

Projet 6

Classifiez automatiquement des biens de consommation

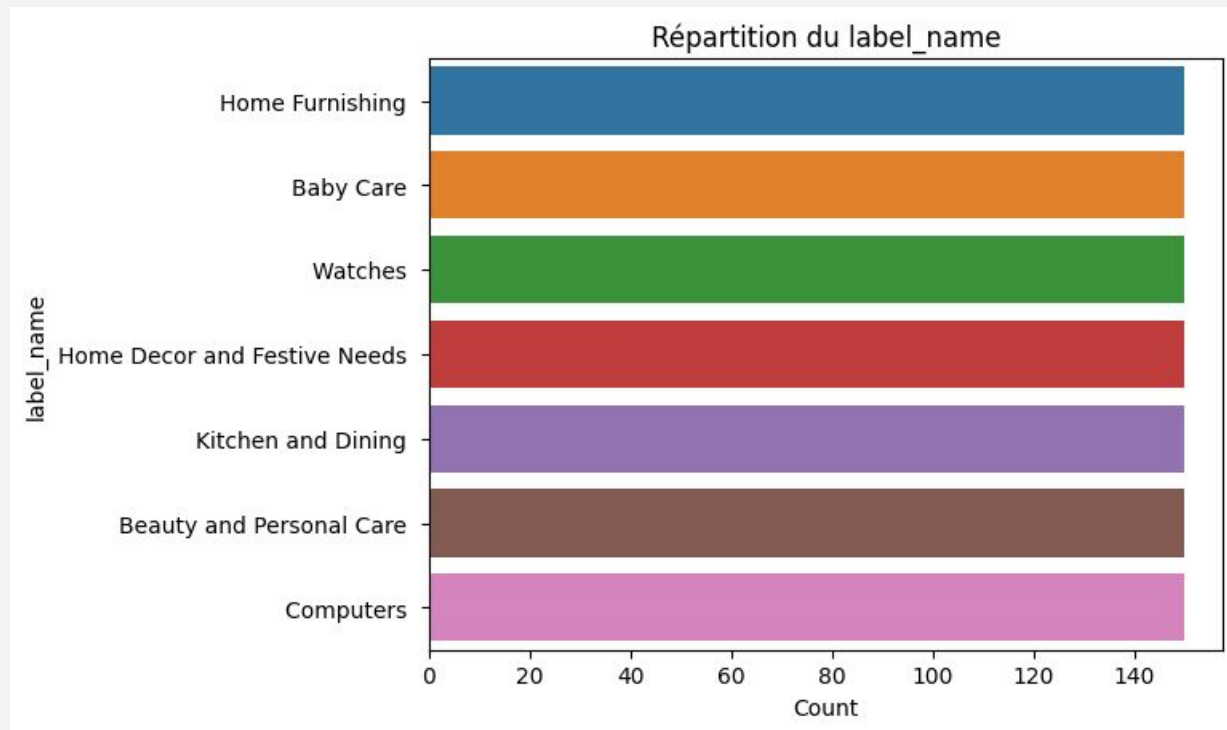


Zeynep Erdem
31-08-2023

Sommaire

- Rappel de la Problématique et Environnement
- Présentation des données
- Etude de faisabilité pour les données textuelles
- Classification supervisé
- Etude de faisabilité pour les données images
- Classification supervisé
- Comparaison des différents modèles
- Présentation du test de l'API
- Conclusion et Recommandations

Présentation du Jeu de Données



- Il y a 1050 lignes et 15 colonnes
- 2 % de valeurs nulles
- product_category_tree
- 7 catégories de produit
- description
- image

Rappel de la Problématique



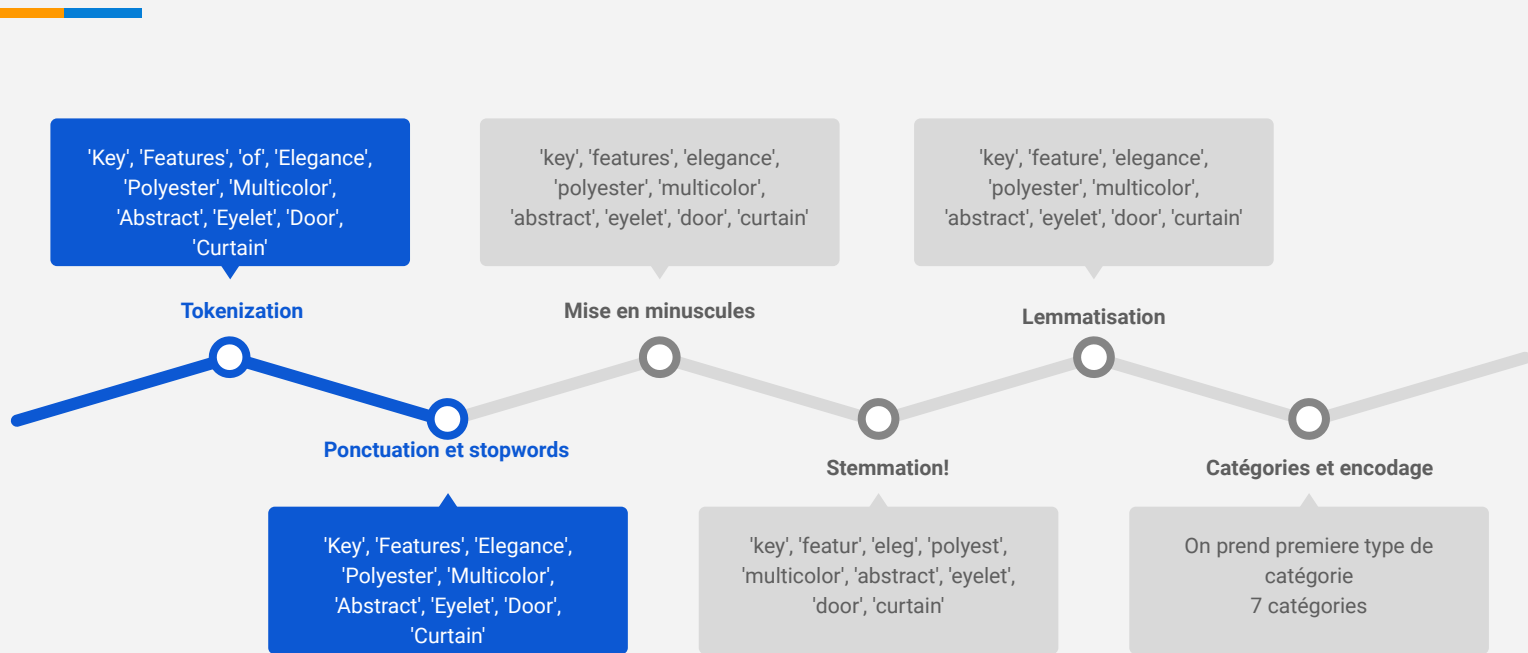
- ❖ Place de marché souhaite lancer une marketplace
 - ❖ Des articles avec une photo et une description
 - ❖ L'attribution de la catégorie manuellement peu fiable
 - ❖ Faciliter la mise en ligne de nouveaux articles
 - ❖ Faciliter la recherche de produits
 - ❖ D'automatiser la tâche
 - ❖ Etudier la faisabilité d'un moteur de classification des articles
 - ❖ Réaliser une classification supervisée à partir des images
 - ❖ Tester la collecte de produits à base de “champagne” via l'API de Edamam Food and Grocery Database
- Python: 3.9.16
 - Pandas: 2.0.2
 - Numpy: 1.24.3
 - Seaborn: 0.12.2
 - Matplotlib: 3.7.1
 - Missingno: 0.5.2
 - Sklearn: 1.2.2
 - Wordcloud: 1.9.2
 - Nltk: 3.8.1
 - Tensorflow: 2.12.0

Les étapes de faisabilité-clustering- pour description (text)

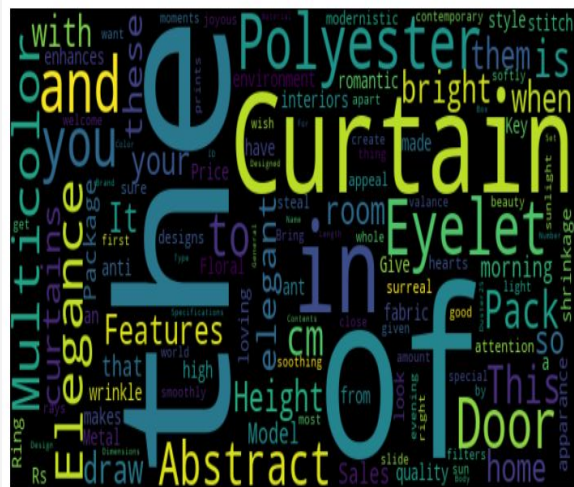


- Récupération des tokens et nettoyage
- Bags of Words :
Count-vectorizer , TF-IDF
- Words Embedding :
Word2Vec, BERT, USE
- TSNE
- Kmeans
- TSNE
- ARI score

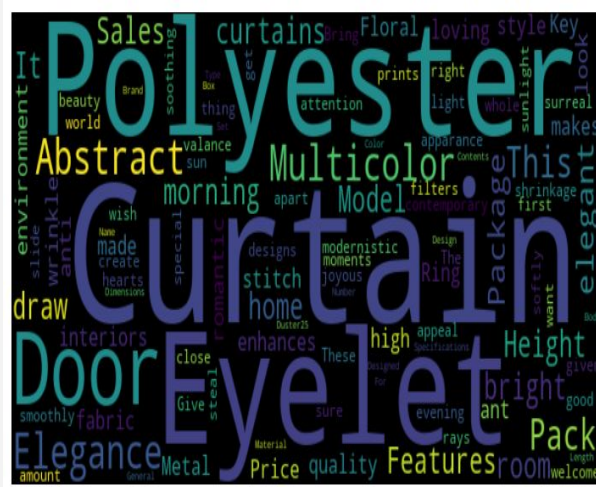
Préparation des Données Textuelles



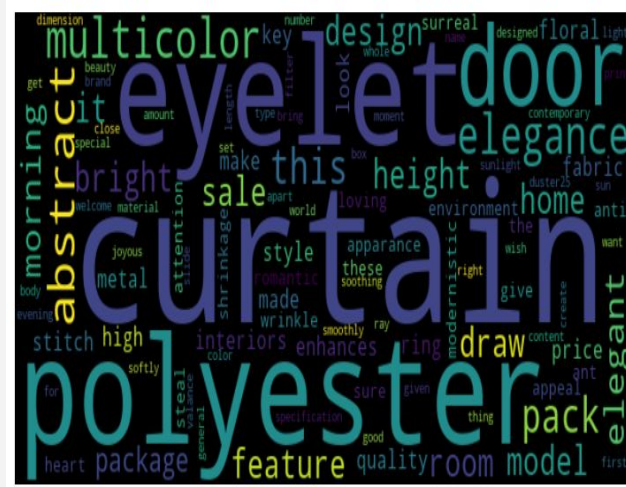
Préparation des Données Textuelles



Tokenization



Ponctuation et stopwords



Mise en minuscules et lemmatisation

Les mots dans les Wordclouds ont changé après chaque traitement

Extractions de features - Bag of Words

Count vectorizer

color	contemporary	content	create	curtain	design	designed	dimension	door	draw	duster25	elegance	elegant	enhances	environment
1	1	1	1	11	2	1	1	5	2	1	4	2	1	1
1	0	1	0	0	2	0	1	0	0	0	0	0	0	0
2	0	1	0	0	1	1	1	0	0	0	0	0	0	0

- Convertit les documents en vecteurs de fréquences de mots.

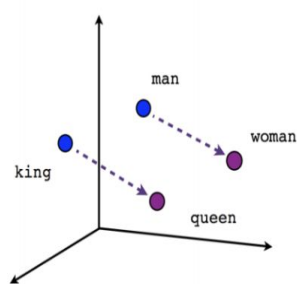
Tf idf (Term Frequency-Inverse Document Frequency)

color	contemporary	content	create	curtain	design	designed	dimension	door	draw	duster25	elegance	elegant	enhances	environment
0.025548	0.054683	0.030088	0.062524	0.542833	0.059766	0.040057	0.029883	0.254176	0.133271	0.073664	0.243843	0.092407	0.060961	0.064373
0.047194	0.000000	0.055580	0.000000	0.000000	0.110403	0.000000	0.055202	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
0.048214	0.000000	0.028390	0.000000	0.000000	0.028197	0.037797	0.028197	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000

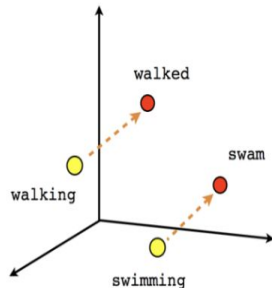
- Ajuste les fréquences de mots en fonction de leur importance globale dans le corpus
- en pénalisant les mots fréquents dans tout le corpus.

La valeur de mot 'dimension' change par rapport au modèle

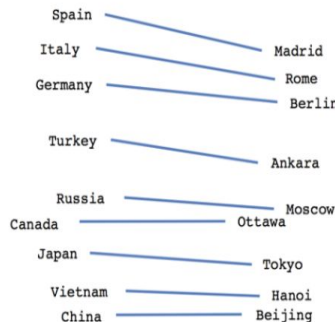
Word Embedding



Male-Female



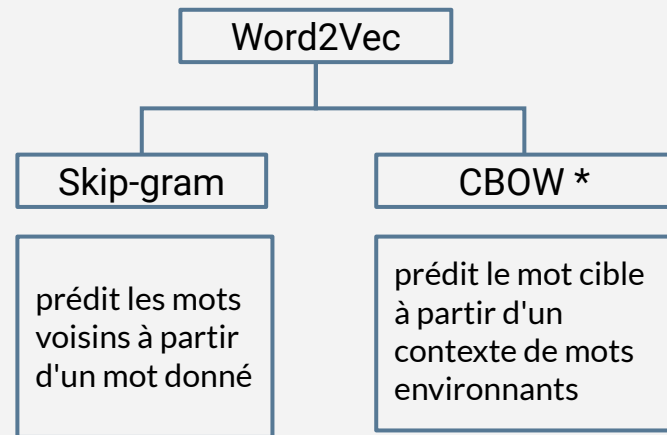
Verb tense



Country-Capital

BERT (Bidirectional Encoder Representations from Transformers)

USE (Universal Sentence Encoder)



Modèle pré-entraîné qui capture les significations contextuelles des mots en analysant les deux directions du texte

Génère des embeddings pour les phrases et les paragraphes entiers, en capturant leur signification sémantique.

Réduction de dimension



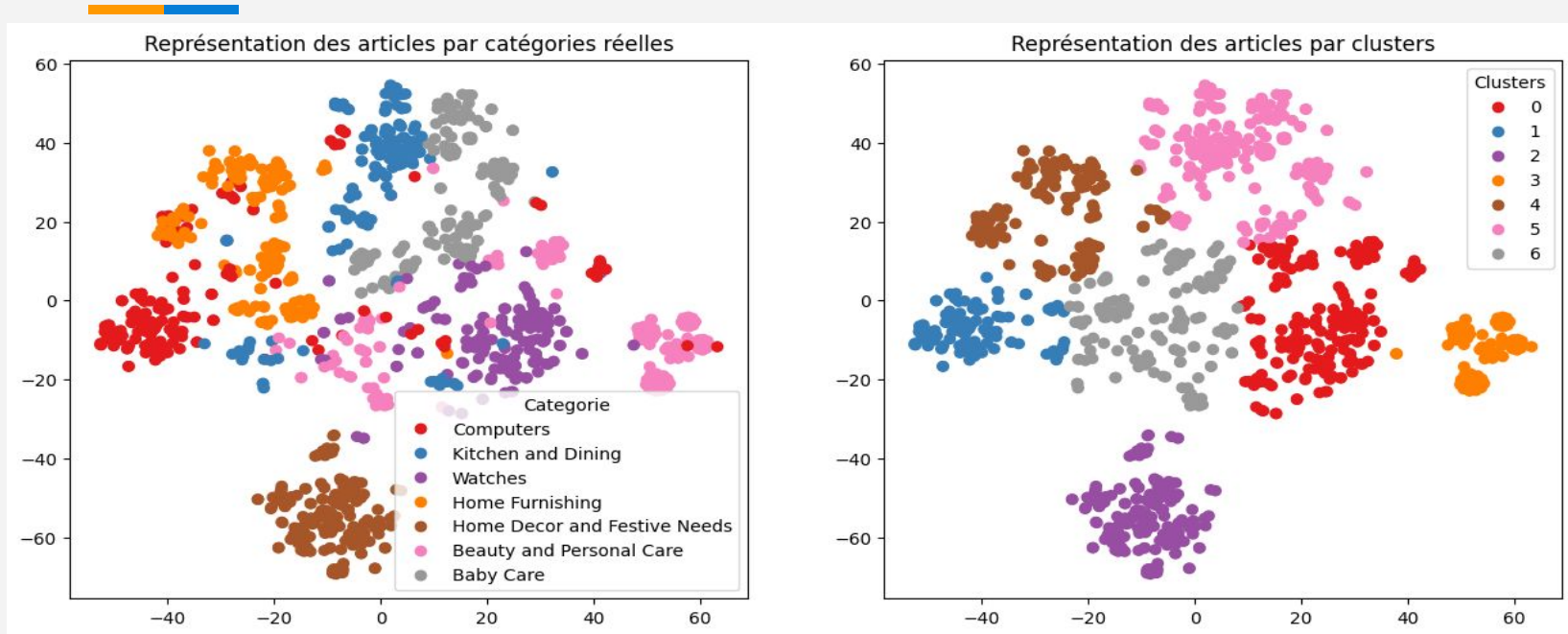
Modèle	Ligne	Colonnes
Count vectorizer	1050	5326
Tf-idf	1050	5326
Word2Vec	1050	300
BERT	1050	768
USE	1050	512



TSNE

En raison du grand nombre de colonnes, une réduction de dimension est nécessaire.

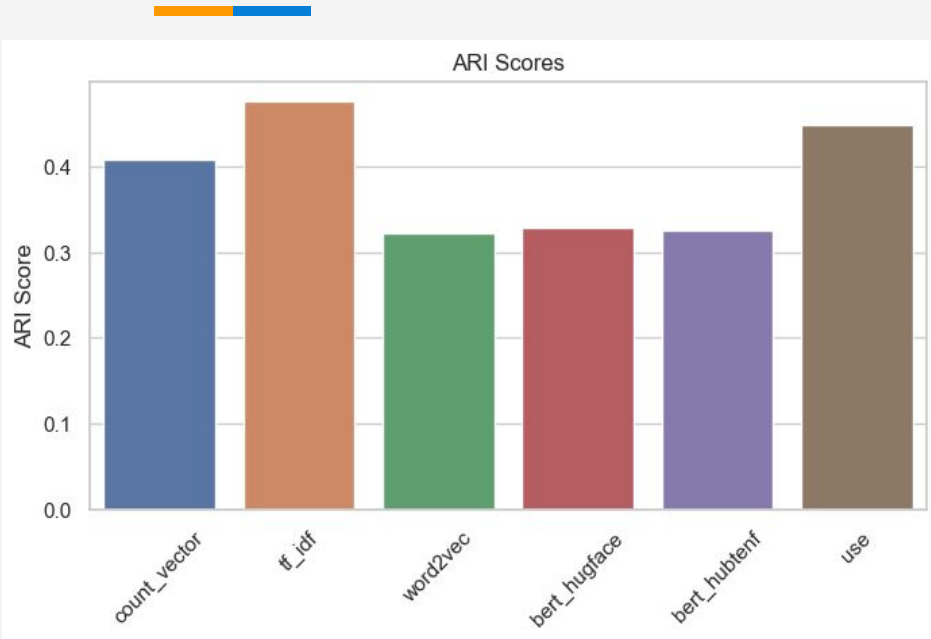
Visualisation - Tf-idf



ARI : 0.47

On a obtenu les clusters cependant certains produits ne sont pas bien distingués.

Comparaison des résultats

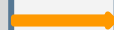


	ARI	time
count_vector	0.4077	4.54
tf_idf	0.4755	3.82
word2vec	0.3217	3.42
bert_hugface	0.3280	3.57
bert_hubtenf	0.3251	3.74
use	0.4486	3.42

On obtient le meilleur résultat avec tf-idf

Classification supervisé

Tf-idf pour extraction des features



PCA pour réduction de dimension



Gridsearch avec 5 fold

mean_fit_time

params mean_test_score mean_train_score

Random Forest	3.805961	{'estimator': RandomForestClassifier(), 'reduction': PCA(n_components=585), 'reduction__n_compon...	0.914068	1.000000
SVC	1.340065	{}	0.923194	0.999492
XGBoost	11.222090	{}	0.911027	1.000000

Les étapes de faisabilité -clustering- pour Image



- SIFT
- VGG16

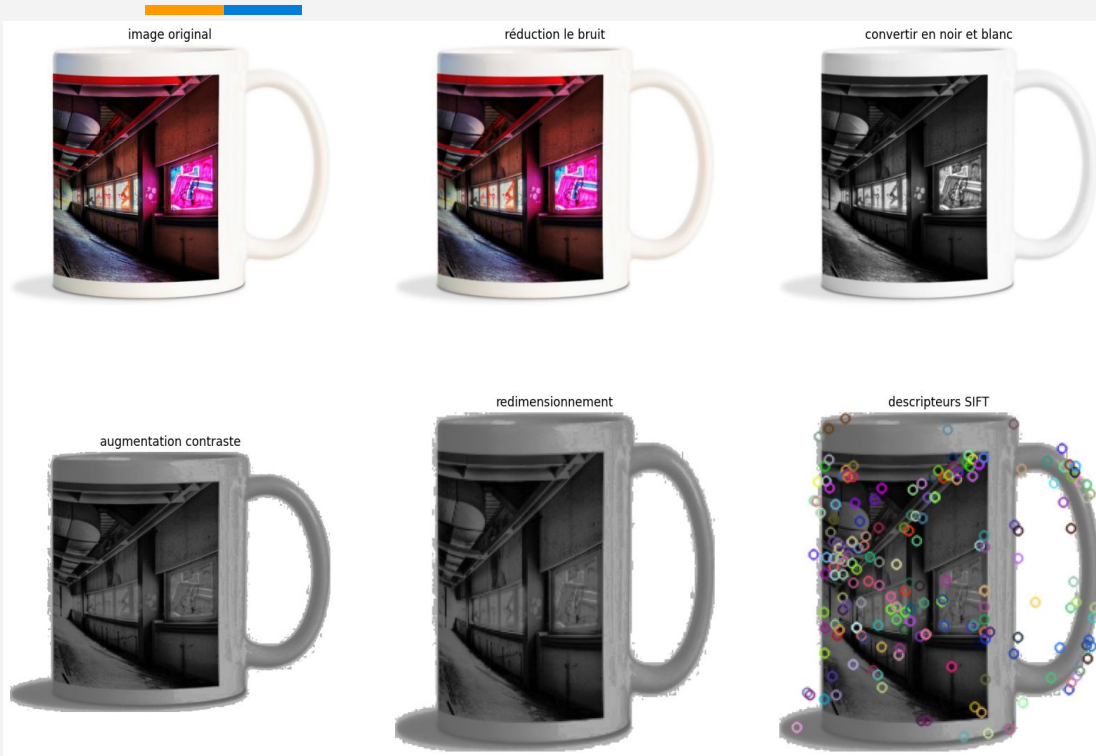
- Bags of visual word :SIFT
- Embedding : CNN
- VGG16 (pré entrainé)

- PCA

- Kmeans

- TSNE
- ARI score

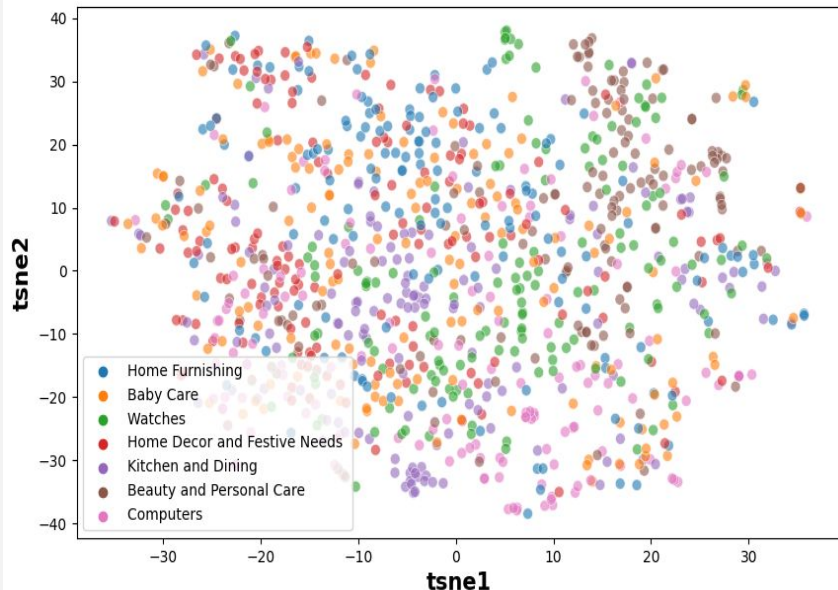
Pré-traitement des images via SIFT



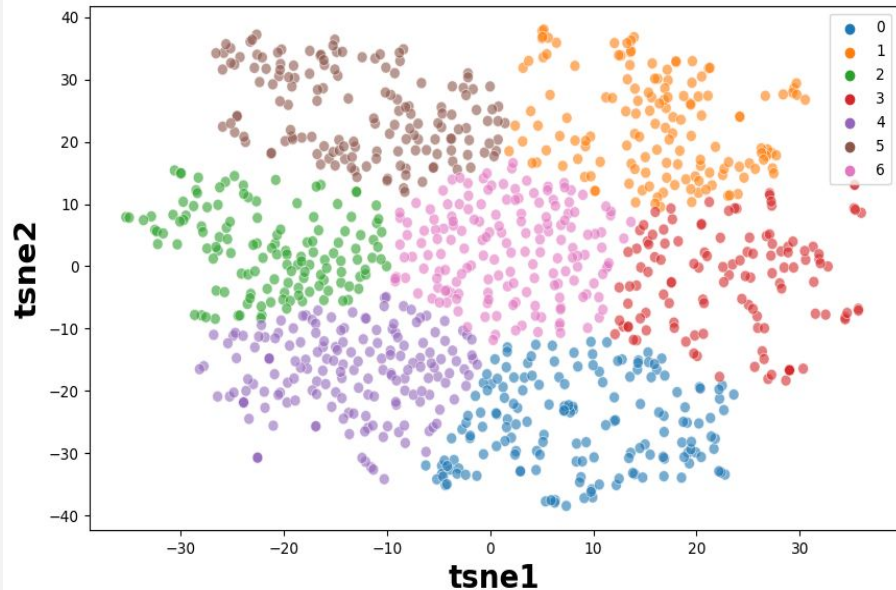
- Création d'une liste de descripteurs pour l'ensemble des images
- Création des clusters de descripteurs avec Kmeans
- Création des features des images
Pour chaque image :
 - prédiction des numéros de cluster de chaque descripteur
 - création d'un histogramme = comptage pour une image du nombre de descripteurs par cluster
- Utilisation d'une ACP en gardant 99% de variance (714 à 495 features)
- Utilisation Tsne pour visualisation

SIFT-Affichage des images selon clusters et calcul ARI

TSNE selon les vraies classes



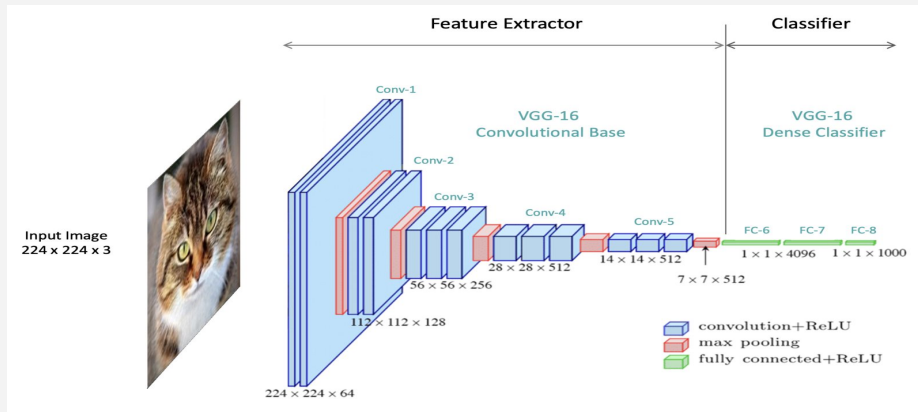
TSNE selon les clusters



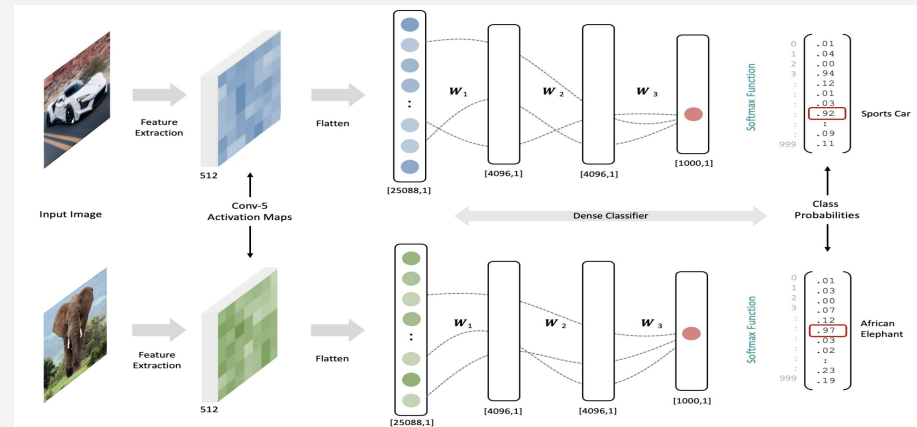
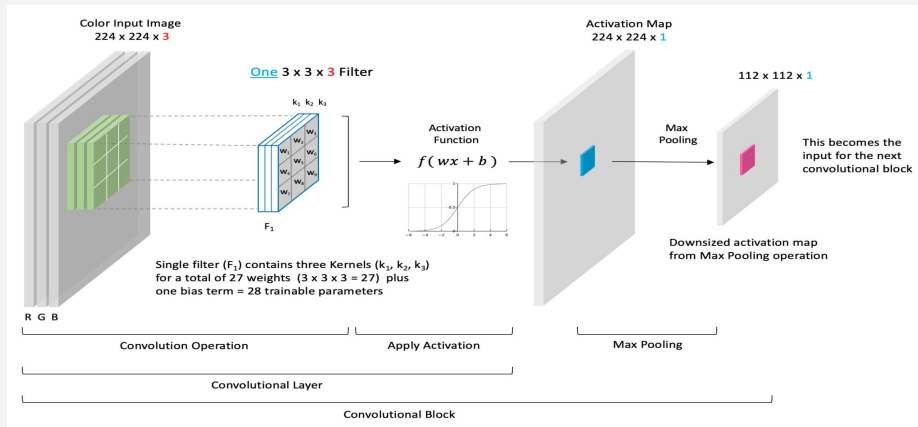
ARI : 0.05

Le score de SIFT n'est pas satisfaisant.

Convolutional Neural Networks - Modèle pré-entraîné

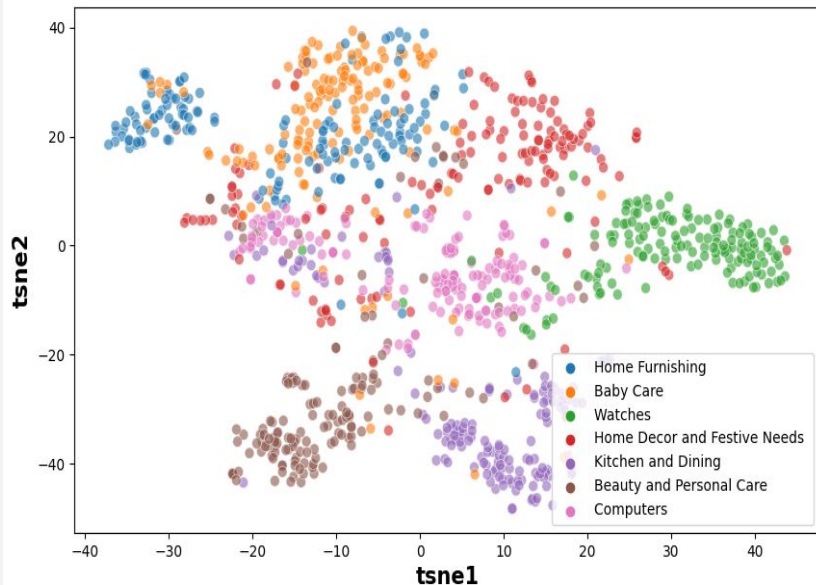


- Extraction des features des images
- Sans les couches fully_connected
- PCA réduction dimension - TSNE visualisation
- Entraînement dernier classifieur pas toutes les couches
- Couche Flatten : aplatissement image : matrice -> vecteur
- Couche Dense : classification à 7 catégories

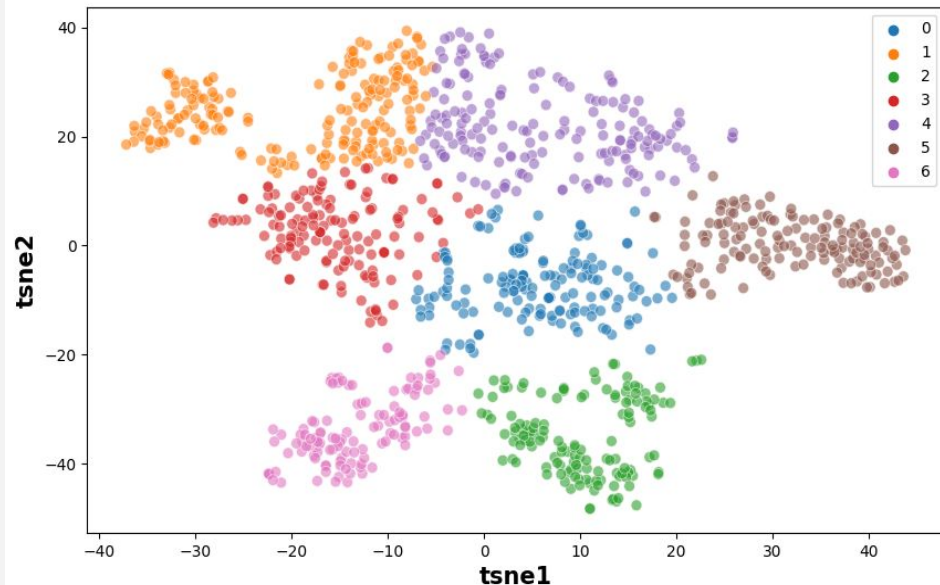


Etude de faisabilité avec VGG16 - Modèle pré-entraîné

TSNE selon les vraies classes



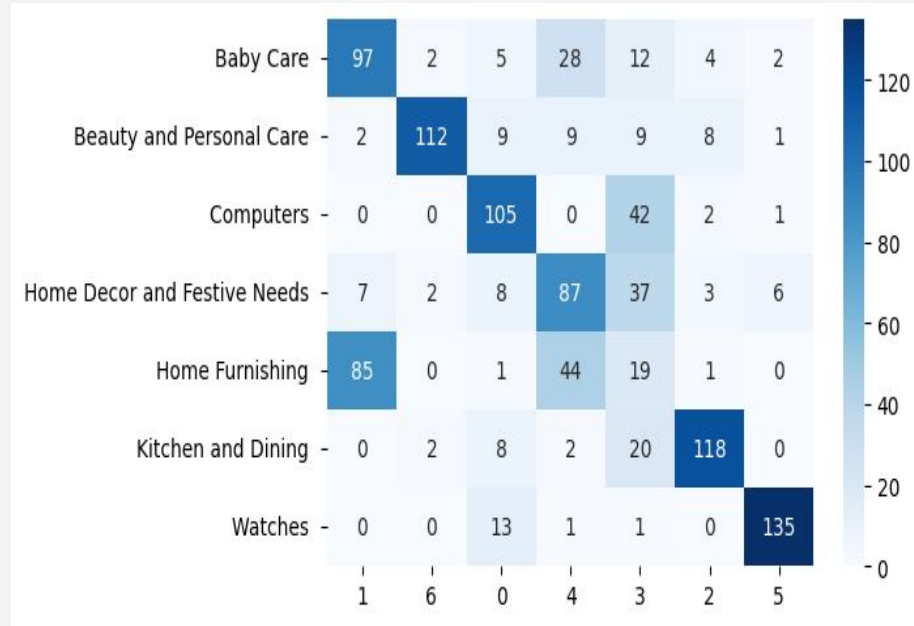
TSNE selon les clusters



ARI : 0.47

Le score de VGG16 est meilleur à celui de SIFT.


Analyse d'une erreur et Confusion Matrix



Affichage image "Kitchen and Dining" considérée comme "Beauty and Personal Care"

Le clustering n'est pas très efficace pour la catégorie "Home Furnishing"

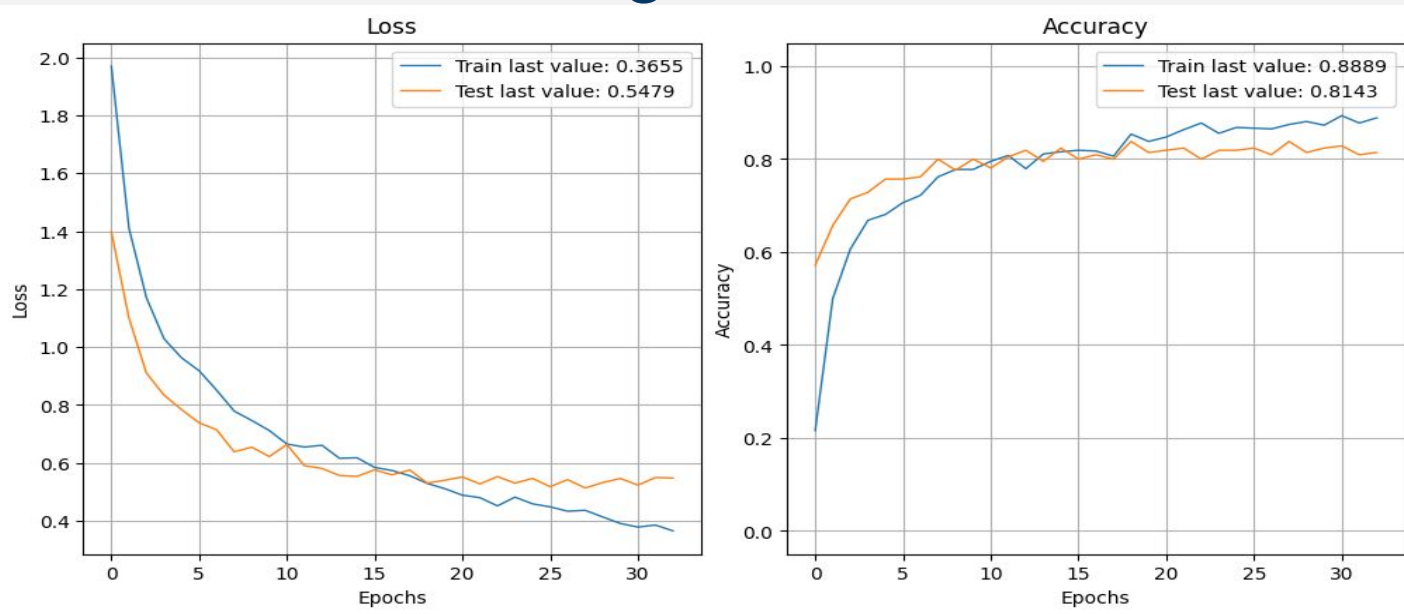
Classification Supervisée

- 
- Splitter notre data à train, validation et test set
 - Création du modèle de classification transfer learning VGG16
 - Layer non entraînaables
 - on garde les poids du modèle pré-entraîné
 - Compilation du modèle
 - `loss = "categorical_crossentropy"`
 - `optimizer = "adam"`
 - `metrics = "accuracy"`
 - Création du callback - EarlyStopping
 - L'entraînement du modèle
 - `epochs = 50`
 - `batch_size = 32`
 - Evaluation (training time - accuracy)
 - 4 approches pour la data preprocessing
 - 4 modèles différents pour comparer les performances

1. Approche préparation initiale des images
2. Approche ImageDatagenerator avec data augmentation
3. Approche nouvelle par Dataset sans data augmentation
4. Approche nouvelle par dataset avec data augmentation intégrée au modèle VGG16

1. VGG16
2. Resnet50V2
3. InceptionResnet50V2
4. EfficientNetV2B0

Approche nouvelle par dataset avec data augmentation intégrée au modèle VGG16



Data augmentation :


- rotation
- décalage de largeur et hauteur
- retournement horizontal
- zoom

Data augmentation évite l'overfitting

Les courbes de train et validation sets suivent les mêmes tendances

train_loss	val_loss	test_loss	train_accuracy	val_accuracy	test_accuracy	training_time_s
0.3021	0.5137	0.7543	0.9143	0.8381	0.7714	224.93

Comparaison des résultats des approches



	train_loss	val_loss	test_loss	train_accuracy	val_accuracy	test_accuracy	training_time_s
Prép initiale img	1.0566	2.1413	2.7927	0.8778	0.7905	0.7905	43.78
ImageDatagenerator data augmentation	0.2587	2.1708	3.0972	0.9571	0.7905	0.7905	253.87
Dataset sans data augmentation	1.2018	2.5776	3.0441	0.9016	0.7952	0.7857	62.97
Data augmentation intégrée au modèle	0.3021	0.5137	0.7543	0.9143	0.8381	0.7714	224.93

On a le meilleur résultat avec Data augmentation intégrée au modèle

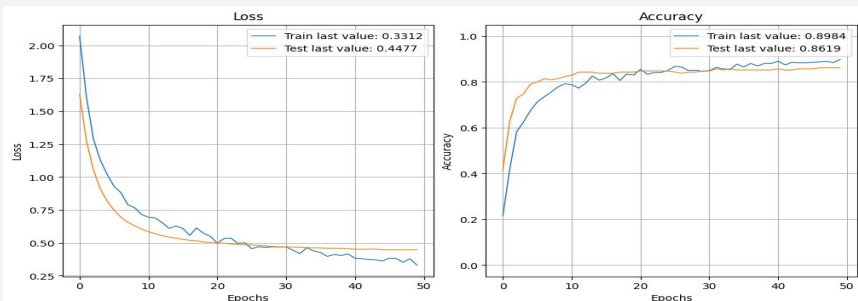
Comparaison des résultats des modèles VGG16, ResNet50V2, InceptionResNetV2, EfficientNetV2Bo

	train_loss	val_loss	test_loss	train_accuracy	val_accuracy	test_accuracy	training_time_s	year	depth	parameters
VGG16	0.3021	0.5137	0.7543	0.9143	0.8381	0.7714	224.93	2015	16	138.4M
ResNet50V2	0.1672	0.5821	0.8822	0.9556	0.8524	0.8238	59.56	2016	103	25.6M
InceptionResNetV2	0.5552	0.7171	0.8658	0.8651	0.8286	0.8000	72.23	2017	449	55.9M
EfficientNetV2B0	0.1100	0.4279	0.5058	0.9603	0.8952	0.8476	70.76	2019	-	7.2M

On a le meilleur résultat avec EfficientNetV2B0
(Smaller Models and Faster Training)

Optimisation des hyperparamètres “optimizer” du modèle EfficientNetV2Bo

résultat sgd



Optimizer ajuste les poids du modèle pendant l'entraînement afin de minimiser la fonction de perte.

SGD (Stochastic Gradient Descent) : Il utilise le gradient de la fonction de perte par rapport à chaque poids. Il peut être efficace, mais sa convergence peut être lente et sensible aux hyperparamètres.

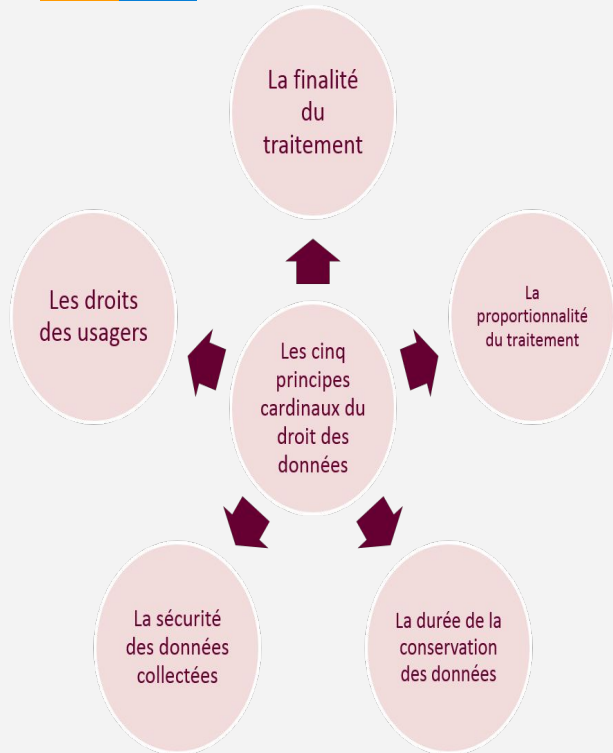
RMSProp : Il utilise une moyenne pondérée des carrés des gradients précédents pour ajuster les poids.

Adam (Adaptive Moment Estimation) : Il ajuste le taux d'apprentissage pour chaque poids individuel en fonction de la moyenne des gradients précédents. Cela peut aider à accélérer la convergence.

	train_loss	val_loss	test_loss	train_accuracy	val_accuracy	test_accuracy	training_time_s
sgd	0.2466	0.4468	0.4998	0.9206	0.8619	0.8571	284.33
adam	0.1100	0.4279	0.5058	0.9603	0.8952	0.8476	70.76
rmsprop	0.1059	0.4702	0.5316	0.9635	0.8476	0.8429	72.72

Malgré le meilleur résultat d'accuracy avec SGD, on ne le préfère pas en raison du temps d'entraînement.

API et RGPD



	foodId	label	category	foodContentsLabel	image
0	food_a656mk2a5dmqb2adiamu6beihduu	Champagne	Generic foods	unknown	https://www.edamam.com/food-img/a71/a718cf3c52...
1	food_b753ithamdb8psbt0w2k9aquo06c	Champagne Vinaigrette, Champagne	Packaged foods	OLIVE OIL; BALSAMIC VINEGAR; CHAMPAGNE VINEGAR...	unknown
2	food_b3dyababjo54xobm6r8jzbghjgqe	Champagne Vinaigrette, Champagne	Packaged foods	INGREDIENTS: WATER; CANOLA OIL; CHAMPAGNE VINE...	https://www.edamam.com/food-img/d88/d88b64d973...
3	food_a9e0ghsamvoc45bwa2ybsa3gken9	Champagne Vinaigrette, Champagne	Packaged foods	CANOLA AND SOYBEAN OIL; WHITE WINE (CONTAINS S...	unknown
4	food_an4jueaucpus2a3u1ni8auhe7q9	Champagne Vinaigrette, Champagne	Packaged foods	WATER; CANOLA AND SOYBEAN OIL; WHITE WINE (CON...	unknown
5	food_bmu5dmkazwuvpaa5prh1daa8jxs0	Champagne Dressing, Champagne	Packaged foods	SOYBEAN OIL; WHITE WINE (PRESERVED WITH SULFIT...	https://www.edamam.com/food-img/ab2/ab24599c2a...
6	food_alp44taoyv11ra0lic1qa8xculi	Champagne Buttercream	Generic meals	sugar; butter; shortening; vanilla; champagne;...	unknown
7	food_byap67hab6evc3a0f9w1oag3s0qf	Champagne Sorbet	Generic meals	Sugar; Lemon juice; brandy; Champagne; Peach	unknown
8	food_am5egz6aq3fpjiaf8xpdkbc2asis	Champagne Truffles	Generic meals	butter; cocoa; sweetened condensed milk; vanil...	unknown
9	food_bcz8rhiajk1fuva0vkfmeakbouc0	Champagne Vinaigrette	Generic meals	champagne vinegar; olive oil; Dijon mustard; s...	unknown

On a respecté des normes RGPD

Conclusion et Recommandations



- On a utilisé la description et l'image des produits pour nos analyses .
 - On a effectué les études de faisabilité pour les deux variables.
 - Selon les résultats des clustering, on a constaté qu'on peut faire une classification.
 - D'après les résultats de classification, on a obtenu les meilleurs résultats avec le modèle CNN EfficientNetV2B0
 - On a essayé d'améliorer la performance du modèle avec hyperparamètres tuning.
 - Les résultats de la classification pour le text et l'image sont satisfaisants.
 - Une extraction des 10 premiers produits de champagne a été obtenu via l'API.
 - Dans toutes phases de la collecte et du stockage des données, on a respecté des normes RGPD.
-
- On peut essayer de faire une classification avec les features de description et l'image du produit ensemble.
 - On peut collecter des données supplémentaires.

