OPENCLASSROOMS

Classifiez automatiquement des biens de consommation

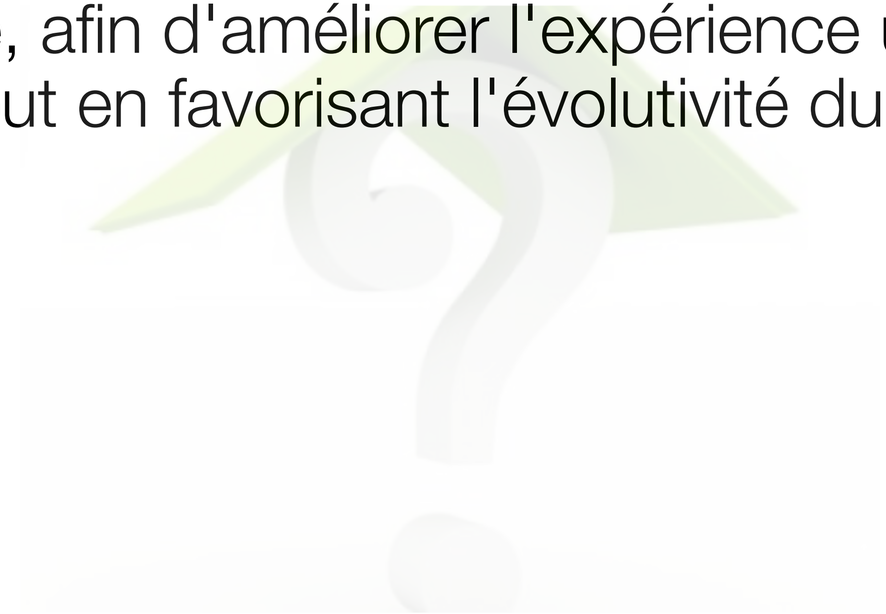
Preparé par: Sidi Teyib BEDDY EL MOUSTAPHA

Mentor : Thiery silberman

Évaluateur: Orhan Yazar

La problématique

- Quelle approche peut être utilisée pour automatiser l'attribution de catégories aux articles sur la marketplace e-commerce de l’entreprise; Place de marché, afin d'améliorer l'expérience utilisateur des vendeurs, et des acheteurs tout en favorisant l'évolutivité du système ?



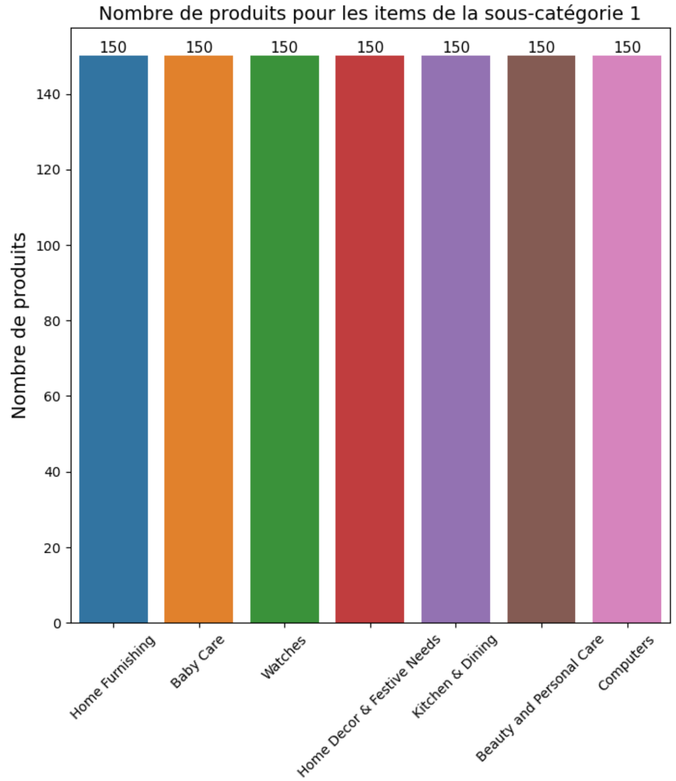
Jeu de données

```
df.shape
```

```
(1050, 15)
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1050 entries, 0 to 1049
Data columns (total 15 columns):
```




```
df.columns
```

```
Index(['uniq_id', 'crawl_timestamp', 'product_url', 'product_name',
       'product_category_tree', 'pid', 'retail_price', 'discounted_price',
       'image', 'is_FK_Advantage_product', 'description', 'product_rating',
       'overall_rating', 'brand', 'product_specifications'],
```

Ce qui nous intéresse

Données textuelles : nom, description

Données visuelles : image des produits

Les catégories des produits

Plan de Présentation

Introduction

Classification des textes

Classification des Images

Test d’une API

Conclusion

Introduction

- Nous étudions la faisabilité d'un moteur de classification des articles pour automatiser l'attribution des catégories sur notre marketplace e-commerce.

les résultats de l'étude

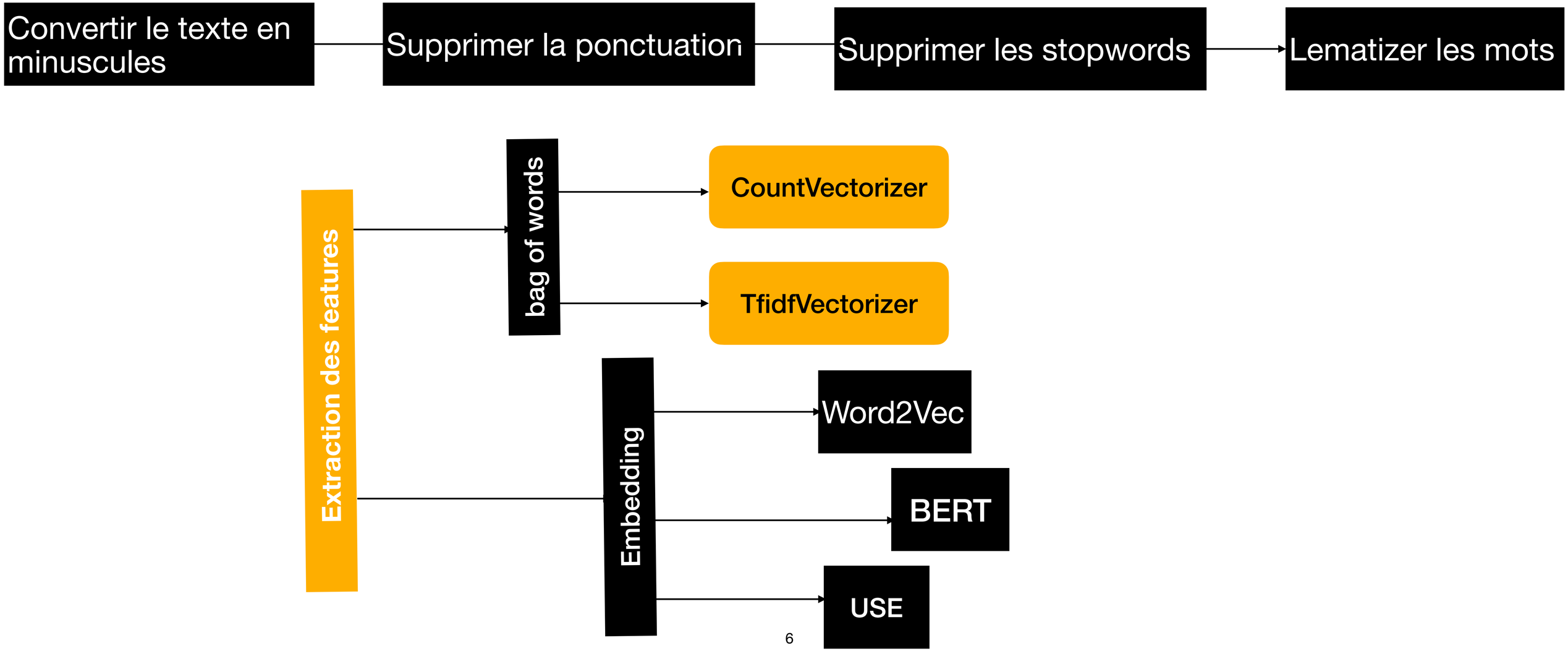
la classification supervisée

les performances de notre API

- L'objectif est d'améliorer l'expérience utilisateur et favoriser la croissance de notre plateforme.

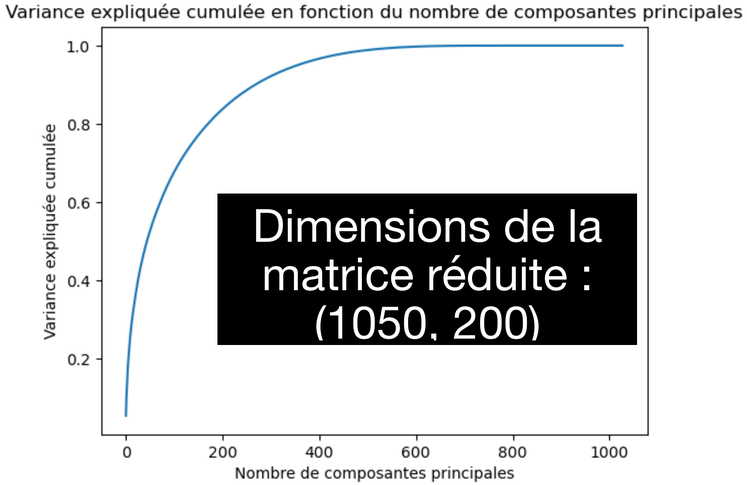
Classification des textes

Le prétraitement de texte



Classification des textes

CountVectorizer



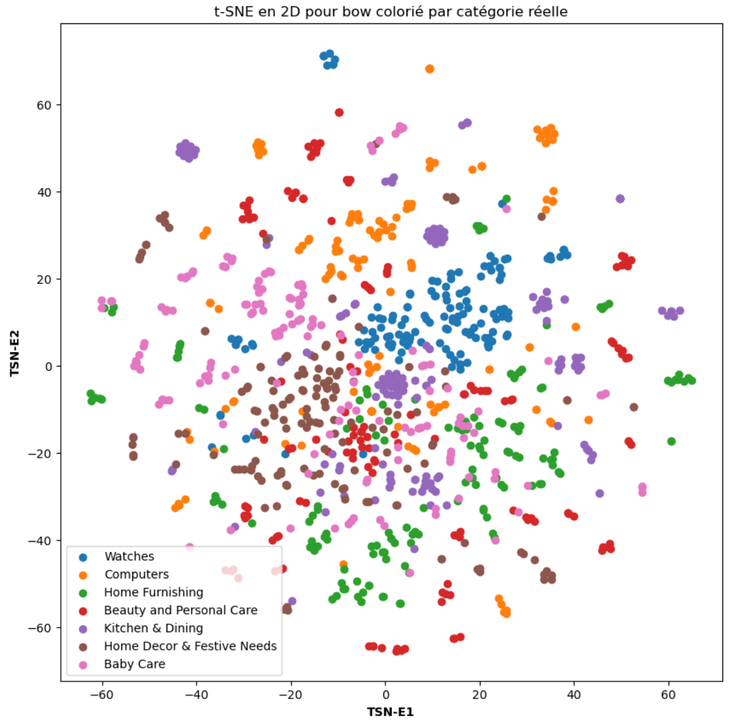
NMF

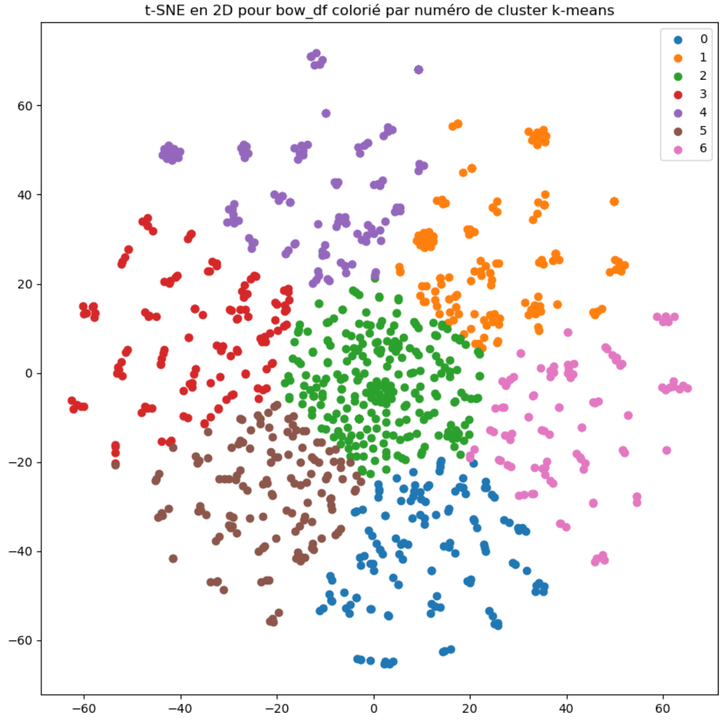
```
from sklearn.decomposition import NMF
no_topics = 20
```

cliquer pour afficher toute la sortie ; double-cliquer pour masquer la sortie

```
nmf = NMF(n_components=no_topics, random_state=1, l1_ratio=.5, init='nndsvd')
nmf.fit(bow_df)

no_top_words = 10
display_topics(nmf, cv.get_feature_names_out(), no_top_words)
```

```
Topic 0:
cm showpiece home brass price ganesha idol lord gold gift
Topic 1:
mug bring coffee perfect design quality make happy special love
Topic 2:
shipping genuine cash free delivery buy product 30 day replacement
Topic 3:
adapter warranty vaio replacement power laptop smartpro charger product designed
Topic 4:
laptop skin shape mouse print pad warranty inch multicolor combo
Topic 5:
baby girl fabric cotton dress boy sleeve neck ideal pattern
Topic 6:
ceramic mug rockmantra design come safe feature day gift material
Topic 7:
battery quality lapguard laptop replacement cell including label high durability
Topic 8:
cm sheet bedsheet cover cotton pillow double inch flat length
Topic 9:
watch analog men india great discount woman dial strap boy
Topic 10:
```





Adjusted Rand Score (ARI) : 0.10

Classification des textes

TfidfVectorizer

LDA

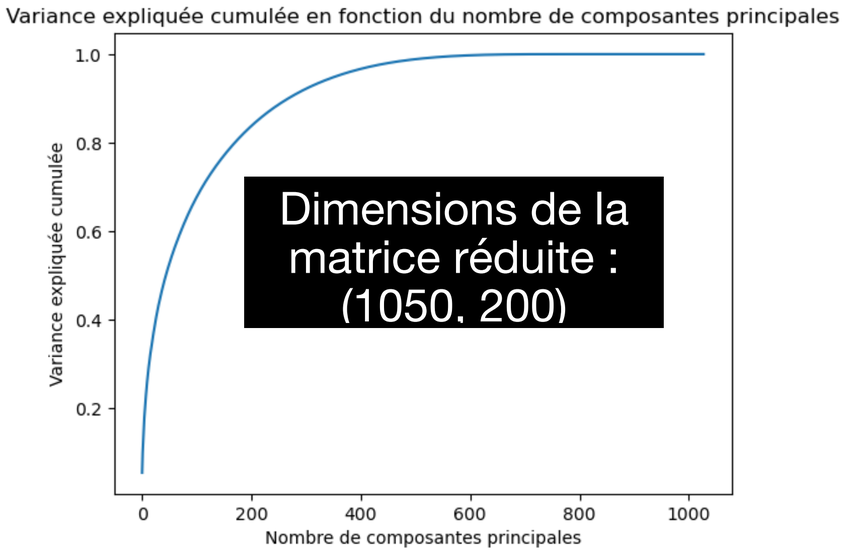
```
from sklearn.decomposition import LatentDirichletAllocation
n_topics = 20

# Créer le modèle LDA
lda = LatentDirichletAllocation(
        n_components=n_topics,
        max_iter=5,
        learning_method='online'
        random_state=0)

# Fitter sur les données
lda.fit(tfidf_bow_df)
```

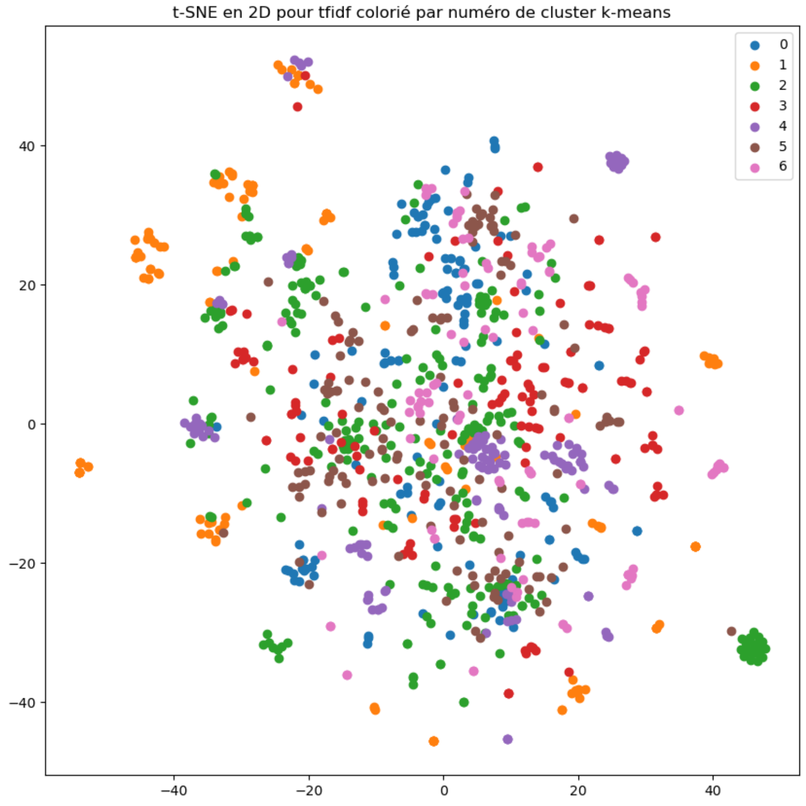
cliquer pour afficher toute la sortie ; double-cliquer pour masquer la sortie

```
LatentDirichletAllocation(learning_method='online', learning_offset=50.0,
                          max_iter=5, n_components=20, random_state=0)
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

```
def display_topics(model, feature_names, no_top_words):
    for topic_idx, topic in enumerate(model.components_):
        print("Topic {}:".format(topic_idx))
        print(" ".join([feature_names[i] for i in topic.argsort()[:-no_top_words - 1:-1]]))


no_top_words = 10
display_topics(lda, tfidf.get_feature_names_out(), no_top_words)
```





Adjusted Rand Score (ARI) : 0.28

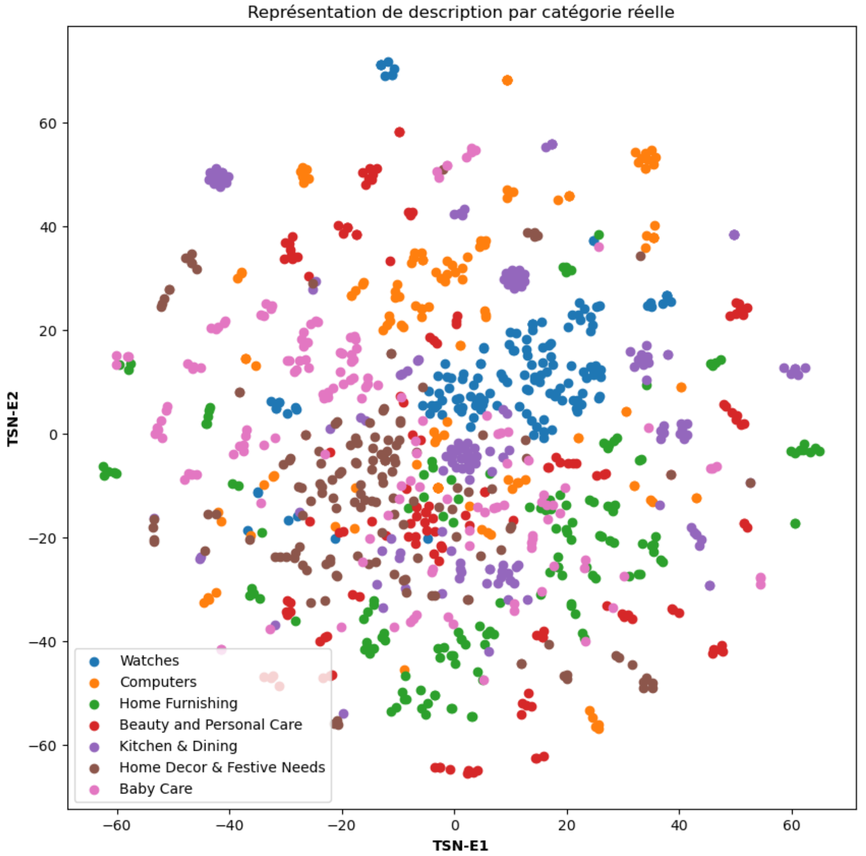
Classification des textes

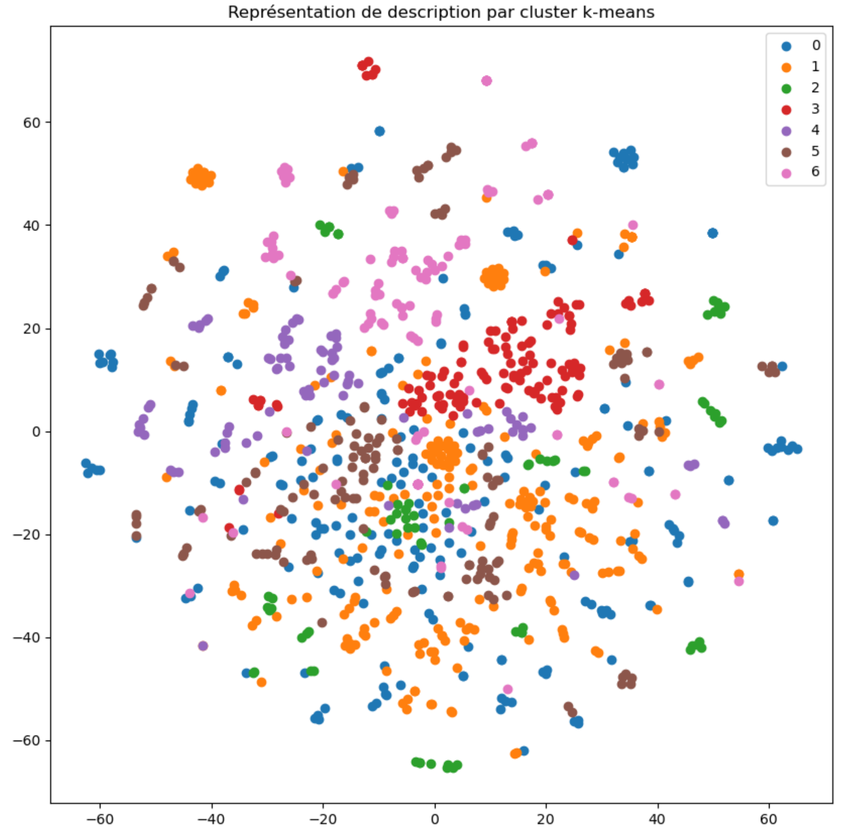
Word2Vec

```
w2v_size=300
w2v_window=5
w2v_min_count=1
w2v_epochs=100
maxlen = 24 # adapt to length of sentences
sentences = df['description'].to_list()
sentences = [gensim.utils.simple_preprocess(text) for text in sentences]
```

```
# Création et entraînement du modèle Word2Vec

print("Build & train Word2Vec model ...")
w2v_model = gensim.models.Word2Vec(min_count=w2v_min_count, window=w2v_window,
                                   vector_size=w2v_size,
                                   seed=42,
                                   workers=1)
#                                    workers=multiprocessing.cpu_count())
w2v_model.build_vocab(sentences)
w2v_model.train(sentences, total_examples=w2v_model.corpus_count, epochs=w2v_epochs)
model_vectors = w2v_model.wv
w2v_words = model_vectors.index_to_key
print("Vocabulary size: %i" % len(w2v_words))
print("Word2Vec trained")
```

```
Build & train Word2Vec model ...
Vocabulary size: 4934
Word2Vec trained
```

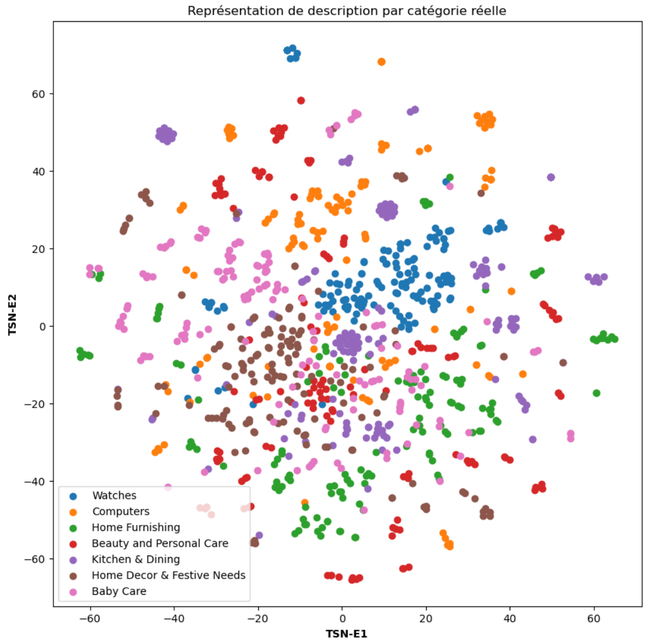


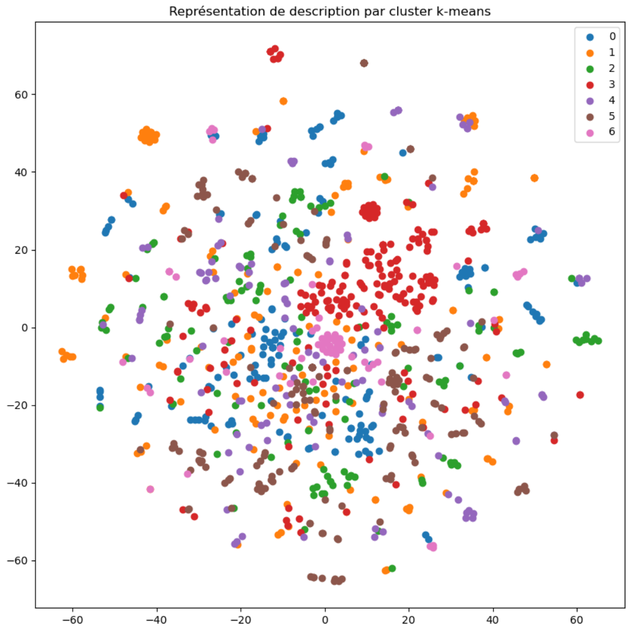


(ARI) : 0.31

Classification des textes

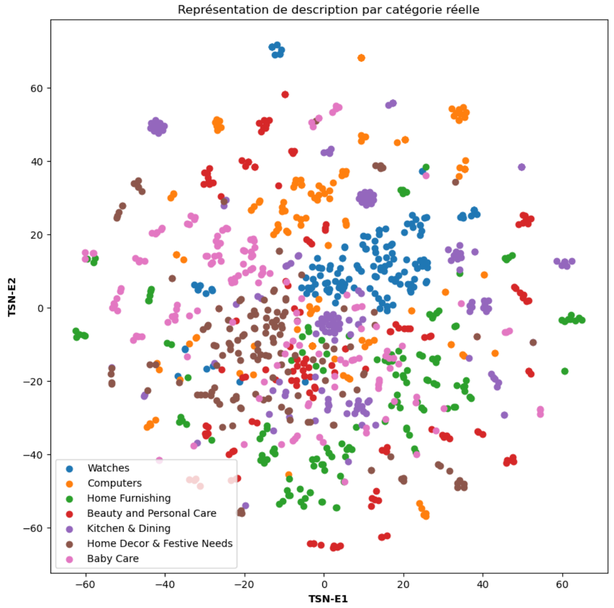
BERT

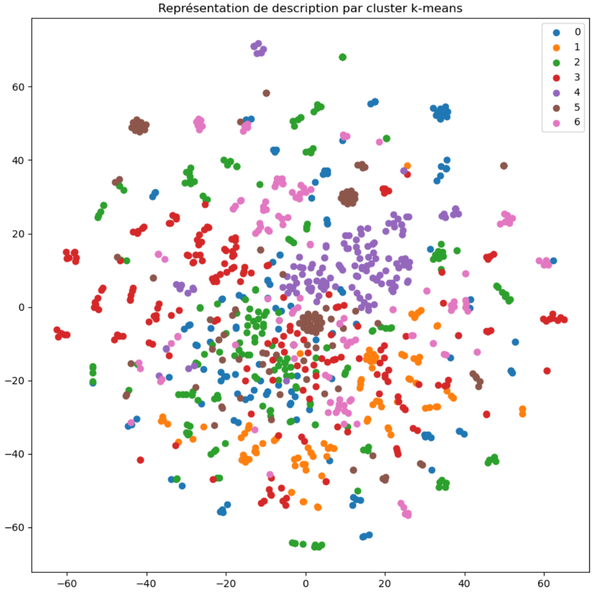




(ARI) : 0.19

USE





(ARI) : 0.39

Classification des Images

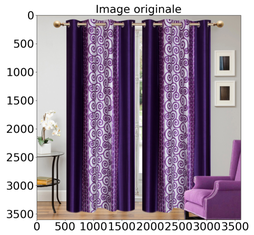
Affichage d'exemples d'images par label

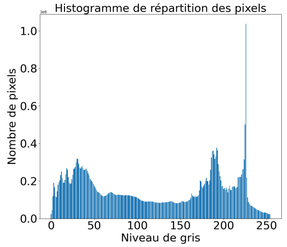
Home Furnishing

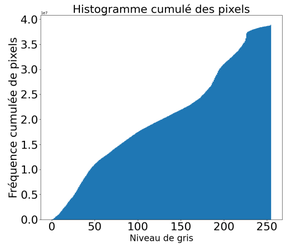

Baby Care

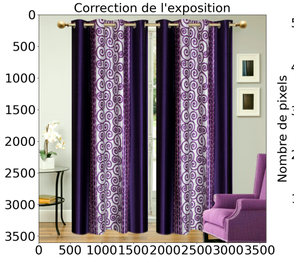

Watches

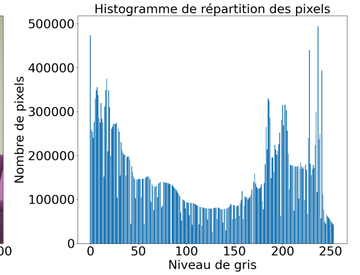

images pour tester les procédures de prétraitement

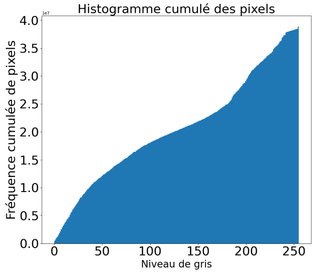


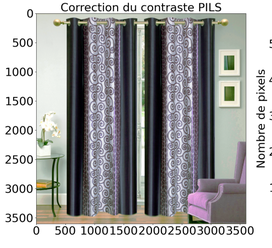


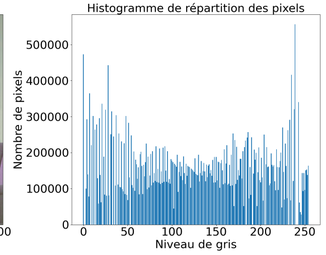


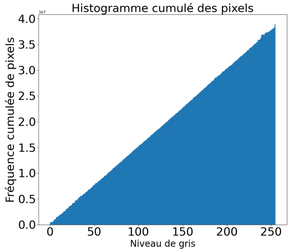


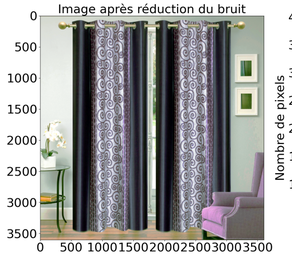


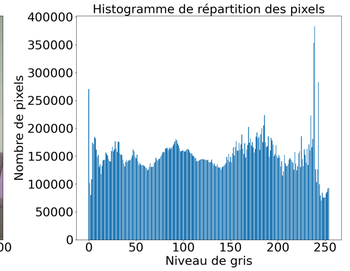


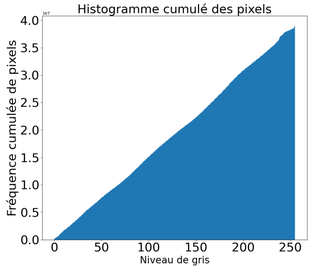


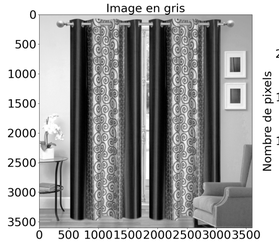


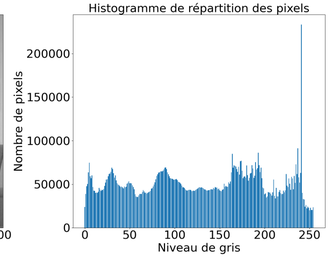


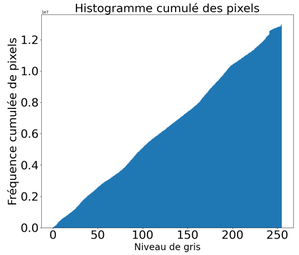


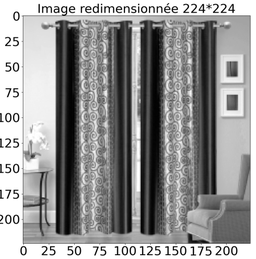


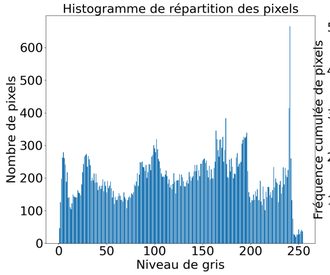


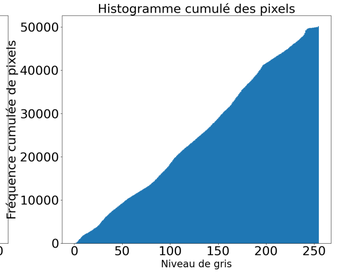












Classification des Images

Méthode SIFT

- Pre-processing des images
- Extraire et réunir l'ensemble des descripteurs SIFT
- Création du Bag Of Virtual Words (Utilisation de MiniBatchKMeans)
- Création des histogrammes
- Réduction de dimension
- Clustering
- Evaluation du score ARI

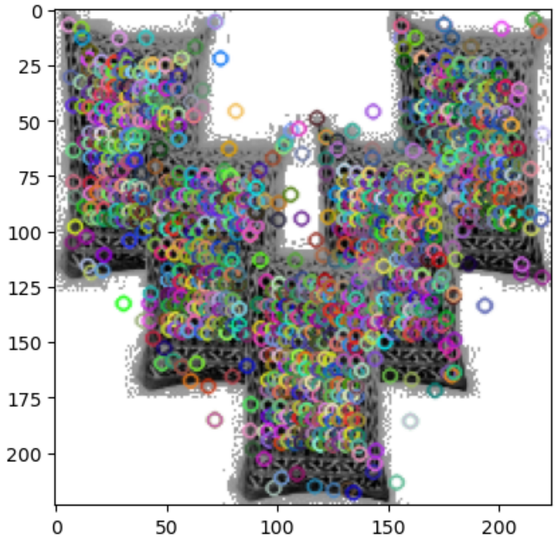
Réductions de dimension

- PCA

La réduction PCA permet de créer des features décorrélées entre elles, et de diminuer leur dimension, tout en gardant un niveau de variance expliquée élevé (99%)

```
print("Dimensions dataset avant réduction PCA : ", im_features.shape)
pca = decomposition.PCA(n_components=0.99)
pca_results_img= pca.fit_transform(im_features)
print("Dimensions dataset après réduction PCA : ", pca_results_img.shape)
```

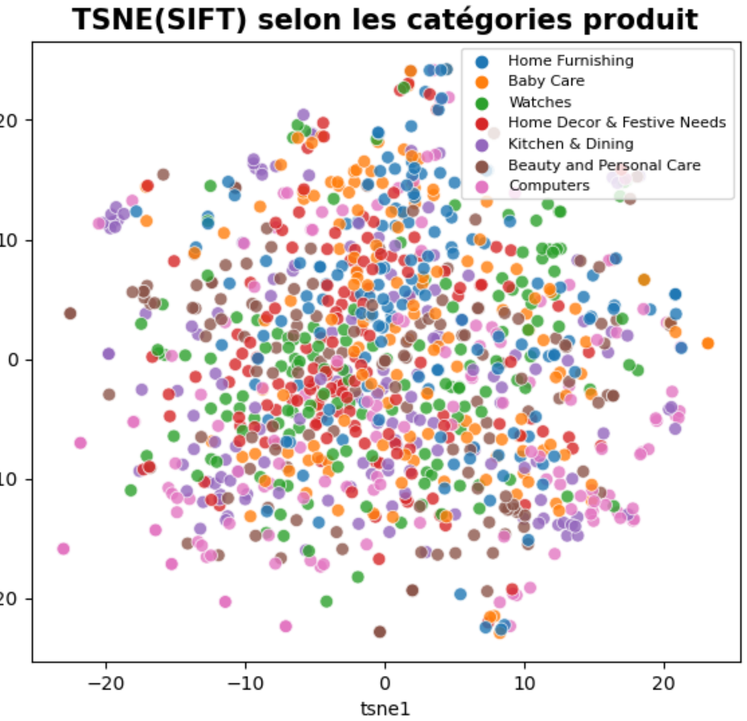
```
Dimensions dataset avant réduction PCA :  (1050, 583)
Dimensions dataset après réduction PCA :  (1050, 477)
```

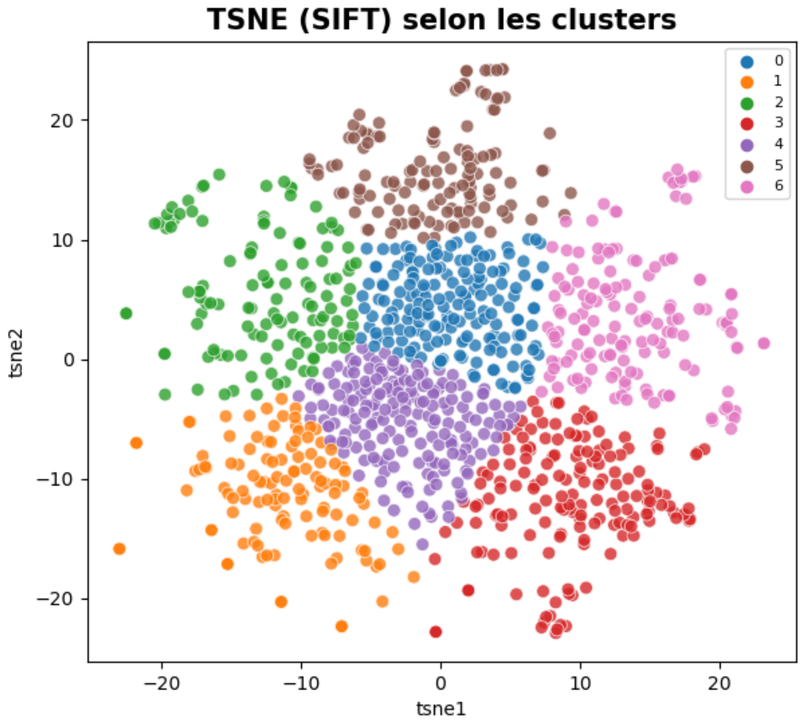



```
Descripteurs :  (1411, 128)

[[  0.   0.   0. ...   0.   0.   0.]
 [  9.   5.   1. ...   0.   0.   2.]
 [  9.   7. 173. ...   0.   0.  10.]
 ...
 [ 71.  13.   1. ...   0.   0.   0.]
 [  0.   0.  22. ...   0.   0.  43.]
 [ 83.  34.  20. ...   1.   4.   1.]]
```

Classification des Images

Méthode SIFT





ARI : 0.0349

Classification des Images

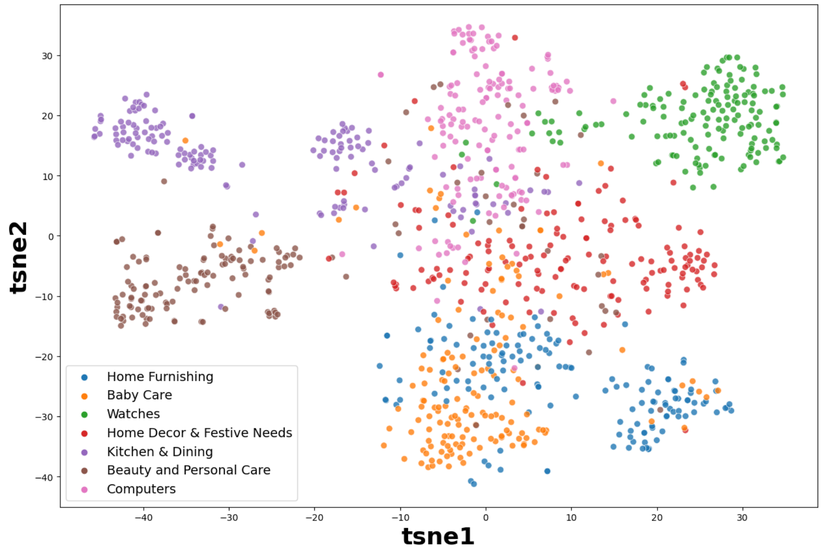
Faisabilité de classification automatique d'images via CNN Transfer Learning

Création du modèle d'étude avec VGG16

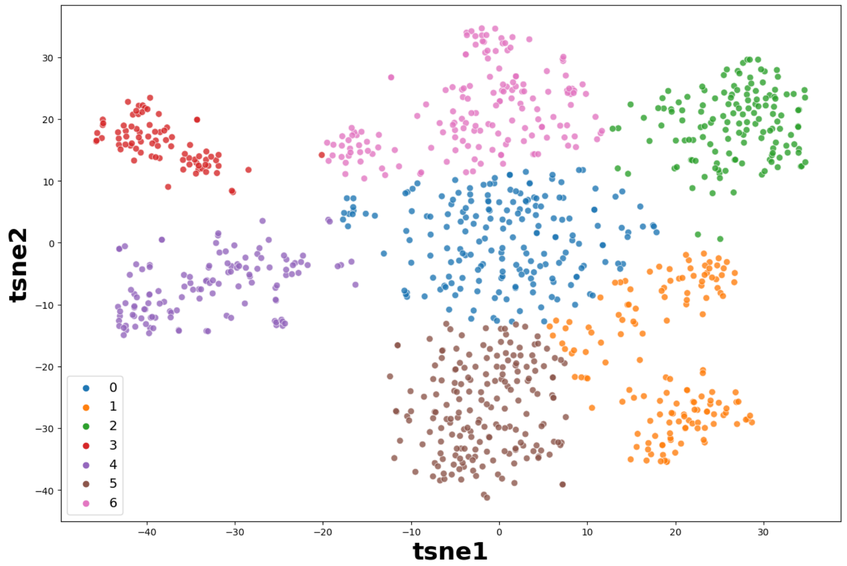
- Chargement des images, extraction des features

```
# Utilisation de VGG16 pré-entraîné sur ImageNet avec les images d'origine

# Liste
vgg16_all_features = []

# Instanciation du modèle
model_vgg16 = VGG16()

# Remove the output layer
model_vgg16= Model(inputs=model_vgg16.inputs, outputs=model_vgg16.layers[-2].output)

# Résumé de l'architecture du modèle
model_vgg16.summary()

for rep_image in df_img['image_loc']:

    # Charger l'image et la redimensionner à la taille
    # requise de 224×224 pixels.

    img = load_img(rep_image, target_size=(224, 224))

    # Convertir les pixels en un tableau NumPy afin de pouvoir travailler
    # avec dans Keras
```

TSNE (VGG-16) selon les catégories produit



TSNE (VGG-16) selon les clusters



Réduction de dimension

- PCA

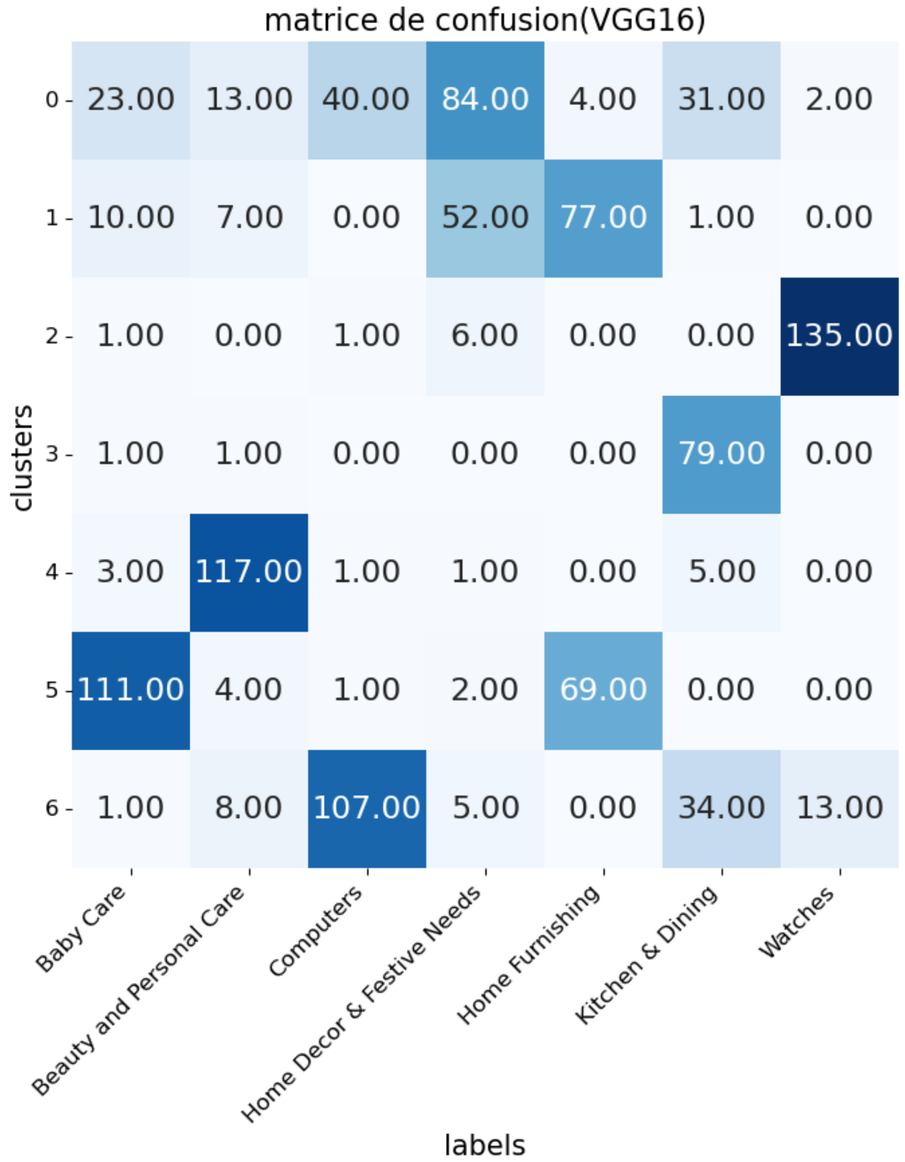
```
print("Dimensions dataset avant réduction PCA : ", df_vgg16_vectors.shape)
pca = decomposition.PCA(n_components=0.99)
pca_results_vgg16 = pca.fit_transform(df_vgg16_vectors)
print("Dimensions dataset après réduction PCA : ", pca_results_vgg16.shape)
```

```
Dimensions dataset avant réduction PCA :  (1050, 4096)
Dimensions dataset après réduction PCA :  (1050, 803)
```

ARI : 0.456

Classification des Images

Faisabilité de classification automatique d'images via CNN Transfer Learning



	modèle	ARI
0	BOW	0.100
1	TF-IDF	0.280
2	Word2Vec	0.310
3	BERT	0.180
4	USE	0.390
5	SIFT	0.034
6	CNN	0.450

Classification des Images

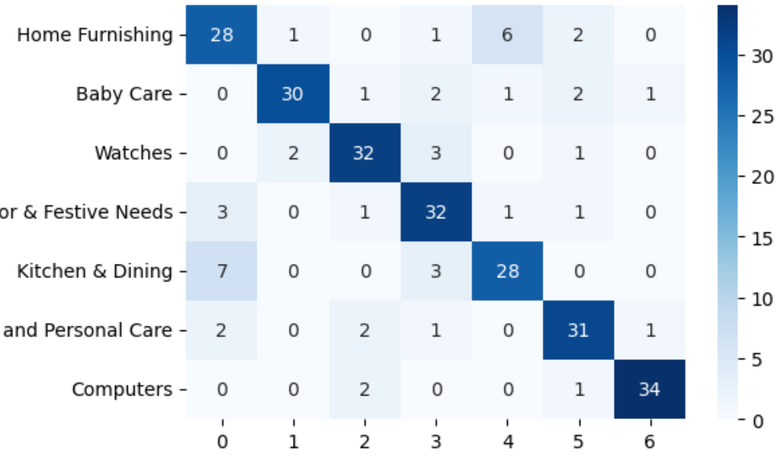
Classification supervisée

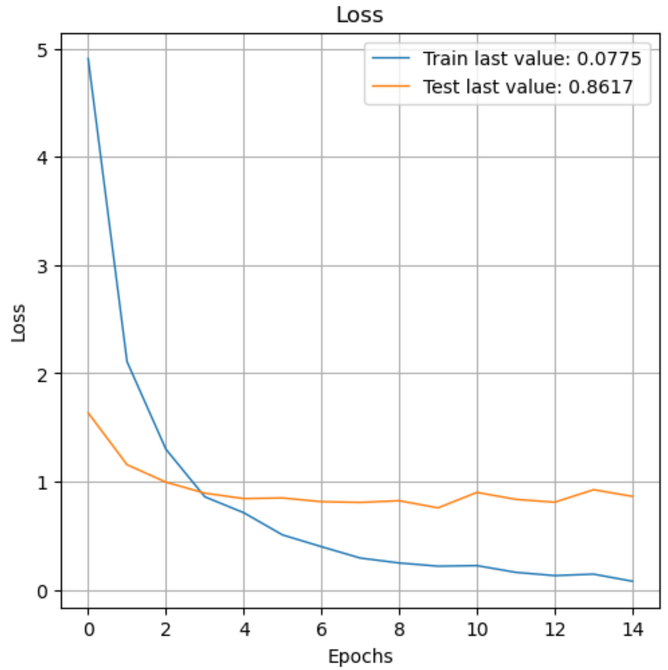
Approche préparation initiale des images

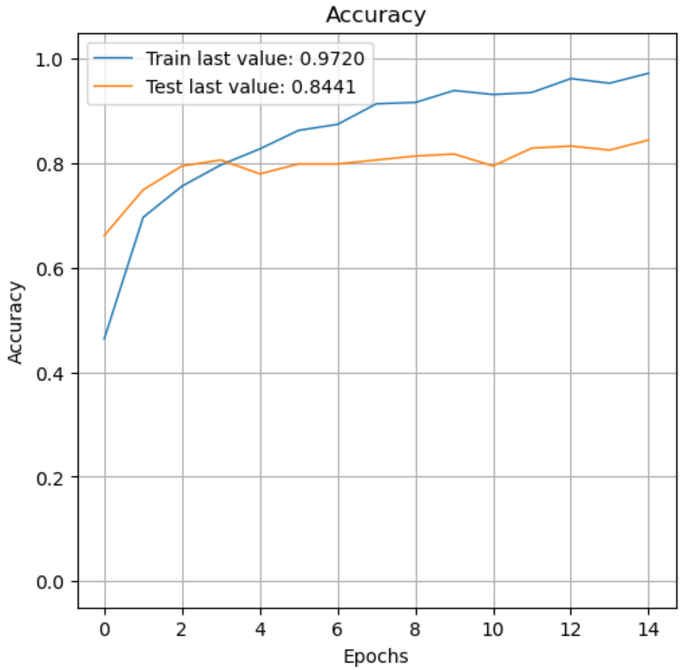
```
def image_prep_fct(data) :
    prepared_images = []
    for image_num in range(len(data['image_loc'])) :
        img = (load_img(
            data['image_loc'][image_num],
            target_size=(224, 224)))
        img = img_to_array(img)
        img = img.reshape((img.shape[0], img.shape[1], img.shape[2]))
        img = preprocess_input(img)
        prepared_images.append(img)
        prepared_images_np = np.array(prepared_images)
    return prepared_images_np

images_np = image_prep_fct(df_img)
print(images_np.shape)
```

cliquer pour afficher toute la sortie ; double-cliquer pour masquer la sortie

```
(1050, 224, 224, 3
```







Classification des Images

Classification supervisée

Approche ImageDatagenerator avec data augmentation

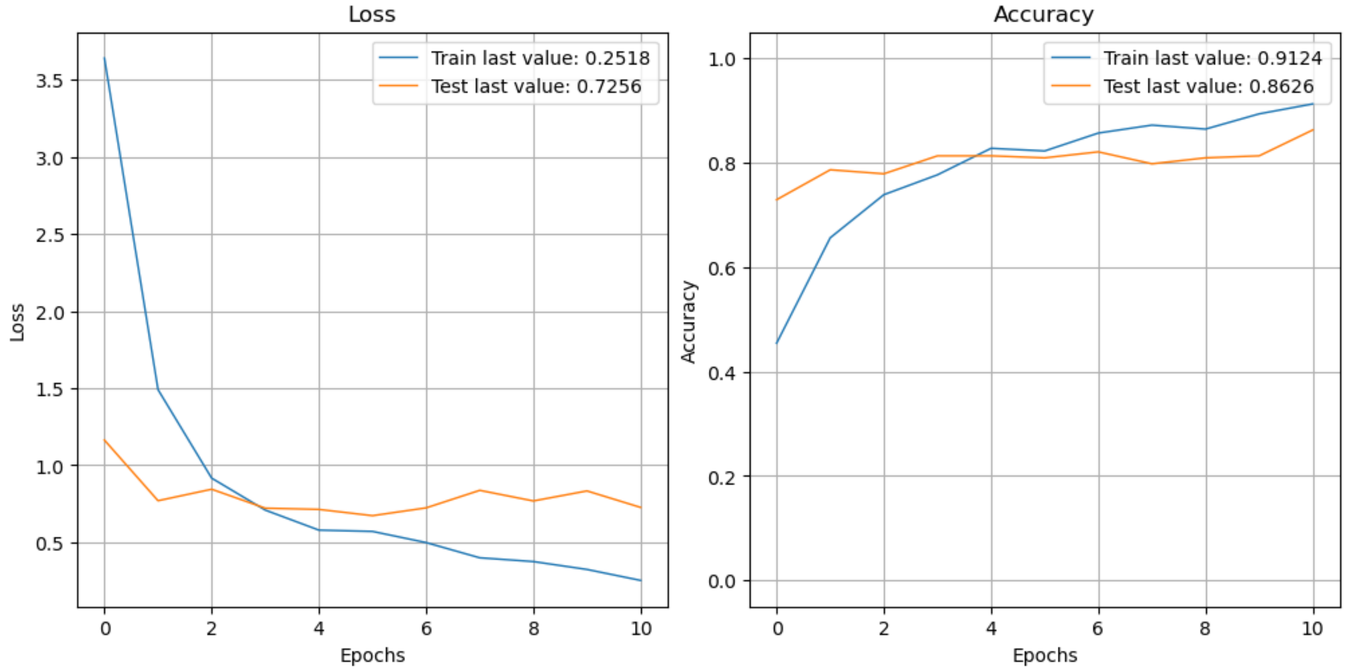
```
from tensorflow.keras.preprocessing.image import ImageDataGenerator
```

```
# Pour mélanger les images, classées initalement par classe
data = df_img.sample(frac=1, random_state=42).reset_index(drop=True)
```

```
list_labels = data["categ_1"].unique()
label_no_name = "no_name"
```

```
def photo_class(photo) :
    for label in list_labels :
        if label in photo[24:] :
            return label
    return label_no_name

data["label_name"] = data["categ_1"]
batch_size = 32

def data_flow_fct(data, datagen, data_type=None) :
    data_flow = datagen.flow_from_dataframe(data, directory='',
                                x_col='image_loc', y_col='label_name',
                                weight_col=None, target_size=(256, 256
                                classes=None, class_mode='categorical'
                                batch_size=batch_size, shuffle=True, s
                                subset=data_type
                                )
    return data_flow
```

```
datagen_train = ImageDataGenerator(
```



Test d’une API

Test d’une API de collecte de données de change

```
import requests

url = "https://edamam-food-and-grocery-database.p.rapidapi.com/api/food-database/v2/parser"

querystring = {"ingr":"champagne","nutrition-type":"cooking","category[0]":"generic-foods","health[0]":"alcohol-free"

head
"X-RapidAPI-Key": "a022ca4f70msh9c7f30971be4b65p137713jsn74f4354b06ab",
"X-RapidAPI-Host": "edamam-food-and-grocery-database.p.rapidapi.com"
}

response = requests.get(url, headers=headers, params=querystring)

Reponse = response.json()
```

cliquer pour faire défiler la sortie vers le haut ; double-cliquer pour masquer

	Unnamed: 0	foodId	nutrients	label	category	image
0	0	food_a656mk2a5dmqb2adiamu6beihduu	{'ENERC_KCAL': 82.0, 'PROCNT': 0.07, 'FAT': 0....	Champagne	Generic foods	https://www.edamam.com/food-img/a71/a718cf3c52...
1	1	food_b753ithamdb8psbt0w2k9aquo06c	{'ENERC_KCAL': 571.0, 'PROCNT': 0.0, 'FAT': 64...	Champagne Vinaigrette, Champagne	Packaged foods	No Image
2	2	food_b3dyababjo54xobm6r8jzbghjgqe	{'ENERC_KCAL': 333.0, 'PROCNT': 0.0, 'FAT': 36...	Champagne Vinaigrette, Champagne	Packaged foods	https://www.edamam.com/food-img/d88/d88b64d973...
3	3	food_a9e0ghsamvoc45bwa2ybsa3gken9	{'ENERC_KCAL': 500.0, 'PROCNT': 0.0, 'FAT': 50...	Champagne Vinaigrette, Champagne	Packaged foods	No Image
4	4	food_an4jjueaucpus2a3u1ni8auhe7q9	{'ENERC_KCAL': 194.0, 'PROCNT': 0.0, 'FAT': 16...	Champagne Vinaigrette, Champagne	Packaged foods	No Image
5	5	food_bmu5dmkazwuvpaa5prh1daa8jxs0	{'ENERC_KCAL': 500.0, 'PROCNT': 0.0, 'FAT': 50...	Champagne Dressing, Champagne	Packaged foods	https://www.edamam.com/food-img/ab2/ab2459fc2a...
6	6	food_alpl44taoyv11ra0lic1qa8xculi	{'ENERC_KCAL': 431.28471367026697, 'PROCNT': 0...	Champagne Buttercream	Generic meals	No Image
7	7	food_byap67hab6evc3a0f9w1oag3s0qf	{'ENERC_KCAL': 116.24649258757483, 'PROCNT': 0...	Champagne Sorbet	Generic meals	No Image
8	8	food_am5egz6aq3fpjlaf8xpkdbc2asis	{'ENERC_KCAL': 402.26144434996013, 'PROCNT': 6...	Champagne Truffles	Generic meals	No Image
9	9	food_bcz8rhiajk1fuva0vkfmeakbouc0	{'ENERC_KCAL': 412.10989377242726, 'PROCNT': 0...	Champagne Vinaigrette	Generic meals	No Image

Conclusion

En conclusion, l'approche utilisant Tf-idf montre des résultats prometteurs, mais peut être améliorée en ajustant le prétraitement des descriptions. Le modèle VGG-16 surpasse le modèle SIFT en termes de performances. Le clustering confirme la faisabilité et la satisfaction du moteur de classification avec les données textuelles, tandis que les données visuelles présentent des limitations. L'augmentation des données améliore les performances du modèle. Ces constatations ouvrent des perspectives intéressantes pour le développement futur.

MERCI