorithme d'apprentissage automatique.

```
[38] # Split data into train and test sets
trainset, testset = train_test_split(data, test_size=0.2)
```

Figure 3.12: Technique train-test

- **KNNBasic :** est un algorithme de filtrage collaboratif basé sur l'approche des k-voisins les plus proches. Il prédit les évaluations des articles pour un utilisateur cible en trouvant les k utilisateurs les plus similaires sur la base de leurs évaluations et en agrégeant leurs évaluations pour générer des prédictions.

- Root Mean Square Error : la ^{racine} entre valeurs prédictes et val. observées
- MAE Mean Absolute Error : la moyenne des écarts absolus entre les prédictes et val. réelles.

SYSTEME DE RECOMMENDATION

```
▶ from surprise import accuracy  
  
# Evaluate the model KNN  
predictions = model_KNN.test(testset)  
  
# Calculate RMSE and MAE  
rmse = accuracy.rmse(predictions, verbose=False)  
mae = accuracy.mae(predictions, verbose=False)  
  
print("RMSE:", rmse)  
print("MAE:", mae)  
  
→ RMSE: 0.5067279719303415  
MAE: 0.5001644446082529
```

Figure 3.13: RMSE et MAE de l'algorithme KNNBasic

- **SVD:** est un algorithme de filtrage collaboratif basé sur la factorisation de la matrice. Il décompose la matrice d'interaction en matrices de faible rang à l'aide de la décomposition en valeurs singulières (SVD), puis il fait une approximation de la matrice originale en la reconstruisant à partir des matrices de rang inférieur.

```
▶ from surprise import accuracy  
  
# Evaluate the model SVD  
predictions = model_SVD.test(testset)  
  
# Calculate RMSE and MAE  
rmse = accuracy.rmse(predictions, verbose=False)  
mae = accuracy.mae(predictions, verbose=False)  
  
print("RMSE:", rmse)  
print("MAE:", mae)  
  
→ RMSE: 0.5074791456187236  
MAE: 0.4999530892696848
```

Figure 3.14: RMSE et MAE de l'algorithme SVD

SYSTEME DE RECOMMENDATION

- **NMF:** est un algorithme de filtrage collaboratif basé sur la factorisation des matrices non négatives. Il décompose la matrice d'interaction utilisateur-élément en deux matrices non négatives représentant les utilisateurs et les éléments, respectivement, à l'aide d'un processus d'optimisation itératif.

```
▶ from surprise import accuracy  
  
# Evaluate the model NMF  
predictions = model_NMF.test(testset)  
  
# Calculate RMSE and MAE  
rmse = accuracy.rmse(predictions, verbose=False)  
mae = accuracy.mae(predictions, verbose=False)  
  
print("RMSE:", rmse)  
print("MAE:", mae)  
  
→ RMSE: 0.5095336860388523  
MAE: 0.5000735893657894
```

Figure 3.15: RMSE et MAE de l'algorithme NMF

b. Le filtrage basé sur le contenu:

Le filtrage basé sur le contenu est une technique de système de recommandation qui suggère des produits en fonction de leurs caractéristiques, telles que les catégories et les ingrédients. En analysant les attributs d'un produit scanné, cette méthode recommande des articles similaires qui partagent les mêmes catégories et ingrédients que le produit pour lequel l'utilisateur a manifesté de l'intérêt.

- **La similarité cosinus :** La similarité en cosinus est une mesure de similarité entre deux vecteurs non nuls d'un espace de produit intérieur. Elle est définie comme le cosinus de l'angle entre les deux vecteurs, qui est le produit des points des vecteurs divisé par le produit de leurs magnitudes. Cette métrique est couramment utilisée dans l'analyse de texte et le filtrage basé sur le contenu pour déterminer la similarité entre les documents ou les utilisateurs.

SYSTEME DE RECOMMENDATION

```
# Initialize TF-IDF vectorizer
tfidf_vectorizer = TfidfVectorizer()

# Fit and transform the TF-IDF vectorizer on the features
tfidf_matrix = tfidf_vectorizer.fit_transform(products_df['features'])

# Calculate cosine similarity between products based on features
cosine_sim = cosine_similarity(tfidf_matrix, tfidf_matrix)
```

Figure 3.16: Calcul de la similarité cosinus

- **La distance euclidienne:** La distance euclidienne est une mesure de la distance en ligne droite entre deux points dans l'espace euclidien. Elle est calculée comme la racine carrée de la somme des carrés des différences entre les coordonnées correspondantes des points. Cette mesure est couramment utilisée dans les algorithmes d'apprentissage automatique pour déterminer la similarité ou la dissimilarité entre les points de données.

```
from sklearn.metrics.pairwise import euclidean_distances

# Calculate Euclidean distance between products based on features
euclidean_dist = euclidean_distances(tfidf_matrix, tfidf_matrix)
```

Figure 3.17: Calcul de la distance euclidienne

3.3.5 Evaluation

a. Le filtrage collaboratif:

- **KNNBasic :** L'algorithme KNNBasic présente des performances cohérentes et précises sur les 5 divisions, avec une RMSE de 0,5066 en moyenne et un MAE de 0,5001 en moyenne. Il présente des temps d'ajustement efficaces d'environ 1,18 seconde et des temps de test d'environ 16,09 secondes.

SYSTÈME DE RECOMMANDATION

Evaluating RMSE, MAE of algorithm KNNBasic on 5 split(s).

	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Mean	Std
RMSE (testset)	0.5069	0.5067	0.5068	0.5064	0.5061	0.5066	0.0003
MAE (testset)	0.5003	0.5002	0.5003	0.4999	0.4997	0.5001	0.0002
Fit time	0.75	1.22	1.23	1.44	1.29	1.18	0.23
Test time	14.92	15.28	17.85	17.38	15.01	16.09	1.26
Average RMSE:	0.5065798602198275						
Average MAE:	0.5000537420547282						

Figure 3.18: Validation-croisée pour KNNBasic

- **SVD** : L'algorithme SVD démontre des performances cohérentes et précises sur les 5 splits, avec une RMSE est en moyenne de 0,5078 et l'MAE est en moyenne de 0,5002. Il maintient des temps d'ajustement stables autour de 19,87 secondes et des temps de test d'environ 2,44 secondes.

Evaluating RMSE, MAE of algorithm SVD on 5 split(s).

	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Mean	Std
RMSE (testset)	0.5077	0.5078	0.5079	0.5078	0.5077	0.5078	0.0001
MAE (testset)	0.5002	0.5002	0.5002	0.5002	0.5001	0.5002	0.0000
Fit time	19.49	20.31	19.05	20.85	19.65	19.87	0.63
Test time	2.00	2.88	1.86	2.57	2.90	2.44	0.44
Average RMSE:	0.5077903672258103						
Average MAE:	0.5001767602939557						

Figure 3.19: Validation-croisée pour SVD

- **NMF** : L'algorithme NMF présente des performances cohérentes et précises sur les 5 divisions, avec une RMSE moyenne de 0,5095 et une MAE moyenne de 0,5000. 0,5095 et une MAE de 0,5000 en moyenne. Il maintient des temps d'ajustement stables autour de 36,00 secondes et des temps de test autour de 2,60 secondes.

b. Le filtrage basé sur le contenu:

- **La similarité cosinus** : la précision est de 0,6. Cela signifie que 60% des instances identifiées comme similaires par la mesure de similarité cosinusienne étaient effectivement pertinentes ou correctes. Une précision de 0,6 indique un niveau modéré de précision dans l'identification des instances similaires, ce qui suggère que cette mesure est assez fiable pour discerner les instances

pertinentes des instances non pertinentes dans l'ensemble de données ou l'application en question.

- **La distance euclidienne:** La similarité euclidienne a une précision de 0,2. Cela signifie que seulement 20% des instances qu'elle a identifiées comme similaires étaient pertinentes. Une précision de 0,2 est relativement faible, ce qui indique que la mesure de la similarité euclidienne n'est pas aussi efficace pour identifier avec précision les instances pertinentes. Cette précision plus faible suggère un taux plus élevé de faux positifs, ce qui signifie que de nombreuses instances identifiées comme similaires par la méthode de la similarité euclidienne n'étaient pas réellement pertinentes.

Cosine Similarity - Precision: 0.6
Euclidean Similarity - Precision: 0.2
(0.6, 0.2)

Figure 3.20: Précision des modèles basé sur le contenu

c. Sélection du modèle :

- **Le filtrage collaboratif**

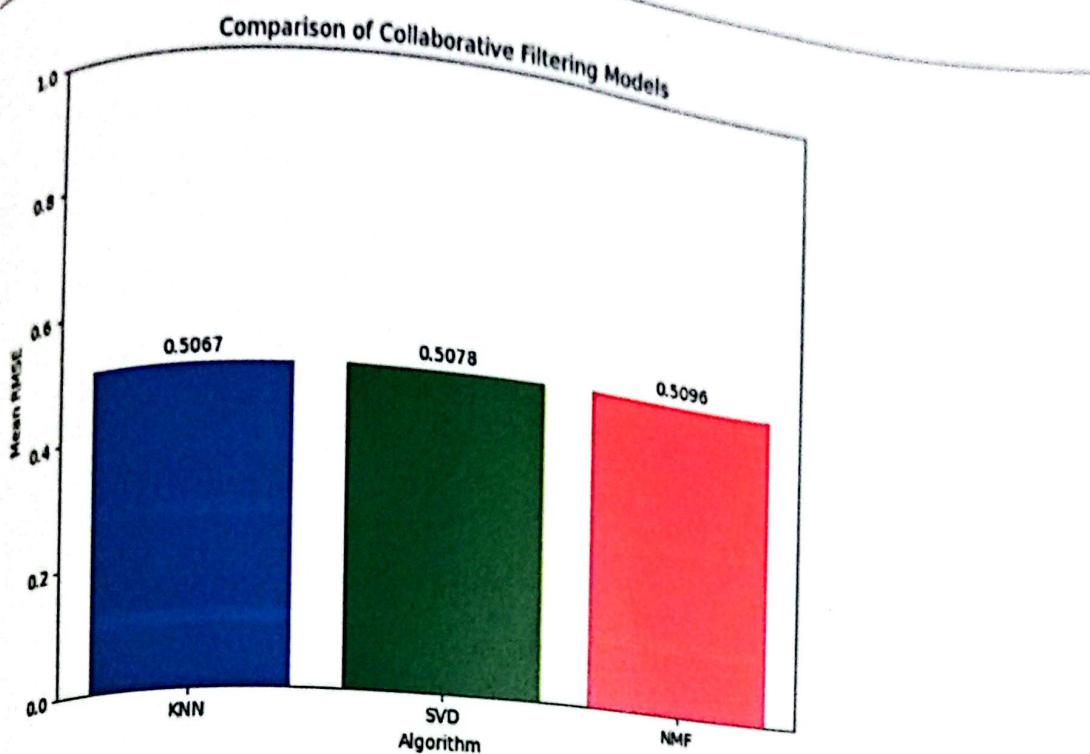


Figure 3.21: Comparaison entre les trois modèles de filtrage collaboratif

Pour déterminer le meilleur algorithme, nous prenons généralement en compte plusieurs facteurs, notamment la précision de la prédiction, l'efficacité de calcul et les exigences spécifiques de la tâche ou de l'application.

Dans le cas présent :

- **RMSE et MAE** : des valeurs plus faibles indiquent une meilleure précision de prédiction.
- **Temps d'adaptation** : Des valeurs plus faibles indiquent une formation plus rapide.
- **Temps de test** : Des valeurs plus faibles indiquent une prédiction plus rapide.

Comparaison des moyennes :

Table 3.1: Comparaison des modèles

Algorithm	RMSE	MAE	Fit time (s)	Test time (s)
NMF	0.5095	0.5000	36.00	2.60
SVD	0.5078	0.5002	19.87	2.44
KNNBasic	0.5066	0.5001	1.18	16.09

=> Si l'on considère tous les facteurs, la SVD semble être le meilleur choix, car elle offre un bon équilibre entre la précision de la prédiction et l'efficacité du calcul.

- Le filtrage basé sur le contenu

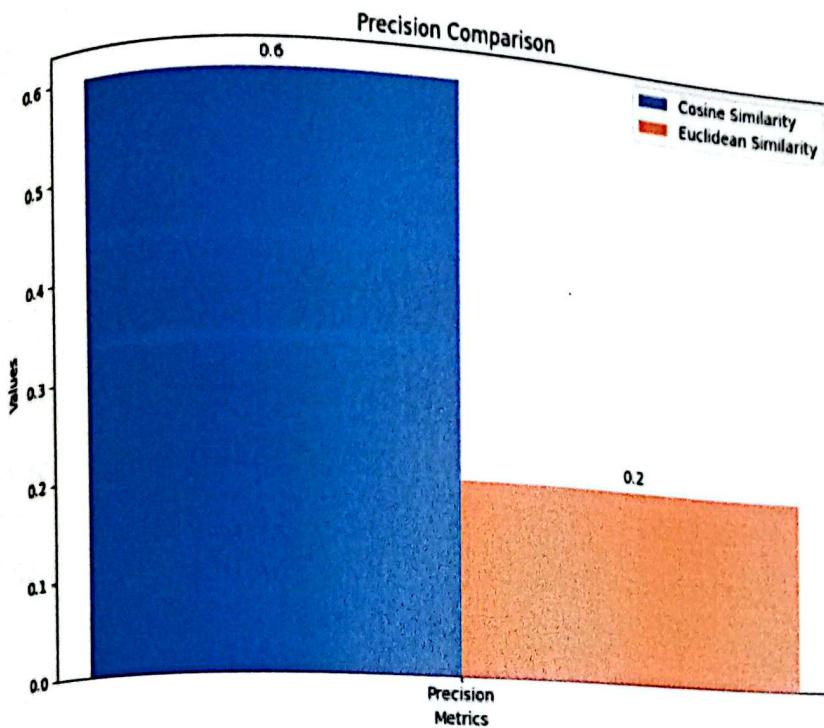


Figure 3.22: Comparaison entre les deux modèles de filtrage basé sur le contenu

Pour déterminer le meilleur algorithme, nous prenons généralement en compte plusieurs facteurs, notamment la précision de la prédiction et l'efficacité du calcul.

=> Tenu compte de ces résultats, la similarité cosinus a été choisie comme modèle préféré pour cette tâche en raison de sa précision nettement plus élevée, garantissant une identification plus fiable et plus précise des instances pertinentes par rapport à la similarité euclidienne.

3.3.6 Déploiement

Dans cette section, nous allons détailler le déploiement et l'exécution de notre modèle. Ce déploiement a impliqué plusieurs étapes clés :

3.3.6.1 Exportation de modèle

Les modèles entraînés ont été exportés de Jupyter Notebook à l'aide de Pickle. Cela nous a permis de sauvegarder l'état de chaque modèle et de les recharger dans l'application FastAPI.

```
[ ] import pickle  
pickle.dump(model_SVD, open('model_SVD.pkl', 'wb'))
```

Figure 3.23: Exportation de modèle SVD

3.3.6.2 Construction de notre modèle en tant qu'API RESTful

FastAPI a été choisi pour ses performances et sa facilité d'utilisation. Nous avons mis en place une application FastAPI pour automatiser le processus de nettoyage et de préparation des données et pour servir les modèles de recommandation.

```
# API endpoint to fetch data from Carrefour dataset and save it to JSON file.  
app.get("/save_to_json/")  
async def save_to_json():  
    if save_carrefour_data_to_file():  
        return {"message": "Data saved to JSON file successfully"}  
    else:  
        raise HTTPException(status_code=500, detail="Failed to save data to JSON file")
```

Figure 3.24: API pour le processus de récupération des données

L'application charge les données brutes, les traite et renvoie des données nettoyées et préparées. En outre, elle fournit des prédictions à partir des modèles formés.

```
def post('/get-scanned-recommendation'):
    n = 10
    user_id = input_parameters.user_id
    scanned_ean = input_parameters.scanned_ean
    # check if user_id exists in user_df
    if user_id not in user_df['user_id'].unique():
        # If user_id doesn't exist, recommend the top liked products
        top_liked_products = liked_products.sort_values(ascending=False).head(n)
        return products_df[products_df['ean'].isin(top_liked_products.index)].to_dict(orient='records')
```

Figure 3.25: API pour le système de recommandation

3.3.6.3 Autres APIs

Quatre APIs principaux ont été créés :

- **GET /save-to-json** : Ce point d'accès permet de récupérer les données de MongoDB et de les enregistrer dans un fichier JSON.
- **GET /clean-data** : Ce point d'accès accepte un fichier JSON et renvoie les données nettoyées et préparées pour la modélisation.
- **GET /add-nutriscore** : Ce point d'accès accepte les données nettoyées et calcule le nutriscore pour chaque produit et le met à jour.
- **POST /get-scanned-recommendation** : Ce point d'accès accepte l'identifiant de l'utilisateur et le produit scanné pour renvoyer une recommandation basée sur l'identifiant de l'utilisateur et le produit scanné.

Conclusion

En conclusion, ce chapitre a détaillé le développement d'un système de recommandation alimentaire intelligent. En adoptant la méthodologie CRISP, nous avons pu structurer notre approche de manière systématique et efficace. Dans ce qui suit, nous allons passer à la phase de réalisation pour mettre en œuvre notre application.