Classification de biens de consommation

Étude de faisabilité

Plan de Présentation

Axes principaux

- ❏ Contexte
- ❏ Approche Globale
- ❏ Natural Language Processing
- ❏ Computer Vision
- ❏ Conclusion & Pistes à creuser

Contexte

Le besoin et la ressource

Besoin

- ❏ Linda - Place de marché (PM)
- ❏ Classification par catégories
- ❏ Automatisation
- ❏ Moteur de classification
- ❏ Test de faisabilité

Ressources

- ❏ PM dataset
- ❏ 150 images de biens PM
- ❏ Connaissances Machine Learning
- ❏ Volonté d’apprendre

Approche Globale

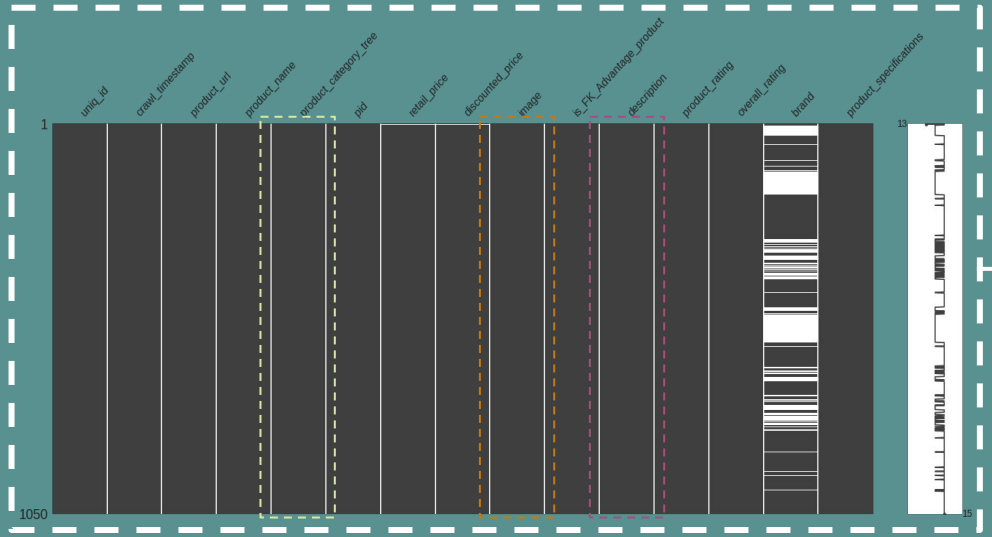
Approche globale

Valable pour NLP & CV

- ❑ Exploration
- ❑ Création des features basiques
- ❑ Vectorization
- ❑ Analyse en Composantes Principales
- ❑ Clustering
- ❑ Observation des résultats

Natural Language Processing

❑ Exploration



product_category_tree

["Home Furnishing >> Curtains & Accessories >>...

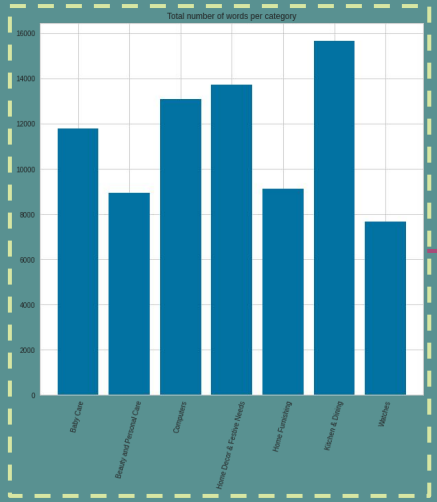
["Baby Care >> Baby Bath & Skin >> Baby Bath T...

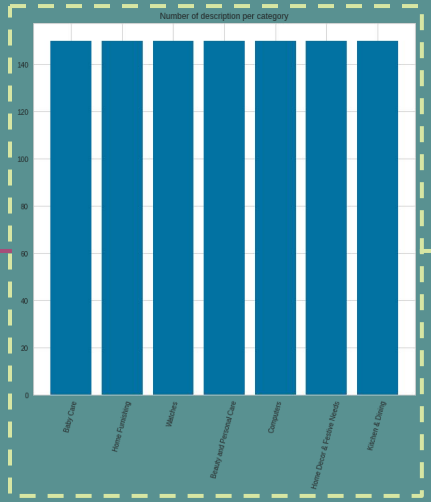
["Baby Care >> Baby Bath & Skin >> Baby Bath T...

```
def clean_it(x):
  t = x.index('"')
  p = x.index('>')
  x = x[t+1:p-1]
  return(x)

data.loc[:,"product_category_tree"] =
data.loc[:,"product_category_tree"].apply(lambda x: clean_it(x))
data.loc[0:3,"product_category_tree"]
```

```
0   Home Furnishing
1        Baby Care
2        Baby Care
3   Home Furnishing
Name: product_category_tree, dtype: object
```





- Exploration
- Basic

Features

```
0: Key Features of Elegance Polyester Multicolor Abstract Eyelet Door Curtain Floral Curtain,Elegance
```

```
0: key features of elegance polyester multicolor abstract eyelet door curtain floral curtain,elegance
```

```
['key',
 'features',
 'of',
 'elegance',
 'polyester',
 'multicolor',
 'abstract',
 'eyelet',
 'door',
 'curtain']
```

```
['key',
 'features',
 'elegance',
 'polyester',
 'multicolor',
 'abstract',
 'eyelet',
 'door',
 'curtain',
 'floral']
```

```
['key',
 'feature',
 'elegance',
 'polyester',
 'multicolor',
 'abstract',
 'eyelet',
 'door',
 'curtain',
 'floral']
```

```python
def clean_sentence(s):
  k  = ' '.join([w for w in s.split() if len(w)>2])
  k = ''.join(' ' if w in string.punctuation else w for w in k)
  k = ''.join('' if w in string.digits else w for w in k)
  return(k)

data.loc[:, "feature_desc"] = data.loc[:, "feature_desc"].apply(lambda x : clean_sentence(x))
data.loc[0, "feature_desc"]
```

```
'key feature elegance polyester multicolor abstract eyelet door curtain floral curtain elegance polye
```

TF-IDF & USE

```
array(['amount', 'ant', 'anti', 'apart', 'apparance', 'appeal',
       'attention', 'beauty', 'body', 'box', 'brand', 'bring', 'close',
       'color', 'contemporary', 'content', 'create', 'designed',
       'dimension', 'duster', 'enhances', 'environment', 'evening',
       'fabric', 'filter', 'first', 'floral', 'general', 'get', 'give',
       'given', 'good', 'heart', 'high', 'interiors', 'joyous', 'key',
       'length', 'light', 'look', 'loving', 'made', 'make', 'material',
       'metal', 'modernistic', 'moment', 'name', 'number', 'price',
       'print', 'quality', 'ray', 'right', 'ring', 'romantic', 'set',
       'shrinkage', 'slide', 'smoothly', 'softly', 'soothing', 'special',
       'specification', 'steal', 'stitch', 'style', 'sun', 'sunlight',
       'sure', 'surreal', 'thing', 'type', 'valance', 'want', 'welcome',
       'whole', 'wish', 'world', 'wrinkle'], dtype=object)
```

```
Topic 0:
usb led fan light portable port flexible mobile keyboard phone
Topic 1:
adapter warranty laptop charger replacement vaio vgn power smartpro cre
Topic 2:
mug ceramic coffee perfect gift design one material give price
Topic 3:
bowl singing oil skin jewellery play reiki cleaning crystal soap
Topic 4:
hair wall sticker bottle applied pot decal surface trait surgical
Topic 5:
laptop color pack skin feature specification type general box package
Topic 6:
product free delivery buy shipping genuine cash day replacement guarantee
```

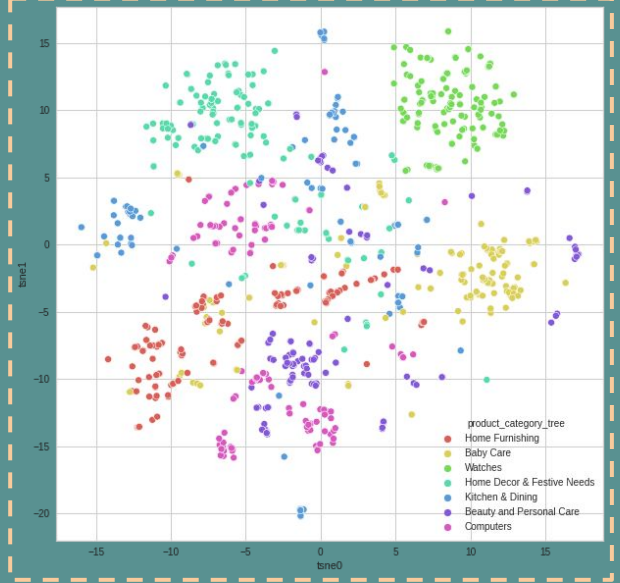
- Exploration
- Basic
- Features
- Vectorization

Term Frequency - Inverse Document Frequency

```
1 tfidf = TfidfVectorizer()
2 res = tfidf.fit_transform(data.loc[:,"feature_desc"])
3 res
```

```
<1050x4717 sparse matrix of type '<class 'numpy.float64'>'
```

min_df 5, max_df 0.8, ngram_range (2,2)



Universal Sentence Encoder

Word embedding versions:

- Model 4
- Model 5 (large)

```
<tf.Tensor: shape=(1050, 512), dtype=float32, numpy=
array([[-0.05352342, -0.05349153,  0.00569391, ...,  0.04749139,
         0.04074656, -0.02473092],
       [-0.04570165, -0.05010772, -0.00906586, ...,  0.02877207,
         0.0305567 ,  0.02340396],
       [-0.05387908, -0.04934131, -0.02578904, ...,  0.0514629 ,
        -0.02304722, -0.04027751],
       ...,
       [-0.01715195, -0.05364091, -0.03726887, ..., -0.02666359,
         0.0432179 , -0.05612453],
       [-0.0214415 , -0.05364957,  0.02223406, ...,  0.01931539,
        -0.0123534 , -0.0035401 ],
       [-0.04491266, -0.05160829,  0.00859575, ...,  0.02589196,
        -0.0143569 ,  0.00464632]], dtype=float32)>
```

- Exploration
- Basic
- Features
- Vectorization
- ACP

Term Frequency - Inverse Document Frequency

```
1 pca = PCA(n_components=0.90)
2 pca.n_components_
```

```
541
```

```
1 svd.explained_variance_ratio_.cumsum()[541]
```

```
0.900313445854019
```

Universal Sentence Encoder

Model 4

```
1 pca = PCA(n_components=0.99)
2 pca_USE = pca.fit_transform(USE)
3 pca.n_components_
```

```
320
```

```
1 pca = PCA(n_components=0.90)
2 pca_USE = pca.fit_transform(USE)
3 pca.n_components_
```

```
121
```

Model 5

```
1 pca = PCA(n_components=0.99)
2 pca_USE = pca.fit_transform(USE)
3 pca.n_components_
```

```
267
```

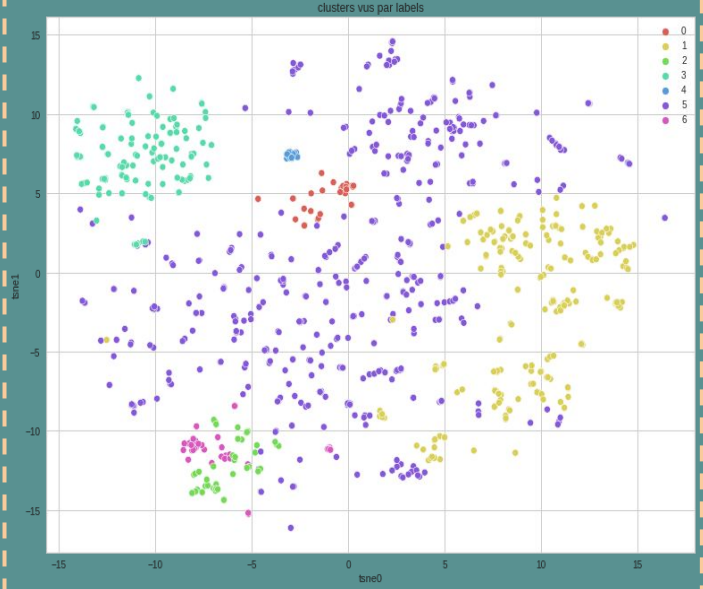
```
1 pca = PCA(n_components=0.90)
2 pca_USE = pca.fit_transform(USE)
3 pca.n_components_
```

```
112
```

category
- Home Furnishing
- Baby Care
- Watches
- Home Decor & Festive Needs
- Kitchen & Dining
- Beauty and Personal Care
- Computers

Term Frequency - Inverse Document Frequency

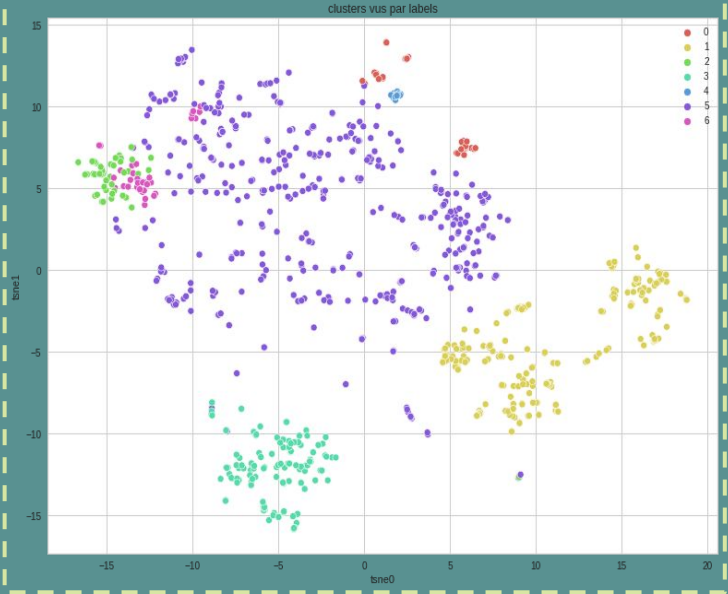
- ❑ Exploration
- ❑ Basic Features
- ❑ Vectorization
- ❑ ACP
- ❑ Clustering



category
- Home Furnishing
- Baby Care
- Watches
- Home Decor & Festive Needs
- Kitchen & Dining
- Beauty and Personal Care
- Computers

Universal Sentence Encoder

- ❑ Exploration
- ❑ Basic Features
- ❑ Vectorization
- ❑ ACP
- ❑ Clustering

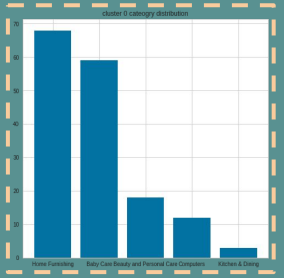


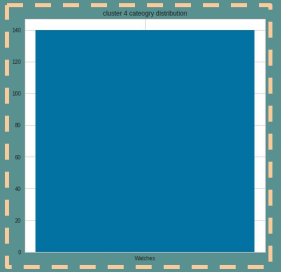
- ❑ Exploration
- ❑ Basic
- Features
- ❑ Vectorization
- ❑ ACP
- ❑ Clustering
- ❑ Résultats

Term Frequency - Inverse Document Frequency

```
1 adjusted_rand_score(data.loc[:,"cluster_category"], data.loc[:,"cluster_label"])
```

```
0.2986582878158066
```





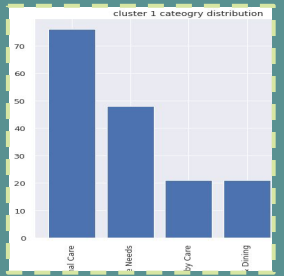
Universal Sentence Encoder

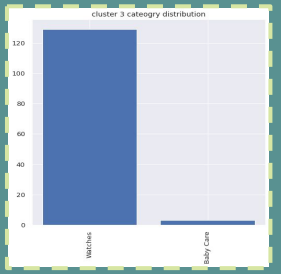
```
1 adjusted_rand_score(data.loc[:,"cluster_category"], data.loc[:,"cluster_label_USE"])
```

```
0.31556804577012343
```

```
1 adjusted_rand_score(data.loc[:,"cluster_category"], data.loc[:,"cluster_label_USE"])
```

```
0.3646122263768904
```

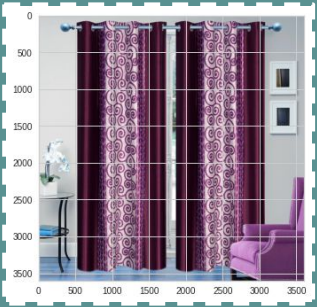




Computer Vision

- Exploration

	image	product_category_tree
0	55b85ea15a1536d46b7190ad6fff8ce7.jpg	Home Furnishing
1	7b72c92c2f6c40268628ec5f14c6d590.jpg	Baby Care
2	64d5d4a258243731dc7bbb1eef49ad74.jpg	Baby Care
3	d4684dcdc759dd9cdf41504698d737d8.jpg	Home Furnishing

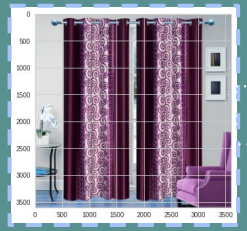


- Exploration
- Basic
- Features

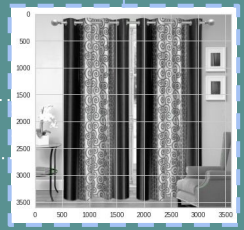
SIFT

ORB

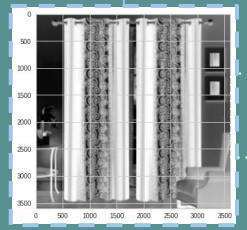
RESNET50



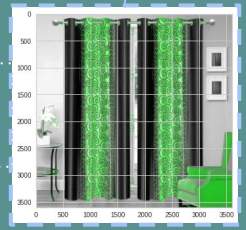
Start



Equaliser



Noise cancelling



Descriptor calculation

```
(9051348, 128)
```

```
(520153, 32)
```

```
5 model = ResNet50(weights='imagenet',
6                   include_top=False,
7                   input_shape=(224, 224, 3))
```

```
[(None, 224, 224, 3
)]
```

```
(None, 7, 7, 2048)   0
```

- Exploration
- Basic
- Features
- Vectorization

SIFT

ORB

RESNET50

```
MiniBatchKMeans(init_size=9027, n_clusters=3009, random_state=0)
```

```
MiniBatchKMeans(init_size=2163, n_clusters=721, random_state=0)
```

```
img = image.load_img(img_path, target_size=(224, 224, 3))
```

```
x = preprocess_input(x)
x = np.expand_dims(x, axis=(0))
```

Histogramme

```
conv5_block3_out (Activation)  (None, 7, 7, 2048)  0
```

```
1 im_features_sift.shape
```

```
(1050, 3009)
```

```
1 im_features_orb.shape
```

```
(1050, 721)
```

```
1 print(retail_list_all.shape)
```

```
(1050, 100352)
```

- Exploration
- Basic Features
- Vectorization
- ACP

SIFT

```
Dimensions dataset avant réduction PCA :  (1050, 3009)
Dimensions dataset après réduction PCA :  (1050, 741)
```

ORB

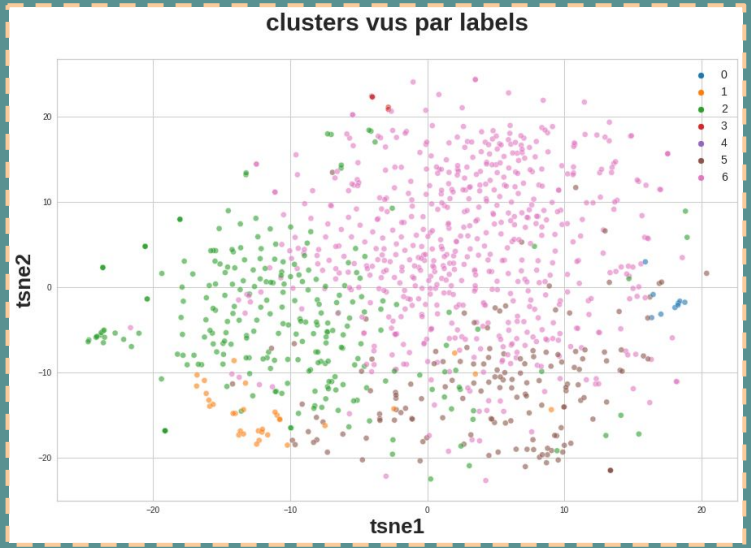
```
Dimensions dataset avant réduction PCA :  (1050, 721)
Dimensions dataset après réduction PCA :  (1050, 578)
```

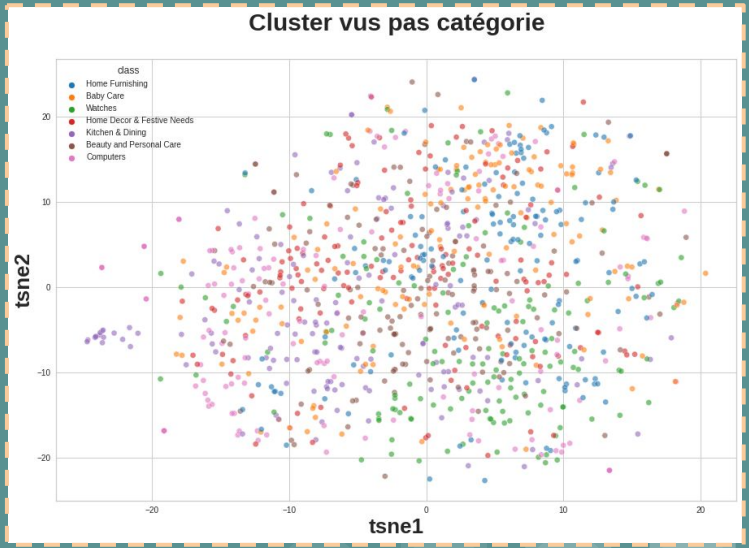
RESNET50

```
Dimensions dataset avant réduction PCA :  (1050, 100352)
Dimensions dataset après réduction PCA :  (1050, 976)
```

SIFT

- ❑ Exploration
- ❑ Basic Features
- ❑ Vectorization
- ❑ ACP
- ❑ Clustering

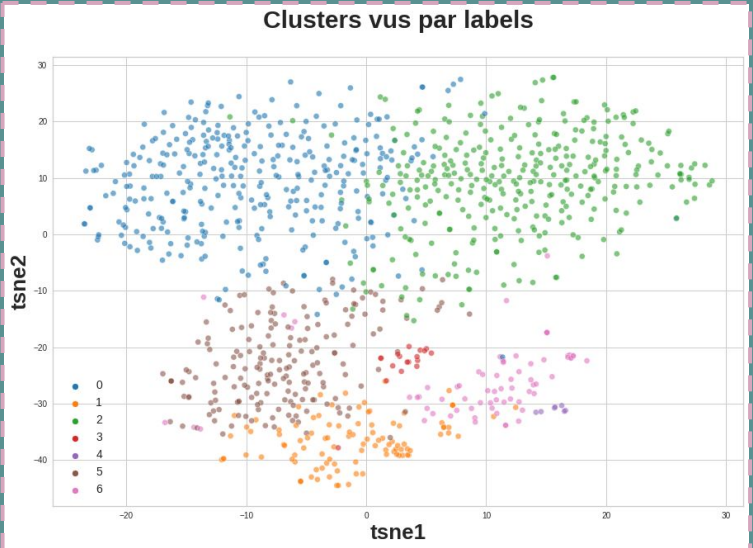


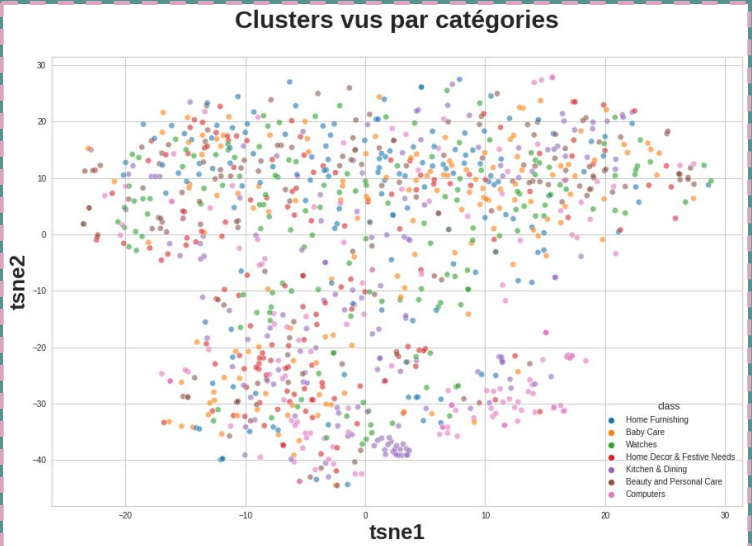


class
- Home Furnishing
- Baby Care
- Watches
- Home Decor & Festive Needs
- Kitchen & Dining
- Beauty and Personal Care
- Computers

ORB

- Exploration
- Basic Features
- Vectorization
- ACP
- Clustering

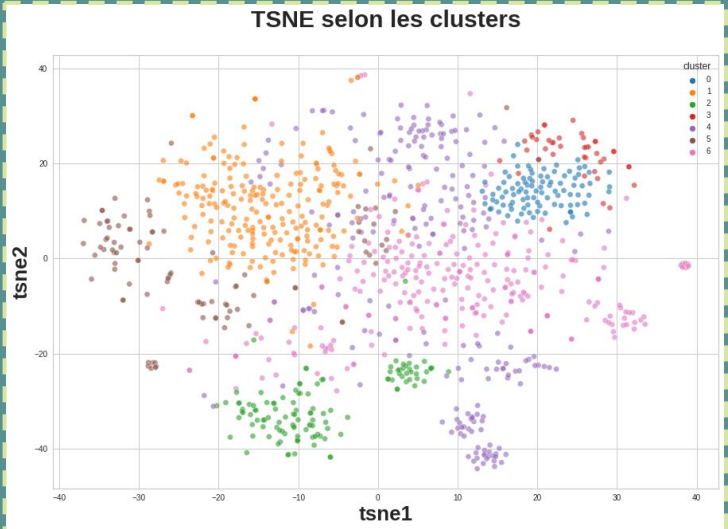


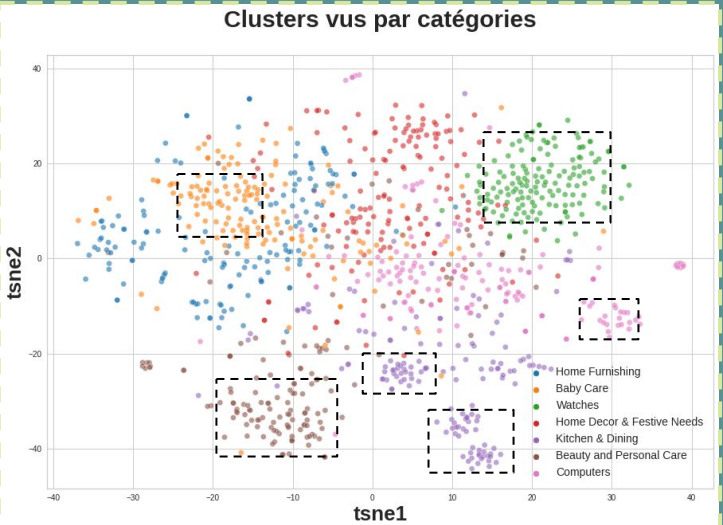


class
- Home Furnishing
- Baby Care
- Watches
- Home Decor & Festive Needs
- Kitchen & Dining
- Beauty and Personal Care
- Computers

RESNET50

- ❑ Exploration
- ❑ Basic Features
- ❑ Vectorization
- ❑ ACP
- ❑ Clustering

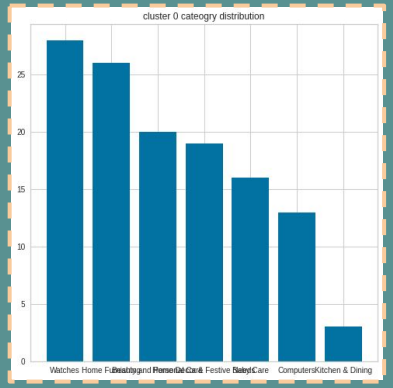


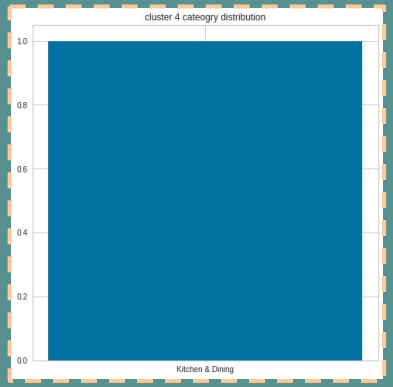


- ❑ Exploration
- ❑ Basic Features
- ❑ Vectorization
- ❑ ACP
- ❑ Clustering
- ❑ Résultats

SIFT

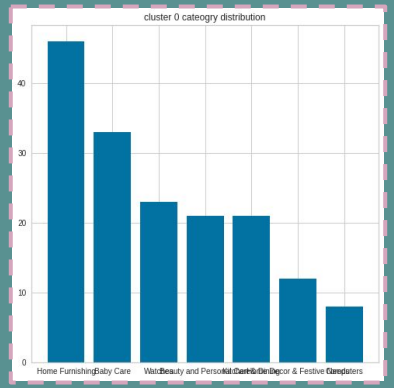
ARI : 0.06581117583105336

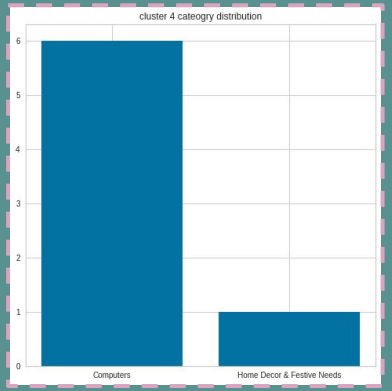




ORB

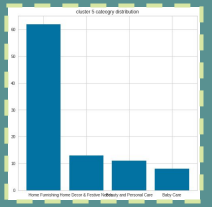
ARI : 0.032785731621232224

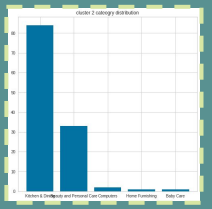


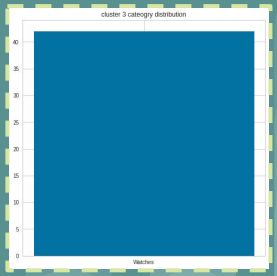


RESNET50

ARI : 0.32334168053878687







Conclusion & Pistes à creuser

- Peu de données, pas complètement propre
- Score ari proche des 0.5, signal positif
- NLP et CV peuvent se compléter (kitchen dining CV, home decor & festive NLP)
- Sans le nettoyage les watches avaient de gros résultats (pipelines par catégories), jeu sur les ngram
- Peu d'affinage sur les modèles et features:
    - longueur des descriptions par exemple
    - seulement 90% de l'info en pca
    - aller plus loin dans les sous catégories pour les groupes qui en ont besoin, etc...
    - pipelines, hyper paramètres...
    - modèles supervisés