# **Formation** : Data scientist openclassrooms

# **Projet 6**: Classifiez automatiquement des biens de consommation

**Préparé par** : Ben Douma Hosni

# Sommaire

Problématique

Description des données

Etude de faisabilité

- Analyse des descriptions textuelles et clustering KMeans
- Analyse des images et clustering KMeans

Classification Supervisée

Test de l'API

Conclusion

# Problématique

L'entreprise "**Place de marché**" aspire à **automatiser l'attribution des catégories** des articles au sein de son marketplace e-commerce.

Dans cette optique, **une étude de faisabilité** est envisagée pour la création d'**un moteur de classification automatique**, exploitant les données **textuelles et les images** associées aux articles.

# Description du dataset

Fichier csv : "flipkart_com-ecommerce_sample_1050.csv" de taille (1050, 15)

| | uniq_id | crawl_timestamp | product_url | product_name | product_category_tree | pid | reta |
|---|---|---|---|---|---|---|---|
| 0 | 55b85ea15a1536d46b7190ad6fff8ce7 | 2016-04-30 03:22:56 +0000 | http://www.flipkart.com/elegance-polyester-mul... | Elegance Polyester Multicolor Abstract Eyelet ... | ["Home Furnishing >> Curtains & Accessories >>... | CRNEG7BKMFFYHQ8Z | |
| 1 | 7b72c92c2f6c40268628ec5f14c6d590 | 2016-04-30 03:22:56 +0000 | http://www.flipkart.com/sathiyas-cotton-bath-t... | Sathiyas Cotton Bath Towel | ["Baby Care >> Baby Bath & Skin >> Baby Bath T... | BTWEGFZHGBXPHZUH | |
| 2 | 64d5d4a258243731dc7bbb1eef49ad74 | 2016-04-30 03:22:56 +0000 | http://www.flipkart.com/eurospa-cotton-terry-f... | Eurospa Cotton Terry Face Towel Set | ["Baby Care >> Baby Bath & Skin >> Baby Bath T... | BTWEG6SHXTDB2A2Y | |

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1050 entries, 0 to 1049
Data columns (total 15 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   uniq_id                 1050 non-null   object
 1   crawl_timestamp         1050 non-null   object
 2   product_url             1050 non-null   object
 3   product_name            1050 non-null   object
 4   product_category_tree   1050 non-null   object
 5   pid                     1050 non-null   object
 6   retail_price            1049 non-null   float64
 7   discounted_price        1049 non-null   float64
 8   image                   1050 non-null   object
 9   is_FK_Advantage_product 1050 non-null   bool
 10  description             1050 non-null   object
 11  product_rating          1050 non-null   object
 12  overall_rating          1050 non-null   object
 13  brand                   712 non-null    object
 14  product_specifications  1049 non-null   object
dtypes: bool(1), float64(2), object(12)
memory usage: 116.0+ KB
```

Ajout d'une variable "category" en se basant sur product_category_tree

```
['Home Furnishing ', 'Baby Care ', 'Watches ',
 'Home Decor & Festive Needs ', 'Kitchen & Dining ',
 'Beauty and Personal Care ', 'Computers '], dtype=object)
```

Les variables a utilisée sont:

```
df['description'].describe()
```

```
count                                                1050
unique                                               1050
top       Key Features of Elegance Polyester Multicolor ...
freq                                                    1
Name: description, dtype: object
```

```
df['product_name'].describe()
```

```
count                                                1050
unique                                               1050
top         Elegance Polyester Multicolor Abstract Eyelet ...
freq                                                    1
Name: product_name, dtype: object
```

# Etude de faisabilité

1. Analyse des variables textuelles:

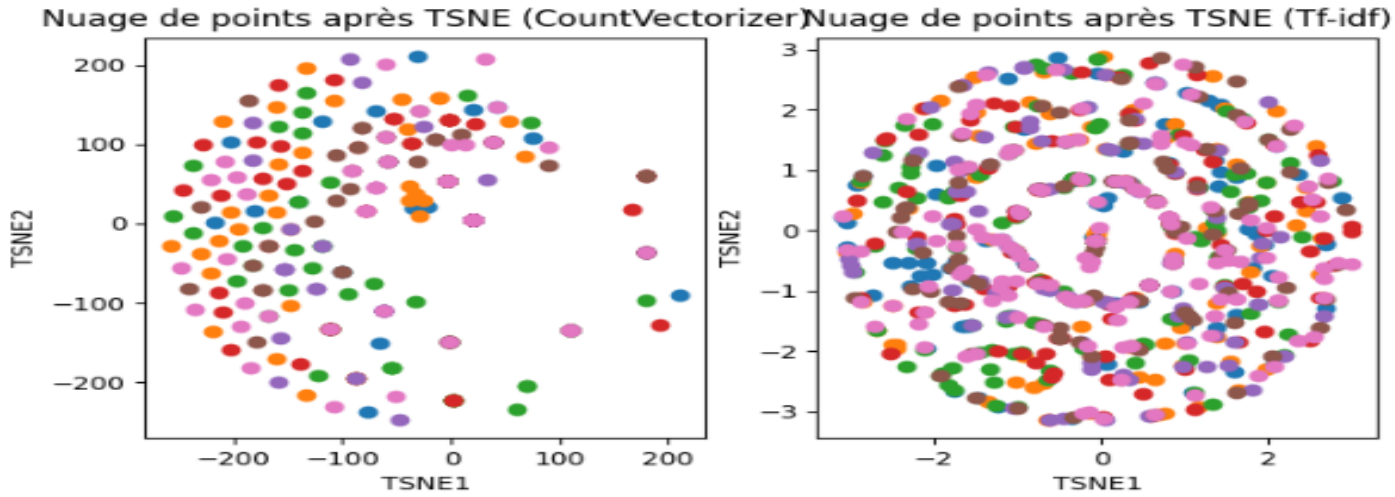
Démarche:

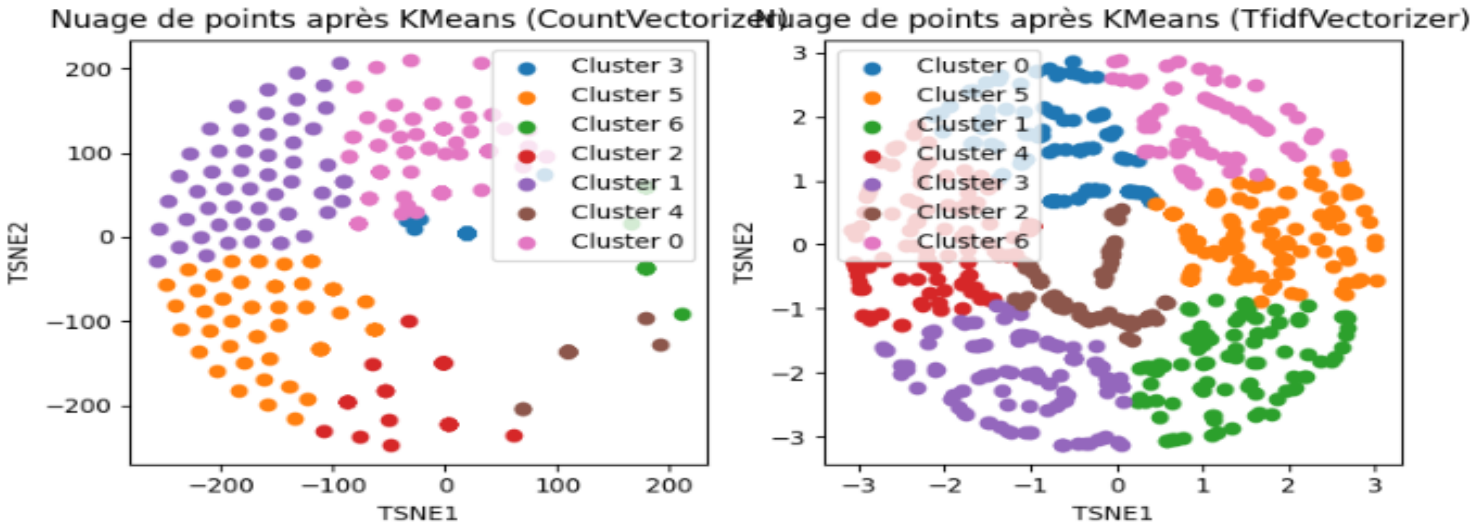
- Nettoyage
- Vectorisation
- Reduction des dimensions
- Clustering avec KMeans
- Calcul ARI

# Variable "description"

1. **CountVectorizer et Tf-idf**

Nettoyage et vectorisation (tokenisation, stopswords,lower,lemmatisation)

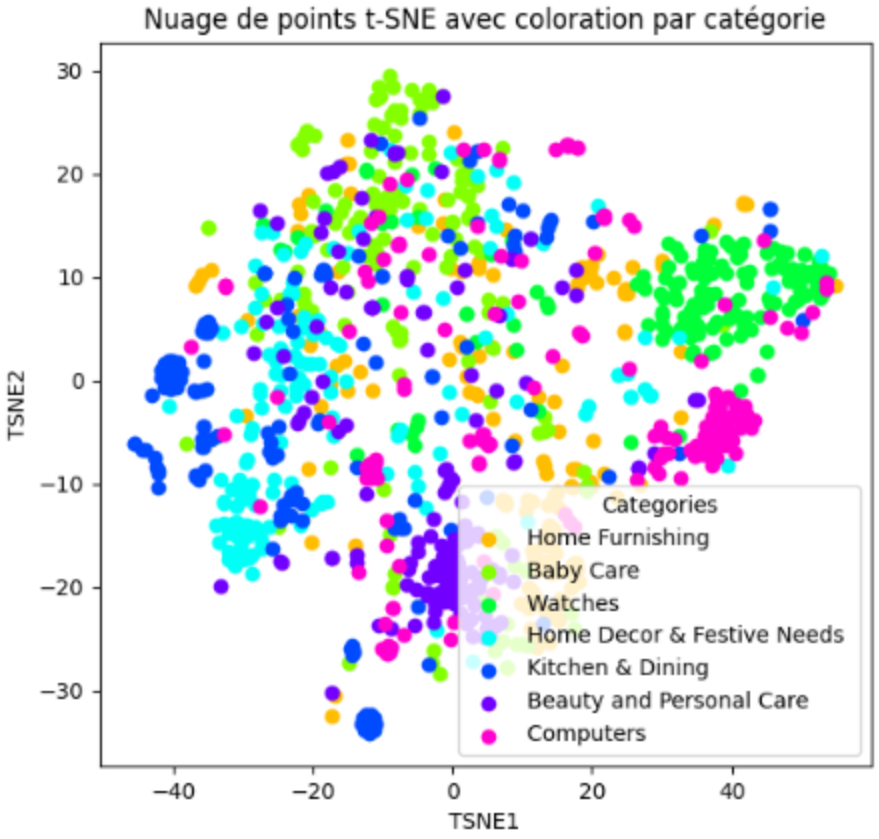


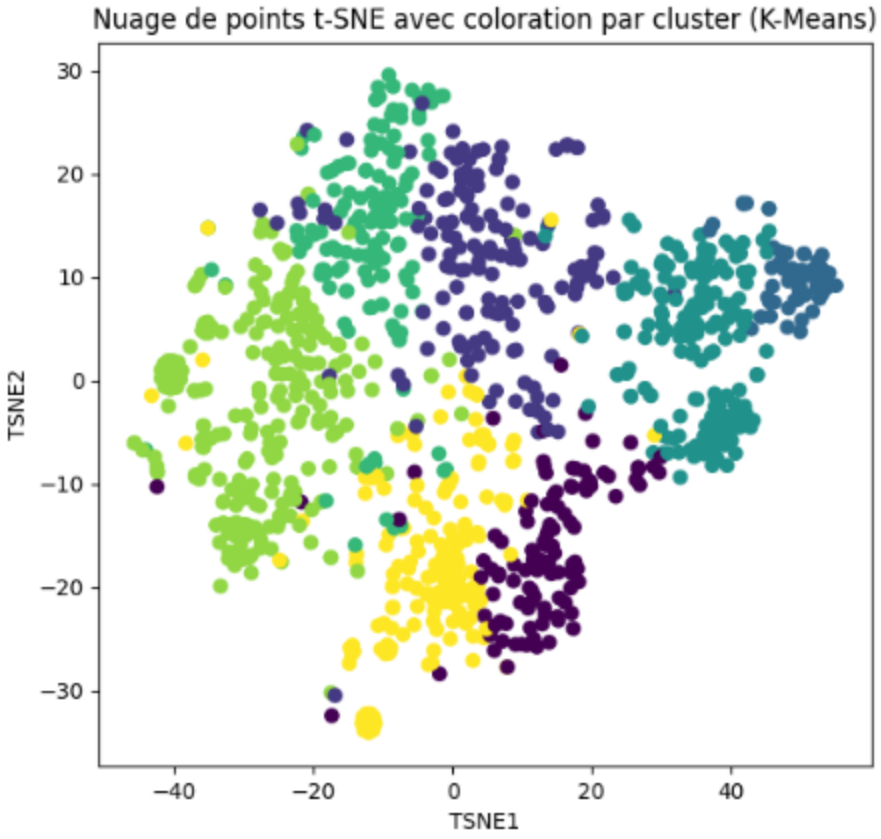




ARI (cv): 0.038
ARI (tf_idf): 0.007

# 2. Word2Vec

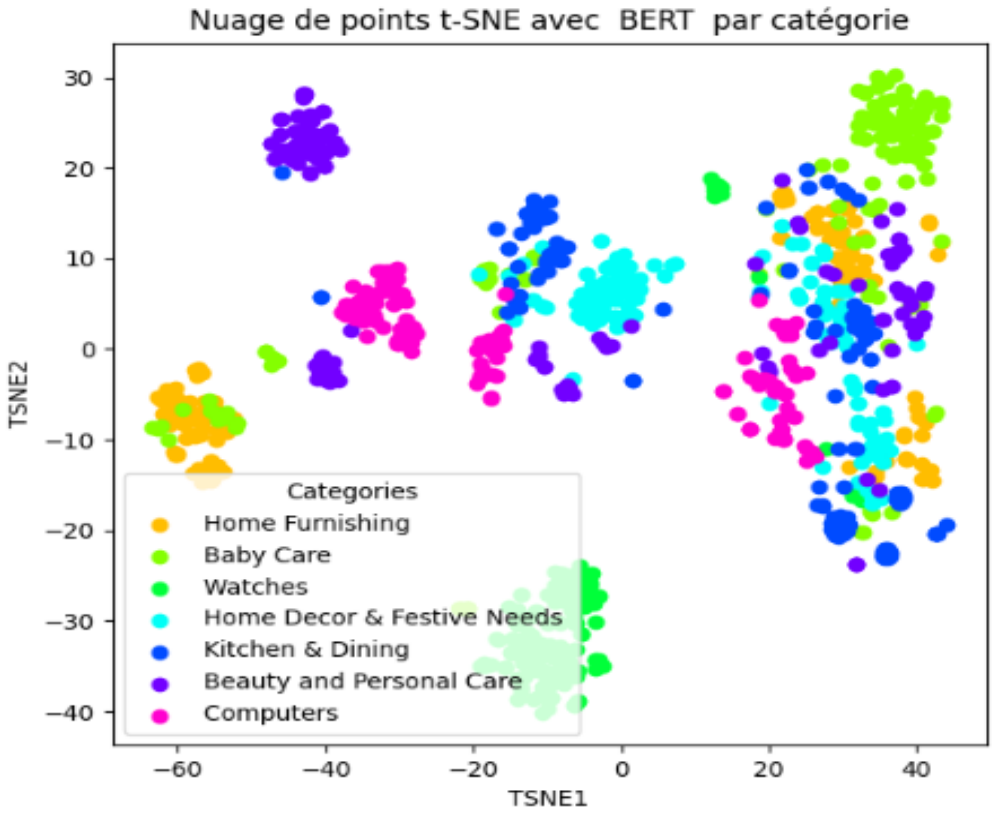
Nettoyage et vectorisation (tokenisation, stopswords, lower, lemmatisation)

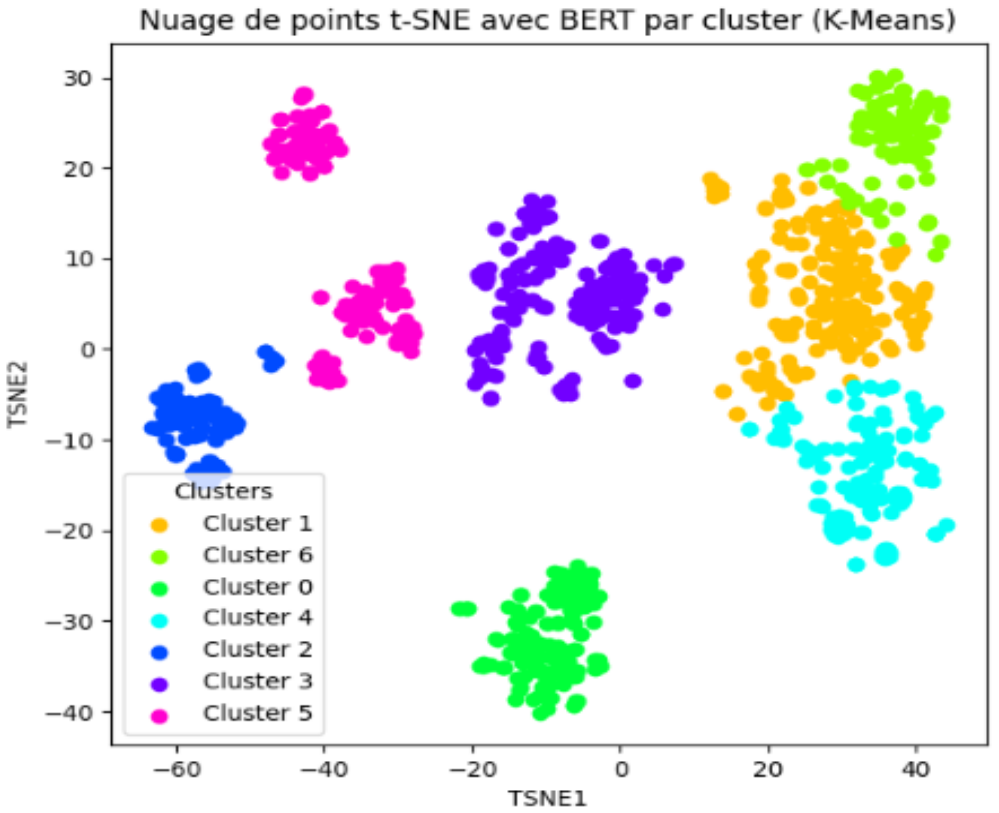



ARI: 0.17

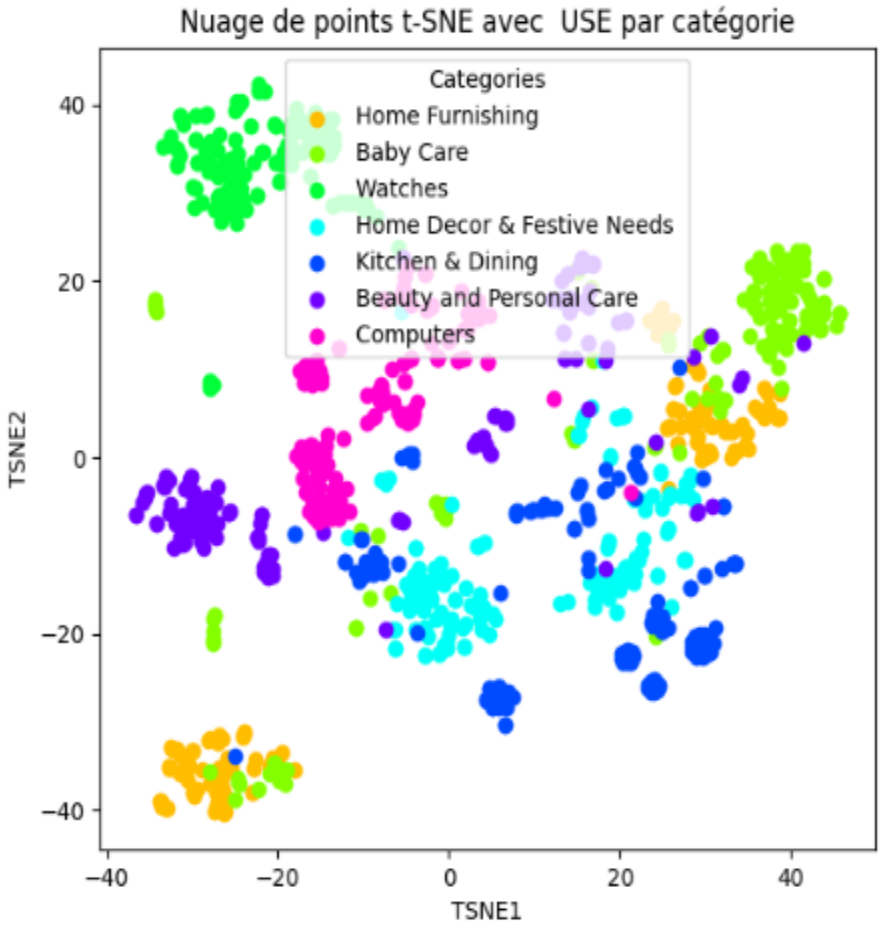
# 3. Bert

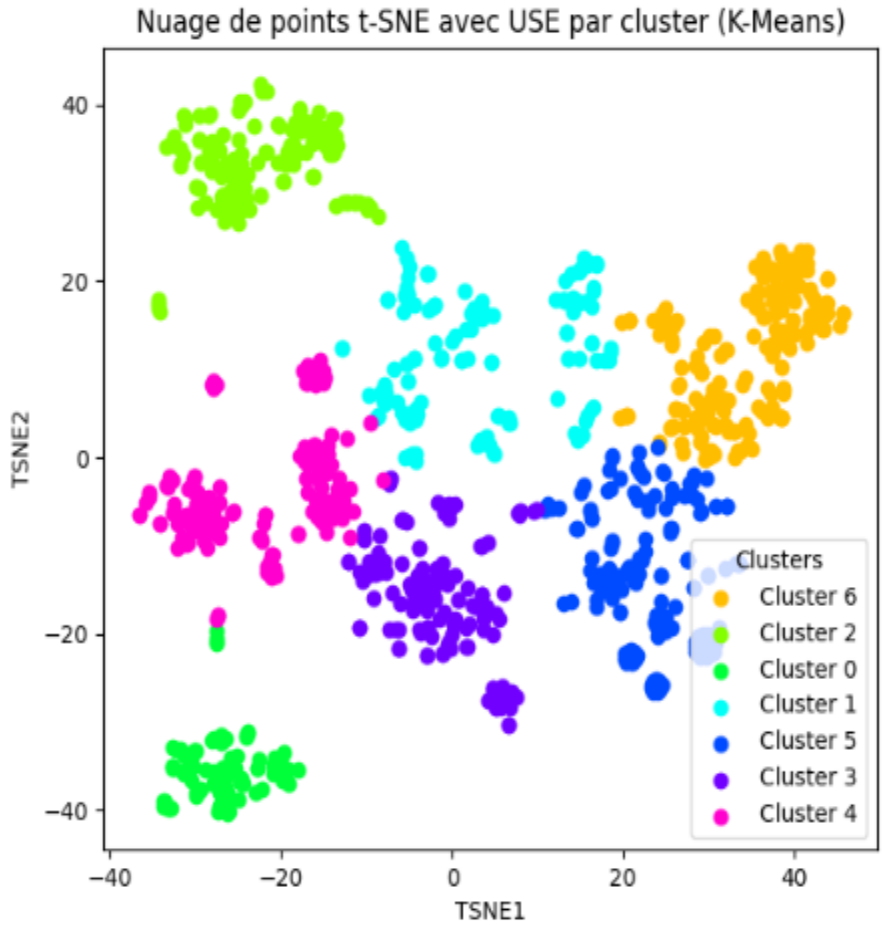



ARI: 0.30

# 4. USE


Nuage de points t-SNE avec  USE par catégorie
Categories
Home Furnishing
Baby Care
Watches
Home Decor & Festive Needs
Kitchen & Dining
Beauty and Personal Care
Computers
TSNE1
TSNE2

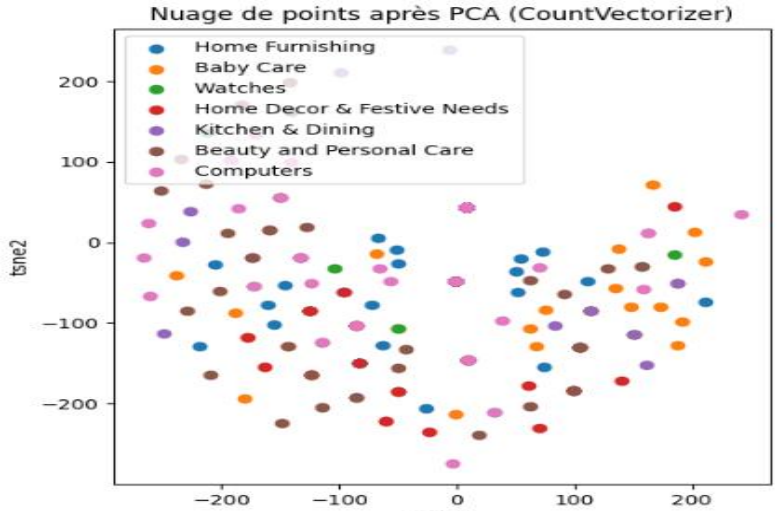

Nuage de points t-SNE avec USE par cluster (K-Means)
Clusters
Cluster 6
Cluster 2
Cluster 0
Cluster 1
Cluster 5
Cluster 3
Cluster 4
TSNE1
TSNE2

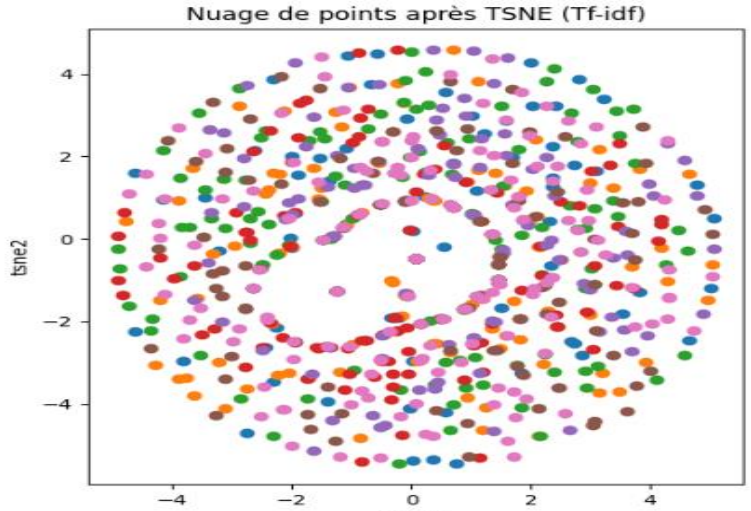
ARI: 0.43

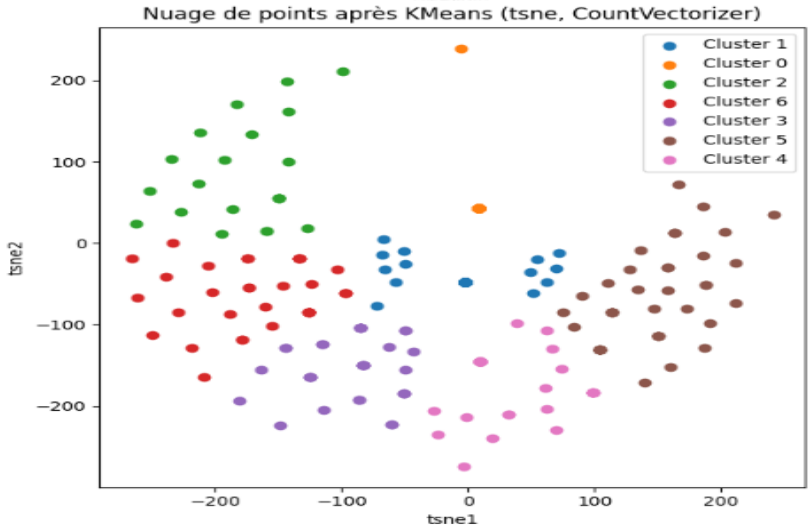
# la variable product_name

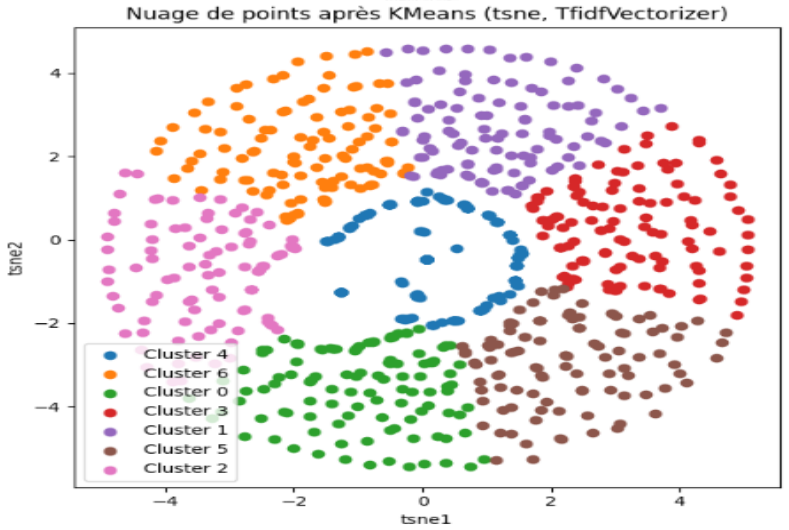
## 1. CountVectorizer et Tf_Idf

Nettoyage et vectorisation (tokenisation, stopswords,lower,lemmatisation)



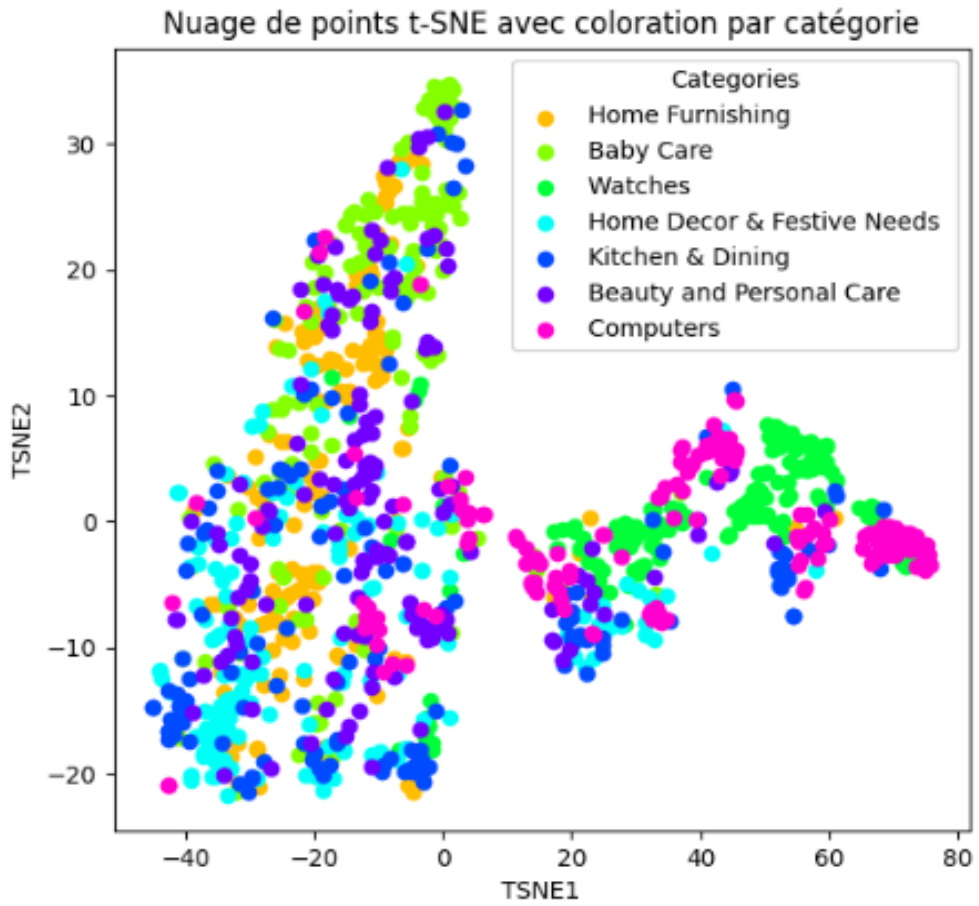



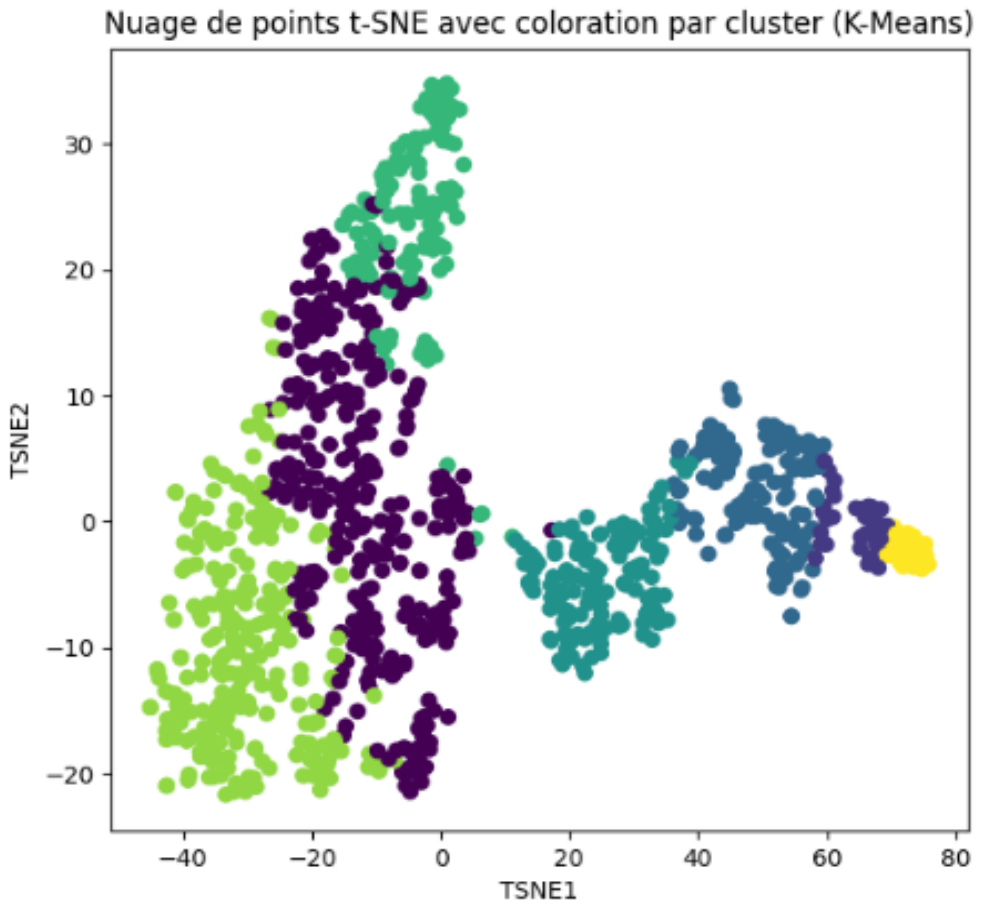




ARI(tf): 0.01
ARI(cv): 0.02

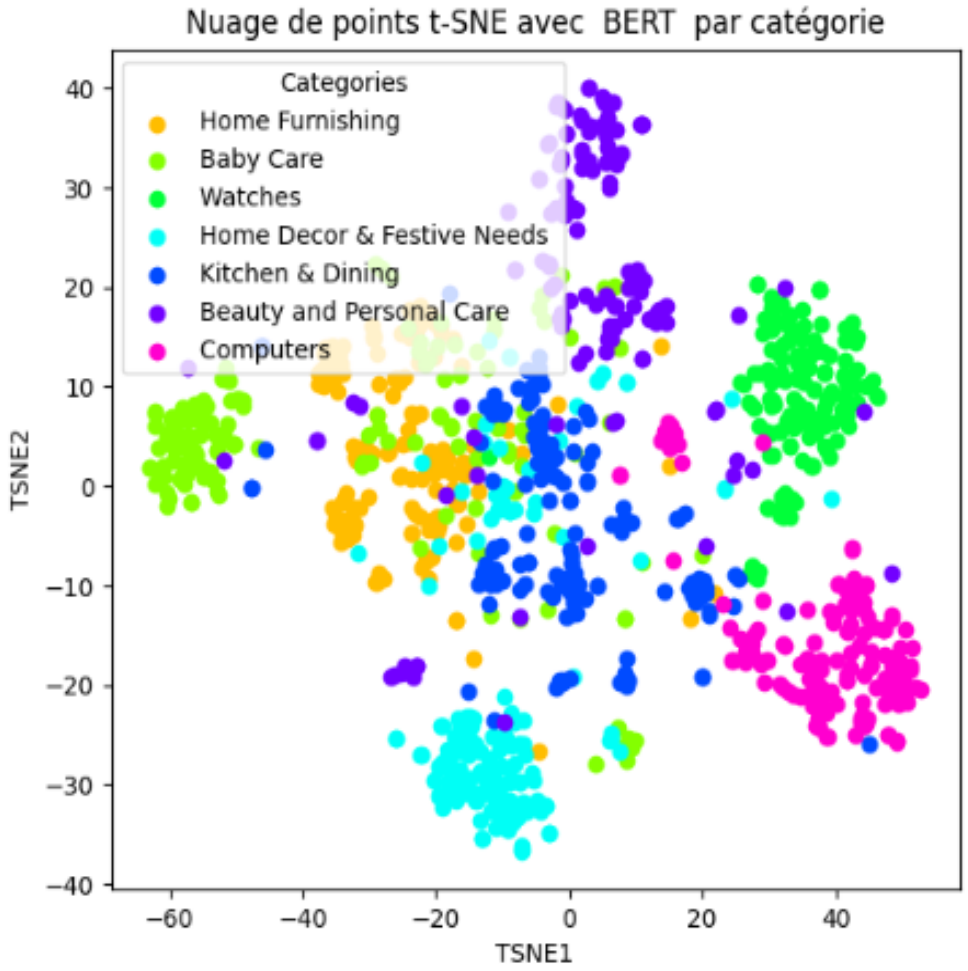
# 2. Word2Vec

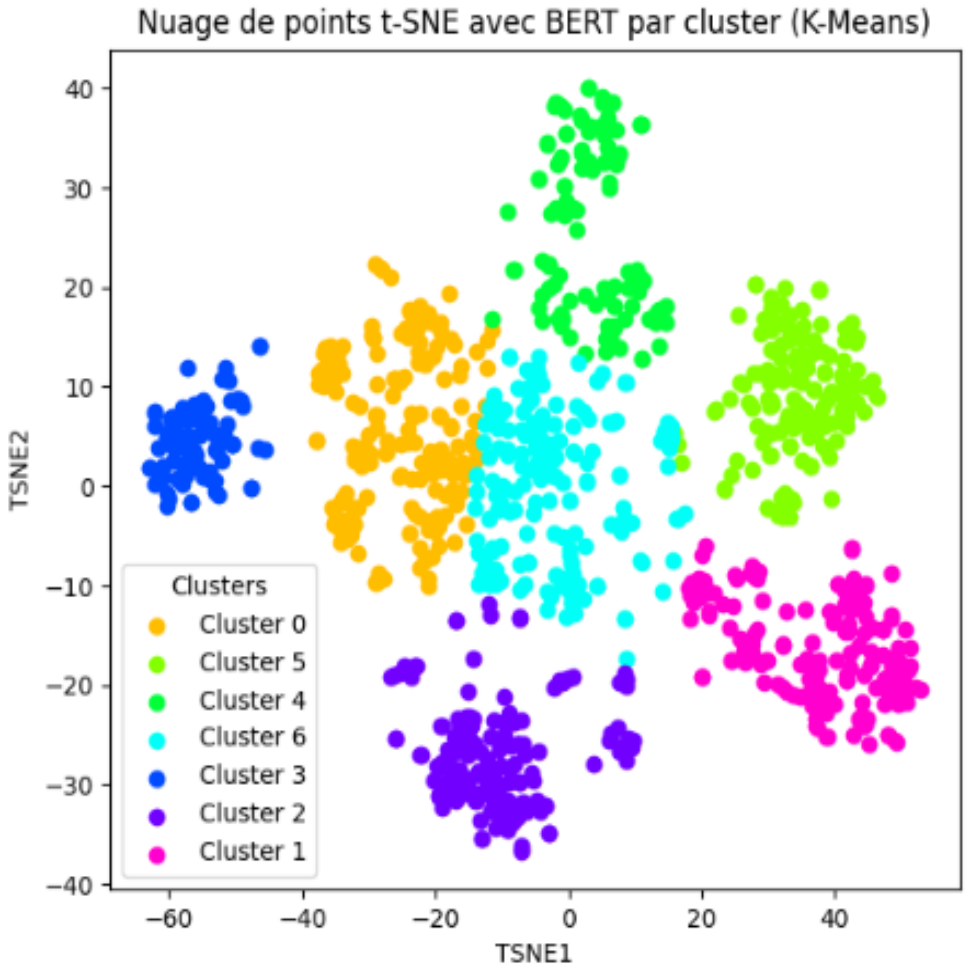



ARI: 0.11

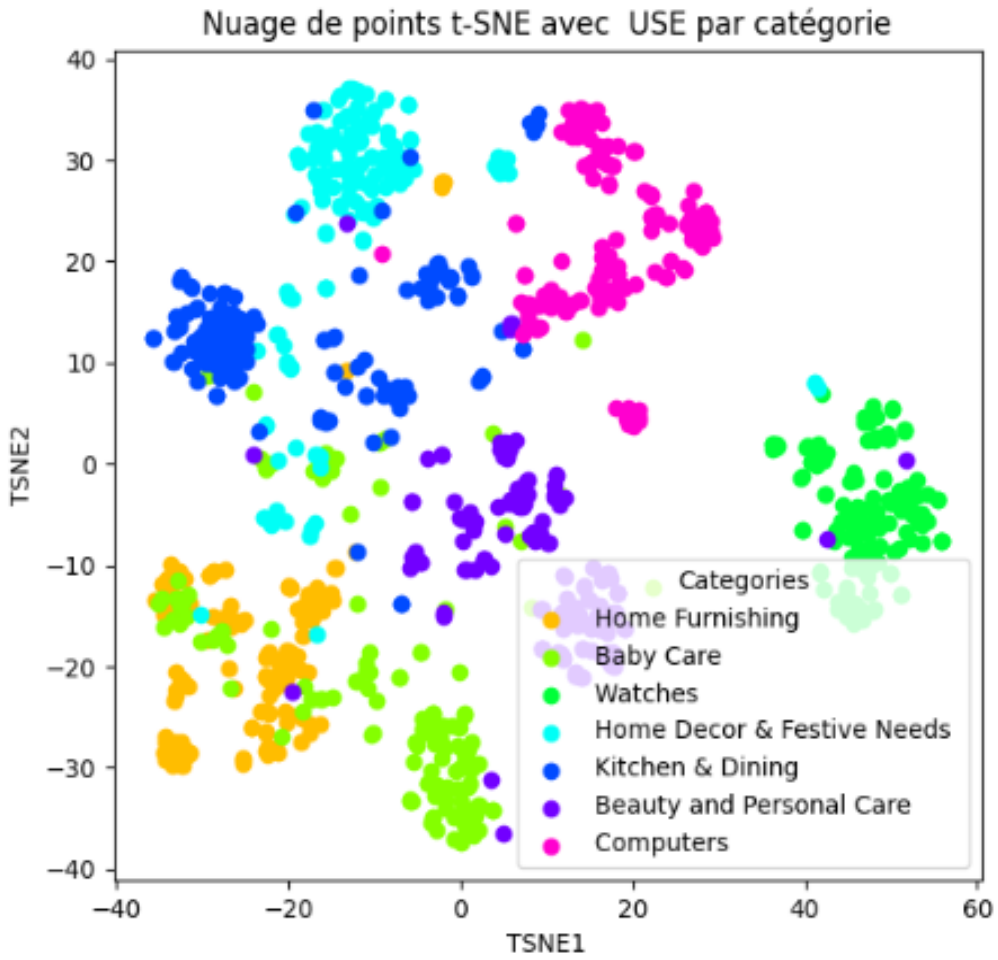
# 3. Bert

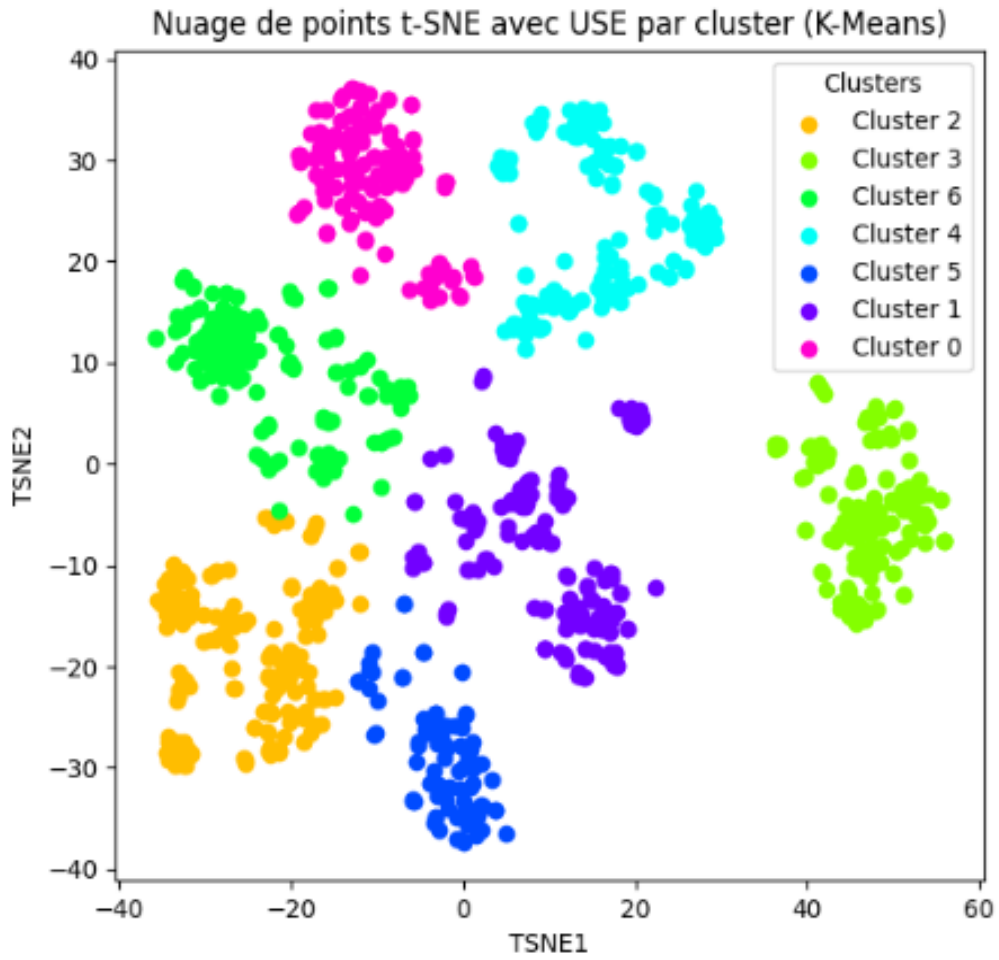



ARI: 0.58

# 4. USE


Nuage de points t-SNE avec  USE par catégorie
TSNE2
TSNE1
Categories
Home Furnishing
Baby Care
Watches
Home Decor & Festive Needs
Kitchen & Dining
Beauty and Personal Care
Computers


Nuage de points t-SNE avec USE par cluster (K-Means)
TSNE2
TSNE1
Clusters
Cluster 2
Cluster 3
Cluster 6
Cluster 4
Cluster 5
Cluster 1
Cluster 0

ARI: 0.67

# Concaténation de deux variables : description et product_name

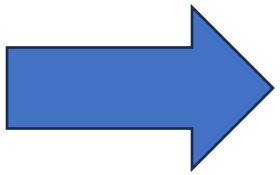
| Modèle | USE | BERT | Word2Vec |
|---|---|---|---|
| ARI | 0.45 | 0.33 | 0.10 |

Les résultats des modèles ne sont pas meilleurs en comparaison avec les résultats précédents

# Tableau comparatif des ARI

| Modele | Ari (variable description) | Ari (Variable product_name) |
|---|---|---|
| CountVectorizer | 0.03 | 0.02 |
| Tf_Idf | 0.007 | 0.01 |
| Word2Vec | 0.17 | 0.11 |
| BERT | 0.30 | 0.58 |
| USE | 0.43 | 0.67 |

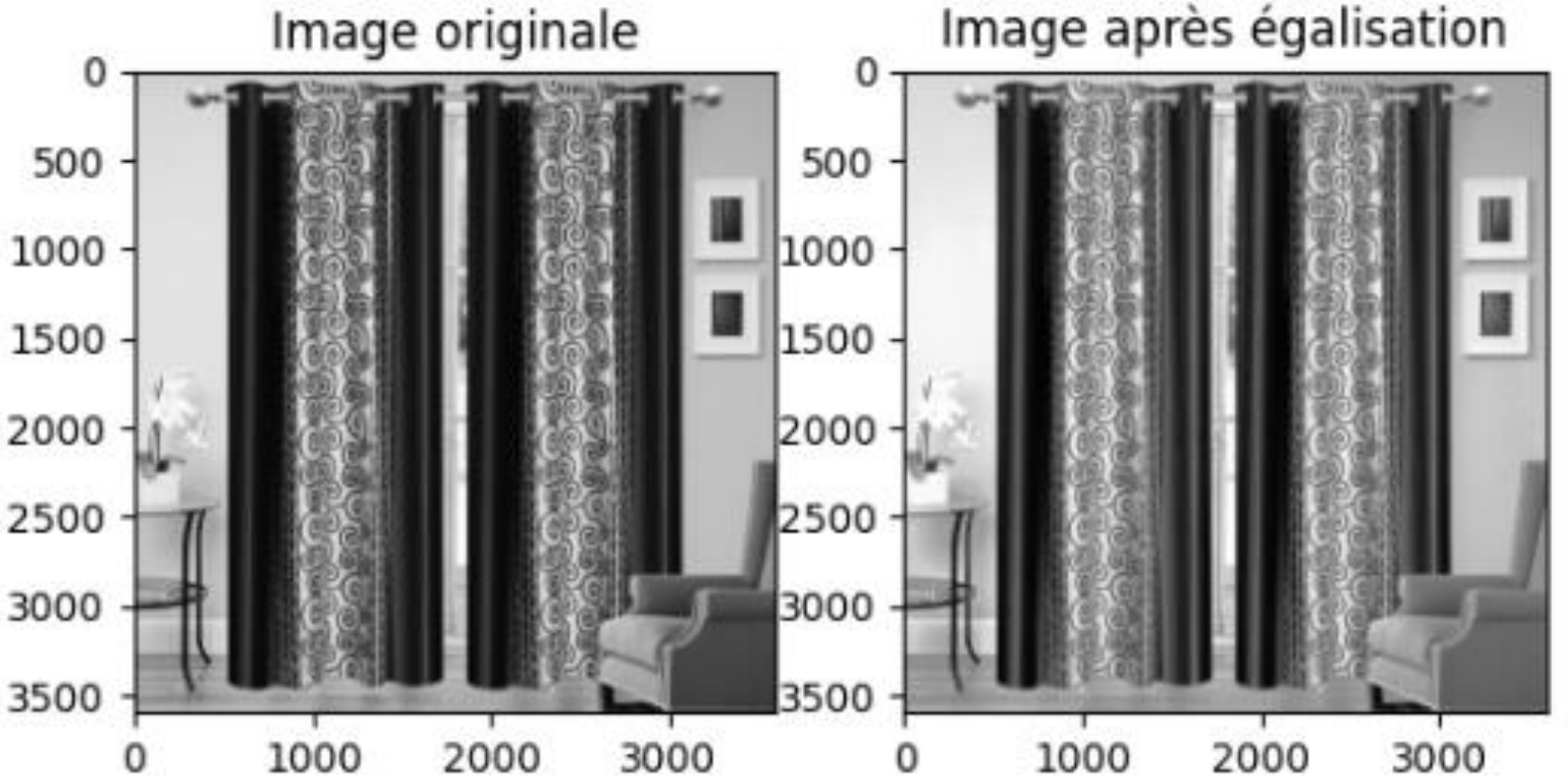
En se basant sur les caractéristiques extraites à travers les modèles Transformers (BERT et USE) Kmeans a réussi à identifier des structures similaires à celles présentes dans la variable "catégory"



On peut conclure à la faisabilité de l'idée d'un moteur de classification en se basant sur les données textuelles pour une catégorisation automatique

## 2. Analyse des images :

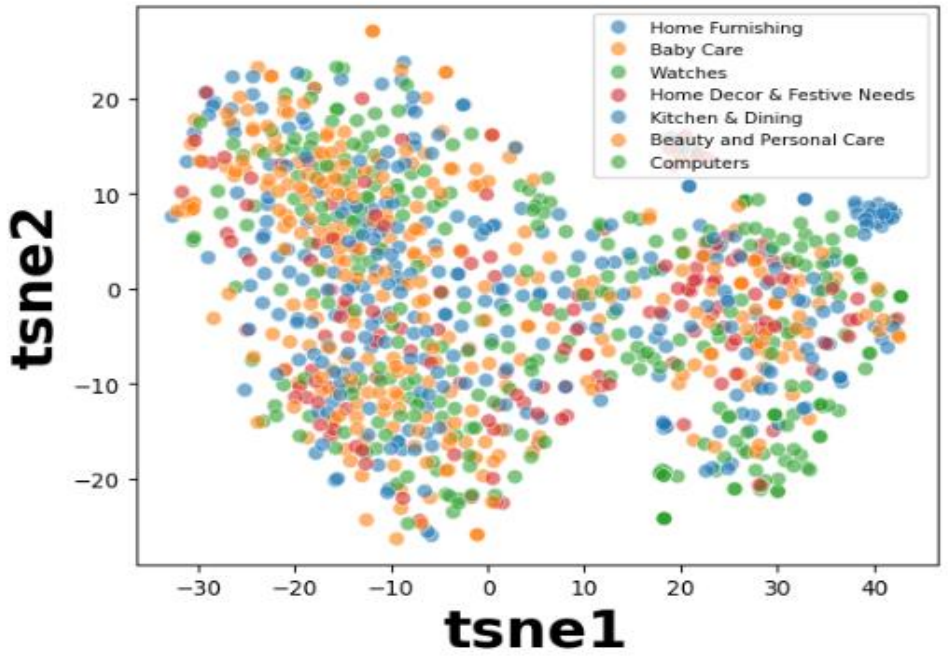
- Application d'une égalisation d'histogramme sur une image



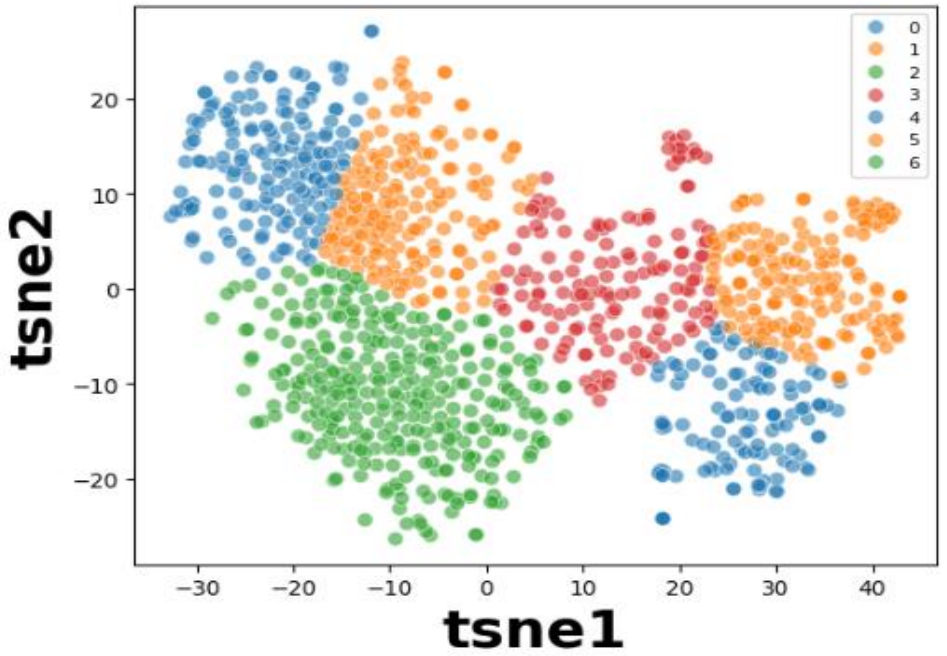
# Test du modèle ORB

- Extraction des caractéristiques pour toutes les images avec ORB
- Création des clusters de descripteurs
- Création des caractéristiques des images par comptage pour chaque numéro de cluster du nombre de descripteurs de l'image
- Réduction de dimension :PCA /TSNE
- Création de clusters (KMeans) à partir du T-SNE

**TSNE selon les vraies classes**



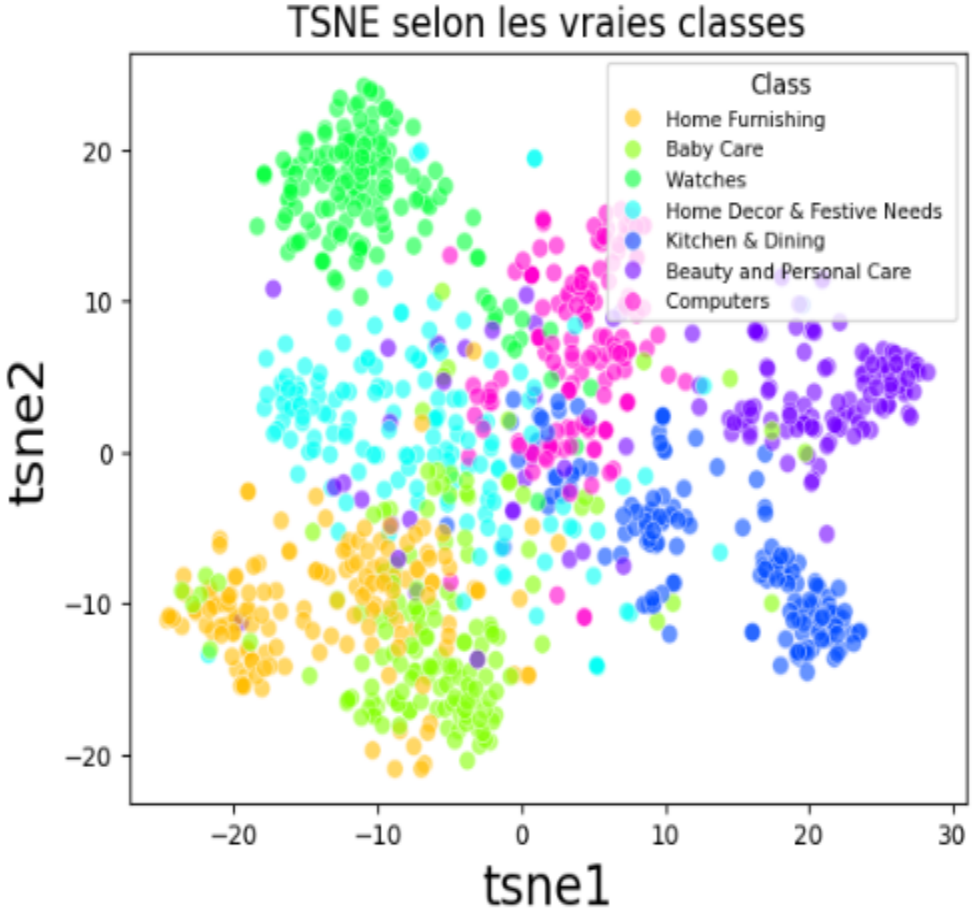

**TSNE selon les clusters**

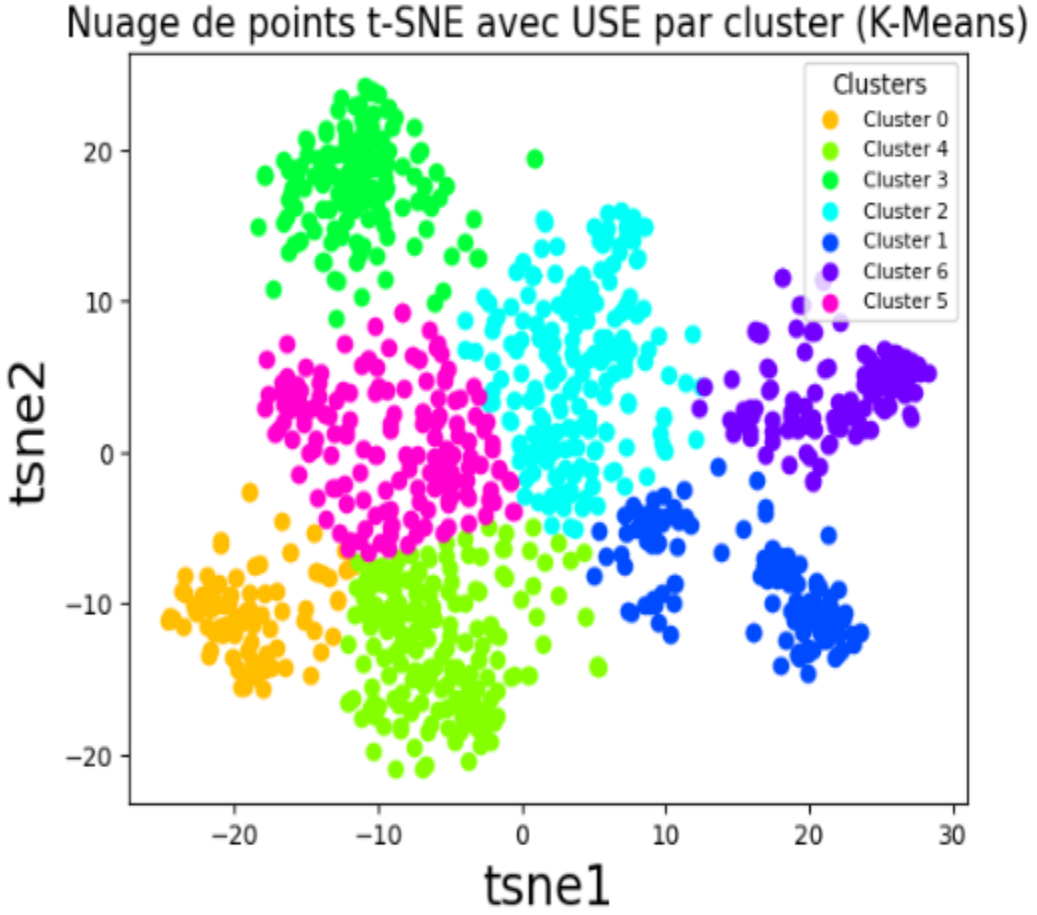



**ARI : 0.03**

# Test du modèle VGG16

- Chargement du modèle
- Extractions des caractéristiques pour toutes les images.
- réduction de dimensions avec PCA/TSNE
- clustering avec KMeans



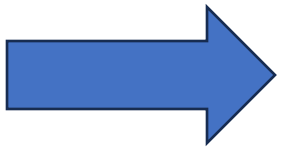




**ARI : 0.55**

# Tableau comparatif des modèles

| Modèle | ORB | VGG16 |
|---|---|---|
| ARI | 0.03 | 0.55 |

Apres comparaison des catégories  et des clusters KMeans on peut voir clairement une certaine similarité surtout avec la vectorisation des images avec VGG16



On peut également conclure à la faisabilité d'un moteur de classification automatique des données de notre jeu de données en utilisant les caractéristiques extraites des images.

# Classification supervisée

- Extraction des caractéristiques avec VGG16
- Entrainement du modèle sur les jeux d'entrainement (batch_size=32, epochs =10)

```
26/26 [==============================] - 185s 7s/step - loss: 0.0525 - accuracy: 0.9950
Epoch 10/10
26/26 [==============================] - 182s 7s/step - loss: 0.0385 - accuracy: 0.9975
Temps total d'entraînement : 0:30:37.255233
```

- Evaluation du modèle sur le jeu de test

```
6/6 [==============================] - 43s 7s/step - loss: 0.7215 - accuracy: 0.7708
[0.7215416431427002, 0.7708333134651184]
```

```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 vgg16 (Functional)          (None, 7, 7, 512)         14714688

 flatten_1 (Flatten)         (None, 25088)             0

 dense_2 (Dense)             (None, 512)               12845568

 dropout_1 (Dropout)         (None, 512)               0

 dense_3 (Dense)             (None, 7)                 3591

=================================================================
Total params: 27563847 (105.15 MB)
Trainable params: 12849159 (49.02 MB)
Non-trainable params: 14714688 (56.13 MB)
_________________________________________________________________
```

| | perte | précision |
|---|---|---|
| Jeu de d'entrainement | 0.02 | 100% |
| Jeu de test | 0.72 | 77% |

un écart très clair entre la performance du modèle sur l'ensemble d'entraînement et sur l'ensemble de test ⇒ peut être une certaine forme de surajustement.

# VGG16 avec data augmentation

Application des transformations (rotation, déformation , étirement ,zoom…) sur les images.

Entrainement du modèle (batch_size=32, epochs =15)

```
Epoch 15/15
26/26 [==============================] - 295s 11s/step - loss: 0.4839 - accuracy: 0.8292
Temps total d'entraînement : 1:14:58.618218
```

Evaluation du modèle

```
6/6 [==============================] - 70s 11s/step - loss: 0.7433 - accuracy: 0.7604
[0.7432541847229004, 0.7604166865348816]
```

| | perte | précision |
|---|---|---|
| Jeu de d'entrainement | 0.48 | 83% |
| Jeu de test | 0.74 | 76% |

# Modification des hyperparamètres

batch_size=64, epochs =15

Jeu d'entrainement:

```
Epoch 15/15
13/13 [==============================] - 309s 23s/step - loss: 0.5327 - accuracy: 0.8125
Temps total d'entraînement : 1:16:58.134149
```

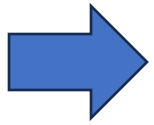
Jeu de test:

```
Matrice de confusion :
[[19  1  4  1  3  1  1]
 [ 1 24  0  0  3  0  2]
 [ 0  3 26  1  0  0  0]
 [ 0  2  1 18  2  4  3]
 [ 5  0  0  1 24  0  0]
 [ 1  0  5  0  0 22  2]
 [ 0  0  0  0  0  0 12]]

Métriques de classification par classe :
                           precision    recall  f1-score   support

                Baby Care       0.73      0.63      0.68        30
  Beauty and Personal Care       0.80      0.80      0.80        30
                Computers       0.72      0.87      0.79        30
Home Decor & Festive Needs       0.86      0.60      0.71        30
          Home Furnishing       0.75      0.80      0.77        30
         Kitchen & Dining       0.81      0.73      0.77        30
                  Watches       0.60      1.00      0.75        12

                 accuracy                           0.76       192
                macro avg       0.75      0.78      0.75       192
             weighted avg       0.77      0.76      0.75       192
```

# Test de l'API

Les 5 principes de RGPD

- **Droits des Individus (RGPD):** le droits d'accès, correction, suppression, transfert et opposition au traitement.
- **Consentement:** un consentement clair et révocable, de manière libre et éclairée, avant de traiter les données personnelles des individus.
- **Responsabilité et Transparence:** Les entreprises doivent assurer la sécurité des données et informer de manière transparente sur leur traitement.
- **Notification des Violations:** En cas de violation, les entreprises doivent signaler l'incident aux autorités et individus concernés dans un délai défini.
- **Sanctions (RGPD):** Des amendes importantes, jusqu'à 4 % du chiffre d'affaires annuel mondial, sont prévues en cas de non-conformité au RGPD



Lors de la récupération des données à partir de l'API ,ces principes ont été respecté.

# Test de API

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

Critère : querystring = {"ingr":"champagne"}

Résultat après traitement :

| | food.foodId | food.label | food.category | food.foodContentsLabel | food.image |
|---|---|---|---|---|---|
| 0 | food_a656mk2a5dmqb2adiamu6beihduu | Champagne | Generic foods | NaN | https://www.edamam.com/food-img/a71/a718cf3c52... |
| 1 | food_b753ithamdb8psbt0w2k9aquo06c | Champagne Vinaigrette, Champagne | Packaged foods | OLIVE OIL; BALSAMIC VINEGAR; CHAMPAGNE VINEGAR... | NaN |
| 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... | NaN |

10 lignes de ce dataframe ont été enregistré dans un fichier csv : "data_rec.csv"

# Conclusion

Les expériences de catégorisation automatique, en utilisant(KMeans ) et les modèles d'extraction de caractéristiques (texte/image), ont démontré la faisabilité d'un moteur de classification automatique.

La classification supervisée des articles en se basant sur les caractéristiques extraites des images a produit des résultats satisfaisants, bien qu'il soit possible de les améliorer.

Merci pour votre attention

Questions ?